KUANT Guides

Guide No.
KUANT 011.3

Missing Data in Large Data Projects:

Two methods of missing data imputation when working with large data projects.

Little, T.D., Howard, W.J., McConnell, E.K., & Stump K.N. (2011)

Preliminaries

This handout outlines *two* methods of missing data imputation when working with *large data* projects. The first one involves imputation from *aggregated scores*, and the second begins with a *principal components analysis.* This guide is recommended *only* for situations in which imputation of the full item set does not work.

In the following two examples, there are 10 scales where each scale consists of 8 - 10 items for a total of 97 items in the data set.

Method One: Imputing at the level of aggregated scales

1) Create aggregate scales of the items on the data set (use the average).

```
ScaleA = mean(of a1 a2 a3 a4 a5 a6 a7 a8 a9 a10);
ScaleB = mean(of b1 b2 b3 b4 b5 b6 b7 b8 b9 b10);
ScaleC = mean(of c1 c2 c3 c4 c5 c6 c7 c8 c9 c10);
ScaleD = mean(of d1 d2 d3 d4 d5 d6 d7 d8 d9 d10);
ScaleE = mean(of e1 e2 e3 e4 e5 e6 e7 e8 e9 e10);
ScaleF = mean(of f1 f2 f3 f4 f5 f6 f7 f8 f9 f10);
ScaleG = mean(of g1 g2 g3 g4 g5 g6 g7 g8 g9 g10);
ScaleH = mean(of h1 h2 h3 h4 h5 h6 h7 h8 h9 h10);
ScaleI = mean(of i1 i2 i3 i4 i5 i6 i7 i8);
ScaleJ = mean(of j1 j2 j3 j4 j5 j6 j7 j8 j9);
```

a) Notes

In this example, notice that the items are *averaged*. If the items for a particular scale are *summed* rather than averaged, the scales should be standardized before they are imputed. However, if the scales are created using averages then standardizing is *not needed* or recommended because the metric of the scales is lost through standardization.

2) Impute at the level of the aggregate scales.
The SAS Proc MI syntax will look something like this:

```
Proc mi data=sample out=outmi nimpute=1 seed=761253;
em maxiter=1000;
mcmc chain=multiple initial=em (maxiter = 1000);
 var ScaleA ScaleB ScaleC ScaleD ScaleE ScaleF ScaleG ScaleH ScaleI ScaleJ;
run;

*For Proc mi options see: http://support.sas.com/onlinedoc/913/docMainpage.jsp;
```

*At this step the **scales** now have NO missing data but the **items** still contain missing data*

3) Use the imputed scales as anchors to impute missing data in the items in a sequential process

Now the key is to put the right variables into the variable list for estimating the missing data:

```
Proc mi data=outmi out=outmi......;
  var ScaleA ScaleB ScaleC ScaleD ScaleE ScaleF ScaleG ScaleH ScaleI
        j1 j2 j3 j4 j5 j6 j7 j8 j9;
* At this step ScaleJ is not in the list but the items for ScaleJ are now imputed;
* ScaleJ needs to be removed because it is a linear combination of its items;

Proc mi data=outmi out=outmi......;
  var ScaleA ScaleB ScaleC ScaleD ScaleE ScaleF ScaleG ScaleH       ScaleJ
        i1 i2 i3 i4 i5 i6 i7 i8;
* At this step ScaleI is not in the list but the items for ScaleI are now imputed;

Proc mi data=outmi out=outmi......;
  var ScaleA ScaleB ScaleC ScaleD ScaleE ScaleF ScaleG        ScaleI ScaleJ
        h1 h2 h3 h4 h5 h6 h7 h8 h9 h10;
* At this step ScaleH is not in the list but the items for ScaleH are now imputed;

Proc mi data=outmi out=outmi......;
  var ScaleA ScaleB ScaleC ScaleD ScaleE ScaleF        ScaleH ScaleI ScaleJ
        g1 g2 g3 g4 g5 g6 g7 g8 g9 g10;
* At this step ScaleG is not in the list but the items for ScaleG are now imputed;

Proc mi data=outmi out=outmi......;
  var ScaleA ScaleB ScaleC ScaleD ScaleE         ScaleG ScaleH ScaleI ScaleJ
        f1 f2 f3 f4 f5 f6 f7 f8 f9 f10;
* At this step ScaleF is not in the list but the items for ScaleF are now imputed;

Proc mi data=outmi out=outmi......;
  var ScaleA ScaleB ScaleC ScaleD        ScaleF ScaleG ScaleH ScaleI ScaleJ
        e1 e2 e3 e4 e5 e6 e7 e8 e9 e10;
* At this step ScaleE is not in the list but the items for ScaleE are now imputed;

Proc mi data=outmi out=outmi......;
  var ScaleA ScaleB ScaleC       ScaleE ScaleF ScaleG ScaleH ScaleI ScaleJ
        d1 d2 d3 d4 d5 d6 d7 d8 d9 d10;
* At this step ScaleD is not in the list but the items for ScaleD are now imputed;

Proc mi data=outmi out=outmi......;
  var ScaleA ScaleB       ScaleD ScaleE ScaleF ScaleG ScaleH ScaleI ScaleJ
        c1 c2 c3 c4 c5 c6 c7 c8 c9 c10;
* At this step ScaleC is not in the list but the items for ScaleC are now imputed;

Proc mi data=outmi out=outmi......;
  var ScaleA        ScaleC ScaleD ScaleE ScaleF ScaleG ScaleH ScaleI ScaleJ
        b1 b2 b3 b4 b5 b6 b7 b8 b9 b10;
* At this step ScaleB is not in the list but the items for ScaleB are now imputed;

Proc mi data=outmi out=outmi......;
  var       ScaleB ScaleC ScaleD ScaleE ScaleF ScaleG ScaleH ScaleI ScaleJ
        a1 a2 a3 a4 a5 a6 a7 a8 a9 a10;
* At this step ScaleA is not in the list but the items for ScaleA are now imputed;
```

*At this point ALL of the **items** are imputed.*

Method Two: Imputing with Principal Components

This section outlines *a* method of using principal component analysis (PCA) to obtain auxiliary variables for *large data* sets. The principal components method is then applied in a manner similar to the aggregated scales approach illustrated above.

Large data projects can present an additional challenge because it is possible to have an excess number of potential auxiliary variables. Because large data sets can be difficult to impute, including additional auxiliary variables may not seem attractive. However, research has shown that auxiliary variables can provide extremely important information to the imputation process (see Enders, 2010). Further, when researchers incorporate non-linear information (which is likely to be important to the imputation model), the number of auxiliary variables alone can become unwieldy. For example, there are 45 second order and 120 third order interaction terms associated with just 10 auxiliary variables.

The current method addresses these limitations by using PCA to reduce the dimensionality of the auxiliary variables and nonlinear information in a data set. A new smaller set of auxiliary variables are created (e.g., principal component scores) that contain all the useful information from the original data set. These principal component scores are then used to inform the missing data handling procedure.

It is important to note that this method is not limited to an imputation model with aggregated scores; that is, PCA auxiliary variables can be used in typical FIML or multiple imputation procedure where auxiliary variables are included.

1) Use the file import drop down menu or SAS proc import to get your data into SAS.
The SAS Proc Import syntax for importing a SPSS file will look something like this:

```
PROC IMPORT OUT= WORK.sample
            DATAFILE= "C:\Users\myname\Desktop\mydata.sav"
            DBMS=SPSS REPLACE;
RUN; *Check your data to ensure that it imported correctly;
```

a) Note:

Check your data to ensure that it imported correctly (see illustration below). For this example assume that this is a portion of the data set used previously (including the following variables: a1-a10, b1-b10, c1-c10, d1-d10, e1-e10, f1-f10, g1-g10, h1-h10, i1-i8, and j1-j9) with the addition of 10 auxiliary variables (aux1-aux10).

Original Data (data=sample)

	a1	a2	a3	a4	a5	aux1	aux2	aux3
1	2	3	.	3	4	4	5	4
2	4	1	3	1	3	3	3	2
3	5	3	1	4	5	4	3	4
4	5	3	1	4	5	.	4	4
5	5	2	.	1	5	5	3	3
6	5	1	1	2	5	5	2	4
7	3	2	2	1	2	.	2	2
8	5	3	.	4	5	3	5	3

Notice that in this example both the analysis variables and the auxiliary variables contain missing data.

2) Capture non-linear information

a) Step 1:

Use the SAS MACRO below to obtain squares and interaction terms for any set of variables. First, run the following code to load the SAS MACRO. (*Do not edit %MACRO code below*).

```
/*Do not edit this syntax*/
%MACRO interact(vars, quadr = 1, prefix = INT);
%LET c=1;
%DO %WHILE(%SCAN(&vars,&c) NE);
     %LET c=%EVAL(&c+1);
%END;
%LET nvars=%EVAL(&c-1);
%DO i = 1 %TO &nvars;
%DO j = %EVAL(&i+1-&quadr) %TO &nvars;
      &prefix._%SCAN(&vars,&i)_%SCAN(&vars,&j) =
                %SCAN(&vars,&i) * %SCAN(&vars,&j);
%END; %END;
%MEND;
```

b) Step 2:

Next, enter your data set name and your list of variables. *This step calls the SAS MACRO and uses it with your data*.

```
/*You must edit this syntax*/
DATA myinterdata; SET sample;
%INTERACT(myvariables ,quadr=1);
/* (1) replace "sample" with your data set name, (2) Replace "my variables" with
your analysis and auxiliary variable names, (3) set quadr option: 1 = all possible
quadratic terms and 2-way interactions OR 2 = all possible cubic terms and 3-way
interactions */
RUN;
```

c) Note:

Again, check your data to ensure that it is correct (see illustration below). Consider the variables *INT_a1_a1* and *INT_a1_a2* in this example. *INT_a1_a1* is the a1 variables squared (i.e., $2^2 = 4$) and *INT_a1_a2* is the interaction term for a1 and a2 (i.e., 2 * 3 = 6). Also, note that missing values are generated when at least one term is missing (see *INT_a1_a3* below).

Nonlinear Terms Data (data=myinterdata)

	aux1	aux2	aux3	INT_a1_a1	INT_a1_a2	INT_a1_a3	INT_a1_a4	INT_a2_a2
1	4	5	4	4	6	.	6	9
2	3	3	2	16	4	12	4	1
3	4	3	4	25	15	5	20	9
4	.	4	4	25	15	5	20	9
5	5	3	3	25	10	.	5	4
6	5	2	4	25	5	5	10	1
7	.	2	2	9	6	6	3	4
8	3	5	3	25	15	.	20	9

Rather than entering all of your variable names in the above INTERACT macro, you can have SAS insert them for you (see the Helpful SAS Code at the end of this guide).

3) Estimate any missing data on the auxiliary variables and items.
The SAS Proc MI syntax will look something like this:

```
Proc mi data=myinterdata out=outmi nimpute=1 seed=461981;
em maxiter=1000; mcmc chain=multiple initial=em (maxiter=1000);
run;
```

a) Notes

Because auxiliary variables are likely to contain missing values, a single regression imputation can be used as an intermediate step to acquire a complete data set for the principal component analysis. If this step fails to converge, consult the Helpful SAS Code for the PCA method at the end of this guide. At this step the **auxiliary variables** and the **items** now have NO missing data (see illustration below).

Initial Imputation Data (data=outmi)

	a1	a2	a3	a4	a5	aux1	aux2	aux3
1	2	3	1.6186855347	3	4	4	5	4
2	4	1	3	1	3	3	3	2
3	5	3	1	4	5	4	3	4
4	5	3	1	4	5	5.1578012925	4	4
5	5	2	3.9287720842	1	5	5	3	3
6	5	1	1	2	5	5	2	4
7	3	2	2	1	2	2.8917554355	2	2
8	5	3	3.6038575247	4	5	3	5	3

4) Standardize the imputed data set.
The SAS Proc STANDARD syntax will look something like this:

```
proc standard data=outmi out=mystddata mean=0 std=1;
run;
```

a) Notes

The data are standardized because there are potentially large differences in the auxiliary variable variances. These differences would allow variables with large variances to contribute more information to the PC scores than equally important variables with smaller variances. Standardization removes the problem of scale dependence from the PC scores. See illustration below.

Standardized Data (data=mystddata)

	a1	a2	a3	a4	a5	aux1	aux2	aux3
1	-0.983797151	0.2226900321	-0.956903617	0.108527669	0.3711850773	0.4241969774	1.5446428665	0.7730139232
2	0.5249188794	-1.457029067	0.0670805009	-1.536150558	-0.439797363	-0.470998465	-0.252171397	-0.938425019
3	1.2792768948	0.2226900321	-1.415542238	0.9308667827	1.1821675175	0.4241969774	-0.252171397	0.7730139232
4	1.2792768948	0.2226900321	-1.415542238	0.9308667827	1.1821675175	1.4606554179	0.6462357349	0.7730139232
5	1.2792768948	-0.617169517	0.7555898065	-1.536150558	1.1821675175	1.3193924201	-0.252171397	-0.082705548
6	1.2792768948	-1.457029067	-1.415542238	-0.713811445	1.1821675175	1.3193924201	-1.150578528	0.7730139232
7	-0.229439136	-0.617169517	-0.674230869	-1.536150558	-1.250779803	-0.567898506	-1.150578528	-0.938425019
8	1.2792768948	0.2226900321	0.5147269494	0.9308667827	1.1821675175	-0.470998465	1.5446428665	-0.082705548

*At this step the **auxiliary variables** and the **items** are all standardized. This is equivalent to preforming a PCA on a correlation matrix rather than a covariance matrix. Note that SAS proc factor and SAS proc princomp can be used to obtain principal component scores from a covariance or correlation matrix; however, the eigenvectors obtained are multiplied by the data matrix rather than the raw data. Therefore, SAS proc factor and SAS proc princomp produce variable specific component scores rather than case specific component scores.*

1) Do a principal components analysis of the items and auxiliary variables.
The *SAS Proc PRINCOMP* syntax (*Proc FACTOR* may also be used) will look something like this:

```
Proc princomp data=mystddata out=myprindata;
run;

*For Proc princomp options: http://support.sas.com/onlinedoc/913/docMainpage.jsp;
```

a) Notes

At this point all of a large number of principal components are added to the end of the data set (see illustration below). Note that SAS will provide output information to aid in determining the number of components to retain.

Principal Component Scores with Data (data=myprindata)

	aux2	aux3	Prin1	Prin2	Prin3	Prin4	Prin5	Prin6
1	1.5446428665	0.7730139232	0.7599735627	-1.79340938	-0.502645214	-0.359436657	-0.373847921	-0.256350445
2	-0.252171397	-0.938425019	-1.789316245	0.0394019324	1.0143849886	0.7817345192	1.0966721097	0.1761981236
3	-0.252171397	0.7730139232	1.0569373611	-0.018111219	1.4655038214	-0.633826484	-1.332034371	-1.159349902
4	0.6462357349	0.7730139232	1.761674152	-0.893327721	1.6038257894	-0.990733297	-0.739055131	-0.974528088
5	-0.252171397	-0.082705548	0.4967430979	-0.149344187	1.8280502832	0.752960578	1.58476285	-0.323269863
6	-1.150578528	0.7730139232	-0.245573084	-0.950973414	2.5050245011	0.3199734544	-0.437738629	-1.430360686
7	-1.150578528	-0.938425019	-2.572259951	0.0822712341	0.3295091934	-0.201037402	0.2973378813	0.4560704901
8	1.5446428665	-0.082705548	1.7489136286	0.661429563	0.6133016002	0.3043379269	0.2837334166	-0.342765087

2) Output the meaningful components as component scores.
The SAS DATA step syntax will look something like this:

```
/* keep the important principal component scores */
DATA data=myNEWprindata set=myprindata;
Keep Prin1-Prin12;
RUN;
```

a) Notes:

In this example, let's assume that the first 12 components accounted for a meaningful amount of variance and any more components would not yield much more reliable information. Notice that the data set below contains only the important principal component scores (i.e., no analysis or auxiliary variables are included).

Only Principal Component Scores Data (data=myNEWprindata)

	Prin1	Prin2	Prin3	Prin4	Prin5	Prin6	Prin7	Prin8
1	0.7599735627	-1.79340938	-0.502645214	-0.359436657	-0.373847921	-0.256350445	-0.873946498	-0.317961913
2	-1.789316245	0.0394019324	1.0143849886	0.7817345192	1.0966721097	0.1761981236	-0.183869562	0.1067270067
3	1.0569373611	-0.018111219	1.4655038214	-0.633826484	-1.332034371	-1.159349902	0.0206474652	0.0964413246
4	1.761674152	-0.893327721	1.6038257894	-0.990733297	-0.739055131	-0.974528088	-0.008525803	0.3889709838
5	0.4967430979	-0.149344187	1.8280502832	0.752960578	1.58476285	-0.323269863	1.0032739677	-0.597125521
6	-0.245573084	-0.950973414	2.5050245011	0.3199734544	-0.437738629	-1.430360686	1.3238059977	0.0164650951
7	-2.572259951	0.0822712341	0.3295091934	-0.201037402	0.2973378813	0.4560704901	0.3784403602	-0.356942927
8	1.7489136286	0.661429563	0.6133016002	0.3043379269	0.2837334166	-0.342765087	-1.581846426	0.4056345189

At this point the dimensionality of the original data set, consisting of a large number of interrelated analysis and auxiliary variables, has been reduced. Most of the variance (linear and non-linear) among the original variables has been retained. As illustrated above, this can be achieved by transforming the original variables to a new set of uncorrelated variables (PC scores), which are ordered with the first few components retaining most of the variation present in all of the original variables.

3) Combine the important *component scores* with the *original* data.
The SAS DATA step syntax will look something like this:

```
/* merge raw variables with principal component variables */
DATA mydatatoimpute;
MERGE sample myNEWprindata;
RUN;
```

a) Notes

Now the newly created PC scores are added to the end of the original data file. If you plan to use FIML or multiple imputation without using aggregated scores, you can now export the data and treat the component scores (e.g., Prin1 - Prin12) as auxiliary variables. In the illustration below, notice that the principal components are simply added to the end of the original data set.

Merged Data (data= mydatatoimpute)

	a1	a2	a3	a4	a5	Prin1	Prin2	Prin3
1	2	3	.	3		0.7599735627	-1.79340938	-0.502645214
2	4	1	3	1		-1.789316245	0.0394019324	1.0143849886
3	5	3	1	4		1.0569373611	-0.018111219	1.4655038214
4	5	3	1	4		1.761674152	-0.893327721	1.6038257894
5	5	2	.	1		0.4967430979	-0.149344187	1.8280502832
6	5	1	1	2		-0.245573084	-0.950973414	2.5050245011
7	3	2	2	1		-2.572259951	0.0822712341	0.3295091934
8	5	3	.	4		1.7489136286	0.661429563	0.6133016002

4) Use the estimated component scores as anchors to estimate the missing data in the items on the dataset.

```
Proc mi data=mydatatoimpute out=mydatatoimpute......;
  var Prin1-Prin12 a1-a10 b1-b10;
* At this step we plug approximately 20 items into the variable list to estimate
  their missing values;

Proc mi data=mydatatoimpute out=mydatatoimpute......;
  var Prin1-Prin12 c1-c10 d1-d10;
* At this step we plug in approximately 20 more items into the list to estimate
  their missing values;

Proc mi data=mydatatoimpute out=mydatatoimpute......;
  var Prin1-Prin12 e1-e10 f1-f10;
* At this step we plug in approximately 20 more items into the list to estimate
  their missing values;

Proc mi data=mydatatoimpute out=mydatatoimpute......;
  var Prin1-Prin12 g1-g10 h1-h10;
* At this step we plug in approximately 20 more items into the list to estimate
  their missing values;

Proc mi data=mydatatoimpute out=mydatatoimpute......;
  var Prin1-Prin12 i1-i8 j1-j9;
* At this step we plug the remaining items to estimate their missing values;
```

*At this point ALL of the **items** are imputed.*

What about Multiple Imputations:

1) Change the number of imputations option (*nimpute*=) in the Proc MI syntax

The SAS Proc MI syntax will look something like this:

```
Proc mi data=outmi out=outmi nimpute=20.....;
  var ScaleA ScaleB ScaleC ScaleD ScaleE ScaleF ScaleG ScaleH ScaleI
        j1 j2 j3 j4 j5 j6 j7 j8 j9;
* At this step ScaleJ is not in the list but the items for ScaleJ are now imputed;

Proc mi data=outmi out=outmi nimpute=20.....;
  var ScaleA ScaleB ScaleC ScaleD ScaleE ScaleF ScaleG ScaleH       ScaleJ
        i1 i2 i3 i4 i5 i6 i7 i8;
* At this step ScaleI is not in the list but the items for ScaleI are now imputed;
```

a) Notes

Multiple imputations should be carried out during **Step 3** of the aggregate scale example and **Step 5** of the principal component example. It is not suggested to multiply impute the initial aggregate scales or the initial component scores, as in Step 2. Therefore, multiple imputations are created *only* for the imputed items.

Note that the input and output datasets in the SAS syntax do *not* change with each imputation step; this stacks the imputed datasets on top of each other 1 – 20. If the original dataset contained 200 observations then the new data set with 20 imputations will contain 4000 observations with the 201st observation being the first observation in the 2nd imputation. SAS will create a new variable called "_imputation_" with a value of 1 for the first imputation, 2 for the second imputation, and so on to a value of 20 for the last imputation.

*At this point ALL of the **items** are imputed 20 times.*

For additional information regarding further analysis of a dataset with multiple imputations in LISREL see the Rubin's Rules Pooler for LISREL Estimates:

http://www.quant.ku.edu/resources/tools.html

Helpful SAS code for the aggregate scale method:

a) Creating a scale based on item means.
The SAS Proc Standard syntax will look something like this:

```
Data sampleNew ; set sample ;
ScaleA = mean(of a1 a2 a3 a4 a5 a6 a7 a8 a9 a10);
ScaleB = mean(of b1 b2 b3 b4 b5 b6 b7 b8 b9 b10);
... ;
run;
```

b) Standardize before imputing (when a scale is created from item SUM).
The SAS Proc Standard syntax will look something like this:

```
Proc standard data=sample mean=0 std=1 out=zsample;
var ScaleA ScaleB ScaleC ScaleD ScaleE ScaleF ScaleG ScaleH ScaleI ScaleJ;
run;
```

Helpful SAS code for the PCA method:

When the initial imputation (*see Step 3*) fails to converge see below:

1) Use Proc Standard to standardize data set.
The SAS Proc MI syntax will look something like this:

```
proc standard data=myinterdata out=mystddata mean=0 std=1;
run;
```

2) Do a principal components analysis of the items and auxiliary variables.
The *SAS Proc PRINCOMP* syntax (*Proc FACTOR* may also be used) will look something like this:

```
Proc princomp data=mystddata out=myprindata;
run;
```

3) Estimate any missing data on the principal components.
The SAS Proc MI syntax will look something like this:

```
Proc mi data=mystddata out=outmi nimpute=1 seed=461981;
em maxiter=1000; mcmc chain=multiple initial=em (maxiter=1000);
run;
```

a) Notes

Typically, when a PCA is calculated from raw data the variables that contain missing values are omitted listwise (and the associated principal component scores are missing). This method only imputes the missingness among the PC scores, rather than imputing prior to creating the PC scores. This procedure results in PC scores that are not technically principal components because they are no longer uncorrelated and it is therefore not possible to recreate the original variables from the PC scores and eigenvalues.

When all possible power terms and interactions are needed:

4) Use a SAS Macro to call all variable names.
The SAS Proc MI syntax will look something like this:

```
PROC CONTENTS DATA = work.sample /*<---replace with a data set name*/
OUT=VAR_NAMES(KEEP = NAME) ; *NOPRINT;
RUN;

DATA _NULL_;
FILE 'C:\Users\Public\Desktop\build'; /*<---replace with a location*/
SET VAR_NAMES END = FINIS;
IF _N_ = 1 THEN PUT "%LET VAR_LIST = ";
PUT NAME;
IF FINIS THEN DO;
CALL SYMPUT('NUM' , COMPRESS(_N_));
PUT ';' ;
PUT 'RUN;' ; END; RUN;
%INCLUDE 'C:\Users\Public\Desktop\build'; /*<---same location as above*/

DATA work.sample; SET work.myinterdata;
%INTERACT(&VAR_LIST,quadr=1);
RUN;
```