

Creating RTF Tables with Univariate Analyses of Multiply Imputed Data

Marie Gantz, Ph.D., RTI International, Research Triangle Park, NC

ABSTRACT

Simple univariate analyses acquire an additional level of complexity when multiply imputed data are involved. Directly applied, chi-square and Student's t-tests do not account for between- and within-imputation variance. Instead, appropriate tests must be performed for each imputation separately and the individual estimates combined into one overall result. The macros described in this paper automate the process of performing univariate tests for differences between the two levels of a binary variable in a multiple imputation setting. The GLM procedure is used to test for differences with respect to a continuous variable, and the GENMOD procedure is used to test for differences with respect to a binomial or ordinal multinomial variable. In either case, the MIANALYZE procedure is then used to combine the parameter and covariance estimates from the individual imputations into one overall result. An additional macro outputs the combined results into an attractive RTF table.

INTRODUCTION

Univariate analyses are often conducted as a preliminary step in analyzing a data set. For example, consider a hypothetical study in which pregnant women diagnosed with depression are randomly assigned to two groups. Participants in the intervention group receive counseling, while women in the control group do not. Women are interviewed at baseline and at a follow-up time point toward the end of pregnancy.

Before analyzing this data set, one might wish to find out whether the two treatment groups differ with respect to baseline characteristics such as age, education, marital status, etc. Generally, this might be accomplished using simple statistical methods such as Student's t-tests and chi-square tests. When the data set consists of multiply imputed data, however, straightforward application of these simple tests is no longer appropriate. To make valid inferences from multiply imputed data, both between- and within-imputation variance must be taken into account. This can be accomplished using the MIANALYZE procedure available in SAS®.

The first step to analyzing multiply imputed data is to analyze each imputation individually using standard statistical techniques. Many SAS procedures, including GLM, LOGISTIC, GENMOD and MIXED can be used to accomplish this step. The results from each analysis are output into ODS data sets. PROC MIANALYZE reads this output and calculates combined parameter and standard error estimates that are valid for the imputed data as a whole.

This paper describes a collection of macros that automate the process of testing for differences between the two levels of a binary independent variable with respect to continuous, binomial, or ordinal multinomial dependent variables in a multiple imputation setting. The macros build on the work of former RTI International colleagues Brinda Baskar and Kennan Murray (Bhaskar and Murray 2004) who created macros to generate univariate analysis of non-imputed data. PROC GLM is used for the analysis of continuous dependent variables, and PROC GENMOD is used for binomial or multinomial dependent variables. Then the univariate comparisons are output to an attractive RTF table using ODS and the REPORT procedure.

DATA STRUCTURE

Several requirements for the structure of the data set are necessary in order for the macros described in this paper to run correctly. First, all of the imputed data sets must be in the same SAS data file, and there must be a variable named `_IMPUTATION_` which identifies the observations in each imputation. The `_IMPUTATION_` variable must take on the values 1, 2, ..., m , where m is the total number of imputations. Second, the binary independent variable must take on the values 0 and 1. The number of observations belonging to each level of the binary variable may differ from imputation to imputation. Finally, all dependent variables used in the analysis, whether continuous or categorical, must be numeric. If a categorical numeric variable is formatted, those formats will label the levels of the variable in the RTF table.

MACROS

Several macros work together to perform the univariate analyses and output the results to the RTF table. They are the DATAINFO, CONT, CAT and TABLEOUT macros. The DATAINFO macro is run once, at the beginning of the program. It obtains information about the analysis data set, including the number of imputations, number of observations per imputation, and number of observations at each level of the binary independent variable. Global variables are defined based on this information and are later used in the CONT and CAT macros.

The CONT macro performs univariate analyses for continuous outcomes. The macro is executed separately for each continuous variable, thus, it may be used multiple times in the creation of one table. It performs separate analyses for each imputation using PROC GLM, and combines the results with PROC MIANALYZE. Means and standard errors for each level of the binary independent variable and for the data overall are combined into one table, along with a p-value for the significance of the independent variable. Those results are formatted for inclusion in the final RTF table.

The CAT macro performs similar analyses for categorical dependent variables. The outcomes must be either binomial or ordinal multinomial. PROC GENMOD is used to analyze each imputation, and PROC MIANALYZE combines the results. A table is created with counts and percentages for each level of the outcome crossed with each level of the binary independent variable, and a p-value for the difference between groups. The values are formatted for inclusion in the final RTF table. The CAT macro is executed once for every categorical dependent variable.

The TABLEOUT macro combines the results output by each execution of the CONT and CAT macros into one final data set. Then PROC REPORT generates the table, and ODS outputs the table to a file specified by the user.

DATAINFO MACRO

The DATAINFO macro obtains information from the analysis data set, such as the number of imputations and the number of records per imputation. The user must provide four parameters, described below.

```
%macro datainfo(dataset,testvar,g0label,g1label);
```

Parameters:

dataset	The name of the analysis data set, including SAS library if applicable.
b	The binary variable that will be the independent variable in each of the univariate analyses.
g0label	A label for the group for which the binary variable has the value 0.
g1label	A label for the group for which the binary variable has the value 1.

STEP 1: CREATE IVAL DATA SET

The first step of the DATAINFO macro creates a SAS data set named IVAL which will hold an integer value VNUM. Each time a new univariate analysis is performed, it represents a row of the RTF table. An integer value will be assigned to each row, starting with the value one. The integer will appear in the name of the SAS data set containing the results of the univariate analysis. Each time a new univariate analysis is run, VNUM will be updated by adding one to the current value, and this will be the row number for the analysis.

```
data ival;
  length vnum 8;
  vnum=0;
run;
```

STEP 2: SORT BY IMPUTATION

The analysis data set must be sorted by imputation so that BY statements included in the analysis procedures will generate a separate analysis for each imputation.

```
proc sort data=&dataset;
  by _imputation_;
run;
```

STEP 3: OBTAIN NUMBER OF IMPUTATIONS

The MEANS procedure is used to obtain the number of imputations in the analysis data set. It is assumed that the maximum value of the _IMPUTATION_ variable is the total number of imputations.

```
proc means data = &dataset noprint;
  var _imputation_;
  output out=numimp max=numimp;
run;

data _null_;
  set numimp;
  call symput ("nimp",trim(left(numimp)));
run;
```

STEP 4: OBTAIN TOTAL NUMBER OF SUBJECTS AT EACH LEVEL OF THE INDEPENDENT VARIABLE

The FREQ procedure is used to determine how many subjects belong to each level of the independent variable. This count may vary from imputation to imputation. If so, the number of subjects in each category will be an average over the imputations. PROC MEANS adds the number of subjects across all categories to obtain the total number of observations per imputation. A DATA step with CALL SYMPUT routines creates macro variables to hold the total number of subjects and the number of subjects in each category defined by the independent variable. A macro variable for the total number of subjects minus two is also created. This number will be used in the CONT macro as the complete data degrees of freedom for combining the individual imputation analyses.

```
proc freq data = &dataset;
  tables &testvar / out=count;
  title "Count by &testvar";
run;

proc means data = count noprint;
  var count;
  output out=total sum=count;
run;

data count;
  set count total;
  count=count/&nimp;
  drop _type_ _freq_;
run;

data _null_;
  set count;
  if &testvar=0 then call symput ("numg0",trim(left(count)));
  if &testvar=1 then call symput ("numg1",trim(left(count)));
  if &testvar="" then call symput ("numtot",trim(left(count)));
  if &testvar="" then call symput ("dfcont",trim(left(count-2)));
run;
```

STEP 5: CREATE GLOBAL VARIABLES

Finally, the DATAINFO macro creates global variables for the name of the analysis dataset, the name of the independent binary variable, the number of imputations, the number of observations in each imputation, the number of subjects in each category of the independent variable, labels for each level of the independent variable, and the complete data degrees of freedom for continuous outcomes. The global variables will be used in the other macros.

```
%global ds b m ntot ng0 ng1 g0lab g1lab ndfcont;
%let ds=&dataset;
%let b=&testvar;
%let m=&nimp;
%let ntot=&numtot;
%let ng0=&numg0;
%let ng1=&numg1;
%let g0lab=&g0label;
%let g1lab=&g1label;
%let ndfcont=&dfcont;

%mend datainfo;
```

CONT MACRO

The CONT macro tests for differences between the levels of the independent binary variable with respect to a continuous dependent variable. The macro has one parameter, which is the name of the continuous variable.

```
%macro cont(a);
```

STEP 1: OBTAIN ROW NUMBER

The IVAL data set is accessed and the current value of VNUM is increased by one. This value will be the row number for the univariate analysis.

```
data ival;
  set ival;
  vnum=vnum+1;
run;
```

```

data _null_;
  set ival;
  call symput ("i",trim(left(vnum)));
run;

```

STEP 2: PERFORM SEPARATE ANALYSES FOR EACH IMPUTATION

PROC GLM analyzes each imputation separately. In the GLM model, the continuous variable is the dependent variable and the binary variable is the predictor. Parameter and covariance estimates for each imputation are output to SAS data sets using ODS.

```

proc glm data=&ds ;
  model &a=&b /inverse;
  by _Imputation_;
  title "Individual Imputation Analysis for &b";
  ods output ParameterEstimates=glmparms&i
              InvXPX=glmxpxi&i;
run;

```

STEP 3: OBTAIN MULTIPLE IMPUTATION INFERENCE

PROC MIANALYZE reads the SAS data sets output by PROC GLM and combines the results from the individual imputation analyses into parameter and standard error estimates that apply to the imputed data as a whole. The EDF option specifies the complete data degrees of freedom, which, for a model with one intercept and one binary independent variable, is the total number of subjects, minus two. This value is plugged in to the macro using the global variable created by the DATAINFO macro. PROC MIANALYZE will use the complete data degrees of freedom to calculate the adjusted degrees of freedom recommended by Barnard and Rubin (1999). Results from PROC MIANALYZE are output to a new SAS data set using ODS.

```

proc mianalyze parms=glmparms&i xpxi=glmxpxi&i edf=&ndfcont;
  modeleffects Intercept &b;
  title "Combined MI Analysis for &b";
  ods output ParameterEstimates=combined&i;
run;

```

STEP 4: CALCULATE ESTIMATES FOR EACH LEVEL OF THE BINARY INDEPENDENT VARIABLE

Because the independent variable is binary (and not specified as a class variable), the two parameter estimates that result from the analysis are for an intercept and for the effect of the independent variable. The intercept corresponds to the level of the binary variable that has the value zero. The mean for the level of the binary variable with value one must be calculated by adding the estimates for the intercept and the independent variable. The standard error for this group must also be calculated. This is accomplished using the TRANSPOSE procedure and a DATA step.

```

proc transpose data=combined&i out=tcomb;
  id parm;
  var estimate stderr;
run;

data tcomb;
  set tcomb;
  if _name_="Estimate" then grp1comb=&b+Intercept; else
  if _name_="StdErr" then grp1comb=sqrt(&b**2-Intercept**2);
  rename Intercept=grp0comb;
run;

proc transpose data=tcomb out=ttcomb;
  id _name_;
  var grp0comb grp1comb;
run;

```

STEP 5: FORMAT GROUP ESTIMATES

A DATA step creates new variables containing the mean and standard error together. PROC TRANSPOSE creates a new table with one column for each category of the independent variable.

```

data ttcomb;
  set ttcomb;
  length cmean $7. cstderr $6. meanse $17.;
  cmean=left(put(estimate,8.2));

```

```

cstderr=left(put(stderr,8.2));
meanse=right(cmean)||' ± '||left(cstderr);
drop estimate stderr cmean cstderr;
run;

proc transpose data=ttcomb out=tttcomb;
  id _name_;
  var meanse;
run;

```

STEP 6: OBTAIN ESTIMATES OF OVERALL MEAN AND STANDARD ERROR

The UNIVARIATE procedure is used to calculate the overall mean and standard error for the data set, for each imputation. Results are output to a SAS data set which is then input into PROC MIANALYZE. The overall mean and standard error estimates for the multiply imputed data as a whole are output by PROC MIANALYZE into another data set.

```

proc univariate data=&ds noprint;
  var &a;
  output out=uni&i mean=m&a stderr=s&a;
  by _Imputation_;
run;

proc mianalyze data=uni&i;
  modeleffects m&a;
  stderr s&a;
  title "Overall Mean and SE for &b";
  ods output ParameterEstimates=meancomb&i;
run;

```

STEP 7: FORMAT OVERALL ESTIMATES AND P-VALUE

The first DATA step below formats the overall mean and standard error. The second formats the p-value for the significance of the difference between the levels of the independent variable, based on the multiple imputation analysis. Then the p-value, overall estimates, and means and standard errors for each level of the independent variable are combined into one data set.

```

data meancomb;
  set meancomb&i;
  length cmean $7. cstderr $6. totcomb $17.;
  cmean=left(put(estimate,8.2));
  cstderr=left(put(stderr,8.2));
  totcomb=right(cmean)||' ± '||left(cstderr);
  overallp=probt;
  keep totcomb overallp;
run;

data combprob;
  set combined&i;
  where upcase(parm)=upcase("&b");
  probt=round(probt,0.0001);
  if probt ne . and probt<=0.05 then pvalcomb=trim(left(probt))||"*"; else
  if probt>0.05 then pvalcomb=put(probt,8.4);
  if probt=0.0000 then pvalcomb='<0.0001'||"*";
  if probt=1 then pvalcomb='1.0000';
  keep pvalcomb;
run;

data allcomb&i(drop=_name_);
  merge tttcomb meancomb combprob;
run;

```

STEP 8: OBTAIN RESULTS FROM THE FIRST IMPUTATION

In the event that there is no between-imputation variation, PROC MIANALYZE will be unable to generate an overall p-value for the difference between the levels of the independent variable. This occurs when the neither the independent nor dependent variable contains imputed values. If this is the case, results will be obtained from the analysis of the first imputation, and these will appear in the RTF table. The code below performs steps 4-7 for the first imputation alone.

```

*Calculate results for each level of the binary variable using
parameter estimates for the intercept and binary variable effect;
proc transpose data=glmparms&i out=timp1;
  id parameter;
  var estimate stderr;
  where _imputation_=1;
run;

data timp1;
  set timp1;
  if _name_="Estimate" then grp1imp1=&b+Intercept; else
  if _name_="StdErr" then grp1imp1=sqrt(&b**2-Intercept**2);
  rename Intercept=grp0imp1;
run;

proc transpose data=timp1 out=ttimp1;
  id _NAME_;
  var grp0imp1 grp1imp1;
run;

*Format mean and SE estimates;
data ttimp1;
  set ttimp1;
  length cmean $7. cstderr $6. meanse $17.;
  cmean=left(put(estimate,8.2));
  cstderr=left(put(stderr,8.2));
  meanse=right(cmean)||' ± '||left(cstderr);
  drop estimate stderr cmean cstderr;
run;

proc transpose data=ttimp1 out=tttimp1;
  id _name_;
  var meanse;
run;

*Obtain mean and SE for overall sample using first imputation;
proc means data=&ds noprint;
  var &a;
  where _imputation_=1;
  output out=imp1mean&i mean=totmean stderr=totse n=ntot;
run;

data imp1mean;
  set imp1mean&i;
  length ctotmean $7. ctotse $6. totimp1 $17.;
  ctotmean=left(put(totmean,8.2));
  ctotse=left(put(totse,8.2));
  totimp1=right(ctotmean)||' ± '||left(ctotse);
  length formats $ 60.;
  formats="Mean ± SE (n="||trim(left(ntot))||")";
  keep totimp1 formats;
run;

*Format p-value from first imputation;
data imp1prob;
  set glmparms&i;
  where _imputation_=1 and upcase(parameter)=upcase("&b");
  probt=round(probt,0.0001);
  if probt ne . and probt<=0.05 then imp1pval=trim(left(probt))||""; else
  if probt>0.05 then imp1pval=put(probt,8.4);
  if probt=0.0000 then imp1pval='<0.0001'||"";
  if probt=1 then imp1pval='1.0000';
  keep imp1pval;
run;

*Combine means, SE and p-value from first imputation;
data imp1all&i(drop=_name_);
  merge tttimp1 imp1mean imp1prob;
run;

```

STEP 9: CREATE FINAL TABLE

The DATA step below creates a macro variable containing the label of the dependent variable. Then the analysis results from the multiply imputed data as a whole and the results from the first imputation alone are combined into one data set. If there is no p-value for the multiple imputation analysis, indicating that there was no between-imputation variance, the results of the first imputation will be used. Otherwise, the multiple imputation results will appear in the RTF table. In the case where there is between-imputation variance due to variation in the independent variable across imputations, but there was no imputation done on the dependent variable, the multiple imputation p-value and group estimates will be used, but the single imputation results must be used for the overall mean and standard error.

```
data _null_;
  set &ds;
  call symput('name',vlabel(&a));
run;

data final&i;
  merge allcomb&i imp1all&i;
  rownum=&i;
  name="&name";
  if pvalcomb ne "" then do;
    group0=grp0comb;
    group1=grp1comb;
    testvalue=pvalcomb;
  end;
  if pvalcomb="" then do;
    group0=grp0imp1;
    group1=grp1imp1;
    testvalue=imp1pval;
  end;
  if overallp=. then totcol=totimp1; else totcol=totcomb;
  label group0="&g0lab*(N=&ng0)"
        group1="&g1lab*(N=&ng1)"
        testvalue="P-value"
        totcol="Total*(N=&ntot)"
        formats="Value"
        name="Variable";
  keep rownum name formats group0 group1 totcol testvalue;
run;

%mend cont;
```

CAT MACRO

The CAT macro tests for differences between the levels of the binary independent variable with respect to a categorical dependent variable. The macro has one parameter, which is the name of the categorical variable. If the dependent variable has more than two levels, its values must be ordinal.

```
%macro cat(a);
```

STEP 1: OBTAIN ROW NUMBER

The IVAL data set is accessed and the current value of VNUM is increased by one. This value will be the row number for the univariate analysis.

```
data ival;
  set ival;
  vnum=vnum+1;
run;

data _null_;
  set ival;
  call symput ("i",trim(left(vnum)));
run;
```

STEP 2: CALCULATE AND FORMAT FREQUENCIES AND PERCENTAGES

PROC FREQ generates cross-tabulations between the dependent and independent variables for each imputation. Cell counts and percentages are output to a SAS data set. Then PROC MEANS averages the cell counts and

percentages over the imputations and outputs the results to a new data set. The percentages retained are those that give the proportion of subjects at each level of the dependent variable, within each level of the independent variable. Thus, the percentages add up to 100 for each category of the independent variable. A DATA step creates new variables containing the count and the percentage together. PROC TRANSPOSE creates a new table with one column for each category of the independent variable.

```
proc freq data=&ds;
  tables &a * &b / out=freq&i outpct;
  by _imputation_;
  title "Cross-tabulation of &a and &b";
run;

proc sort data=freq&i;
  by &a &b;
proc means data=freq&i noprint;
  var count pct_col;
  by &a &b;
  output out=avg&i mean=count pct_col;
run;

data avg;
  set avg&i;
  where &a ne .;
  group="group"||left(trim(&b));
  length ccount $4. cpct_col $5. colpct $17.;
  cpct_col=left(put(pct_col,8.2));
  ccount=left(put(count,8.));
  colpct=right(ccount)||" ("||right(cpct_col)||"%";
  drop _TYPE_ _FREQ_ pct_col count &b cpct_col ccount;
run;

proc transpose data=avg out=tavg;
  id group;
  var colpct;
  by &a;
run;
```

STEP 3: CALCULATE DEPENDENT VARIABLE SUBTOTALS, GRAND TOTAL, AND NUMBER OF OUTCOME CATEGORIES

PROC MEANS sums the counts from the previous cross-tabulations in order to obtain the total number of subjects at each level of the dependent variable, averaged over the imputations. Those results are output to a SAS data set, and a second application of PROC MEANS sums over the dependent variable categories to obtain the grand total. The grand total will differ from the total number of subjects calculated by the DATAINFO macro when the outcome does not apply to all subjects. For example, a variable such as previous miscarriage might be missing for women who have not had a previous pregnancy. The MEANS procedure also calculates the number of levels of the dependent variable.

A DATA step creates macro variables to hold the value of the grand total and the number of dependent variable categories. Another macro variable holds the number of categories minus one, which will be the number of intercept terms in the statistical model. A third macro variable is created for the complete data degrees of freedom which will be used by PROC MIANALYZE in combining the analyses of the individual imputations. This value is the total number of subjects minus the number of model parameters. Since the model parameters consist of one binary variable and intercepts for each value of the dependent variable except one, the total number of model parameters is equal to the number of categories of the outcome variable.

```
proc means data=avg&i sum noprint;
  where &a ne .;
  var count;
  by &a;
  output out=tot(drop=_TYPE_ _FREQ_) sum=total;
run;

proc means sum data=tot noprint;
  var total;
  output out=total1 sum=tot n=numcat;
run;

data _null_;
  set total1;
```

```

call symput ("grandtot",tot);
call symput ("ncat",trim(left(numcat)));
call symput ("nint",trim(left(numcat-1)));
call symput ("ndfcat",trim(left(tot-numcat)));
run;

```

STEP 4: COMBINE DEPENDENT VARIABLE SUBTOTALS AND GRAND TOTAL

A DATA step obtains the format for the dependent variable and creates a new macro variable for the format. Another DATA step combines the cross-tabulations and dependent variable subtotals into one data set, calculates the percent of subjects at each level of the dependent variable, and creates new variables containing the count and percentages together. The format from the dependent variable is used to label the levels of the outcome in the table.

```

data _null_;
  set tavg;
  call symput('varfmt',vformat(&a));
run;

data totavg;
  merge tavg tot;
  where &a ne .;
  by &a;
  rownum=&i;
  grandtotal=&grandtot
  tot_pct=round((total/grandtotal)*100,0.01);
  length ctotl $4. ctot_pct $5. totcol $17.;
  ctot_pct=left(put(tot_pct,8.2));
  ctotl=left(put(total,8.));
  totcol=right(ctotl)||" ("||right(ctot_pct)||"%";
  name=vlabel(&a);
  length formats $ 60.;
  formats=put(&a,&varfmt);
  drop &a _name_ grandtotal tot_pct total ctot_pct ctotl;
run;

```

STEP 5: PERFORM STATISTICAL TESTS FOR EACH ITERATION

PROC GENMOD tests the effect of the binary independent variable on the categorical dependent variable for each imputation. A multinomial distribution is specified for the dependent variable, which is appropriate for both binomial and multinomial ordinal outcomes. The RORDER=INTERNAL option is specified so that the dependent variable will be ordered according to its unformatted values. PROC GENMOD will model the probability of being in a category with a lower ordered value.

Resulting parameter and covariance estimates are output to SAS data sets using ODS. A third ODS data set is output with the name of the parameters, including intercepts, and the order in which they appear in the model. Unfortunately, when the DIST=MULT option is used, these three ODS tables are not output in a format which is recognized by PROC MIANALYZE. The FIX macro, which is included as an appendix, is used to reformat the tables so that PROC MIANALYZE will accept them as input. For further information on the FIX macro, refer to Gantz (2006).

```

proc genmod data=&ds rorder=internal;
  model &a=&b /dist=mult covb;
  by _Imputation_;
  title "Individual Imputation Analysis for &b";
  ods output ParameterEstimates=gmparms&i
             Covb=gmcovb&i
             ParmInfo=pinfo&i;
run;

%fix(&ncat);

```

STEP 6: OBTAIN MULTIPLE IMPUTATION INFERENCE

PROC MIANALYZE combines the parameter and standard error estimates from the individual imputation analyses to obtain results for the multiply imputed data as a whole. PROC MIANALYZE requires that all independent variables be listed in the MODELEFFECTS statement, including all intercepts. The LISTINT macro is used to list the appropriate number of intercepts, depending on the number of outcome categories. The required parameter for

the LISTINT macro is the number of intercept terms in the model. This value is fed into the macro by means of the macro variable created in step 3. The EDF option specifies the complete data degrees of freedom. This value is also plugged in using a macro variable created in step 3. PROC MIANALYZE will use the complete data degrees of freedom to calculate the adjusted degrees of freedom recommended by Barnard and Rubin (1999). The results of the multiple imputation analysis are output into a SAS data set. A DATA step formats the p-value for the significance of the independent variable.

```
%macro listint(k);
  %do j=1 %to &k;
    Intercept&j
  %end;
%mend listint;

proc mianalyze parms=gmparms&i covb=gmcovb&i parminfo=pinfo&i edf=&ndfc&i;
  modeleffects %listint(&nint) &b;
  title "Combined MI Analysis for &b";
  ods output ParameterEstimates=combined&i;
run;

data combstat;
  set combined&i;
  where upcase(parm)=upcase("&b");
  rownum=&i;
  P_PCHI=round(Probt,0.0001);
  if P_PCHI ne . and P_PCHI<=0.05 then testvalue2=trim(left(P_PCHI))||""; else
  if P_PCHI>0.05 then testvalue2=put(P_PCHI,8.4);
  if P_PCHI=0.0000 then testvalue2='<0.0001'||"";
  if P_PCHI=1 then testvalue2='1.0000';
  keep testvalue2 rownum;
run;
```

STEP 7: OBTAIN P-VALUE FROM THE FIRST IMPUTATION

In the event that there is no between-imputation variation, PROC MIANALYZE will be unable to generate an overall p-value for the multiply imputed data. In this case, the p-value from the analysis of the first imputation will appear in the RTF table. A DATA step obtains and formats this p-value. Then the data sets containing the p-values from the multiple imputation analysis and from the first imputation are merged and the appropriate p-value is selected.

```
data imp1parms;
  set gmparms&i;
  where _imputation_=1 and upcase(parameter)=upcase("&b");
  rownum=&i;
  probchisq=round(probchisq,0.0001);
  if probchisq ne . and probchisq<=0.05 then
    testvalue1=trim(left(probchisq))||"";
  else if probchisq>0.05 then testvalue1=put(probchisq,8.4);
  if probchisq=0.0000 then testvalue1='<0.0001'||"";
  if probchisq=1 then testvalue1='1.0000';
  keep testvalue1 rownum;
run;

data pval;
  merge imp1parms combstat;
  by rownum;
  if testvalue2="" then testvalue=testvalue1; else testvalue=testvalue2;
  keep testvalue rownum;
run;
```

STEP 8: CREATE FINAL TABLE

A DATA step creates the final table containing the counts and percentages for each level of the dependent variable crossed with the independent variable, the subtotals and percentages for the dependent variable, and the appropriate p-value. The number of subjects in each category of the independent variable and the total number of subjects are incorporated into the column labels using global variables defined by the DATAINFO macro.

```
data final&i;
  merge totavg pval;
  by rownum;
  if not first.rownum then do;
```

```

        name='';
        testvalue='';
    end;
    label group0="&g0lab*(N=&ng0)"
          group1="&g1lab*(N=&ng1)"
          testvalue="P-value"
          totcol="Total*(N=&ntot)"
          formats="Value"
          name="Variable";
run;

%mend cat;

```

TABLEOUT MACRO

The TABLEOUT macro combines the final tables from each execution of the CONT and CAT macros into one SAS data set. Then the REPORT procedure generates a table from that data set. The macro has four parameters, as described below.

```
%macro tableout(tnum,firstrow,lastrow,title,file);
```

Parameters:

tnum	An integer value used to identify the table. This number will appear in the final SAS data set containing all of the data for the RTF table.
firstrow	An integer value corresponding to the row number of the first dependent variable to be included in the table. Generally, this will be one. However, at times it may be desirable to create multiple small tables instead of one large table.
lastrow	An integer value corresponding to the row number of the last dependent variable to be included in the table.
title	A title for the RTF table.
file	Location and name of the RTF file to which the final table will be output.

The NAMES macro creates a list of the CAT and CONT output data sets to be included in the RTF table. This macro was created by Bhaskar and Murray (2004). A DATA step merges the tables into the FINAL T_n data set, where n is the table number. Then the REPORT procedure creates a formatted table from the data set, in a style defined by the TEMPLATE procedure. ODS is used to output the table to the RTF file specified by the user.

```

%macro names(j,k);
    %do i=&j %to &k;
        final&i
    %end;
%mend names;

data finalT&tnum;
    set %names(&firstrow,&lastrow);
run;

proc template;
    define style unistyle;
        parent=styles.printer;
        replace fonts /
            'titlefont2' = ("Arial", 9pt, Bold)
            'titlefont' = ("Arial", 10pt, Bold)
            'strongfont' = ("Arial", 8pt, Bold)
            'emphasisfont' = ("Arial", 8pt, Bold)
            'fixedemphasisfont' = ("Arial", 8pt, Bold)
            'fixedstrongfont' = ("Arial", 8pt, Bold)
            'fixedheadingfont' = ("Arial", 8pt, Bold)
            'batchfixedfont' = ("Arial", 8pt, Bold)
            'fixedfont' = ("Arial", 8pt, Bold)
            'headingemphasisfont' = ("Arial", 8pt, Bold)
            'headingfont' = ("Arial", 8pt, Bold)
            'docfont' = ("Arial", 8pt);
        end;
run;

```

```

ods listing close;
ods rtf file="&file..rtf" style=unistyle;
proc report data=finalT&tnum nowd split='*';
  column rownum name formats group0 group1 testvalue totcol ;
  define rownum /order order=data noprint;
  compute before rownum;
  line ' ';
  endcomp;
  define name/order=data style(column)=[just=left];
  define formats/ order=data style(column)=[just=left];
  define group0/order=data style(column)=[just=right];
  define group1/order=data style(column)=[just=right];
  define testvalue/order=data style(column)=[just=center];
  define totcol/order=data style(column)=[just=right];
  title color=black font=arial " &title";
run;
ods rtf close;
ods listing;

%mend tableout;

```

EXAMPLE

The code below generates a table from a SAS data set named PREG in the WORK library. The independent binary variable is CARE_GROUP, which consists of two levels. The label for CARE_GROUP=0 is Usual Care, and the label for CARE_GROUP=1 is Intervention. The table will test for differences between the levels of CARE_GROUP with respect to three baseline characteristics. Age (AGE) is a continuous variable, marital status (MARSTAT) is a binomial variable, and education (EDUC) is a three-category ordinal variable. Table 1, titled BASELINE CHARACTERISTICS is output to the file C:\\Tables\\Table_1.rtf.

```

%datainfo(work.preg,care_group,Usual Care,Intervention);
%cont(age);
%cat(marstat);
%cat(educ);
%tableout(1,1,3,BASELINE CHARACTERISTICS,C:\\Tables\\Table_1);

```

The resulting table is shown below.

BASELINE CHARACTERISTICS

Variable	Value	Usual Care (N=234)	Intervention (N=229)	P-value	Total (N=463)
Age	Mean ± SE (n=463)	24.51 ± 0.36	24.93 ± 0.37	0.4136	24.72 ± 0.26
Education level	1 = < High school	74 (31.62%)	75 (32.75%)	0.5390	149 (32.18%)
	2 = HS graduate/GED	95 (40.60%)	98 (42.79%)		193 (41.68%)
	3 = At least some college	65 (27.78%)	56 (24.45%)		121 (26.13%)
Marital status	1 = Unmarried	179 (76.50%)	173 (75.55%)	0.8108	352 (76.03%)
	2 = Married	55 (23.50%)	56 (24.45%)		111 (23.97%)

CONCLUSION

Routine univariate tests, which are usually simple to execute, become much more complex when multiply imputed data are involved. The macros described here provide tools that make it easy to test for differences between the levels of one binary independent variable with respect to multiple continuous and categorical dependent variables using multiply imputed data. In addition, they provide a way of exporting these tests to an attractively formatted RTF table.

APPENDIX

Below is the FIX macro used to reformat ODS tables output by PROC GENMOD so that they can be used as input for PROC MIANALYZE. For more details, refer to Gantz (2006).

```
%macro fix(numcat);

data gmcovb&i;
set gmcovb&i;
rename
Prm1=Prm&numcat;
if RowName="Prm1" then do;
  Rowname="Prm&numcat";
end;
%do j=1 %to &numcat-1;
  rename
  Intercept&j=Prm&j;
  if RowName="Intercept&j" then do;
    Rowname="Prm&j";
  end;
%end;
run;

data int;
length Parameter $30. Effect $30.;
%do k=1 %to &m;
  %do j=1 %to &nint;
    _Imputation_=&k;
    Parameter="Intercept&j";
    Effect="Intercept&j";
    output;
  %end;
%end;
run;

data Pinfo&i;
set int Pinfo&i;
if Parameter="Prm1" then do;
  Parameter="Prm&numcat";
end;
%do j=1 %to &numcat-1;
  if Parameter="Intercept&j" then do;
    Parameter="Prm&j";
  end;
%end;
run;

%mend fix;
```

REFERENCES

Barnard, J. and Rubin, D.B. 1999. "Small-Sample Degrees of Freedom with Multiple Imputation." *Biometrika*, 86:948-955.

Bhaskar, Brinda, and Kennan Murray. 2004. "Generating Customized Analytical Reports from SAS Procedure Output." *Proceedings of the Seventeenth Annual Northeast SAS Users Group Conference*, Baltimore, MD, paper number AP15.

Gantz, Marie. 2006. "Combining PROC GENMOD Models with Multinomial Outcomes Using PROC MIANALYZE." *SESUG Proceedings 2006*, Atlanta, GA, paper number ST12.

CONTACT INFORMATION

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

Marie Gantz, Ph.D.
RTI International
PO Box 12194
Research Triangle Park, NC 27709-2194

Work Phone: 919-485-7780
Fax: 919-485-7762
Email: mgantz@rti.org
Web: www.rti.org

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.