HRS/AHEAD Documentation Report

IMPUTE: A SAS Application System for Missing Value Imputations

--- With Special Reference to HRS Income/Assets Imputations

Honggao Cao

Survey Research Center
Institute for Social Research
University of Michigan, Ann Arbor

July 2001

Draft Version 3.0

IMPUTE: A SAS Application System for Missing Value Imputations

--- With Special Reference to HRS Income/Assets Imputations

Honggao Cao

Institute for Social Research
University of Michigan

July 2001

This document is a companion file for a SAS program developed at the Institute for Social Research, the University of Michigan to perform missing value imputations for the Health and Retirement Study. Comments on the document and the program are welcome.

CONTENTS

IMPUTE: A SAS Application System for Missing Value Imputations

--- With Special Reference to HRS Income/Assets Imputations

Honggao Cao

I. Introduction

Survey questions that ask respondents to report amounts are often subject to high rates of missing data (Heeringa, Hill, and Howell, 1995; Hill, 1999; Juster and Smith, 1998). To avoid losing altogether valuable information from those who are unable or unwilling to provide answers to amount questions, more and more researchers rely on special questionnaire designs to collect "bracket" information, trying to infer exact amounts from the brackets collected. The work involving an inference of exact amount from bracket information may be called "missing value imputation with brackets". When the bracket information is not available, it is reduced to a general type of imputation, called here "missing value imputation without brackets".

In this document, we describe a SAS application system, IMPUTE, designed for missing value imputation without brackets in general, and for inferring exact amount from bracket information in particular. To navigate our tour, and as a motivation for this project, we shall pay special attention to the Health and Retirement Study (HRS) income and assets imputations. Among many other advantages, the HRS income and asset data sets provide a good example for illustrating the uses of, and the limits to, the system.

II. Invocation

An easy way to invoke IMPUTE is to include a system file, "impute.sas", in a SAS program preparing and organizing input data sets. This calling program is normally

called “control file”, and will be discussed in detail later.

Suppose that the system file is stored in “C:\HRSINC98”, then an execution of the following control file would effectively invoke IMPUTE:

Box 1. A Control File Invoking IMPUTE

```
* Control.sas;
* This is an example of control file invoking “IMPUTE”;

filename impute “C:\HRSINC98\impute.sas”;

%inc impute;
```

Once the system is invoked, and assuming that an input data set containing all of the required variables is available, one can execute the system by calling a macro %impute, supplying to it appropriate parameters. For example, the following macro call uses an input data set, “income98”, and a “mixed” imputation procedure to impute a HRS98 income variable “Q4634x” (a respondent’s self-employment income received in the last calendar year):

Box 2. Example of a Macro Call

```
%impute(data=income98,reshape=1,hhid=hhid2,rvar=Q4634x,
  rcontrol=Q4633,topv=99999996,points=4,p1=5000,p2=10000,
  p3=25000,p4=100000,finr=finr,svar=Q4670x,scontrol=Q4669,
  rvar1=Q4635a,rvar2=Q4636a,rvar3=Q4637,rvar4=Q4638,
  svar1=Q4674,svar2=Q4671,svar3=Q4672,svar4=Q4673,
  covar=”age1-age3 educ1-educ3 male marst”,method=mixed)
```

The parameter structure of this macro call seems to be intimidating (In fact, this is the most complicated parameter structure that the current version of the system allows). A simple imputation, however, may only need a much simpler set of parameters (see the examples in Box 10, Section VII). The example in Box 2 is chosen intentionally to show

the great flexibility possessed by IMPUTE, suggesting that the system can deal with a quite complicated imputation problem.

III. The Parameter Structure of the System

At the core of the system, IMPUTE, is a SAS macro program supported by a set of mini SAS macros. The program allows for 15 sets of parameters, some of which are required, others optional.

(a). The Required Parameters

Eight sets of parameters are required by IMPUTE, including *data*, *method*, *finr*, *reshape*, *rvar*, *rcontrol*, *topv*, and *points*.

Box 3. The Required Parameters

data = name of input data, a SAS data set with a structure stipulated below (Section IV). It has to contain the variable(s) for which imputations are done, as well as related bracket variables (Caution: Variable name "id" is reserved).

method = imputation method. Currently, this parameter may take on one of the four values: *median*, *hotdeck*, *score*, and *mixed*. A detailed description of each method is given in the Appendix of this document.

finr = name of a dummy variable (0,1) distinguishing an information-providing respondent (R) from his/her spouse.

reshape = one of two integers (0, 1) indicating if the input data set needs to be reshaped before imputation. The data set needs to be reshaped when *reshape* = 1; and otherwise when *reshape* = 0.

rvar = variable name, the name of the variable for which (amount) imputations are done.

rcontrol = control variable name, the name of the variable indicating ownership of the amount variable, *rvar*.

Box 3. The Required Parameters (Continued)

topv = the maximum, valid value for variable *rvar*, and, if *reshape* = 1, for an optional parameter, *svar*.

points = number of cutoff (bracketing) points. It has to be non-negative. When *points* = 0, the program would essentially do imputations for variable *rvar* without brackets.

(b). The Optional Parameters

The following seven sets of parameters are optional: *hhid, svar, scontrol, p's (p1, p2, …, p10), rvar's (rvar1, rvar2, …, rvar10), svar's (svar1, svar2, …, svar10),* and *covar*.

Box 4. The Optional Parameters

hhid = the name of a unique household identification variable. It is required only when *reshape* = 1.

svar = variable name, the name of the variable comparable to *rvar* (i.e., a spouse's counterpart variable of *rvar*). It is required only when *reshape* = 1.

scontrol = control variable name, the name of the variable indicating ownership of the amount variable, *svar*.

p1, p2, p3, ... = cutoff (bracketing) values. They have to be in an increasing order. They are required only when *points* > 0.

rvar1, rvar2, rvar3, ... = variables determining the brackets of *rvar*. They have to be in the same order as cutoff values. They are required only when *points* > 0.

svar1, svar2, svar3, ... = variables determining the brackets of *svar*. They have to be in the same order as cutoff values. They are required only when *points* > 0, and, *reshape* = 1.

covar = a vector of explanatory variables. This parameter is needed only when *method* = score or mixed. Delimited by a single or multiple spaces, the variables as a group have to be put inside quotation signs.

IV. The Structure of the Input Data Set

IMPUTE has various requirements for the structure of an input data set, depending on the values of two required parameters: *reshape* and *method*. These requirements dictate what variables are in the data set, and how they should be arranged. In this section, we discuss four basic data structures, including (in the order of increasing complexity) (1) *reshape* = 0 and *method* = median or hotdeck; (2) *reshape* = 0 and *method* = mixed or score; (3) *reshape* = 1 and *method* = median or hotdeck; and (4) *reshape*=1 and *method* = mixed or score. We shall do so with the variables listed below.

Box 5. Possible Variables Used in an Input Data Set

ID	hhid	unique household id
Variable	pn	person id
	finr	dummy variable, information-providing R
	rssi	R social security income last month
	spssi	SP social security income last month
Amount	rssiyr	year R started to receive ssi
Variable	rssimo	month R started to receive ssi
	spssiyr	year SP started to receive ssi
	spssimo	month SP started to receive ssi
	rssicon	control variable for rssi
	spssicon	control variable for spssi
Control	rssiyrco	control variable for rssiyr
Variable	rssimoco	control variable for rssimo
	spssiyco	control variable for spssiyr
	spssimco	control variable for spssimo
	rssi1	low bracket ($500) for rssi
	rssi2	middle bracket ($1000) for rssi
Bracket	rssi3	high bracket ($1500) for rssi
Variable	spssi1	low bracket ($500) for spssi
	spssi2	middle bracket ($1000) for spssi
	spssi3	high bracket ($1500) for spssi
	age1-age3	dummy variables: age
Explanatory	educ1-educ3	dummy variables: education
Variable	male	dummy variable: sex
	marst	dummy variable: marital status

(a). Possible Variables in an Input Data Set

As indicated in Box 5, an input data set may contain five types of variables. **Identification variable** is self-evident, identifying respondents and/or households in the data set. One special identification variable, "finr" (short for "financial respondent"), is required by IMPUTE. It is a (1, 0) dummy distinguishing an information-providing respondent (a financial or family R in HRS, for example) from one who is related to the respondent but does not provide information on the variable to be imputed (e.g., the spouse of a financial or family R). When there is only one respondent from a household (or other type of unit of analysis), he or she has to be an information-providing respondent, with "finr" = 1.

Amount variable is one for which imputations are done. By default, an amount variable is bounded by zero at the bottom, and a value specified by the parameter *topv* at the top. Any value outside the range *0 – topv* would be treated as missing, and thus requiring some kind of imputations.[1]

For reasons noted later (in Section V), an amount variable may not have a name with more than 7 letters (or numbers). Thus, while a valid variable name in SAS, Q1234567 is not a valid name for an amount variable used by IMPUTE.

Each amount variable needs a **control variable**, indicating whether the respondent has the asset, income, expenditure or other amount type. It controls how a missing value on the amount variable should be imputed: a zero or a positive value? A control variable may take on one of the following values: 1, 5, any values greater than 5, and **.** (missing).

[1] More often than not in social science studies, and almost always in HRS, amount variables are top-bounded by values greater than the largest, valid values that the variables may take on (999999 for "don't know" or "refusal", for example). For each of those variables, the parameter *topv* needs to be set in such a way that "don't know" or "refusal" would become missing automatically.

When a control variable is 1, a missing value on the corresponding amount variable would be imputed as a positive value. Imputations would be done for it based on a donor pool formed by all positive values of the amount variable. Conversely, when a control variable is 5, a missing value on the amount variable would be imputed as zero.

A control variable with any value greater than 5 brings in uncertainty for the amount variable. IMUPTE would first determine if the control variable should take on 1 or 5, based on one of two imputation procedures: hotdeck and score. The amount imputation would then be done based on the result of this first-step, "ownership" imputation. Finally, when a control variable is missing, a missing value on the corresponding amount variable would be assumed to be legitimate.[2] By default, a legitimate missing on the amount variable is set to zero.[3]

Bracket variable contains bracket information for a corresponding amount variable. Always tied to a break point value, a bracket variable provides the answer to the question that the amount variable is less than (1), about the same (3), or greater than (5) the break point value.[4] In principle, there can be no limit to the number of bracket variables that an amount variable may have. But the current version of IMPUTE sets the

[2] For example, a missing value on the spouse's social security income for a respondent who is single would be legitimate. Therefore, the control variable for the spouse's social security income for this respondent should be missing.

[3] A control variable is often created from a single "yes/no" question, and may be illustrated in the following example.

Question: "Did you receive any social security income last month?"

1	Yes
5	No
8	Don't Know
9	Refused
.	Inappropriate (Missing)

[4] Other values that a bracket variable may take on are "don't know" (8), "refused" (9), and missing (.).

limit to 10. Naturally, bracket variables are needed only when they exist, or when the bracket points (*points*) greater than zero. (Note: In Box 5, there are no bracket variables for “rssiyr”, “rssimo”, “spssiyr” or “spssimo”.)

Explanatory variables are those IMPUTE uses to help imputations. They are required only when the imputation method (*method*) is “mixed” or “score”. While IMPUTE has no specific requirements for explanatory variables, it is expected that the variables are closely related to the variable to be imputed, and that each individual in the data set has non-missing values on all of the variables.

(b). Why and How the Parameters *method* and *reshape* May Influence the Input Data Structure?

The role of the parameter *method* plays in IMPUTE can be better seen in the Appendix. When *method* = mixed or score, some or all of the imputations would be done based on a regression procedure. When *method* takes on such a value, therefore, an appropriate set of explanatory variables needs to be included in the input data set.

A primary function of the parameter *reshape* is to determine how to construct the donor pool for a variable to be imputed. Normally, survey questions regarding amounts are addressed to individual respondents who themselves are the unit of analysis. For example, a respondent may be asked about social security income he or she received last month, or insurance premium he or she paid for last year. When all respondents are asked about this question, imputations may be done simply based on their answers (i.e., the answers to the respondent question alone forms the donor pool). There is no need for data reshaping, or in the programming language recognized by IMPUTE, *reshape* = 0.

In other cases, however, a respondent may be asked not just about his or her own

information, but also about information of some related persons (e.g., social security income his or her spouse received last month, or insurance premium his or her spouse paid for last year). Imputations based on this type of data may have two different strategies. First, one may treat a respondent's own question independent of, and separate from, a spouse question, imputing respondent and spouse information separately (i.e., forming two different donor pools with the answers to the respondent and spouse questions). No data reshaping would then be required, and `reshape` = 0. Alternatively, one may treat the respondent and the spouse questions as a single question answered by two different, but closely related, respondents, pooling them together before imputation (i.e., forming a joint donor pool with the answers to the respondent and the spouse questions). If this latter strategy is taken, the original data need to be reshaped (`reshape` = 1), and the respondent and spouse questions need to be combined.[5]

Data reshaping before imputations (or imputations with a joint donor pool) is often preferable when there is no systematic mechanism selecting information-providing respondents. Should such a mechanism exist, a question about an information-providing respondent may be very different from the same question about his or her spouse. Pooling the respondent and the spouse questions in this scenario may invite imputation bias for both the respondent and the spouse.

(c). Four Basic Structures of the Input Data Set

(1). *reshape* = 0 and *method* = median or hotdeck

[5] IMPUTE can do data reshaping internally. But it requires that the input data are arranged in a proper way (see Box 8). It is important to note that combining a respondent question and a spouse question is possible only when the two questions share the same bracket structure with the same break point values.

The structure of the input data set is the simplest when *reshape* = 0 and *method* = median or hotdeck. Explanatory variables need not to be included; and there is no need for data reshaping. In the example shown in Box 6, for each household there may be more than one respondent, but all the respondents are of information-providing. There are three amount variables in the data set: one ("rssi") with brackets, and the others ("rssiyr"

Box 6. The Data Structure when *reshape* = 0, and *method* = median or hotdeck

hhid	1	1	2	2	3	4	5	5	6
pn	1	2	1	2	1	1	1	2	1
finr	1	1	1	1	1	1	1	1	1
rssi	100	.	500	9999	.	752	600	800	.
rssicon	1	5	1	1	8	1	1	1	9
rssi1	.	5	.	.	.	.	.	.	.
rssi2	.	1	.	.	.	.	.	.	.
rssi3	.	8	.	8	.	.	.	.	.
rssiyr	1996	1997	1997	9999	.	1994	1995	1994	.
rssiyrco	1	1	1	9	9	1	1	1	9
rssimo	7	11	2	99	.	3	9	12	.
rssimoco	1	1	1	9	9	1	1	1	9

and "rssimo") without. Each amount variable has a control variable. The bracket structure for "rssi" is defined in such a way that all the bracket variables would be missing when "rssi" is not, mimicking the survey design in HRS. The top values for "rssi", "rssiyr" and "rssimo" (9999, 9999, and 99, respectively) are special codes for "don't know" or "refused". The parameter *topv* for these variables, therefore, needs to be set as, say, 9996, 9996, and 96, values slightly less than their corresponding top values.

(2). *reshape* = 0 and *method* = mixed or score

When *method* = mixed or score, explanatory variables need to be included in the input data set (Box 7). No missing values are allowed for these variables.

(3). *reshape* = 1 and *method* = median or hotdeck

When *reshape* = 1 and *method* = median or hotdeck, IMPUTE expects that the

Box 7. The Data Structure when *reshape* = 0, and *method* = mixed or score

hhid	1	1	2	2	3	4	5	5	6
pn	1	2	1	2	1	1	1	2	1
finr	1	1	1	1	1	1	1	1	1
rssi	100	.	500	9999	.	752	600	800	.
rssicon	1	5	1	1	8	1	1	1	9
rssi1	.	5	.	.	.	.	.	.	.
rssi2	.	1	.	.	.	.	.	.	.
rssi3	.	8	.	8	.	.	.	.	.
rssiyr	1996	1997	1997	9999	.	1994	1995	1994	.
rssiyrco	1	1	1	9	9	1	1	1	9
rssimo	7	11	2	99	.	3	9	12	.
rssimoco	1	1	1	9	9	1	1	1	9
age1	1	1	0	0	0	0	0	0	0
age2	0	0	0	1	0	1	1	0	1
age3	0	0	0	0	1	0	0	1	0
educ1	0	0	1	0	0	0	1	0	0
educ2	0	1	0	1	1	0	0	1	0
educ3	1	0	0	0	0	1	0	0	1
male	1	0	0	1	0	1	1	0	0
marst	1	1	1	1	0	0	1	1	0

Box 8. The Data Structure when *reshape* = 1, and *method* = median or hotdeck

hhid	1	1	2	2	3	4	5	5	6
pn	1	2	1	2	1	1	1	2	1
finr	1	0	1	0	1	1	0	1	1
rssi	100	100	500	500	.	.	600	600	.
rssicon	1	1	1	1	8	8	1	1	9
rssi1	.	.	.	.	.	.	.	.	.
rssi2	.	.	.	.	.	.	.	.	.
rssi3	.	.	.	.	.	.	.	.	.
spssi	.	.	9999	9999	752	752	800	800	.
spssicon	5	5	1	1	1	1	1	1	9
spssi1	5	5	.	.	.	.	.	.	.
spssi2	1	1	.	.	.	.	.	.	.
spssi3	8	8	8	8	.	.	.	.	.
rssiyr	1996	1996	1997	1997	.	1994	1995	1995	.
spssiyr	1997	1997	9999	9999	.	.	1994	1994	.
rssiyrco	1	1	1	1	9	1	1	1	9
spssiyco	1	1	9	9	.	.	1	1	.
rssimo	7	7	2	2	.	3	9	9	.
spssimo	11	11	99	99	.	.	12	12	.
rssimoco	1	1	1	1	9	1	1	1	9
spssimco	1	1	9	9	.	.	1	1	.

input data set has a structure shown in Box 8. Specifically, all the amount and bracket variables should be treated as if they were the characteristics of a household (or other type of unit of analysis). That is, both an information-providing respondent and his or her spouse have the same values on all of the related amount, control and bracket variables. It is important to note that this type of data structure, while awkward in its appearance, can be easily prepared. For example, one can merge by "hhid" an individual-specific data set containing all the identification variables and, when needed, explanatory variables, with a household-level data set containing "hhid" and all the related amount and bracket variables.

(4). *reshape* = 1 and *method* = mixed or score

The most complicated of the four basic structures, the input data set in this case (Box 9) is just a natural extension of the one in Box 8. Explanatory variables are added, as a mixed or score method requires estimations of regression models.

(d). Imputing Multiple Variables

While imputations for a single variable are permissible by IMPUTE, in reality, they are often done sequentially for multiple variables in a single run. In imputing HRS income and assets variables, for example, one may prepare a single input data set that contains all of the income and assets variables in Section J (about 100 of them), calling macro %impute repeatedly to do imputations for all the amount variables.

When a single input data set is needed for imputing multiple variables, it is necessary to have the data set arranged to satisfy the strictest requirements. In fact, unless told otherwise (via appropriate parameter assignments), IMPUTE generally expects the input data set to be the most complicated (e.g., with the structure in Box 9). It would be a

Box 9. The Data Structure when *reshape* = 1, and *method* = mixed or score

hhid	1	1	2	2	3	4	5	5	6
pn	1	2	1	2	1	1	1	2	1
finr	1	0	1	0	1	1	0	1	1
rssi	100	100	500	500	.	.	600	600	.
rssicon	1	1	1	1	8	8	1	1	9
rssi1	.	.	.	.	.	.	.	.	.
rssi2	.	.	.	.	.	.	.	.	.
rssi3	.	.	.	.	.	.	.	.	.
spssi	.	.	9999	9999	752	752	800	800	.
spssicon	5	5	1	1	1	1	1	1	9
spssi1	5	5	.	.	.	.	.	.	.
spssi2	1	1	.	.	.	.	.	.	.
spssi3	8	8	8	8	.	.	.	.	.
rssiyr	1996	1996	1997	1997	.	1994	1995	1995	.
spssiyr	1997	1997	9999	9999	.	.	1994	1994	.
rssiyrco	1	1	1	1	9	1	1	1	9
spssiyco	1	1	9	9	.	.	1	1	.
rssimo	7	7	2	2	.	3	9	9	.
spssimo	11	11	99	99	.	.	12	12	.
rssimoco	1	1	1	1	9	1	1	1	9
spssimco	1	1	9	9	.	.	1	1	.
age1	1	1	0	0	0	0	0	0	0
age2	0	0	0	1	0	1	1	0	1
age3	0	0	0	0	1	0	0	1	0
educ1	0	0	1	0	0	0	1	0	0
educ2	0	1	0	1	1	0	0	1	0
educ3	1	0	0	0	0	1	0	0	1
male	1	0	0	1	0	1	1	0	0
marst	1	1	1	1	0	0	1	1	0

good programming practice, therefore, to make an input data set comply with the most complicated structure whenever the data set is used for imputing multiple variables.

V. The Control File

An imputation routine in SAS, IMPUTE will not work without a companion SAS calling program or **control file.** Generally, this file invokes the system IMPUTE, prepares input data sets, manages amount, control, and bracket variables, makes macro calls (%impute), and collect relevant variables after imputations. Box 10 displays a typical control file doing all of these tasks.

(a). Managing Amount, Control and Bracket Variables

One of the most important tasks for a control file is to manage amount, control and bracket variables. To accomplish this task, one needs to comply with the following rules.

(1). For each amount variable (Q123456, for example), create an imputation or "x" variable (e.g., Q123456x)[6] that takes on the original value of the amount variable. When an original value is valid (i.e., in the range *0 – topv*), the imputation variable would retain the value. Otherwise, it would be replaced with an imputed value. The advantage of having this new variable is that, once imputations are completed, the original amount variable will not be overwritten. When necessary, therefore, comparisons can be made between the original and imputed values to check the validity of the imputations.

(2). For each amount variable, create a control or "c" variable (e.g., Q123456c) indicating whether a respondent has a valid value on the amount variable. Frequently, the "c" variable may be created from a single "yes or no" question. When that is the case, a straightforward assignment as shown in Box 10 is enough. In some complicated situations, however, one may need to integrate information from several questions to get the "c" variable created. Since IMPUTE itself has no capability of verifying a "c" variable, when writing a control file, users need to make sure that the "c" variable is

[6] We suggest that an imputation variable be created by directly adding an "x" to an original amount variable. This way, a one-to-one correspondence between the original and imputation variables can easily be established. For the same reason, four other variables, the control (or "c") variable, the summary-bracket (or "s") variable, the bottom break point value or the "d" variable, and the top break point value or the "n" variable that are either required or generated by IMPUTE will be created in the same way: adding "c", "s", "d", and "e", respectively, to an original amount variable. It is important to note that, since SAS does not allow for a variable name with more than 8 letters (or numbers), the aforementioned convention implies that an original amount variable may not have a name with more than 7 letters (or numbers).

Box 10. A Typical Control File

```
* Control.sas;
* This is an example of control file;

libname hrs98 “c:\HRSINC98”;
filename impute “C:\HRSINC98\impute.sas”;

* Task 1. Invoke the system IMPUTE;
%inc impute;

* Task 2. Prepare Input Data;
proc sort data=hrs98.demog;
  by hhid pn;
  run;

proc sort data=hrs98.income;
  by hhid;
  run;

data income;
  merge demog income;
  by hhid;
  run;

* Task 3. Manage Amount, Control, and Bracket Variables;
data income;
  set income;

/* “x” variable: keep the original amount variables intact */
  rssihx=rssi;          /* reshape=0, hotdeck procedure */
  rssimx=rssi;          /* reshape=0, mixed procedure */
  rssinx=rssi;          /* reshape=1, mixed procedure */
  spssihx=spssi;        /* reshape=0, hotdeck procedure */
  spssimx=spssi;        /* reshape=0, mixed procedure */
  spssinx=spssi;        /* reshape=1, mixed procedure */
  rssiyrx=rssiyr;       /* reshape=0, hotdeck procedure */
  rssimox=rssimo;       /* reshape=0, hotdeck procedure */
  spssiyrx=spssiyr;     /* reshape=0, hotdeck procedure */
  spssimox=spssimo;     /* reshape=0, hotdeck procedure */

/* “c” variable: straightforward assignment */
/* Note: Each amount variable has a control variable */
  rssihc=ssicon;
  spssihc=spssicon;
  rssimc=ssicon;
  spssimc=spssicon;
  rssinc=ssicon;
  spssinc=spssicon;
  rssiyrc=rssiyrco;
  rssimoc=rssimoco;
  spssiyrc=spssiyco;
  spssimoc=spssimco;
```

Box 10. A Typical Control File (Continued)

```
/* Bracket variables */
  rssib1=rssi1;    /* Unnecessary here */
  rssib2=rssi2;    /* Unnecessary here */
  rssib3=rssi3;    /* Unnecessary here */
  spssib1=spssi1;  /* Unnecessary here */
  spssib2=spssi2;  /* Unnecessary here */
  spssib3=spssi3;  /* Unnecessary here */

  run;

* Task 4. Make Macro Calls;

/* Call %impute sequentially for multiple variables */
%impute(data=income,reshape=0,hhid=hhid,rvar=rssihx,
 rcontrol=rssic,topv=9996,points=3,p1=500,p2=1000,p3=1500,
 finr=finr,method=hotdeck)          /* Macro Call Example 1 */

%impute(data=income,reshape=0,hhid=hhid,rvar=spssihx,
 rcontrol=spssic,topv=9996,points=3,p1=500,p2=1000,p3=1500,
 finr=finr,method=hotdeck)          /* Macro Call Example 2 */

%impute(data=income,reshape=0,hhid=hhid,rvar=rssimx,
 rcontrol=rssic,topv=9996,points=3,p1=500,p2=1000,p3=1500,
 finr=finr,method=mixed,covar="age1-age3 educ1-educ3
 male marst")                       /* Macro Call Example 3 */

%impute(data=income,reshape=0,hhid=hhid,rvar=spssimx,
 rcontrol=spssic,topv=9996,points=3,p1=500,p2=1000,p3=1500,
 finr=finr,method=mixed,covar="age1-age3 educ1-educ3
 male marst")                       /* Macro Call Example 4 */

%impute(data=income,reshape=1,hhid=hhid,rvar=rssinx,
 rcontrol=rssinc,topv=9996,points=3,p1=500,p2=1000,p3=1500,
 finr=finr,method=mixed,svar=spssinx,scontrol=spssinc,
 rvar1=rssib1,rvar2=rssib2,rvar3=rssib3,svar1=spssib1,
 svar2=spssib2,svar3=spssib3,covar="age1-age3 educ1-educ3
 male marst")                       /* Macro Call Example 5 */

%impute(data=income,reshape=0,hhid=hhid,rvar=rssiyrx,
 rcontrol=rssiyrc,topv=9996,points=0,finr=finr,method=hotdeck)
                                    /* Macro Call Example 6 */
%impute(data=income,reshape=0,hhid=hhid,rvar=rssimox,
 rcontrol=rssimoc,topv=96,points=0,finr=finr,method=hotdeck)
                                    /* Macro Call Example 7 */
%impute(data=income,reshape=0,hhid=hhid,rvar=spssiyrx,
 rcontrol=spssiyrc,topv=9996,points=0,finr=finr,method=hotdeck)
                                    /* Macro Call Example 8 */
%impute(data=income,reshape=0,hhid=hhid,rvar=spssimox,
 rcontrol=spssimoc,topv=96,points=0,finr=finr,method=hotdeck)
                                    /* Macro Call Example 9 */
```

Box 10. A Typical Control File (Continued)

```
* Task 5. Collect Relevant Variables;
data hrs98.income;
  set income;
  if finr=1;
  keep hhid rssihx rssihs rssihc rssihd rssihe rssiht rssimx rssims
   rssimc rssimd rssime rssimt rssinx rssins rssinc rssind rssine
   rssint spssihx spssihs spssihc spssihd spssihe spssiht spssimx
   spssims spssimc spssimd spssime spssimt spssinx spssins spssinc
   spssind spssine spssint rssiyrx rssiyrs rssiyrc rssiyrt rssimox
   rssimos rssimoc rssimot spssiyrx spssiyrs spssiyrc spssiyrt
   spssimox spssimos spssimoc spssimot;
  run;
```

created correctly.

(3). An amount variable may (*points>0*) or may not (*points=0*) have bracket variables. When it does, each of the bracket variables must represent a unique break point value. When there are two or more bracket variables representing a break point value,[7] one needs first to integrate these variables into a single variable, and then treat the integrated variable as the bracket.

(b) Calling %impute

The "x", "c" and the bracket variables are not the only information required by IMPUTE. Before calling %impute, one also needs to know the values of other relevant parameters of the macro. Specifically, for each "x" variable, what imputation method should be chosen (*method* =)? Does the input data need to be reshaped (*reshape* =)?

[7] In HRS, a break point value in a bracket sequence is sometimes covered by two or more bracket variables. When asked about their assets, for example, different HRS respondents may be asked the bracket questions according to different break-point orders, which are based on their asset status in the previous wave. Thus, a break point value may be covered by one variable for some respondents, and by another variables for other respondents.

What is the variable that uniquely identifies a household (*hhid* =) or distinguishes an information-providing respondent from a non-information-providing respondent (*finr* =)? How many break points are there (*points* =), and what are the break point values (p1 =, p2 =, p3 =, …)? And, if necessary, what are the explanatory variables included in the vector "covar" (*covar* =)?

With the above parameter information, the next step in a control file is to call %impute, supplying to it the relevant information. IMPUTE will then do imputations as directed by the information provided. In principle, there is no limit to the number of macro calls in a control file, as long as all of the required information is available before the calls.

The structure of a macro call is quite complicated, the cost we have to pay for its ability to deal with a complicated imputation problem. In fact, the complexity of the structure of a macro call is exactly proportional to the complexity of the imputation problem the call intends to resolve. Generally, a macro call would have the most complicated structure, if (i) data reshaping is needed before imputation (*reshape*=1); (ii) the amount variable has bracket variables (*points*>0); and (iii) a mixed or score method is chosen for imputation (*method*=mixed or score) (see Example 5 in Box 10). Conversely, a macro call would have the simplest structure if (i) there is no need for data reshaping before imputation (*reshape*=0); (ii) the amount variable has no bracket variables (*points*=0); and (iii) a median or hotdeck method is chosen for imputation (*method*=median or hotdeck) (see Examples 6-9 in Box 10).

(c). Collecting Output Data

After imputations for each amount variable, IMPUTE would add to the original input data set four variables. They are the "x" variable, the "c" variable, a summary bracket or "s" variable, and an imputation type or "t" variable. When an amount variable has any brackets (i.e., *points* > 0), it would also add to the original data set two variables explicitly characterizing the bracket pattern of each data record: the "d" variable for the bottom break point value, and the "e" variable for the top break point values. The definitions of the "s", "d", "e", and "t" variables, along with their relationship to the "x" and "c" variables, will be discussed in Section VI. As the core of an output data set, these "added" variables should be collected when all the imputations are completed.[8]

Depending on the structure of the input data, an output data set may have records for non-information-providing respondents. When writing the final output data set, however, we expect that only the records for information-providing respondents are kept (i.e., *finr* = 1).

VI. Understanding the "Added" Variables in an Output Data Set

(a). The "s" Variable

The "s" or summary-bracket variable is an index summarizing the bracket pattern for each record of an amount variable in the input data set.[9] Three special codes have been reserved for the "s" variable: "-2" for legitimate missing (see Section IV or footnote 2 for its definition), "-1" for original value, and "0" for imputed value with no bracket

[8] More precisely, only the "s", "d", "e", and "t" variables are generated by IMPUTE. The "x" and "c" variables, required before calling %impute, are normally created in a control file.

[9] Don't confuse a summary-bracket ("s") variable with the bracket variables of an amount variable. An amount variable may have zero, one, or more bracket variables, but it has only one summary-bracket variable that integrates information based on the bracket variables.

information. Other codes, for other types of missing values on the amount variable, are generated in the way illustrated in the following example.

Assume that the missing values on amount variable Q123456 are determined by four bracket variables A, B, C, and D, which correspond to break point values 5000, 10000, 25000, and 100000, respectively. Each bracket variable may take on six values: 0 (missing), 1 (less than), 3 (about the same as), 5 (greater than), 8 (don't know), and 9 (refusal). The corresponding summary-bracket or "s" variable is then defined as:

$$Q123456s = A + 10*B + 100*C + 1000*D \qquad (1).$$

A value "1550" on the "s" variable, therefore, stands for a bracket pattern "less than 100000, greater than 25000, greater than 10000, and missing on 5000".

(b). The "d" and "e" Variables

The "d" and "e" variables identify, respectively, the bottom and top break point values of the bracket associated with a data record. The following rules apply: (i) if the amount variable is not missing, both the "d" and "e" variables would be equal to the original value; (ii) if the amount variable is missing with no bracket information, both the "d" and "e" variables would be missing; (iii) if a missing value on the amount variable has a bottom-open bracket (e.g.,"– 5000"), the "d" variable would be missing; (iv) if a missing value on the amount variable has a top-open bracket (e.g., "5000-"), the "e" variable would be missing; and (v) a missing value on the amount variable with a closed bracket would have non-missing values on both "d" and "e" variables. In addition, a special code, "-1", has been reserved for the two variables if the amount variable is about the spouse of an information-providing respondent and the respondent has no spouse.

(c). Imputation Type: The "t" Variable

Coupled with the control ("c") variable, the "s", "d", and "e" variables summarize all the relevant information needed for differentiating the corresponding "x" variable. Box 11 lists all possible relationships among these variables. To make better use of these relationships, IMPUTE creates a "t" variable, identifying the imputation type for each "x" variable. The code for the "t" variable is listed in the last column of Box 11.

Box 11. Differentiate the "x" Variable Using the "s", "c", "d", and "e" Variables

Original Amount Variable (Q1234)	Imputation Variable (Q1234x)	Control Variable (Q1234c)	Summary Bracket Variable (Q1234s)	Break point Value Bottom (Q1234d)	Top (Q1234e)	Imputation Type (Q1234t)
legitimate missing	0	.	-2	(., -1)	(., -1)	1
valid value	>=0	(1,5)	-1	>=0	>=0	2
closed bracket	>0	1	>0	>=0	>0	3
bottom-open bracket	>=0	1	>0	.	>0	4
top-open bracket	>0	1	>0	>=0	.	5
bracket with no information	>=0	1	>=0	.	.	6
imputed ownership	>=0	8,9	0	.	.	7

VII. The Output Files

Regardless of the structure of an input data set or the way a control file is created, IMPUTE automatically produces four summary reports: (i) "dataflag", (ii) "meanbrkt",

(iii) “compare1”, and (iv)“compare2”. The first file checks the consistency of bracket patterns before imputation, identifying any logical contradictions among the bracket variables. The second file checks imputation results against the bracket variables that have guided the imputations. The third contrasts the mean, median, and number of observations of actual, valid data with those of imputed, while the forth compares the distributions of valid data and imputed values based on the relevant brackets. A careful study of these files may help correct errors in the input data set, select appropriate imputation methods, as well as identify some potential problems in the control file.

(a). Checking the Consistency of Bracket Variables in the Input Data: “dataflag”

In general, IMPUTE expects that the input data are error-free, but it does have the feature of identifying potential logical contradictions among bracket variables. One example of such contradictions may be this: a respondent received last month less than $500 but greater than $1,000 from social security income. The contradictions are sometimes correctable, if independent, relevant information is available. When the contradictions are not correctable, however, IMPUTE would ignore the “bad” values of the bracket variables, treating the corresponding records on the amount variable as missing without any bracket information.

To make possible the correction of any potential logical contradictions, IMPUTE lists in “dataflag” all the records contaminated with contradictions. All the variables represented by the parameters *hhid, rvar1, rvar2,* …, and when necessary, *svar1, svar2,* …, are displayed, so that the contradictions may easily be seen. When there are no logical contradictions, however, a simple message “consistency check passed” are printed in the file.

(b). Checking against the Bracket Variables: “meanbrkt”

One critical indicator of possible errors in an imputation process is that the imputed results are not consistent with the bracket variables guiding the imputations. For example, if the bracket variables signal that a missing value on the corresponding amount variable should be greater than 5000 and less than 20000, any imputation outside this range is wrong.

To capture potential imputation errors, IMPUTE lists in a summary file the number of observations, mean, standard deviation, minimum and maximum values of all the imputations associated with each bracket pattern represented by a summary bracket (“s”) variable. Needless to say, the number of observations, mean, and standard deviation are all informative about the property of the imputations. But the most useful piece of information is the two extreme values, which may help reveal some obvious (if not all of the) errors.[10]

(c). Comparing Overall Distributions of Actual and Imputed Data: “compare1”

A fundamental assumption behind any imputation efforts is that actual observations may be used to predict missing records. Depending on the relationship between the two groups, the statistical characteristics of actual (observed) and imputed data may either be very similar, or quite different. When the relationship is known, therefore, a comparison of actual data with imputed data may help check the validity and/

[10] To illustrate, use the example in Section VI, (a). As noted, a value “1550” on the “s” variable stands for a bracket pattern “less than 100000, greater than 25000, greater than 10000, and missing on 5000”. If the minimum and maximum values of all the imputations in this summary bracket group are not consistent with this pattern, there must be errors somewhere in the imputation process.

or reliability of the imputations.[11]

In the second summary file, IMPUTE documents, side by side, the mean, median, and number of observations of valid and imputed *positive* values, along with the ratio of the number of imputed values to the number of valid values. One record per imputation call, these statistics are calculated based on the data actually used for imputations (i.e., after data reshaping whenever needed). When `reshape` = 1, therefore, they summarize the information not only for information-providing respondents, but also for their spouses or other related persons. The focus here is only on positive values, because other values (e.g., zeroes) are neither directly related to nor the direct product of amount imputations.

The results in this summary file may be useful in two ways. First, they portray the overall distributions of actual and imputed data, offering a way to gauge the distributional relationship between the two data groups and, possibly, the adequacy of the imputations. Second, the ratio of the number of imputed values to the number of observed values directly tells one how much valid information has been used for imputations. Understandably, with an adequate imputation method, the more the valid information is used (or the smaller is the ratio), the more reliable are the imputations. When the ratio of imputed values to valid values for an amount variable is very high, therefore, one knows that the reliability of the imputations is probably not very high.

(d). Comparing the Distributions of Actual and Imputed Data across Brackets: "compare2"

[11] For example, if respondents with valid (actual) data are known to be the same as – or not very different from -- those with missing data, imputed data are expected to have characteristics similar to those of actual data. Any substantial differences between the two groups would suggest inadequacy of the imputation procedure.

Imputations with brackets are critically dependent on the bracket structure, which is normally defined by the number of break points, as well as the break point values. To examine the effect of the bracket structure on imputations, and as a way to diagnose any potential irregularities with the imputations, the third summary file lists the distributions of valid and imputed data based on the structure. Each imputation call creates one record in the file when `reshape` = 0. When `reshape` = 1, however, there are two records per imputation call, one for the amount variable for the information-providing respondents, and the other for the same amount variable for their spouses. For each record, the total number of observations is always equal to the number of the information-providing respondents in the input data set. This number is decomposed into three groups: legitimate missing, valid and imputed zeroes, and valid and imputed (positive) amounts. The mean and media values listed in the file are the statistics for the positive amounts only. When reshape = 0, therefore, these values should be the same as those listed in the second summary file.

VIII. Limitations

(i). Without significant modifications, IMPUTE cannot effectively handle a data set in which there are more than two respondents in a household;

(ii). Negative value on an amount variable is not allowed, and negative imputation is not possible without conversions before imputation;

(iii). “Don’t Know” and “Refused” are treated as the same answer to a question creating a control or amount variable;

(iv). The variances of imputations are not available;

(v). When predicting ownership, a simple linear model (rather than a logistic model) is used. Although preliminary tests show no big difference between the alternative models, a change from a simple linear model to a logistic model may cause some programming errors;

(vi). When predicting amount using mixed or score method, no transformation on the amount variable is done; and

(vii). The effects of the individual or household weight, if any, on the imputations are ignored.

References

Herringa, S. G., Hill, D. H., and Howell, D. A. "Unfolding Brackets for Reducing Item Non-Response in Economic Surveys," Health and Retirement Study Working Paper, No. 94-029, June 1995.

Hill, D. H. "Unfolding Bracket Method in the Measurement of Expenditures and Wealth," in J. P. Smith, and R. J. Willis (eds.), Wealth, Work, and Health, University of Michigan, 1999.

Juster, F. T., and Smith, J. P. "Improving the Quality of Economic Data: Lessons from the HRS and AHEAD," *Journal of the American Statistical Association*, Vol. 92, No. 440, 1998.

Appendix. Alternative Methods for Imputing a Variable Determined by a Set of Brackets

(a). Four alternative methods are allowed for the current version of IMPUTE: median, hotdeck, score, and mixed. Each of the methods has a different approach to the two stages of predictions inherent in an imputation process: ownership prediction and amount prediction. The ownership prediction focuses on the control variable, trying to determine if a respondent with an uncertain answer to the control-variable question ("don't know" or "refused") should be treated as "owning" a positive value on the corresponding amount variable. Conditional on the fact that a respondent has—is predicted to have--a positive value on the amount variable, and based on the bracket information available, the amount prediction determines the positive value for the respondent if he or she is unwilling or unable to provide exact amount. Generally, a complete round of amount prediction consists of imputations for four different bracket types: closed brackets (e.g., 5000-10000), bottom-open brackets (e.g., -10000), top-open brackets (e.g., 10000-), and no brackets. Depending on the bracket structure, each bracket type may have multiple bracket patterns (e.g., 0-5000, 5000-10000, and 10000-25000 are all closed brackets). The imputation procedure for each bracket type may not be the same, but within a bracket type, it is the same for all possible bracket patterns.

Box A1 displays the imputation procedures for each imputation method. A median procedure uses the median value of a group of valid observations as the imputations for the missing observations of same bracket pattern. By contrast, a hotdeck or regression procedure uses an immediate neighbor rule based on random assignment or predicted score. An example of this immediate neighbor rule follows.

Box A1. Alternative Imputation Methods

		Amount Prediction			
Method	**Ownership Prediction**	Closed Brackets	Bottom-Open Brackets	Top-Open Brackets	No Brackets
Median	hotdeck	median	median	median	median
Hotdeck	hotdeck	hotdeck	hotdeck	hotdeck	hotdeck
Score	regression	regression	regression	regression	regression
Mixed	regression	hotdeck	hotdeck	regression	regression

(b). Suppose that one has an amount variable with values shown in column 1 of Box A2. A hotdeck (regression) procedure would generate a random number (predicted score) for each of the records, listed in column 2 (3). The immediate neighbor rule first sorts the data in column 1 based on the random number or predicted score in the ascending order, replacing a missing value with the valid value immediately before it. Sorting the data in the descending order, and repeating the procedure would have all the missing data imputed.

Box A2. The Immediate Neighbor Rule
Used in the Hotdeck and Regression Procedures

(1) Match amount variable with random number (hotdeck) or predicted score (regression)

Amount Variable	Random Number	Predicted Score
1	10	6
2	7	13
.	2	9
3	11	2
4	1	8
.	4	1
.	8	11
5	9	7
6	5	14
7	14	3
.	6	10
8	12	5
9	3	4
10	13	12

(2) Sort the data – first in an ascending, and then descending, order -- by the random number or predicted score, replacing a missing value with the valid value immediately before it

	Hotdeck		Regression		
	Before Imputation	After Imputation	Before Imputation	After Imputation	
1	4	4	.	.	(3)
2	.	4	3	3	
3	9	9	7	7	
4	.	9	9	9	
5	6	6	8	8	
6	.	6	1	1	
7	2	2	5	5	
8	.	2	4	4	
9	5	5	.	4	
10	1	1	.	4	
11	3	3	.	4	
12	8	8	10	10	
13	10	10	2	2	
14	7	7	6	6	