NATIONAL CENTER FOR EDUCATION STATISTICS

Working Paper Series

The Working Paper Series was initiated to promote the sharing of the valuable work experience and knowledge reflected in these preliminary reports. These reports are viewed as works in progress, and have not undergone a rigorous review for consistency with NCES Statistical Standards prior to inclusion in the Working Paper Series.

U. S. Department of Education
Office of Educational Research and Improvement

NATIONAL CENTER FOR EDUCATION STATISTICS

Working Paper Series

Comparison of Proc Impute and Schafer's Multiple Imputation Software

Working Paper No. 2001–10 September 2001

Contact: Sam Peng
Office of the Commissioner
E-mail: samuel.peng@ed.gov

U. S. Department of Education
Office of Educational Research and Improvement

U.S. Department of Education
Rod Paige
Secretary

Office of Educational Research and Improvement
Grover J. Whitehurst
Assistant Secretary

National Center for Education Statistics
Gary W. Phillips
Acting Commissioner

The National Center for Education Statistics (NCES) is the primary federal entity for collecting, analyzing, and reporting data related to education in the United States and other nations. It fulfills a congressional mandate to collect, collate, analyze, and report full and complete statistics on the condition of education in the United States; conduct and publish reports and specialized analyses of the meaning and significance of such statistics; assist state and local education agencies in improving their statistical systems; and review and report on education activities in foreign countries.

NCES activities are designed to address high priority education data needs; provide consistent, reliable, complete, and accurate indicators of education status and trends; and report timely, useful, and high quality data to the U.S. Department of Education, the Congress, the states, other education policymakers, practitioners, data users, and the general public.

We strive to make our products available in a variety of formats and in language that is appropriate to a variety of audiences. You, as our customer, are the best judge of our success in communicating information effectively. If you have any comments or suggestions about this or any other NCES product or report, we would like to hear from you. Please direct your comments to:

National Center for Education Statistics
Office of Educational Research and Improvement
U.S. Department of Education
1990 K Street NW
Washington, DC 20006

September 2001

The NCES World Wide Web Home Page is
http://nces.ed.gov

Suggested Citation

U.S. Department of Education. National Center for Education Statistics. *Comparison of Proc Impute and Schafer's Multiple Imputation Software*. Working Paper No. 2001–10, by Ming-xiu Hu and Sameena Salvucci. Project Officer, Sam Peng. Washington, DC: 2001.

Foreword

In addition to official NCES publications, NCES staff and individuals commissioned by NCES produce preliminary research reports that include analyses of survey results, and presentations of technical, methodological, and statistical evaluation issues.

The ***Working Paper Series*** was initiated to promote the sharing of the valuable work experience and knowledge reflected in these preliminary reports. These reports are viewed as works in progress, and have not undergone a rigorous review for consistency with NCES Statistical Standards prior to inclusion in the Working Paper Series.

Copies of Working Papers can be downloaded as pdf files from the NCES Electronic Catalog (http://nces.ed.gov/pubsearch/), or contact Sheilah Jupiter at (202) 502–7444, e-mail: sheilah_jupiter@ed.gov, or mail: U.S. Department of Education, Office of Educational Research and Improvement, National Center for Education Statistics, 1990 K Street NW, Room 9048, Washington, DC 20006.

Marilyn M. Seastrom
Chief Mathematical Statistician
Statistical Standards Program

Ralph Lee
Mathematical Statistician
Statistical Standards Program

This page intentionally left blank.

Comparison of

Proc Impute and

Schafer's Multiple Imputation Software

Prepared by:

Ming-xiu Hu
Sameena Salvucci
Synectics for Management Decisions, Inc.

Prepared for:

U.S. Department of Education
Office of Educational Research and Improvement
National Center for Education Statistics

September 2001

(Originally delivered September 1996)

This page intentionally left blank.

Table of Contents

This page intentionally left blank.

I. Introduction

In many censuses and sample surveys, missing values commonly exist because some subjects contacted do not respond to some items being asked. These missing values not only mean less efficient estimates because of the reduced size of the database but also that standard complete-data methods cannot be immediately used to analyze the data. Moreover, possible biases exist because the respondents are often systematically different from the nonrespondents; of particular concern, these biases are difficult to eliminate since the precise reasons for nonresponse are usually not known.

Basically, there are four types of imputation procedures to handle missing values: *superficial methods*, such as assigning the mean or mode for all missing cases; *weighting methods*, in which missing values are implicitly filled in by increasing the weights assigned to similar cases that responded; *single imputation* and *multiple imputation*, which are described below.

Single imputation, that is, filling in a value for each missing value, is probably the most common method for handling item nonresponse in current survey practice. There are three major attractive features supporting this practice. First, standard complete-data methods of analysis can be used on the filled-in data set. Second, imputation will obviously be more accurate when close relations exist among variables present and those missing. Variables with close relations can provide information for each other. Third, imputation can incorporate data collector's knowledge. Because the data collectors usually have much better information about and understanding of the process that creates nonresponse than the typical user, it is possible that data analysts, even those with a full arsenal of modern statistical tools, might reach better inferences by trusting the data collector's imputations than by applying sophisticated statistical models to a less rich data base.

Just as these advantages are rather obvious and important, there are equally obvious and important disadvantages of single imputation. When we apply complete-data methods to imputed data sets, inferences based on the imputed data set will be too sharp since the extra variability due to the unknown missing values is not being taken into account. Also, quantities such as correlations that depend on variabilities can be badly biased. Furthermore, when nonreponse is not really understood, no account is being taken of the uncertainty arising from not knowing which nonresponse models for imputation are appropriate. These flaws can be corrected by multiple imputation.

Multiple imputation, replacing each missing value with two or more acceptable values representing a distribution of possibilities, retains the virtues of single imputation and has three extremely important advantages to multiple imputation over single imputation. First, when imputations are randomly drawn in an attempt to represent the distribution of the data, multiple imputation increases the efficiency of estimation. Second, when the multiple imputations represent repeated random draws under a model for nonresponse, valid inferences—that is, ones that reflect the additional variability due to the missing values under that model reflect the additional variability due to the missing values under that model—are

obtained simply by combining complete-data. Third, by generating repeated randomly drawn imputations under more than one model, multiple imputation allows the straightforward study of the sensitivity of inferences to various models for nonresponse simply using complete-data methods repeatedly. However, multiple imputation needs more work to produce multiple imputations and to analyze the multiply-imputed data set, and more space to store a multiply-imputed data set.

This task evaluated two existing imputation software products: *Proc Impute*, created by Statistical Analysis Group in Education (SAGE) and modified by Dr. Wise and Dr. McLaughlin of American Institute for Research (AIR), and *Schafer's Multiple Imputation Software*, created by Dr. Schafer of Pennsylvania State University. The most recent version of *Proc Impute* is a stand-alone Fortran program which can be run under a DOS environment. This version allows a user to generate multiple imputations, but the results may not be "proper" in the sense of Rubin's definition (see I (2)). *Schafer's Multiple Imputation Software* consists of three independent parts for multivariate normal variables, categorical variables and mixed variables, respectively. This software is for a multiple imputation purpose and cooperates with Rubin's "proper" criterion if the sample is a simple random sample.

Detailed evaluations of these two software packages are described in section II and section III, respectively, and a comparison of these two packages and some simulation results are given in section IV. Different imputations for three variables from 1990-91 SASS administrator data file by NCES, *Schafer's Multiple Imputation Software* and *Proc Impute* are also attached in appendix 4. Section V gives our suggestions for next steps.

II. Evaluation of *Proc Impute*

This subtask evaluated the *Proc Impute* imputation software in terms of its usability/performance in a DOS or Windows 486 environment, its suitability for generating multiple imputations, its adaptability to different surveys conducted by the National Center for Education Statistics (NCES), and its feasibility to interface with SAS.

When we use the term "*Proc Impute*" in this report we are referring to the stand-alone FORTRAN program (*PC Impute*) and not the SAS procedure. The stand-alone program is an improved version of the SAS procedure[1] developed by the Statistical Analysis Group in Education (SAGE) under contract with NCES.

In this report all discussions about specific performance standards of *Proc Impute* are based upon runs conducted on the NCES data set "National Survey of Postsecondary Faculty" (NSOPF);[2] these runs were performed in a Pentium (586) environment—90 MHZ clock speed, 16 megabytes of memory, and 600 megabytes of hard disk space.

A. ALGORITHM AND ITS IMPLEMENTATION

Description of Algorithm

Proc Impute is a distributional estimation procedure that is believed to be more general and to produce more accurate results than a standard "hot deck" procedure. Basically, this procedure assumes that relations among variables are constant for observed cases and missing cases, and considers each variable on the file in turn as a "target" variable whose missing values are to be filled in and uses information on other variables to minimize the error in imputing each target variable. For each "target" variable, regression analysis is used to find the best combination of predictors, and cases with the target variable present are divided into subsets based on values of the regression function. All cases in a given subset that are missing the target variable then are imputed with weighted averages of two values drawn from that regression function value subset and an adjacent subset with probability proportional to the distribution of reported values for that variable within these two subsets. The basic assumption of this algorithm is that within these homogenous subsets, the missing value cases will have the same target value distribution as the cases with reported values on the target variable. More specifically, *Proc Impute* makes three passes through the data file.

During the first pass through the data, the program computes basic univariate and bivariate statistics, such as the mean, standard deviation, minimum, maximum, and the number of missing values for each variable, the intercorrelations among the variables, and the number of cases missing one variable but not the other for each pair of variables. Then it determines the best linear predictor of each variable in terms of the remaining variables and an optimal order for imputing missing values. In order to ensure that *Proc*

Impute accurately reflects the strengths of relations among variables on the file, the program allows for two imputation equations for variables—a first "ghost" imputation equation to provide values to use in equations that produces the value to be included in the output data set. Predictors are allowed to enter the equations only if they make significant contributions to reducing error variance.

During the second pass through the data, the bivariate frequency distributions of the regression function values and their associated target variables are estimated by counting the number of cases in each regression value subset at each level of the target variable. Then, each bivariate frequency distribution converted to separate probability distributions of target variables for each regression subset. Moreover, the mean regression function value in each subset is also computed to provide information for interpolation between the distributions in adjacent regression subsets.

During the final pass through the data, missing values are imputed for each case. For each missing value, two subsets are identified: the regression value subset and the adjacent subset. One observed value from each subset is selected with the probability proportional to the relative frequency of that value in that subset. Then the two values drawn from the two adjacent subsets are averaged according to the distance of the mean regression value in each subset from the regression value for the case being imputed. This average value is rounded to an integer if the integer flag is set for the target variable.

After all missing values have been imputed for a case, the case is written to the output file with all of the missing values filled in. Missing data flags are also created and set for each variable with a value of "I" corresponding to imputed values, "R" corresponding to real values and "A" corresponding to skip missing values.

Implementation of Algorithm

The original version of *Proc Impute* created by SAGE is used as a SAS procedure on a JCL mainframe, but the recent PC version modified by Dr. Wise and Dr. McLaughlin of American Institutes for Research is a stand-alone Fortran program which can only be run under a DOS environment. But it is easy to interface with SAS for Windows (see B(4) for details).

B. EVALUATION

Our evaluation consists of answering three main questions (1)-(4) described in the following paragraphs. Answers to questions (3) and (4) are based on our own version of *Proc Impute* that has been slightly modified.

(1) Is it feasible to perform all imputations for a "typical" NCES survey with *Proc Impute* on a 486 PC?

Yes, it is. *Proc Impute* is a multiple-regression-based algorithm. It can impute all types of variables by treating them as continuous, but the imputation values generated by this procedure are within the range of observed values. Categorical variables are treated as if they were ordered, and it may be desirable to recode categorical variables into a series of dichotomous indicators prior to use *Proc Impute*.

Speed and storage are not serious problems to run this software on a 486 PC, but it should be noted that it is less expensive to make a series of calls to *Proc Impute* on small blocks of variables than a single call on a large number of variables, and it is more efficient to put highly correlated variables into the same block and to include key predictor variables in every block. It is advisable to include no more than 30 variables for each call. The user must construct a control file to run the program just once to carry out all the imputations as a batch job.

More specific features about the feasibility of the software are described in the following five questions (a)-(e).

(a) How many runs would it take to impute all variables in a survey?

First, it should be noted that *Proc Impute* uses a regression-based imputation algorithm. Second, in our experience, *Proc Impute* can effectively incorporate no more than 30 variables into any one regression model. Therefore, any "large" data sets (i.e., data sets containing more than 30 variables) must be partitioned into subsets containing no more than 30 variables each before being processed by *Proc Impute.*

Given those considerations, how many runs does it take to impute all variables in a survey? In short, **one** run is all that is required to impute all missing values (for all variables) in any survey. However, in the case of "large" data sets, one run would consist of the following four steps:

(i) First, the analyst must decide upon the specific regression models to use in partitioning the variables. Let's use the 1993 National Study of Postsecondary Faculty (NSOPF) data as an example. If the strategy is to merely use "adjacent" variables in the regression models, then the

analyst would partition the NSOPF data set into about 14 (i.e., 400/30 . 13.3) subsets. If the strategy is to use key predictor variables (e.g., sex, race, and region) in every regression model (in addition to the adjacent variable strategy), then the analyst would partition the NSOPF data set into about 15 (i.e., 397/27 . 14.7) subsets. Obviously, how much the regression models are customized determines not only the number of subsets to be processed by *Proc Impute*, but also the amount of time devoted to the overall imputation process (see part **(e)** below).

(ii) Second, an ASCII data file must be created for each subset of variables (i.e., for each regression model). This can be accomplished by running one SAS program.

(iii) The third step involves running *Proc Impute* on each data subset. To perform an imputation, a control file must be constructed. Fortunately, one can specify all of the regression models in the same control file and, thus, run *Proc Impute* on the entire data set as a single batch job.

(iv) The final step in the process is combining the output (imputed) files into a single file that contains both the original and the imputed values for all variables, with flags indicating imputed values.

While *Proc Impute* will impute **all** missing values in any data (sub)set that is specified in the control file in **one** run and, thus, will impute all missing values for any data file in one run (possibly processed as subsets of the file and run as a batch job), the amount of pre- and post-processing of a given data file is dependent upon the size of the file, the number of variables in the file, the relationships among the variables, etc.

(b) Can *Proc Impute* handle all types of variables (i.e., continuous, ordinal, and categorical) correctly?

Proc Impute will only process "numerical" data; that is, it will only process variables that are both coded and stored as strings of numbers—not variables whose coding and storage allows for character strings. Any "character" variables in the data set must be either recoded to "numeric" or removed from the data set. Once the data set holds the proper coding, each variable will have a continuous, an ordinal, or a categorical distribution—categorical variables can be either dichotomous or polytomous. *Proc Impute's* ability to process each of the distribution types is as follows:

Continuous: Because *Proc Impute* uses a regression-based algorithm, it assumes that each variable is continuous, is distributed normally, and has homogeneous variance. A "standard" linear regression that is run on continuous variables which violate the distribution or variance assumptions often yields high probabilities of generating "out-of-range" predictions. However,

in attempts to avoid imputing "out-of-range" values, *Proc Impute* uses knowledge about conditional frequency distributions along with its regression algorithm when imputing missing values.

Basically, *Proc Impute* considers each variable with missing values as a "target" variable, and uses step-wise regression to identify the best combination of predictor variables for each "target" variable based solely upon those cases where the value of the "target" variable is **not** missing. Once the regression models are constructed and regression values are computed for **all** cases, *Proc Impute* partitions the range of regression values into subsets. For each missing value, two subsets are identified: the regression value subset and the adjacent subset. One observed value from each subset is selected with probability proportional to the relative frequency of that value in that subset. Then the two values obtained from the two adjacent subsets are averaged according to the distance of the mean regression value in each subset from the regression value for the case being imputed. This average value is rounded to an integer if the integer flag is set for the target variable. Hence, all imputed values not only are within the range of observed values but also exhibit distributions similar to the observed values.[4]

In short, *Proc Impute* encounters little difficulty in imputing a "reasonable" set of missing values for continuous variables.

Ordinal: *Proc Impute* handles ordinal variables as if they were continuous. Therefore, all imputed values are within the range of observed values, and all imputed values exhibit distributional properties similar to those of the observed values. We experienced no difficulties in *Proc Impute's* handling of ordinal variables.

Dichotomous: Again, *Proc Impute* assumes that all variables are continuous—if the analyst is willing to assume normality and homogeneous variance for dichotomous variables, then the set of imputed values will have the "nice" properties listed above. We experienced no difficulties in *Proc Impute's* handling of dichotomous variables.

Polytomous: For each polytomous categorical variable, the analyst needs to create an appropriate number of dummy (0/1) variables,[5] and then run *Proc Impute* on the dummy variables. *Proc Impute* handles the dummy variables in the same fashion that it handles dichotomous variables. Since *Proc Impute* does not understand that the dummy variables are grouped as sets of variables, the imputed values may be meaningless;[6] however, since the dummy variables in any set representing a given polytomous variable are highly correlated, that should rarely happen. Fewer than four percent of the imputed values of our "reconstructed" polytomous variables were bad.

In summary, once all "character" variables either have been converted to "numeric" or have been removed from the data set, there exists no special pre- or post-processing for continuous,

ordinal, or dichotomous categorical variables. Also, imputed values for all such variables are reasonable—reasonable in the sense that the imputed values both fall within the range of the observed values and mimic the distributional properties of the observed values.

Polytomous categorical variables are the only type that are potentially troublesome to *Proc Impute*. Special pre- and post-processing is required for polytomous variables—the mapping and inverse mapping of the polytomous variables to and from their associated sets of dummy variables. And, although it is uncommon, *Proc Impute* may impute values into the dummy variables that are meaningless after performing the inverse mapping to the original polytomous variable. In such cases "hot-deck" procedures may be appropriate to impute any remaining missing values.

(c) How much special processing is required to handle skip patterns?

It is very easy for *Proc Impute* to handle skip patterns. The analyst only need to set the skip missing values to "A" in the ASCII data file. If the ASCII data file is created from a SAS data file, then the skip missing values should be set to ".A".

(d) How much memory and disk space is needed?

The amount of required disk space is predominantly a function of the size of your data file. It requires about 480 Kb conventional memory to run *Proc Impute*. With a 586 Pentium PC (16 Mb of total memory and 636 Kb conventional memory), we did not experience memory problems after we remove some programs to generate 488 Kb free conventional memory to run *Proc Impute*. Considering the rate of technological developments, we do not foresee future difficulties.

(e) How fast is it?

For the actual imputation processing, speed is not a serious issue. For the NSOPF survey, a run with 12,000 cases and 30 variables took less than 20 minutes—this translates into a processing time of less than 280 minutes if the entire data set was run as a batch job (14 subsets multiplied by 20 minutes/subset—see question 1a(i)).

However, an analyst will devote the majority of his/her processing time to pre- and post-imputation file management. This time will be spent performing a subset of the following tasks:

- Changing "character" variables to "numeric" variables and/or removing "character" variables from the file.
- Creating ASCII flat file(s)—if working with data in some other format, and
- Constructing control programs (IMPUTE.CON).

For "large" data sets,

- Determining the appropriate partitioning scheme (i.e., determining the regression model subsets), and
- Combining the output (imputed) subsets into one overall completed file.

For polytomous variables,

- Creating dummy variables and then performing the mappings and inverse mappings between the original polytomous variables and the associated dummy variables, and
- Using another method of imputation for individual cases where *Proc Impute* generated "meaningless" values.

For cases with valid skips,

- Removing the cases from the data file before imputation processing, and
- Merging the removed cases with the imputed data file, and
- Using another method of imputation to generate outcomes for variables with "true" missing values.

Working through this list of pre- and post-imputation tasks could easily consume more than a full work-week's worth of the analyst's time. Hence, the non-imputation portion of the processing is easily the most time-consuming part of generating a complete data set using *Proc Impute.*

(2) Does *Proc Impute* perform "proper" imputations in the sense of Rubin?[7] If not, can *Proc Impute* be adapted to perform multiple imputations?

Multiple imputation involves imputing each missing value (in the incomplete data set) multiple times. Hence, in performing a multiple imputation, one creates multiple files of complete data, wherein each of the multiple data files has a different set of imputed values. Once the multiple files have been constructed, the analyst should **replicate** all subsequent analyses by using the information from **all** of the multiple files to assess the impact of random variation (of missing values) on statistical inferences.

Rubin listed three criteria to satisfy a Proper Multiple Imputation (PMI): (i) the multiple imputation procedure provides randomization-valid inferences for the complete-data statistic $\hat{O}$ (the conditional mean of the objective of the study O), (ii) the average of the multiple complete-data variance is centered at U (the conditional variance of O) with variability of a lower order than that of Ö (the variance of the posterior mean of O), and (iii) over repeated samples, the variability of B (the variance of the posterior mean) is also centered at U and is of a lower order than that of $\hat{O}$. These criteria are based on m 6 4 (where m is the number of imputations on the data set).

Rubin's PMI criteria are based on the asymptotic properties of the multiple imputation statistics; hence, the concept of "proper" imputation is exclusively suitable to **multiple imputation** approaches. Since *Proc Impute* uses a **single imputation** procedure based upon a (non-

Bayesian) distributional estimation, *Proc Impute* cannot meet Rubin's criteria for "proper" imputation. However, it is the case that *Proc Impute* is designed to assess the impact of random variation (of missing values) on statistical inferences.

Even though the two methods cannot be compared in the "proper" sense (as introduced by Rubin), we can still examine the criteria for the optimalities of these two methods—the randomization-valid inferences for PMI are based on the concept of the Central Limit Theorem whereas the distributional estimation method employed in *Proc Impute* is based on Pitman's Closeness Criterion.[8]

Proc Impute allows a user to generate m sets of imputations by setting the option "multiple=m" in the control file (see (3) for details). Then the question arises: "as the number of imputations increases, do these sets of imputed values adhere to Rubin's PMI criteria?" The answer depends upon the data, since *Proc Impute* uses regression to find the optimal combination of predictors. If the involved errors agree with the Gauss-Markov assumption then the least-squares estimator gives an optimal fit of the observations to theoretical models. It would not be difficult to verify that multiple imputations generated by *Proc Impute* are "proper," since both the observed "combination of predictors" and the observed "distribution of the cases in the range" would converge to the true "combination of predictors" and the true "distribution in the range," respectively. It should also be noted that the average of m estimators based on the m sets of imputed data is asymptotically unbiased (conditionally on the observed data) if the multiple imputation procedure is randomization-valid.[9]

Since the design and structure of *Proc Impute* are fixed, it would not be easy to incorporate Rubin's strategies into the program.

(3) How to use *Proc Impute*?

This PC version of *Proc Impute* is a stand-alone Fortran program and is invoked by calling the executable file IMPUTE.EXE from DOS. *Proc Impute* expects to find a DOS ASCII control file (the default control file name is IMPUTE.CON) which specifies the imputation problem and the input and output data sets. So there are two steps to run this software: First, construct a control file; then, call IMPUTE.EXE. The format of the control file is given below and an example of a control file is given after the description.

Format of the control file for version 2.0 of *Proc Impute*

PROC IMPUTE options;
    TITLE statement;
    BY statement (optional);

VAR statement;

If there are more than one data subsets which require imputation, we will repeat these statements in the same control file such that all the imputation can be carried out as a single batch job. “PROC IMPUTE” statement and “VAR” statement are required by every subset. It always starts with a “PROC IMPUTE” statement and ends with a semi-colon. The program stops reading statements when it comes to the end of the control file or encounters another “PROC IMPUTE” line, indicating a new imputation request. Detailed description of the statements and the options follows.

PROC IMPUTE Statement

PROC IMPUTE options;

The options that can appear in the PROC IMPUTE statement are given below. These options can be in any order, and most options have default values and therefore need not to be specified. But the input, output, and printout files must be specified through the options “DATA=filename,” “OUT=filename,” and “PRINTOUT=filename,” respectively.

DATA=filename
: specifies the directory and the name of the input ASCII data file to be imputed by *Proc Impute*. If no directory is specified, Proc Impute expects the file is located on the directory where *Proc Impute* is installed.

OUT=filename
: specifies the directory and the name of the output ASCII data file outputed by *Proc Impute*. If no directory is specified, Proc Impute expects the file is located on the directory where *Proc Impute* is installed.

PRINTOUT=filename
: specifies the name of the printout file which reports missing data frequencies and univeriate statistics, characteristics of cases with missing values, correlations between reported values, regression equations, conditional distributions and error analysis.

EQNS=filename
: specifies the name of the output file that contains more detailed information about the regression equations than that given in the PRINTOUT file. But the file is not so readable. The default file name is IMPUTE.EQN.

DISTS=filename
: specifies the name of the output file that contains more detailed information about the conditional distributions than that given in the PRINTOUT file. But the file is not so readable. The default file name is IMPUTE.DIS.

TERSE|VERBOSE

controls the PRINTOUT file. The default value is TERSE. VERBOSE will lead to a much longer PRINTOUT file, and may be used when some unusual results occur.

FLEVEL=number

sets a threshold for letting variables into the prediction equation. A higher FLEVEL will let in more variables. Dr. McLaughlin, one author of *Proc Impute*, claims that the value 0.25 works best with his experience.

HOTDECK|SIMPLE|REGRESS

specifies the method to impute the missing values. The default method is HOTDECK which imputes values selected randomly from the empirical distribution as described in II(A). SIMPLE method imputes the mean value for all missing cases. And REGRESS method imputes the predicted values from the regression equation for the missing cases.

SEED=number

sets random generator seed.

MULTIPLE=number

specifies the number of multiple imputation sets. The number can not be too big due to the storage limitation. If MULTIPLE=4, say, is used as an option, and the output data file is specified as OUTFILE.TXT, for example, then Proc Impute will create four sets of output files named OUTFILE.TXT, OU2FILE.TXT, OU3FILE.TXT and OU4FILE.TXT. That is, for imputations other than one, it replaces the third character in the file name by the index of the multiple.

RECL=number

specifies the record length. RECL does not necessarily equal to the exact length of the records. But it should be noted that the missingness flags are outputed at the position of RECL+2. If the RECL is set to be too small, the missingness flags may overwrite the data. On the other hand, if the RECL is set to be too large, it wastes storage space.

BY Statement

BY var col len;

A BY statement can be used in the control file to allow selection of values for imputation from different distributions for different subsets of the data defined by the BY variables. The syntax is "BY var col len", where "var" gives the variable name, "col" gives the starting column of a record where the BY variable is located, and "len" gives the length of the BY variable. For example, if the BY variable "school" is located in columns 11-15 of a record, we should use "BY school 11 5". The input data file must be pre-sorted by the BY variable if this statement is used.

TITLE Statement

TITLE ‘characters’;

A TITLE statement can be used in the control file to specify a title for the PRINTOUT file.

VAR Statement

VAR variablel col format
variable2 col format
.....
;

A VAR statement must be used in the control file to specify the “target” variables to be imputed by *Proc Impute*. Variables with no missing value may also be specified in this statement so that they can be used in the regression model to provide information for predicting missing values of other variables. The syntax is “VAR variable col format”, where “variable” gives the variable name, “col” gives the starting column where the variable is located, and “format” specifies the format of the values for that variable. None of these three elements can be omitted.

Here is an example of a control file that impute two data sets verif1.dat and verif2.dat, where verif1.dat has 4 variables for imputation and verif2.dat has 6 variables for imputation.

An example of the control file

```
PROC IMPUTE DATA=a:\verif1.dat OUT=verif1.out
     PRINTOUT=verif1.prn  EQNS=verif1.eqn  DISTS=verif1.dis
     TERSE  HOTDECK  SEED=11111  FLEVEL=0.25
     MULTIPLE=1  RECL=66;
     TITLE 'Imputations for data set verif1 with 4 variables';
     VAR   Y1      1   8.3
           Y2  9   8.3
           Y3    17    8.3
           Y4    25    8.3
     ;

  PROC IMPUTE DATA=a:\verif2.dat OUT=a:\verift2.out
     PRINTOUT=verif2.prn EQNS=verif2.eqn DISTS=verif2.dis
     TERSE HOTDECK  SEED=11111  FLEVEL=0.25
     MULTIPLE=1 RECL=65;
```

```
TITLE 'Imputations for data set verif2 with 5 variables';
VAR   Y1     1   8.3
      Y2     9   8.3
      Y3    17   8.3
      Y4    25   8.3
      Y5    33   8.3
;
```

(4) How to interface *Proc Impute* with SAS for Windows?

Proc Impute can be run from within SAS so that an analyst can do both imputation and analysis for the imputed data within the same SAS session. Suppose that we want to impute an incomplete SAS data file named *incomp.sd2*, and output a imputed SAS data file named *comp.sd2*, which keeps the original variable names, labels, and formats, etc, as in *incomp.sd2*. We will need the following five simple steps to do the job:

(i) Create an ASCII input file (named as a:\temp1.dat) for *Proc Impute* with SAS statements:

data tmp1; set incomp;
file "a:\tmp1.dat";
put variable list; (with fixed formats)
run;

(ii) Construct a control file as described in B(3) above. An easiest way to specify the control file is to open an old control file into SAS editor or somewhere else and modify it. Any convenient variable name (such as X1, X2, ..., etc) can be used in the VAR statement in the control file. They do not have to be the same as in *incomp.sd2*.

(iii) Use SAS File|Run pull-down menu to run *Proc Impute* from within SAS. Let us call the output data file as *a:\tmp2.dat*.

(iv) Create a complete SAS data file from imputed data file *a:\tmp2.dat*.

data tmp; infile "a:\tmp2.dat";
input variable list; (the same variable names as in incomp.sd2)
run;

(v) Create the target SAS complete data file which keeps the variable names, labels, format, etc, as in the original SAS data file *incomp.sd2*:

data comp; merge incomp tmp; run;

For a "large" data set that has over 30 variables, we may have to divide the data set into several subsets, and perform steps (i) once for each subset but we only need to perform steps (ii) and (iii) once for all subsets. If the single imputation option is selected, we need to execute steps (iv) and (v)

once for each subset, and then use another MERGE statement to combine all these complete SAS data files into one complete SAS data file if necessary; if the multiple imputation option is selected, we need to execute steps (iv) and (v) multiple times for each subset and use MERGE statement multiple times to combine the corresponding data files (one file from each subset each time) to generate multiple imputed data files.

C. NOTES

[1] *PC Impute* contains three refinements—McLaughlin, Donald H. (1991), "Imputation for Non-Response Adjustment," Internal Report. American Institutes for Research: Palo Alto, California.

[2] NSOPF (faculty survey) contains approximately 12,000 cases and 400 variables.

[3] We ran *Proc Impute* on a sample consisting of 872 cases and 31 variables on both a 486 machine (33 MHZ speed, 16 Mb memory, 110 Mb storage) and the above 586 machine. The processing times were 55 (± 1) seconds and 15 (± 1) seconds, respectively. These outcomes indicate that the 586 is 3.4 to 4 times faster than the 486 in processing a *Proc Impute* run. However, it should be noted that this conversion factor is a function of many things: the number of cases, the number of variables, the number of missing values, the pattern of missing values, the correlations among the variables, etc.

[4] The basic assumption of this algorithm is that within these homogeneous subsets, the missing value cases have "target" variable distributions identical to the "target" variable distributions of cases with reported values—SAGE (1980), "Guidebook for Imputation of Missing Data," prepared for NCES (contract #300-78-150). American Institutes for Research: Palo Alto, CA.

[5] For example, for a ten-category variable one needs to create nine 0/1 dummy variables, where for each case either: (a) eight of the dummy variables are coded with the value 0 and the remaining dummy variable is coded with the value 1 (indicating that the original polytomous variable case belongs to the dummy category coded "1") or (b) all nine of the dummy variables are coded with the value 0 and no dummy variable is coded with the value 1 (indicating that the original polytomous variable case belongs to the "missing" dummy category). In general, an n-category variable would be associated with (n-1) dummy variables having coding schemes analogous to the above example.

[6] In the above example, for a specific case, it may be the situation that more than one of the nine dummy variables will be imputed with the value "1"—this would indicate that the original polytomous variable case assumes multiple categories simultaneously!

[7] Rubin, Donald B.(1987), Multiple Imputation for Nonresponse in Surveys. New York: John Wiley & Sons.

[8] For an estimation problem with parameter space 1, an estimator *$_1$ is said to be Pitman closer (to ›) than *$_2$, if, for every ›01, P›(**$_1$(X)-›*<**$_2$(X)-›*)>0.5. This criterion is called Pitman closeness or Pitman nearness or Pitman domination.

[9] Rubin, Multiple Imputation for Nonresponse in Surveys, 116.

III. Evaluation of Schafer's Multiple Imputation Software

Schafer's Multiple Imputation Software consists of three independent parts: the first part uses a multivariate normal model to impute continuous variables; the second part uses a saturated multinomial model and a constrained loglinear model (Bishop, Fienberg and Holland, 1975) to impute categorical variables; and the third part uses restricted and unrestricted general location models (Olkin & Tate, 1961) to impute mixed variables (include both categorical variables and continuous variables in one model). We implemented all three parts on a PC environment and evaluated them in terms of its usability/performance in a Windows 486 PC environment, and its adaptability to different surveys conducted by NCES.

All discussions about specific performance standards of this software are based on runs conducted on the NCES data set "1993-94 School and Staffing Survey Administrator Component" (SASS.AS). These runs were performed in a Pentium (586) environment—90 MHZ clock speed, 16 megabytes of memory, and 520 megabytes of hard disk space.

A. ALGORITHM AND ITS IMPLEMENTATION

Description of the Algorithms

Schafer's multiple imputation software uses multivariate normal models, multinomial models and general location models to impute missing values for continuous variables, categorical variables and mixed variables, respectively. All models assume that the missing mechanism is ignorable; that is, missing values occur at random. Brief descriptions of the three types of models follow. Details about these models can be found in Schafer (1991).

(1) Algorithms for Incomplete Continuous Data

Suppose that a random vector $\mathbf{X}=(Y_1, Y_2, ..., Y_p)$ has a multivariate normal distribution MN($\boldsymbol{\mu}$,$\boldsymbol{\Sigma}$), the prior distribution of $\boldsymbol{\mu}$, given $\boldsymbol{\Sigma}$, is multivariate normal MN($\boldsymbol{\mu}_0$, $J^{-1}\boldsymbol{\Sigma}$), and the prior distribution of $\boldsymbol{\Sigma}$ is normal-inverted Wishart W^{-1} (m, $\boldsymbol{\Psi}$). Then the posterior distributions of $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are also multivariate normal MN ($\boldsymbol{\mu}'_0$, $(J')^{-1}\boldsymbol{\Sigma}$), and normal-inverted Wishart W^{-1} (m′, $\boldsymbol{\Psi}'$), respectively, where the updated hyperparameters are

$$t' = t + n$$

$$m' = m + n$$

$$m_0' = \left(\frac{n}{t+n}\right)\bar{y} + \left(\frac{t}{t+n}\right)m_0$$

and

$$(\Lambda')^{-1} = \Lambda^{-1} + nS + (\frac{tn}{t+n})(\bar{y} - m_0)(\bar{y} - m_0)^T .$$

Where n stands for the sample size and S stands for the sample variance.

First, this software uses the EM algorithm (Dempster, Laird, and Rubin, 1977; Little and Rubin, 1987) to find the Maximum Likelihood Estimates of **μ** and **G**, which may be used as the starting values in the iterative simulation step. Then, the software applies the iterative simulation method to simulate one or more iterations of a single Markov chain (Schafer, 1995). Each iteration consists of a random imputation of the missing data drawn from multivariate normal distribution with current parameter values (I-step), followed by a random draw from the posterior distributions of the parameters, multivariate normal distribution of **μ** and normal-inverted Wishart distribution of **G,** given the observed data and the imputed data (P-step).

(2) Algorithms for Incomplete Categorical Data

Let Y_1, Y_2, ..., Y_p denote the p categorical variables recorded for n units (rows) in the n×p data matrix Y. Denote the possible values of Y_j by the positive integers 1, 2, ..., d_j for j=1, 2, ..., p. Each row of Y can be assigned to a unique cell of the p-dimensional contingency table that cross-classified the data by Y_1, Y_2, ..., Y_p . Denote the total number of cells in this table by D= **J** d_j, the cell probabilities by $2=\{2_d: d=1, 2, ..., D\}$, and the cell counts by $\{x_d: d=1, 2, ..., D\}$, after we re-index the cells by 1, 2, ..., D in such an order that the first variable varies the fastest, and the second variable varies the second fastest, and so on. The software considers two types of models for categorical data.

(2.1) Saturated Multinomial Model

Suppose Y has a multinomial distribution with a density function

$$f(y|q) = \frac{n!}{\prod_{d=1}^{D} x_d! I_S} \prod_{d=1}^{D} q_d^{x_d} I_S, \qquad (A.2.1)$$

where I_S is the indicator for the simplex $S=\{2: 2_d >0, \ 3 2_d=1 \}$. If a cell probability is 0, we call it *structural zero* and exclude it from any further calculation. Assume the priors for the cell probabilities 2 are Dirichlet distribution with hyperparameters $\{<_d\}$(the natural conjugate prior),

$$p(q) \propto \prod_{d=1}^{D} q_d^{\nu_d - 1} I_{S,} \qquad (A.2.2)$$

then the posterior distribution for 2 is also Dirichlet with hyperparameters $\{x_d + <_d - 1\}$,

$$f(q \mid y) \propto \prod_{d=1}^{D} q_d^{x_d + \nu_d - 1} I_S \qquad (A.2.3)$$

To apply this saturated multinomial model to incomplete categorical data, the software first use the EM algorithm to find maximum likelihood estimate or posterior mode of cell probabilities 2, which may be used as the starting values in the iterative simulation step. Then, the software applies the iterative simulation method to simulate one or more iterations of a single Markov chain. Each iteration consists of an I-step and a P-step. The I-step draws a random imputation for the missing data from multinomial distribution with current parameter estimates, and the P-step draws parameter estimates of the cell probabilities 2 from the posterior Dirichelet distribution. Details about the implementation of I-step can be found in Schafer (1991, pp. 79-80). The P-step is very straightforward.

(2.2) The constrained loglinear model

The saturated multinomial model fits the full set of D-1 parameters in 2 and may be appropriate when the number of cases n is large relative to the number of cells D. As the number of variables p grows, however, D quickly becomes enormous, and it may be undesirable to estimate all D-1 parameters. In such cases, it is customary to reduce the dimensionality of the problem by requiring 2 to satisfy a set of loglinear constraints. Now let $2_{ijk...t}$ denote the cell probability for the cell where $Y_1 = i$, $Y_2 = j$, ..., $Y_p = t$. We may impose the loglinear constraints

$$\log q_{ijk...t} = u + u_{1(i)} + u_{2(j)} + \ldots + u_{p(t)} + u_{12(ij)} + u_{13(ik)} + \ldots + u_{123(ijk)} + \ldots \qquad (A.2.4)$$

on the cell probabilities, where, for identifiability, the u-terms are constrained to sum to zero over any subscripts; for example,

$$\sum_{i=1}^{d_1} u_{12(ij)} = \sum_{j=1}^{d_2} u_{12(ij)} = 0.$$

By setting some of u-terms to zero, especially the higher order interactions, we can often capture the essential features of the data set without resorting to the estimation of all D-1 parameters.

The maximum likelihood estimates of 2 can be obtained through the algorithm of Iterative Proportion Fitting (Bishop, Fienberg and Holland, 1975). The GIBS algorithm, a stochastic version of IPF called Bayesian IPF by Gelman, Meng and Rubin (1991), is used to simulate the posterior distribution of 2 under this loglinear model.

To apply this constrained loglinear model to incomplete categorical data, the software first use the ECM algorithm (Meng and Rubin, 1991) to find the maximum likelihood estimate or posterior mode of the cell probabilities, which may be used as the starting values in the iterative simulation step. Then, the software applies the iterative simulation method to simulate one or more iterations of a single Markov chain. Each iteration also consists of an I-step and a P-step. The I-step is identical to the I-step of the saturated multinomial model, while the P-step uses the Bayesian IPF algorithm to draw parameter estimates for the cell probabilities.

(3) Algorithms for Incomplete Mixed Data

Partition the complete data matrix Y as Y=(W, Z), where W is an n×p matrix of categorical variables, and Z is an n×q matrix of continuous variables. Let W_1, W_2, ..., W_p and Z_1, Z_2, ..., Z_q denote the variables in W and Z, respectively. As in A(2) above, denote the possible values of W_j by the positive integers 1, 2, ..., d_j, the total number of cells by D=A d_j, the cell probabilities by B={B_d: d=1, 2, ..., D}, and the cell counts by {x_d: d=1, 2, ..., D}, after we re-index the cells by 1, 2, ..., D in such an order that the first variable varies the fastest, and the second variable varies the second fastest, and so on. Let U be the n×D matrix with rows u_i^T, i=1, 2, ...n, where u_i is a D-vector containing a 1 in position d if the ith row of W falls into cell d, and 0's in all other positions.

(3.1) The general location model (Olkin and Tate, 1961)

This model assumes that the marginal distribution of W is a multinomial distribution on the cell counts {x_d: d=1, 2, ..., D},

$$f(y \mid p) = \frac{n!}{\prod_{d=1}^{D} x_d! I_S} \prod_{d=1}^{D} p_d^{x_d} I_S, \qquad (A.3.1)$$

where I_S is the indicator for the simplex S={B: B_d >0, 3B_d=1 }. If a cell probability is 0, it will be excluded from any further calculation. Given W, the rows of Z are conditionally modeled as multivariate normal. Let E_d be a D-vector containing a 1 in position d, and 0's elsewhere. The conditional distribution of the ith row of Z, given u_i =E_d (i.e., ith row falls in cell d), is assumed to be MN(μ_d, E), where μ_d is a q-vector of means corresponding to cell d. This model allows the means of Z_1, Z_2, ..., Z_q to vary freely from cell to cell, but assumes a common covariance structure E for all cells.

Let $\Psi=(\Pi, \mu, \Sigma)$ denote all the parameters in this model, where $\mu=(\mu_1, \mu_2,..., \mu_D)^T$ is a D×q matrix of means. Assume the prior of Π is a Dirichlet distribution with hyperparameters $\alpha=\{\alpha_d\}$, and the prior distribution of (μ, Σ) is the diffuse Jeffreys prior $P(\mu, \Sigma) \propto |\Sigma|^{-(q+1)/2}$. Then the posterior distribution of Π, $P(\Pi|W)$, is a Dirichlet distribution with hyperparameters $\{x_d+\alpha_d-1\}$, and the posterior distribution of μ and Σ is multivariate normal and normal-inverted Wishart; that is,

$$P(\sum | Z, W) = W^{-1}(n-D, (\hat{\epsilon}^T \hat{\epsilon})^{-1})$$
$$P(m_d | \sum, Z, W) = N(\hat{m}_d, x_d^{-1} \sum) \qquad (A.3.2)$$

for d=1, 2, ..., D, where

$$\hat{m} = (U^T U)^{-1} U^T Z$$
$$\hat{\epsilon} = Z - U\hat{m}. \qquad (A.3.3)$$

If any cell count x_d is zero, the matrices U and $U^T U$ will have deficient rank, and (A.3.3) will not be defined. In this case , the posterior distribution will be improper due to the inestimability of μ_d. When this occurs, an analysis under this prior may proceed by omitting the inestimable parameters μ_d from the model or by reducing the dimensionality of the parameter though a constrained model as described in section (3.2) below.

To apply this unrestricted general location model to incomplete mixed data, the software first use the EM algorithm to find the maximum likelihood estimates of the cell probabilities, the cell means and the covariances, which may be used as the starting values in the iterative simulation step. Then, the software applies the iterative simulation method to simulate one or more iterations of a single Markov chain. Each iteration consists of an I-step and a P-step. The I-step draws a random imputation for the missing categorical data and missing continuous data from the predicted multinomial distribution and multivariate normal distribution, respectively, with current parameter estimates. Details about the implementation of this step can be found in Schafer (1991, pp. 111-115). The P-step simulates parameter estimates of cell probabilities Π, cell means μ, and covariances Σ from their posterior distributions, which are Dirichelet, multivariate normal and Wishart (A.3.2), respectively. This step is very straightforward.

(3.2) The restricted general location model

The unrestricted general location model has (D-1)+Dq+q(q+1)/2 free parameters, and is useful when n is large relative to the number of the free parameters. D and then D×q become enormous

very quickly when the number of categorical variables p grows. A restricted model is more desirable in practice for mixed data. The same loglinear constrains can be applied to the cell probabilities as in section (2.2). Here we discuss the constrains on the within-cell means μ of the continuous variables. Let A be a D×r design matrix and μ=A$, where $ is a r×q matrix. We assume that rank(A)=r#D. So we only need to estimate rq parameters of $ instead of Dq parameters of μ. This constrained model still allow the means μ_i to vary from cell to cell, but now require that each of the q columns of the matrix μ lies in the r-dimensional linear subspace of R^D spanned by the columns of A. By saturating the loglinear model for B and taking $A=I_{D\times D}$ (identity matrix), we obtain the general location model as a special case.

In this restricted model, the same prior is assumed and the same posterior will be obtained for the cell probabilities as in section (2.2) above, since we apply the marginal distribution to cell probabilities which is a separate factor in the full likelihood in the model. For parameters ($, E), we still assume the Jeffreys prior; that is, $B(\$, E)\%|E|^{-(q+1)/2}$, and the posterior distributions are

$$P(\sum | Z,W) = W^{-1}(n-4,(\hat{\epsilon}^T\hat{\epsilon})^{-1}$$
$$P(b | \sum, Z, W) = N(\hat{b}, \sum \otimes V), \qquad (A.3.4)$$

where

$$b = V^{-1}A^TU^TZ, \quad \hat{\epsilon} = Z - UA\hat{b}, \quad V = (A^TU^TUA)^{-1},$$

and E¼V is the Kronecker product of E and V (Anderson, 1984, pp. 599-601).

To apply this restricted general location model to incomplete mixed data, the software first use the ECM algorithm to find the maximum likelihood estimates or posterior modes of cell probabilities, the cell means and the covariances, which may be used as the starting values in the iterative simulation step. Then, the software applies the iterative simulation method to simulate one or more iterations of a single Markov chain. Each iteration also consists of an I-step and a P-step. The I-step is identical to the I-step of the unrestricted general location model, while the P-step draws parameter estimates for the cell probabilities B through the Bayesian IPF algorithm, and simulates parameter estimates for $ and E from their posterior distribution (A.3.4).

Implementation of algorithm

Dr. Schafer uses S-PLUS functions and Fortran subroutines, which support the S-PLUS functions, to implement the above algorithms. The software works in an S-plus environment. When we use the term “the software” in this report, we will usually refer to the S-plus functions rather than the Fortran subroutines. There are a total of twelve S-PLUS functions for dealing

with normal continuous variables, eighteen for categorical variables and nine for mixed variables. But not all are needed for imputation purposes (see appendix 1-appendix 3 for details). These functions can be classified into four categories: preliminary data manipulation functions, functions for EM or ECM algorithm, simulation and imputation functions, and multiple imputation inference functions.

Due to a problem with the random generators in the original version of the software, none of the simulation and imputation functions works. We also need to fix some storage mode errors in the Fortran subroutines in order that the WATCOM Fortran compiler can successfully compile the Fortran source files in the software. Actually, fixing all these problems takes us more time than the evaluation process. However, the software still has some problems with the constrained loglinear model for categorical variables, and the constrained general location model for mixed variables. These two constrained models try to increase the number of variables in one run by imposing constrains on the parameters. But, because the programs for these two models work so slowly and still need the same amount of space to store the parameters as their corresponding saturated models, we can only put one or two more categorical variables in those constrained models. Furthermore, both models have variable ordering problems. The variables stored in a particular order according to the missingness pattern after the preliminary data manipulation, while users will usually specify the u-terms (interactions) in (A.2.4) for the programs in the original order of the variables. Then the program will mis-match the variables and therefore the results could be wrong. The results will be correct if the u-terms are symmetrically designed; that is, if one interaction of a certain order in (A.2.4) is included in the model, all interactions of that order must be included. The user may call those functions in the software for the constrained loglinear model (ECM.CAT and DABIPF.CAT) if the u-terms are symmetrically selected, but the functions in the software for the restricted general location model (ECM.MIX and DABIPF.MIX) are not recommended in any circumstance because they work very slowly and some unexpected errors, such as “overflow range error,” may happen for some specific priors.

Dr. Schafer is working on a new version of the software which will not re-order the variables in the preliminary data manipulation so that the variable ordering problems described here will not exist any more in the coming new version.

B. EVALUATION

Our evaluation consists of answering three main questions (1)-(3) described in the following paragraphs.

(1) Is it feasible to perform all imputations for a "typical" NCES survey with this software on a 486 PC?

Yes, it is. Combining all three parts of the software with five types of models as described in section A, we can impute all types of variables for a “typical” NCES survey. A large data set may have to

be divided into several subsets. The number of variables in each subset depend on the number of cases and the number of free parameters in the models. Higher correlated variables with similar scales should be put into the same subset. We may use the first part of the software to impute continuous variables, and the second part to impute for the categorical variables. If some categorical variables are highly correlated with some continuous variables, we may want to use the third part of the software to impute the missing values for these mixed variables.

Speed and storage are not very serious problems to run this software on a 486 PC. Furthermore, this software is easy to use and convenient to handle skip patterns. More specific features about the feasibility of this software are discussed in the following five questions (a)-(e).

(a) How many runs would it take to impute all variables in a survey?

S-PLUS deals with a data set through a matrix: the rows represent cases and the columns represent variables. Because object sizes and dynamic memory are limited in S-PLUS, a large data set must be partitioned into several subsets. The partition strategy is to put highly correlated variables with close scales (for continuous variables) into the same subset. This makes the convergence criterion for the EM or ECM algorithms easier to set up and very likely produces more accurate results. The number of variables in each subset depends on the number of cases and the number of free parameters to be estimated in the model, which may include cell probabilities, cell means and variance-covariances. On one hand, more cases can estimate more free parameters so that we can include more variables in the model; on the other hand, more cases leads to a bigger S-PLUS object (data matrix) so that we can have less variables in the model. The number of variables in each subset should be determined such that (1) the size of the data matrix and the dynamic memory requirement must be under S-PLUS limitation, and (2) the number of cases must be relatively large to the number of free parameters.

With the multivariate normal model for continuous variables, we do not have too many free parameters, and the number of free parameters is not a crucial factor to decide the number of variables in each subset. The software can incorporate with 30 variables in one subset if the number of cases is less than 15,000. With all other four models (saturated multinomial model, constrained loglinear model, unrestricted and restricted general location model), both factors, the number of cases and the number of free parameters, are crucial to determine the number of variables in each subset. Generally speaking, an analyst may want to include as many variables in a subset as the software can correctly handle since more variables in the model will provide more information for each other to predict the missing values, and including more variables in each subset will lead to less subsets and therefore less runs. But we do not recommend more than 30 variables in any model due to consideration of speed, storage and memory requirement, and the number of free parameters which need to be estimated.

The SASS.AS data set has 9,415 observations, 15 categorical variables, and 56 continuous variables, which are appropriate for imputation. There are a lot of ways to partition these variables. For example, we may use three saturated multinomial models for the 15 categorical variables and two multivariate normal models for the 56 continuous variables; or we may use two constrained loglinear model without third or higher order interactions for the 15 categorical variables and two multivariate normal models for the 56 continuous variables; or we may use three unrestricted general location models with 5 categorical variables and 19 continuous variables apiece, etc. But we do not recommend the user to use the restricted general location model because it works very slowly and some unexpected errors, such as "overflow range error," may happen for some specific priors. We mean that the S-PLUS functions for this model in this version of the software do not work so well.

After the variables have been divided into a certain number of subsets and each subset of data has been read into an S-PLUS data matrix, the following runs are required to impute the variables for each subset:

(i) Call function **prelim.norm** for multivariate normal models, **prelim.cat** for saturated multinomial models and constrained loglinear models, or **prelim.mix** for restricted and unrestricted general location models, to perform some preliminary manipulations, such as centering, scaling, and sorting by missingness patterns on a matrix of incomplete data.

(ii) Call function **em.norm** for multivariate normal models, **em.cat** for multinomial models, **ecm.cat** for constrained loglinear models, **em.mix** for unrestricted general location models, and **ecm.mix** for restricted general location models, to find the maximum-likelihood estimates of the parameters with the incomplete data using the EM or ECM algorithm. These parameter estimators of cell probabilities (if categorical variables are present), means, and variance-covariances, will usually be used as starting values of parameters for the iterative simulation functions **da.norm**, **da.cat**, **dabipf.cat**, **da.mix** and **dabipf.mix**.

(iii) Call function **da.norm** for multivariate normal models, **da.cat** for saturated multinomial models, **dabipf.cat** for constrained loglinear models, **da.mix** for unrestricted general location models, or **dabipf.mix** for restricted general location models, to simulate one or more iterations of a single Markov chain under a normal-inverted Wishart prior. These functions draws parameter estimates from their posterior distributions. These parameter estimates will be used by step (iv) below to generate imputations for missing values.

(iv) Call function **imp.norm** for multivariate normal models, **imp.cat** for saturated multinomial models and constrained loglinear models, or **imp.mix** for general location models (both restricted and unrestricted), to impute the missing values of the data matrix under user-

supplied values of the parameters (usually use the parameter estimates from step (iii)). These functions will return a matrix of complete data.

Steps (iii) and (iv) can be performed multiple times to generate multiple imputations.

(b) Can it handle all types of variables?

Yes, it can. We may use multivariate normal models to impute continuous variables, saturated multinomial models and/or constrained loglinear models to impute for categorical variables, and restricted and/or general location models to impute mixed variables when categorical variables are highly correlated with continuous variables. We believe that the models for categorical variables and mixed variables in this software are more appropriate than *Proc Impute* which fits usual regression models by treating all types of variables as continuous variables.

The multivariate normal assumption is crucial to the Schafer's normal imputation models. If the variables departure too far from normality, the imputations generated by Schafer's software could be very bad. Theoretically speaking, any continuous variable can be transferred to a variable with a normal distribution. But we may not be able to do so in practice since the true distribution of a variable rarely known to us, or the exact transformation may be too complicated to perform even if we know the distribution of the variable. Therefore, in practice, we may first use some common transformations (e.g., logarithm, exponential, square root, square, etc.) to make the variables as close to normal variables as possible, then apply Schafer's multivariate normal imputation models to the transferred variables.

It should be noted that all four types of models which involve categorical variables do not take explicit account of any ordering of the categories; that is, they regard the possible levels of each categorical variable as unordered categories. In some case, incomplete ordinal data can approximately be handled by pretending that they are normally distributed and applying the multivariate normal model. In other cases, we may disregard the ordering and apply the multinomial model. Although the multinomial model may result in some loss of information and may be less satisfactory for the development of scientifically meaningful models, it may be a perfectly reasonable approach if our goal is merely to produce plausible multiple imputations of the missing data for future analyses (Schafer, 1991, p. 71).

(c) How much special processing is required to handle skip patterns?

It is very easy to handle skips in S-PLUS. For example, suppose that "NA" represents the real missing values, "99999" represents" the valid skips, and x is the data matrix, then the following four statements will do the job:

(i) Record positions of valid skips: **pos_(x==99999)**,

(ii) Treat all valid skips as "true" missing values: **x[pos]_NA**

(iii) Use Schafer's software to impute all missing values including real missing values and valid skips.

(iv) Remove all imputed values for valid skips: **x[pos]_99999**

Only one minute may be needed to handle all valid skips in this way. If the data set includes several different "skip flags," more statements will be needed, but they will be similar to (i), (ii), and (iv).

(d) How much memory and disk space would be required?

The amount of required memory and disk space depends on the size of the data matrix. Due to the object size limitation and the dynamic memory limitation in S-PLUS for Windows, the PC environment where Schafer's software works, the number of variables in each run is limited. For a 486 PC, we do not recommend more than 30 variables in any model. Otherwise, it may run out of dynamic memory or exceed the object limitation. As the number of cases is 20,000 or more, we recommend 20 variables or less in one run.

For a model involving categorical variables (saturated multinomial model, constrained loglinear model, restricted/unrestricted general location models), the number of variables in each model not only depends on the number of cases, but also depends on the number of free parameters in the model. The number of cases must be relatively large to the number of free parameters. The number of cells becomes enormous as the number of categorical variables grows. So the saturated multinomial model and the unrestricted general location model can only include a few categorical variables. Ten categorical variables will have at least 2^{10} =1,024 cells (and therefore at least 1,024 free parameters in the saturated models) if all variables have only 2 levels. In a real survey, 10 variables usually have much more than 1,024 cells. For example, the first 10 categorical variables in the SASS.AS data set include 4 variables with 3 levels, 5 variables with 4 levels, and one variable with 5 levels, which will lead to $3^4 \times 4^5 \times 5$=414,720 cells. Definitely that is too much for one model.

Theoretically, the constrained loglinear model and the restricted general location model can include much more categorical variables in each run since we can control the number of the free parameters as we want to. However, we actually can not put too much more variables into these models due to two reasons: (1) the S-PLUS functions for these models in the software work so slowly that we can not afford a big model (see next section for details about speed), and (2) they require the same amount of space to store the parameters. The software store all the parameters in one vector called "theta" in double precision. The size of "theta" grows as twice fast as the number of cells when the number of categorical variables grows. It will exceed

the object size limitation (default value is 5 MB) very quickly. Although we can increase the maximum object size in S-PLUS to run Schafer's software with a "theta" larger than 5 MB, it will be very likely that the program runs out of dynamic memory if we do so. Even if we have enough memory and space to run with such a big object, the processing time is un-affordable. For example, it will take several hours to run the constrained loglinear model without third or higher order interactions for those 10 categorical variables in the SASS.AS data set mentioned in the previous paragraph. While it only takes about a minute to run two saturated multinomial models with 5 categorical variables apiece, and merge the two imputed subsets back into one subset. So it is really not a good idea to run this software with too many categorical variables in terms of cost.

Here we try to give our recommendation on the number of variables that each model should include in for data sets with 5,000 to 12,000 cases, but it should not be surprising if they are not appropriate for some situations. A user really should determine this issue on his/her own based on the data set he/she got.

Model	**# of variables in one run** (5,000 to 12,000 cases)
Multivariate normal model	30 continuous variables
Saturated multinomial model	5-8 categorical variables
Constrained loglinear model (Without 3rd or higher interactions)	7-10 categorical variables
Unrestricted general location model	about 5 categorical variables & 15-20 continuous variables
Restricted general location model (without 3rd or higher order interactions)	better not use it with this version

As mentioned earlier, a large data set must be partitioned into smaller subsets, and run the software on one subset at a time. We experienced that the second run was hung up when we made two runs of the software in the same S session. It is advisable for a user to quit an S session after running Schafer's software for one subset in that session, and enter another S session to run the software for the second subset if both subsets have more than 25 variables and over 10,000 cases.

(e) How fast is it?

The imputation processing time depends on the size of the data matrix, the number of iterations specified for the iterative simulation algorithm and that for EM or ECM algorithms. Usually, 25 iterations will generate quite stable results for both algorithms.

We have run each model on one or two subsets for the SASS.AS survey data set with 9,415 cases. All runs are supposed to take 25 iterations for both iterative simulation algorithm and EM or ECM algorithm. The imputation processing time for each model is given as follows.

S-Functions	**Time**	**Description**
Multivariate Normal Model (with 30 continuous variables)		
prelim.norm:	1'10"	preliminary data manipulations
em.norm:	2'15"	initial parameter estimates by EM algorithm
da.norm:	4'20"	iterative simulation (data augmentation)
imp.norm:	1'55"	imputation of the missing values
Total:	9'40"	
Saturated Multinomial Model (6 variables with 4×5×3×4×3×4=2880 cells)		
prelim.cat:	5"	preliminary data manipulations
em.cat:	10"	initial parameter estimates by EM algorithm
da.cat:	5"	iterative simulation (data augmentation)
imp.cat:	2"	imputation of the missing values
Total:	22"	
Constrained Loglinear Model (8 variables with 4×3×3×3×4×3×4×4=20736 cells and no higher than 2nd order interactions)		
prelim.cat:	10"	preliminary data manipulations
ecm.cat:	3'35"	initial parameter estimates by ECM algorithm
dabipf.cat:	3'38"	Bayesian iterative proportional fitting
imp.cat:	5"	imputation of the missing values
Total:	7'28"	

Unrestricted General Location Model (20 continuous variables and 5 categorical variables with 3×5×3×4×3=540 cells)

prelim.mix:	1'15"	preliminary data manipulations
em.mix:	2'15"	initial parameter estimates by EM algorithm
da.mix:	1'20"	iterative simulation (data augmentation)
imp.mix:	1'30"	imputation of the missing values
Total:	6'20"	

Restricted General Location Model (20 continuous variables and 5 categorical variable with no higher than 2nd order interactions and design matrix $A=I_{D\times D}$)

prelim.mix:	1'15"	preliminary data manipulations
ecm.mix:	6 hrs	initial parameter estimates by ECM algorithm
dabipf.mix:	forever	Bayesian iterative proportional fitting

From these results, we can see that the actual imputation processing speed is fast for all models except the restricted general location model. It is worth to point out that a model including one more variable may cost a lot more running time. For example, for the constrained loglinear model, a run (including all 4 steps) with 7 categorical variables takes 1 minute and 23 seconds, with 8 variables takes 7 minutes and 28 seconds, while with 9 variables takes several hours (we run DABIPF.CAT and ECM.CAT for a couple of iterations and estimate the total time for 25 iterations). The speed is really sensitive to the size of data set, especially the number of categorical variables. We should not put too many variables into any single model, but we also should not put too few variables into the model since more variables in the model will provide more information for each other to generate more accurate imputations. In the above example, it is appropriate to put 7 or 8 variables into the constrained loglinear model, while 9 are too many and 6 are too few.

It should also be noted that the Bayesian Iterative Proportional Fitting Algorithm works very slowly with this version of the software and it really can not increase too many variables in the models than the saturated models due to the storage limitation of "theta", the container for the parameters (see (B.1.d) above for details). Moreover, the time is unaffordable to run a constrained model with third order (or higher) interactions for a data set with a reasonable size.

In summary, it will not take too much imputation processing time to impute all missing values for a survey if we choose the "appropriate" models in Schafer's software. However, an analyst will devote the majority of his processing time to pre- and post- imputation file-management. This time will be spent performing a subset of the following tasks:

- Checking the distributions of the variables, and the correlations between the variables;
- Performing data transformations for those continuous variables with sever violation of the normality assumption;
- Re-coding the levels of the categorical variables with positive integers starting with 1 if necessary (may use function “categorize” provided by the software);
- Partitioning the data set into subsets according to the rules of putting highly correlated variables with close scales into the same subsets;
- Performing inverse transformations and/or re-coding process to transfer the imputed variables to the original variables with original scales (may use function “uncategorize” in the software);
- Combining the output (imputed) subsets into one overall complete file.

Working through this list of pre- and post-imputation data file management may take much more time than the actual imputation processing.

(2) How well documented is the software? Is it difficult to use?

How to install and use the software will be described before answering these questions. Installation instructions provided by Schafer’s software are for a UNIX workstation and some are not applicable to a PC system. Before performing the following installation steps for a PC environment, a bunch of storage mode errors (over 60 places) in the FORTRAN subroutines have to be corrected in order for them to be successfully compiled.

Installation:

(i) Use WATCOM FORTRAN 77 compiler (WATCOM International Corporation, 1993) to compile the FORTRAN source files “norm.for”, “cat.for” and “mix.for” to create object files “norm.obj”, “cat.obj” and “mix.obj”, respectively;

(ii) Copy files "norm.obj", "norm.s", “cat.obj”, “cat.s”, “mix.obj” and “mix.s” to the S-PLUS working directory "c:\spluswin\home" (if S-plus for Windows is installed on the C drive);

(iii) Create a subdirectory "_help" of "c:\spluswin\home_data" and copy all the help files into this subdirectory "c:\spluswin\home_data_help";

(iv) In an S session, define the function **.First** as follows:

```
.First_function() {
        dyn.load(“norm.obj”)      # load file “norm.obj”
        source(“norm.s”)          # load file “norm.s”
        dyn.load(“cat.obj”)       # load file “cat.obj”
        source(“cat.s”)           # load file “cat.s”
```

```
        dyn.load("mix.obj")        # load file "mix.obj"
        source("mix.s")            # load file "mix.s"
        rngseed(6534288)           # initialize the random generator seed
}
```

(v) Quit S session.

Then the next time we enter an S session, Schafer's software will automatically be loaded into it. Since the three parts of the software work independently, a user can load only part of the software through the **.First** function if the other part is not needed for his/her purpose.

Using the program:

After correctly installing the program, the 40 S-functions in the software (12 for continuous variables, 19 for categorical variables and 9 for mixed variables) can be called just like any other S-PLUS functions. See *Appendix 1—Appendix 3* for brief descriptions of these functions. More complete descriptions of these functions are found in the help files by typing "help(filename)" from within S session.

So the software is easy to use if the user is familiar with S language.

To recapitulate, the question was "How well documented is the software?" By and large, the software is well documented, and the algorithm for the software is especially well developed. However, as the author said, the software is at its early stage and improvement will be made to its future version. The most serious problem is with the constrained loglinear model and the restricted general location model. The programs for these models have variable ordering problems, work very slowly, and require a huge vector to store the parameters, which limits their capacity of dealing with large number of categorical variables to the level of their corresponding saturated models. Actually, no model in this software can deal with more than 8 categorical variables of the SASS.AS data set. We believe that, if the parameters are stored in several vectors instead of one vector "theta", those constrained models may be able to deal with more categorical variables. Moreover, as already mentioned, no simulation or imputation function work due to its random generator problems. Some storage mode errors are also needed to be corrected. Figuring out these program problems took much more time than the evaluation process.

As a close note for this section, we would like to quote some idiosyncrasies from Dr. Schafer as a caution, although we did not experience these problems:

(i) These S-plus functions do not supply many error messages, so if something does not seem to work, it's largely up to the user to figure out why.

(ii) If the EM algorithm bombs, it could be that the ML estimate is on the boundary of the parameter space. Similarly, if Data Augmentation under the default non-informative prior does not seem to work, it could be that the posterior distribution does not exit. These problems may arise with data sets that are sparse; that is, having lots of missing values, or having a number of observations not substantially larger than the number of variables. The remedy for sparse data is to supply a proper prior distribution, or simplify the model by eliminating variables that are not crucial to the analysis.

(3) Can the software be adapted to interface easily with SAS and SPSS?

The immediate answer to this question is "no." Schafer's software is written in S language and run under an S-PLUS environment, and S-PLUS and SAS (SPSS) can not interface with each other. However, with the help of software DBMS/COPY (Conceptual Software, Inc. 1994), it is very easy to make transformations between SAS (SPSS) data files and S-PLUS data matrices so that an analyst can use Schafer's software to impute the data under an S-PLUS environment and analyze the imputed data under a SAS or SPSS environment. We may use S-PLUS "File" pull-down menu to import and/or export SAS (SPSS) data files from within S sessions. In order to use this S-PLUS "File" pull-down menu, we need to add a statement "DBMSCOPY=C:\DBMSCOPY" to the S-PLUS initial file "SPLUS.INI" if the software DBMS/COPY is installed on the C directory.

After we have correctly installed DBMS/COPY software and modified the S-PLUS initial file as described above, we can use the following four simple steps to import a SAS or SPSS incomplete data file for Schafer's software to impute, and then output a complete SAS or SPSS data file for SAS or SPSS to analyze:

(i) Click the "File" menu and select "import" from the S-PLUS tool bar to transfer the incomplete SAS or SPSS data file to an S-PLUS data frame, say, X;
(ii) Change the storage mode of X to "single"[1] with statements:
X_as.matrix(X); storage.mode(X)_ "single";
(iii) Apply Schafer's software to the data matrix X to impute the missing values;
(iv) Click the S-PLUS "File" menu again and select "export" to transfer the imputed S-PLUS data matrix to a SAS or SPSS data file.

[1]When a SAS or SPSS data file is transferred to an S-PLUS data frame through DBMS/COPY, the output data frame has a "list mode", original S-PLUS functions can be applied to this type of data frame, while Schafer's software can not because it requires a "single mode" of a data matrix.

In the case that DBMS/COPY is not available to the user, the following three steps can transfer a SAS file to a S-PLUS data matrix (we may use similar steps for SPSS data files):

(i) In a SAS environment, use a number to represent missing value '.', for example, '-1', providing that there is no other value equal to -1.

(ii) Transfer the SAS file to an ASCII file (may use PUT statement in SAS), and send the ASCII file to "c:\spluswin\home" (may use FILE statement in SAS).

(iii) In an S session, use function "scan" to read the data into an S object. Then let "NA" stand for missing values and make a data matrix.

After all missing values have been imputed by the software, the complete data will be stored in a matrix. The following statement can be used to transfer the imputed data matrix into an ASCII file which can be read directly by SAS:

write(t(x), "filename", ncol=ncol(x))

where x is the imputed data matrix, t(x) is the transpose of x, and "filename" is the name of the target ASCII file which will be located in the S-plus working directory "c:\spluswin\home".

C. REFERENCES

Anderson, T. W (1984), *An Introduction to Multivariate Statistical Analysis*. New York: John Wiley & Sons.

Bishop, Y. M. M., Fienberg, S. E. and Holland, P. W. (1975), *Discrete Multivariate Analysis: Theory and Practice*, Cambridge, MA: MIT Press.

Dempster, A.P., Laird, N. M., and Rubin, D. B. (1977), *Maximum Likelihood Estimation from Incomplete Data via the EM Algorithm (with discussion).* Journal of the Royal Statistical Society Series B, **39**, 1-38.

Gelman, A., Meng, X. L. and Rubin, D. B. (1991), *Simulating the Posterior Distribution of Loglinear Contingency Table Models*. Technical Report, Department of Statistics, University of California, Berkeley, CA.

Little, R.J.A. and Rubin, D.B. (1987), *Statistical Analysis with Missing Data.* New York: John Wiley & Sons.

Meng, X. L. and Rubin, D. B. (1991), *IPF for Contingency Tables with Missing Data via the ECM Algorithm*. Proceedings of the Statistical Computing Section, American Statistical Association, 244-247.

Olkin, I. and Tate, R. F. (1961), *Multivariate correlation models with mixed discrete and continuous variables*. Annals of Mathematical Statistics, **32**, 448-465.

Rubin, Donald B. (1987), *Multiple Imputation for Nonresponse in Surveys.* New York: J. Wiley & Sons, 76-77.

Schafer, J. L. (1991), *Algorithms for Multiple Imputation and Posterior Simulation from Incomplete Multivariate Data with Ignorable Nonresponse*. Unpublished Ph.D. dissertation.

Schafer, J.L (1995), *Analysis of Incomplete Multivariate Data*. To be published by Chapman and Hall.

This page intentionally left blank.

IV. A Comparison of *Proc Impute* and *Schafer's Software*

This section consists of three subsections. Subsection A describes a small simulation study which is designed to compare the two imputation software packages in terms of generating more accurate imputations for continuous data. Subsection B discusses some statistical arguments on theoretical comparisons of the algorithms used in the software; Subsection C presents the imputations generated by NCES, *Proc Impute* and *Schafer's Software* for a few selected variables from 1990-91 School and Staffing Survey Administrator Component.

A. A SIMULATION STUDY FOR CONTINUOUS VARIABLES

We simulate three types of data sets: independent normal variables, correlated normal variables and contaminated independent variables. Each data set has 8 variables and 2000 cases. The first 7 variables have about 10% missing values apiece and the 8th variable has no missing value. Three types of missing mechanisms are considered: (1) X is randomly missing; (2) X is missing when Z<c (a constant) and corr(X, Z)=0.6; and (3) X is missing when Z<c and corr(X, Z)=0.9. Table 1-Table 3 compare *Proc Impute* and *Schafer's Software* in terms of average imputing error and mean bias for independent normal data, correlated normal data and contaminated data, respectively. Here the average imputing error is defined as

$$\sqrt{\frac{1}{m}\sum_{i=1}^{m}(I_i \& R_i)^2},$$

where I_i and R_i are the imputed value and the real value, respectively, for the ith missing case, and m is the number of missing values.

Table 1 shows that, when the variables (X_1-X_8) in the model are normal and independent, the performance of *Proc Impute* and that of *Schafer's Software* are very close in terms of average imputing error and mean bias. It also shows that neither of imputation methods improves any mean bias comparing to the un-imputed sample mean. This is supposed to be because the variables in the model provide no information for each other for predicting the missing values if they are independent. In this kind of situation, imputation methods may be used for the purpose of analytical convenience when some statistical approaches can only be applied to complete data sets, or the statistical approaches become too complicated for incomplete data sets. Of course, this kind of situation is rare in a real survey. Variables are correlated with each other more or less.

When the variables in the model are normal and correlated (with correlation between 0.3-0.9), table 2 demonstrates that *Schafer's Software* always has better performance than *Proc Impute*, whether the

Table 1 *A Comparison of Proc Impute and Schafer's Software for Independent Normal Variables through a Simulation with 2000 cases (std=1)*

Variables:	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8
true mean (μ):	-3	-2	-1	0	0.5	1	2	3
(1) Missing at random								
# of missing values:	191	187	193	219	194	222	198	0
average imputing error:								
Schafer:	1.414	1.397	1.418	1.417	1.394	1.387	1.422	0
Proc Impute:	1.414	1.437	1.419	1.383	1.414	1.472	1.324	0
$\mu_I - \mu^*$:								
Schafer:	-.007	0.028	0.002	0.009	-.002	-.003	-.008	0.046
Proc Impute:	-.012	0.029	-.002	- .009	0.005	-.001	0.004	0.046
$\mu_M - \mu^*$:	-.012	0.033	0.001	0.001	0.006	-.004	-.004	0.046
(2) X_i is missing if $Z_i<c_i$ and corr(X_i, Z_i)=0.6								
# of missing values:	202	198	199	202	208	198	195	0
average imputing error:								
Schafer:	1.780	1.685	1.660	1.817	1.641	1.804	1.780	0
Proc Impute:	1.676	1.613	1.790	1.782	1.680	1.731	1.628	0
$\mu_I - \mu$:								
Schafer:	0.107	0.133	0.110	0.130	0.117	0.133	0.120	0.046
Proc Impute:	0.105	0.133	0.119	0.121	0.118	0.130	0.099	0.046
$\mu_M - \mu$:	0.107	0.135	0.118	0.120	0.116	0.138	0.103	0.046
(3) X_i is missing if $Z_i<c_i$ and corr(X_i, Z_i)=0.9								
# of missing values:	204	199	186	207	200	200	175	0
average imputing error:								
Schafer:	2.023	2.102	2.001	2.115	2.072	2.108	1.986	0
Proc Impute:	2.100	2.011	2.077	2.020	2.012	2.131	2.073	0
$\mu_I - \mu$:								
Schafer:	0.169	0.203	0.163	0.191	0.175	0.194	0.148	0.046
Proc Impute:	0.176	0.197	0.169	0.179	0.180	0.197	0.151	0.046
$\mu_M - \mu$:	0.169	0.199	0.167	0.186	0.173	0.192	0.148	0.046

Note: * μ_I is the imputed sample mean; μ_M is the sample mean without any imputation; and μ is the true mean.

Table 2 *A Comparison of Proc Impute and Schafer's Software for Correlated Normal Variables through a Simulation with 2000 cases (std=1 and corr(X_i, X_j)=1-0.1*|i-j|)*

Variables:	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8
true mean (μ):	-3	-2	-1	0	0.5	1	2	3
(1) Missing at random								
# of missing values:	199	185	197	193	191	190	190	0
average imputing error:								
Schafer:	0.635	0.426	0.464	0.456	0.443	0.450	0.467	0
Proc Impute:	0.760	0.537	0.449	0.487	0.599	0.654	0.657	0
$\mu_I - \mu^*$:								
Schafer:	0.000	-.003	0.002	-.006	-.004	0.002	0.002	0.000
Proc Impute:	0.005	-.001	0.002	-.001	0.002	0.003	0.009	0.000
$\mu_M - \mu^*$:	0.002	0.003	-.012	0.000	0.001	0.007	0.004	0.000
(2) X_i is missing if $Z_i<c_i$ and corr(X_i, Z_i)=0.6								
# of missing values:	196	192	186	196	208	201	202	0
average imputing error:								
Schafer:	0.915	0.673	0.544	0.513	0.491	0.510	0.488	0
Proc Impute:	0.947	0.784	0.804	0.836	0.617	0.740	0.542	0
$\mu_I - \mu$:								
Schafer:	0.039	0.023	0.022	0.014	0.015	0.020	0.017	0.000
Proc Impute:	0.043	0.024	0.029	0.030	0.017	0.029	0.016	0.000
$\mu_M - \mu$:	0.117	0.113	0.110	0.113	0.109	0.122	0.124	0.000
(3) X_i is missing if $Z_i<c_i$ and corr(X_i, Z_i)=0.9								
# of missing values:	210	181	190	198	195	199	207	0
average imputing error:								
Schafer:	1.143	1.013	0.892	0.804	0.730	0.688	0.566	0
Proc Impute:	1.453	1.354	1.102	0.896	0.938	0.835	0.627	0
$\mu_I - \mu$:								
Schafer:	0.079	0.059	0.059	0.046	0.043	0.036	0.035	0.000
Proc Impute:	0.114	0.084	0.074	0.051	0.060	0.048	0.043	0.000
$\mu_M - \mu$:	0.183	0.158	0.164	0.168	0.169	0.175	0.181	0.000

Note: * μ_I is the imputed sample mean; μ_M is the sample mean without any imputation; and μ is the true mean.

Table 3 *A Comparison of Proc Impute and Schafer's Software for Contaminated Independent Variables through a Simulation with 2000 cases (*90% normal and 10% Cauchy)

Variables:	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8
complete sample mean (μ_C):	-2.979	-2.044	-1.025	0.162	0.438	0.926	2.050	2.969
complete sample std (s_C):	2.042	3.183	1.535	6.288	8.783	8.350	4.459	2.295
(1) Missing at random								
# of missing values:	208	203	202	188	197	206	193	0
average imputing error (in s_C):								
Schafer:	1.169	1.014	1.163	1.066	1.162	1.148	1.047	0
Proc Impute:	1.455	0.493	1.326	2.282	1.343	0.187	0.979	0
μ_I-μ_C*:								
Schafer:	0.009	-.012	-.001	-.069	0.103	-.018	0.052	0
Proc Impute:	0.008	-.018	-.019	-.071	0.212	0.008	0.061	0
μ_M -μ_C* :	-.008	-.001	0.004	0.007	-.010	-.007	0.007	0
(2) X_i is missing if $Z_i<c_i$ and corr(X_i, Z_i)=0.6								
# of missing value:	200	203	198	209	225	220	200	0
average imputing error (in s_C):								
Schafer:	1.858	2.091	2.006	1.962	2.884	2.453	2.253	0
Proc Impute:	1.618	2.616	1.990	1.905	2.898	2.591	2.138	0
μ_I-μ_C:								
Schafer:	0.135	0.180	0.153	0.253	0.241	0.321	0.248	0
Proc Impute :	0.127	0.199	0.133	0.091	0.285	0.377	0.231	0
μ_M -μ_C:	0.126	0.167	0.150	0.158	0.260	0.322	0.238	0
(3) X_i is missing if $Z_i<c_i$ and corr(X_i, Z_i)=0.9								
# of missing values:	200	192	209	202	202	206	196	0
average imputing error:								
Schafer:	2.589	2.316	2.221	1.892	3.025	2.656	2.199	0
Proc Impute:	1.770	2.189	2.329	2.585	3.062	3.188	2.226	0
μ_I-μ_C:								
Schafer:	0.237	0.284	0.239	0.210	0.293	0.367	0.323	0
Proc Impute:	0.195	0.249	0.254	0.220	0.267	0.388	0.284	0
μ_M -μ_C:	0.200	0.242	0.238	0.199	0.295	0.362	0.299	0

Note: * μ_I is the imputed sample mean; μ_M is the sample mean without any imputation; and μ_M is the complete sample mean.

assumption of missing at random (MAR), which is required by *Schafer's Software*, holds or not. This may be because the more important assumption, normal distribution of X, is satisfied for *Schafer's Software* in this case. When missing values occur at random, neither of the two imputation methods makes any improvement on the population mean estimates. However, when the missing values occur with some patterns, both methods dramatically improved the population mean biases. The stronger the correlation between the "target" variable and the predictors is, the more the improvement will be.

For the data set with 90% normal data and 10% Cauchy data, table 3 illustrates that *Proc Impute* is better than *Schafer's Software* in some cases, while it is the other way around in other cases. And the performance of both methods are very unstable. With all three types of missing mechanisms, both imputed sample means perform poorly because the correlation between variables are very small so that little information can be borrowed from other variables to impute the missing values for the "target" variables. In the case of missing at random, missing values cause no bias and the un-imputed sample mean is much better than the imputed means, while the serious bias makes the un-imputed sample mean evenly bad as the imputed mean when the missing values occur with some trend.

Our findings of this small simulation study can be summarized as follows: (1) for independent data, the biases caused by the missing values can not be corrected through any imputation approach; (2) for normal correlated data, *Schafer's Software* always perform better than *Proc Impute* no matter what missingness mechanism is; (3) both imputation approaches can improve the estimator for the population mean dramatically if the missing values occurred with some strong pattern and the variables have moderately high correlations; (4) both imputation approaches are not so robust for the contaminated data.

More simulation studies are definitely needed to compare the two imputation approaches. For example, it is necessary to compare them for all types of variables(continuous, ordinal, categorical, mixed), and compare them with more criteria, such as coverage probabilities, variance estimates, etc; it is also worthwhile to investigate their performance on multiple imputation inference (see section V for details).

B. THEORETICAL ARGUMENTS

Proc Impute is based on an assumption that the relations between variables keep the same for the observed cases and missing cases, and tries to predict the missing values of the "target" variables through a probabilistic relationship between the "target" variables and predicting variables. This assumption is more reasonable in practice than the assumption of missing at random (MAR), which is required by *Schafer's Software*. In other words, *Schafer's Software* assume that the observed values have the same distribution as the missing values. In practice, it is often the case that the distribution of the missing cases has a different pattern from that of observed cases, but the relations among the variables are usually similar for both observed part and missing part. Actually, MAR is a very strong assumption. If data are missing at random, then imputation is useless for improving the statistical results, and we may ignore the missing cases and base our inference on the observed cases only. However, our

simulation study fortunately shows that the assumption of MAR is not so important to *Schafer's Software*. As long as the normal distribution assumption holds, Schafer's Software always do better than Proc Impute no matter what missingness mechanism is. Then the consequent question is, "Can we just use Schafer's Software and ignore this assumption?" We are not in a hurry to answer this question. More simulation studies should be conducted to answer this question.

Schafer's Software assumes normality for continuous variables, while *Proc Impute* assumes that the conditional distribution of the "target" variables given the predictors are normal, or equivalently, the residuals from the regression models are normal. It comes to our attention that *Schafer's Software* is more sensitive to its distributional assumption than *Proc Impute*. The reason is that *Schafer's Software* uses a Bayesian method to generate the imputations directly from the assumed distribution, while the distributional assumption in *Proc Impute* is used only in the middle step to fit the regression models and the imputations are not generated directly from the regression equations. Therefore we should be more careful about the distribution of the data when we use *Schafer's Software* to do the imputations. As we mentioned before, theoretically, any continuous variable can be transferred into a variable with a normal distribution. In practice, we may use common transformations (e.g., logarithm, exponential, square root, square, etc) to make the variables as close to normal variables as possible, and then apply *Schafer's Software* to impute missing values for the transferred normal variable and then transfer back to the original variables. As we see in our simulation, *Schafer's Software* has much better performance if the data are normal. So we recommend *Schafer's Software* for the continuous data if they are approximately normal, or a simple transformation is available to make them approximately normal. Otherwise, we may try *Proc Impute*. It is possible that *Proc Impute* will show better performance than *Schafer's Software* if we use a data set which has further departure from normality, or in the cases that the conditional distributions of the variables (or residuals) are normal but the variables themselves are not.

For categorical variables, *Schafer's Software* uses saturated multinomial models and constrained loglinear models to impute missing values, while *Proc Impute* fits usual regression models by treating all types of variables as continuous variables. The models used in *Schafer's Software* are believed to be more appropriate than those used in *Proc Impute* for categorical variables.

Another advantage of *Schafer's Software* is about multiple imputation. *Proc Impute* is basically a single-imputation algorithm. Although the newest version of *Proc Impute* allows the users to generate multiple imputations, but the number of imputations for each case is very limited and it may not be "proper" in the sense of Rubin's definition. On the other hand, *Schafer's Software* is originally created for multiple-imputation purpose, it adheres to Rubin's "proper" criterion if the sample is a simple random sample.

However, *Schafer's Multiple Imputation Software* draws imputation values for the missing cases from a posterior distribution. The generated imputation values for continuous variables may be out of the range of the observed values (or, say, the domain of the variables), while *Proc Impute* has made an

effort to avoid this kind of problem. Another advantage of Proc Impute is its convenience to perform imputations for large data sets. *Proc Impute* can carry out all imputations as one single batch job no matter how large the data set is. But *Schafer's Software* usually requires more than one S session and much more runs to carry out all the imputations for a large data set due to the limitation of dynamic memory of S-PLUS for Windows. An analyst may need to divide a large data set into more subsets to apply *Schafer's Software*, and, consequently, need to spend much more time on pre- and post-imputation data file management.

In summary, *Schafer's Software* may generate more accurate imputations if its distributional assumptions are approximately satisfied, but *Proc Impute* is much more convenient to use for a large data set.

C. AN APPLICATION TO NCES DATA

Appendix 4 shows the un-imputed data and the data imputed by NCES, by *Schafer's Software* and by *Proc Impute* for three variables: *Year received bachelor's degree (BS/BA)*, *Years of elementary/secondary teaching experience before becoming principal (Tch)*, and *Year of birth (Birth)*, from 1990-91 SASS Administrator data file. It is noticed that all the means and standard deviations for the un-imputed data, NCES imputed data, two sets of imputed data by *Schafer's Software*, and *Proc Impute* imputed data, are very close, although the imputed values for each individual missing case are quite different. One may argues that this is because the numbers of missing cases, 142 for **BS/BA**, 82 for **Tch**, and 114 for **Birth**, are not big enough comparing to the sample size 9054. In fact, we also used these methods to impute another variable with 898 missing cases; the means and standard deviations for that variable are still very close. The distributions of **BS/BA**,**Tch** and **Birth** are close to normal, so the imputations by *Schafer's Software* may be good although it generates a few negative values for **Tch** which are not acceptable for the variable. All imputation values generated by *Schafer's Software* are rounded up to integers because these variables are integer-valued, while *Proc Impute* automatically generates the same type of values for the missing cases as observed values.

This page intentionally left blank.

V. Next Steps

As we mentioned earlier, *Schafer's Software* will generate more accurate imputations than *Proc Impute* if its distributional assumptions are approximately satisfied, and it also has the advantage over *Proc Impute* for generating "proper" multiple imputations if the sample is a simple random sample. But the programs for the two constrained models in *Schafer's Software* has a variable ordering problem. If the u-terms (or interactions) in the loglinear constrains (A.2.4) of section II are not symmetrically designed, the results will not be right. Furthermore, each of the two constrained models is supposed to fit more categorical variables than its saturated model, and allow any order interactions in the models. However, we can usually put no more than 8 categorical variables and no higher than second order interactions. Otherwise, either the vector "theta", which stores all parameters of the model, will exceed the object size limitation, or the run will take un-affordable time. Dr. Schafer is modifying the software to fix these problems (at least the variable ordering problem). We think it is worthwhile for NCES to obtain the new version of *Schafer's Software* and implement it on a PC environment. Actually, we know the whole software very well and it is not so difficult for us to fix these problems for NCES if Dr. Schafer can not complete the modification in timely manner.

We did a small simulation for continuous variables. A thorough simulation study will be necessary to investigate the performance of both imputation software packages in terms of generating more accurate imputations. The following questions will be answered by the simulation. Will *Schafer's Software* still perform better than *Proc Impute* in terms of other popular, important, and more complicated criteria such as coverage probabilities, variance estimates, etc.? Can *Schafer's Software* also generate more accurate imputations for categorical variables, ordinal variables? Is the assumption of MAR really not important to all five types of models in *Schafer's Software*? Can *Schafer's Software* really generate more "proper" multiple imputations than *Proc Impute*?

Our simulation study shows that both *Proc Impute* and *Schafer's Software* are very unstable for contaminated data with outliers. It will be valuable for NCES to explore some more robust imputation algorithms or implement some robust statistical approaches for imputation purposes and develop new imputation software. It will be a huge project. If NCES is interested, we will propose it in detail.

This page intentionally left blank.

Appendix 1: Twelve S-PLUS Functions for Continuous Variables in Schafer's Multiple Imputation Software

1. prelim.norm(x)

Performs preliminary manipulations for x—a matrix of incomplete continuous data. The data are centered, scaled, and sorted by missingness patterns. It returns a list of objects that summarizes various features of the incomplete data matrix.

2. em.norm(s, start, showits=T, maxits=1000, criterion=0.0001, prior)

Performs maximum-likelihood estimation using the EM algorithm, or finds a posterior mode under a normal-inverted Wishart prior supplied by the user. It returns a vector of parameters representing the MLE. The parameter vector stores the means and variance-covariance matrix in this order: -1, means, up-triangular rows of variance-covariance matrix. It is called *packed storage* by the author.

Brief descriptions of the arguments passed in "em.norm" follow:

s: a summary list produced by "**prelim.norm**"
start: optional starting value of the parameter
showits: if "TRUE," reports the iterations of EM
maxits: maximum number of iterations performed
criterion: convergence criterion
prior: optional prior distribution. This is a list of the hyperparameters of a normal-inverted Wishart distribution. In order, the elements of the list are:

' (a scalar),
m (a scalar),
μ_0 (a vector of length p),
Ψ^{-1} (a p×p matrix),

where p is the number of variables. If no prior is supplied, the default is usual noninformative prior for multivariate normal model:

' =0
m=-1
μ_0=arbitrary
Ψ^{-1}=a matrix of zeros

3. **getparam.norm(s, theta, corr=F)**

Returns mean vector and variance-covariance matrix on their original scale and original form from theta which is on packed storage.

Brief descriptions of the arguments passed in “getparam.norm” follow:

s: a summary list produced by “prelim.norm”

theta: parameter estimators in packed storage

corr: if "TRUE", returns standard deviations and correlation matrix; if "FALSE", variance-covariance matrix

4. **makeparam.norm(s, thetalist)**

Does the opposite of "getparam.norm"—makes a parameter list in packed storage

Brief descriptions of the arguments passed in “makeparam.norm” follow:

s: a summary list produced by “prelim.norm”

thetalist: as the results produced by "getparam.norm"

5. **da.norm(s, start, prior, steps=1, showits=F, return.ymis=F)**

Simulates one or more iterations of a single Markov chain. Each iteration consists of a random imputation of the missing data given the observed data and the current parameter value (I-step), followed by a draw from the posterior distribution of the parameter given the observed data and the imputed data (P-step). Returns a value of the parameter, the result of the final P-step.

Brief descriptions of the arguments passed in “da.norm” follow:

s: a summary list produced by “prelim.norm”

start: starting value of the parameter. One obvious choice is the estimator generated by "em.norm"

prior: optional prior distribution—a list of the hyperparameters of a normal-inverted Wishart distribution as in “em.norm”.

steps: number of data augmentation iterations

showits: if "TRUE", reports the iterations

return.ymis: if "TRUE", returns the output of the last I-step as the imputed values of missing data in addition to the output of the last P-step.

6. mda.norm:

Monotone data augmentation which is supposed to converge more quickly than "da.norm" for nearly monotone missingness data sets. Unfortunately this function does not work because of some bad Fortran subroutines.

7. imp.norm(s, theta, x)

Draws missing elements of a data matrix under the multivariate normal model and a user-supplied parameter. Returns a matrix of complete data.

Brief descriptions of the arguments passed in “imp.norm” follow:

s: a summary list produced by “prelim.norm”

theta: a parameter vector in packed storage, such as one created by "em.norm" or "da.norm".

x: the original data matrix

8. rngseed(seed)

Initializes the random number generator seed. If this function has not been called in .First function, it must be called at least once before the simulation functions (e.g., da.norm and imp.norm) can be used. The argument “Seed” is preferred to be a large integer (must be positive integer).

9. loglik.norm(s, theta)

Evaluates the observed-data loglikelihood function at a user-supplied value of the parameter. This function is useful for monitoring the progress of EM and data augmentation.

The arguments passed in “loglik.norm” have the same meaning as in “da.norm”.

10. logpost.norm(s, theta, prior)

Similar to loglik.norm, except it evaluates the log of observed-data posterior density under a normal-inverted Wishart prior (as in da.norm). If no prior is specified by the user, the usual "noninformative" prior for the multivariate normal distribution is used.

The arguments passed in “logpost.norm” have the same meaning as in “da.norm”.

11. ninvwish(s, params)

Simulates a value from a normal-inverted Wishart distribution. This function may be useful for obtaining values of the parameters of a multivariate normal distribution for multiple chains of data augmentation.

Brief descriptions of the arguments passed in "ninvwish" follow:

s: a summary list produced by "prelim.norm"

params: a list of parameters of a normal-inverted Wishart distribution as in da.norm. When using this function to create starting values for data augmentation, μ_0 and ? $^{-1}$ should be chosen in relation to the data matrix after the columns have been centered and scaled to have mean zero and variance one.

12. mi.inference(est, std.err, confidence=0.95)

Combines estimates and standard errors from m complete-data analyses performed on m imputed data sets to produce a single inference. Uses the technique described in Rubin (1987) for multiple imputation inference for a scalar estimand.

Brief descriptions of the arguments passed in "mi.inference" follow:

est: a list of m (at least 2) vectors representing estimates from complete-data analyses performed on m imputed data sets

std.err: a list of m vectors containing standard errors from the complete-data analyses corresponding to the estimates in 'est'

confidence: desired coverage of interval estimates

This function returns a list with the following components:

est: the average of the complete-data estimates

std.err: standard errors incorporating both the between and the within imputation uncertainty

df: degrees of freedom associated with the t reference distribution used for interval estimates

signif: p-values for the two-tailed hypothesis tests that the estimated quantities are equal to zero

lower: lower limits of the confidence interval

upper: upper limits of the confidence interval

r: estimated relative increases in variance due to nonresponse

fminf: estimated fractions of missing information

Appendix II: Eighteen S-PLUS Functions for Categorical Variables in Schafer's Multiple Imputation Software

1. categorize(x)

Recode the levels of categorical variables as consecutive positive integers starting with 1.

The only argument, x, in this function is a matrix of discrete data taking integer values. This function will return a list with the following two components:

x: a matrix corresponding to x, but whose columns have been recoded as consecutive positive integers 1,2,...
levs: a list of length `ncol(x)' whose elements are character vectors giving the original levels of the columns of `x'.

2. uncategorize(x, levs)

Does the opposite of "categorize"—change the categorical variables to their original levels after imputation.

The two required arguments are:

x: a matrix whose columns are categorical data taking values 1,2,...
levs: a list of character vectors of length `ncol(x)' giving the integer codes corresponding to the levels of the columns of `x'.

This function returns a matrix like `x', except recoded to correspond to `levs'.

3. prelim.cat(x, counts, levs)

Performs grouping and sorting operations on categorical data sets with missing values. It creates a list that is needed for input to "em.cat", "da.cat", "imp.cat", etc.

The three arguments are:

x: categorical data matrix containing missing values. The data may be provided either in ungrouped or grouped format. In ungrouped format, the rows of x correspond to individual observational units, so that nrow(x) is the total sample size. In grouped format, the rows of x correspond to distinct covariate patterns; the frequencies are provided through the "counts" argument. In either format, the columns correspond to

variables. The categories must be coded as consecutive positive integers beginning with 1 (1,2,...), and missing values are denoted by NA.

counts: optional vector of length `nrow(x)' giving the frequencies corresponding to the covariate patterns in x. The total sample size is `sum(counts)'. If `counts' is missing, the data are assumed to be ungrouped; this is equivalent to taking `counts' equal to `rep(1,nrow(x))'.

levs: optional vector of length `ncol(x)' indicating the number of levels for each categorical variable. If missing, `levs[j]' is taken to be `max(x[,j],na.rm=T)'.

This function returns a list with 17 components. The key components are:

nmis: a vector of length `ncol(x)' containing the number of missing values for each variable in x.

r: matrix of response indicators showing the missing data patterns in x. Dimension is (m,p) where m is number of distinct missingness patterns in the rows of x, and p is the number of columns in x. Observed values are indicated by 1 and missing values by 0. The row names give the number of observations in each pattern, and the columns correspond to the columns of x.

d: vector of length `ncol(x)' indicating the number of levels for each variable. The complete-data contingency table would be an array with these dimensions. Identical to `levs' if `levs' was supplied.

ncells: number of cells in the cross-classified contingency table, equal to `prod(d)'.

4. em.cat(s, start, prior=1, showits=T, maxits=1000, eps=0.0001)

Finds ML estimate or posterior mode of cell probabilities under the saturated multinomial model. If zero cell counts occur in the observed-data table, the maximum likelihood estimate may not be unique, and the algorithm may converge to different stationary values depending on the starting value. Also, if zero cell counts occur in the observed-data table, the ML estimate may lie on the boundary of the parameter space. Supplying a prior with hyper-parameters greater than one will give a unique posterior mode in the interior of the parameter space. Estimated probabilities for structural zero cells will always be zero.

The arguments are:

s: summary list of an incomplete categorical data set produced by the function `prelim.cat'.

start: optional starting value of the parameter. This is an array with dimensions `s$d' whose elements sum to one. The default starting value is a uniform array (equal probabilities in all cells). If structural zeros appear in the table, `start' should contain zeros in those positions and nonzero (e.g., uniform) values elsewhere.

prior: optional vector of hyperparameters for a Dirichlet prior distribution. The default is a uniform prior distribution (all hyperparameters = 1) on the cell probabilities, which will result in maximum likelihood estimation. If structural zeros appear in the table, a prior should be supplied with `NA's in those cells.

showits: if `TRUE', reports the iterations of EM so the user can monitor the progress of the algorithm.

maxits: maximum number of iterations performed. The algorithm will stop if the parameter still has not converged after this many iterations.

eps: convergence criterion. This is the largest proportional change in an expected cell count from one iteration to the next. Any expected cell count that drops below 1E-07 times the average cell probability (1/number of non-structural zero cells) is set to zero during the iterations.

This function returns a array of dimension `s$d' containing the ML estimate or posterior mode, assuming that EM has converged by `maxits' iterations.

5. ecm.cat(s, margins, start, prior=1, showits=T, maxits=1000, eps=0.0001)

Uses ECM algorithm to find ML estimate or posterior mode of cell probabilities under a constrained loglinear model for incomplete categorical data. This is an iterative algorithm. At each iteration, performs an E-step followed by a single cycle of iterative proportional fitting. If zero cell counts occur in the observed-data tables, the maximum likelihood estimate may not be unique, and the algorithm may converge to different stationary values depending on the starting value. Also, if zero cell counts occur in the observed-data tables, the ML estimate may lie on the boundary of the parameter space. Supplying a prior with hyperparameters greater than one will give a unique posterior mode in the interior of the parameter space. Estimated probabilities for structural zero cells will always be zero.

The arguments are:

s: summary list of an incomplete categorical data matrix `x' produced by the function `prelim.cat'.

margins: optional vector describing the sufficient configurations or margins in the desired loglinear model. A margin is described by the factors not summed over, and margins are separated by zeros. Thus c(1,2,0,2,3,0,1,3) would indicate the (1,2), (2,3), and (1,3) margins in a three-way table; that is, the model of no three-way association. The integers 1,2,... in the specified margins correspond to the columns of the original data matrix `x'. If no margins are given, `ecm.cat' performs EM for the saturated model with no loglinear constraints, and the results will agree with those of ‘em.cat’.

start: optional starting value of the parameter. This is an array with dimensions `s$d' whose elements sum to one. The default starting value is a uniform array (equal probabilities in all cells). If structural zeros appear in the table, `start' should contain zeros in those positions and nonzero (e.g., uniform) values elsewhere.

showits: if `TRUE', reports the iterations of ECM so the user can monitor the progress of the algorithm.

maxits: maximum number of iterations performed. The algorithm will stop if the parameter still has not converged after this many iterations.

eps: convergence criterion. This is the largest proportional change in an expected cell count from one iteration to the next. Any expected cell count that drops below 1E-07 times the average cell probability(1/number of non-structural zero cells) is set to zero during the iterations.

prior: optional vector of hyperparameters for a Dirichlet prior distribution. The default is a uniform prior distribution (all hyperparameters = 1)on the cell probabilities, which will result in maximum likelihood estimation. If structural zeros appear in the table, a prior should be supplied with `NA's in those cells.

This function returns an array of dimension `s$d' containing the ML estimate or posterior mode, assuming that ECM has converged by `maxits' iterations.

6. da.cat(s, start, prior=0.5, steps=1, showits=F)

Uses Markov-Chain Monte Carlo method to simulate draws from the observed-data posterior distribution of underlying cell probabilities under a saturated multinomial model. At each iterations, the missing data are randomly imputed under their predictive distribution given the observed data and the current value of `theta' (I-step), and then a new value of `theta' is drawn from its Dirichlet posterior distribution given the complete data (P-step). After a suitable number of steps are taken, the resulting value of the parameter may be regarded as a random draw from its observed-data posterior distribution.

This function is used in conjunction with `imp.cat' to create proper multiple imputations. It is very IMPORTANT that the random number generator seed must be set at least once by the function `rngseed' before this function can be used.

The arguments are:

s: summary list of an incomplete categorical data set created by the function `prelim.cat'.

start: starting value of the parameter. This is an array of cell probabilities of dimension `s$d', such as one created by `em.cat'. If structural zeros appear in the table, starting values for those cells should be zero.

prior: optional vector of hyperparameters specifying a Dirichlet prior distribution. The default is the Jeffreys prior (all hyperparameters =.5). If structural zeros appear in the table, a prior should be supplied with hyperparameters set to `NA' for those cells.

steps: number of data augmentation steps to be taken. Each step consists of a imputation or I-step followed by a posterior or P-step.

showits: if `TRUE', reports the iterations so the user can monitor the progress of the algorithm.

This function returns an array like `start' containing simulated cell probabilities.

7. mda.cat(s, start, steps=1, prior=0.5, showits=F)

Uses Markov-Chain Monte Carlo method to simulate draws from the observed-data posterior distribution of underlying cell probabilities under a saturated multinomial model. At each iteration, the missing data are randomly imputed under their predictive distribution given the observed data and the current value of `theta' (I-step). Unlike `da.cat', however, not all of the missing data are filled in, but only enough to complete a monotone pattern. Then a new value of `theta' is drawn from its Dirichlet posterior distribution given the monotone data (P-step). After a suitable number of steps are taken, the resulting value of the parameter may be regarded as a random draw from its observed-data posterior distribution. For good performance, the variables in the original data matrix `x' (which is used to create `s') should be ordered according to their rates of missingness from most observed (in the first columns) to least observed (in the last columns). This function is supposed to converge more quickly than `da.cat' when the pattern of observed data is nearly monotone.

This function may be used in conjunction with `imp.cat' to create "proper" multiple imputations. It is very IMPORTANT that the random number generator seed must be set at least once by the function `rngseed' before this function can be used.

The arguments are:

s: summary list of an incomplete categorical data set created by the function `prelim.cat'.

start: starting value of the parameter. This is an array of cell probabilities of dimension `s$d', such as one created by `em.cat'. If structural zeros appear in the table, starting values for those cells should be zero.

steps: number of data augmentation steps to be taken. Each step consists of an imputation or I-step followed by a posterior or P-step.

prior: optional vector of hyperparameters specifying a Dirichlet prior distribution. The default is the Jeffreys prior (all hyperparameters =.5). If structural zeros appear in the table, a prior should be supplied with hyperparameters set to `NA' for those cells.

showits: if `TRUE', reports the iterations so the user can monitor the progress of the algorithm.

This function returns an array like `start' containing simulated cell probabilities.

8. dabipf.cat (s, margins, theta, steps=1, prior=0.5, showits=F)

Performs data augmentation/Bayesian IPF algorithms. Produces a new draw of parameter estimates via an iterative simulation approach. At each iteration, the missing data are randomly imputed under their predictive distribution given the observed data and the current value of `theta' (I-step), and then a new value of `theta' is drawn through Bayesian IPF algorithm (P-step).

The random number generator seed must be set at least once by the function `rngseed' before this function can be used.

The arguments are:

s: summary list of an incomplete categorical data matrix `x' produced by the function `prelim.cat'.

margins: vector describing the sufficient configurations or margins in the desired loglinear model. A margin is described by the factors not summed over, and margins are separated by zeros. Thus c(1,2,0,2,3,0,1,3) would indicate the (1,2), (2,3), and (1,3) margins in a three-way table; that is, the model of no three-way association. The integers 1,2,... in the specified margins correspond to the columns of the original data matrix `x'. The same "margins" as in the function "ecm.cat" should be used if the parameter estimate generated by "ecm.cat" is used as the starting value for "theta".

theta: starting value of the parameter. This is an array of cell probabilities of dimension `s$d', such as one created by `ecm.cat'.

steps: number of data augmentation steps to be taken. Each step consists of an imputation or I-step followed by a cycle of Bayesian IPF or P-step.

prior: optional vector of hyperparameters specifying a Dirichlet prior distribution. The default is the Jeffreys prior (all hyperparameters =.5). If structural zeros appear in the table, a prior should be supplied with hyperparameters set to `NA' for those cells.

showits: if `TRUE', reports the iterations so the user can monitor the progress of the algorithm.

9. ipf(table, margins, start, eps=0.0001, maxits=50, showits=T)

Finds ML estimation for hierarchical loglinear models via iterative proportional fitting. This function is essentially the same as the S-PLUS internal function `loglin', but results are computed to double precision. See `help(loglin)' for more details.

The arguments are:

table: contingency table (array) to be fit by a log-linear model. All elements must be non-negative.

margins: vector describing the marginal totals to be fit. A margin is described by the factors not summed over, and margins are separated by zeros. Thus c(1,2,0,2,3,0,1,3) would indicate fitting the (1,2), (2,3), and (1,3) margins in a three-way table; that is, the model of no three-way association.

start: starting value for ipf algorithm. The default is a uniform table. If structural zeros appear in `table', `start' should contain zeros in those cells and ones elsewhere.

eps: convergence criterion. This is the largest proportional change in an expected cell count from one iteration to the next. Any expected cell count that drops below 1E-07 times the average cell probability (1/number of non-structural zero cells) is set to zero during the iterations.

maxits: maximum number of iterations performed. The algorithm will stop if the parameter still has not converged after this many iterations.

showits: if `TRUE', reports the iterations of IPF so the user can monitor the progress of the algorithm.

This function returns an array like `table', but containing fitted values under the loglinear model. The sum of the elements of this array equals `sum(table)'.

10. bayesipf(table, margins, start, steps=1, showits=F)

Simulates parameter estimates of cell probabilities via Bayesian iterative proportional fitting. This function performs stochastic relaxation on the expected cell counts of a contingency table under a loglinear model. Given a starting value, it cycles through the sufficient configurations, performing the Bayesian IPF algorithm (Gelman and Rubin, 1991). After a large number of steps, the resulting table of counts approximates a draw from its posterior distribution under a Dirichlet prior subject to the loglinear constraints.

The random number generator seed must be set at least once by the function `rngseed' before this function can be used. The starting value should lie in the interior of the parameter space. Hence, caution should be used when using a maximum likelihood estimate (e.g., from `ipf') as a starting value. Random zeros in a table may produce MLE's with expected cell counts of zero, and any zero in a starting value is interpreted by `bayesipf' as a structural zero. This difficulty can be overcome by using as a starting value calculated by `ipf' after adding a small positive prior count (e.g., 1/2) to each cell.

The arguments are:

table: contingency table (array) containing cell counts+prior counts. All elements should be positive, except for structural zeros, which should be zero.

margins: vector describing the sufficient configurations or margins in the desired loglinear model. A margin is described by the factors not summed over, and margins are separated by zeros. Thus c(1,2,0,2,3,0,1,3) would indicate the (1,2), (2,3), and (1,3) margins in a three-way table; that is, the model of no three-way association.

start: starting value for the algorithm. If structural zeros appear in `table', `start' should contain zeros in the same positions. Otherwise, `start' should lie in the interior of the parameter space. The default is a uniform array with zeros corresponding to the zeros in `table'.

steps: number of complete cycles of Bayesian ipf to be performed.

showits: if `TRUE', reports the iterations so that the user can monitor the progress of the algorithm.

This function returns an array like the argument "table," but containing simulated expected cell counts.

11. imp.cat(s, theta)

Performs single random imputation of missing values in a categorical data set under a user-supplied value of the underlying cell probabilities. It is very IMPORTANT that the random number generator seed must be set by the function `rngseed' at least once in the current session before this function can be used.

The two required arguments are:

s: summary list of an incomplete categorical data set created by the function `prelim.cat'.

theta: parameter value under which the missing data are to be imputed. This is an array of cell probabilities of dimension `s$d' whose elements sum to one, such as produced by `em.cat', `ecm.cat', `da.cat', `mda.cat' or `dabipf'.

This function returns an imputed complete data set. If the original incomplete data set was in ungrouped format (`s$grouped=F'), then is returns a matrix like `s$x' except that all `NA's have been filled in. If the original data set was grouped, then it returns a list with the following components:

x: Matrix of levels for categorical variables

counts: vector of length `nrow(x)' containing frequencies or counts corresponding to the levels in `x'.

12. getparam.cat(s, theta)

Convert the sorted parameter vector to an array, which is easier to read.

The two required arguments are:

s: summary list of an incomplete categorical data matrix `x' created by the function `prelim.cat'.
theta: parameter vector in sorted order, such as one produced by the function`ecm.cat'.

This function returns an array of cell probabilities whose dimensions correspond to the columns of the categorical data matrix x. The dimension is`c(max(x[,1]),max(x[,2]),...)'.

13. makeparam.cat(s, theta)

Does the opposite of "getparam.cat"—Convert parameter array to sorted vector

The two arguments are:

s: summary list of an incomplete categorical data matrix `x' created by the function `prelim.cat'.
theta: array of cell probabilities or expected frequencies whose dimensions correspond to the columns of the categorical data matrix x. The dimension should be `c(max(x[,1]),max(x[,2]),...)'.

This function returns a vector in sorted order, suitable for use as a starting value or prior for `da.cat', `mda.cat', and `dabipf'.

14. g2.cat(s, theta)

Calculates G^2 statistic for incomplete categorical data

The two required arguments are:

s: summary list of an incomplete categorical data matrix `x' created by the function `prelim.cat'.
theta: parameter vector in sorted order, such as one produced by the function `ecm.cat'.

This function returns the value of the G^2 likelihood ratio goodness of fit statistic associated with `theta'. When `theta' is the maximum likelihood estimate under the saturated model, this provides a test for the missing data being missing completely at random (MCAR), and provides a standard for testing the significance of models with loglinear constraints.

15. logpost.cat(s, theta, prior)

Calculates the observed-data loglikelihood or log-posterior density for incomplete categorical data under a specified value of the underlying cell probabilities; for example, as resulting from em.cat or ecm.cat.

The arguments are:

s: summary list of an incomplete categorical data set created by the function `prelim.cat'.
theta: an array of cell probabilities of dimension `s$d'
prior: optional vector of hyperparameters for a Dirichlet prior distribution. The default is a uniform prior distribution (all hyperparameters = 1) on the cell probabilities, which will result in evaluation of the loglikelihood. If structural zeros appear in the table, a prior should be supplied with NAs in those cells and ones (or other hyperparameters) elsewhere.

This function returns the value of the observed-data loglikelihood or log-posterior densityfunction at `theta'. This is the loglikelihood or log-posterior density that ignores the missing-data mechanism.

16. loglik.cat(s, theta)

Calculates loglikelihood for incomplete categorical data

The arguments are:

s: summary list of an incomplete categorical data matrix `x' created by the function `prelim.cat'.
theta: parameter vector in sorted order, such as one produced by the function`ecm.cat'.

The function returns the value of the loglikelihood function at `theta'.

17. mi.inference(est, std.err, confidence=0.95)

Performs multiple imputation inference. Uses the method described on pp. 76-77 of Rubin (1987) for combining estimates and standard errors from m complete-data analyses performed on m imputed data sets to produce a single inference for a scalar estimand. Significance levels and interval estimates are approximately valid for each one-dimensional estimand, not for all of them jointly.

The arguments are:

est: a list of m (at least 2) vectors representing estimates (e.g., vectors of estimated regression coefficients) from complete-data analyses performed on m imputed data sets.
std.err: a list of m vectors containing standard errors from the complete-data analyses corresponding to the estimates in `est'.
confidence: desired coverage of interval estimates.

This function returns a list with the following components, each of which is a vector of the same length as the components of `est' and `std.err':

est: the average of the complete-data estimates.
std.err: standard errors incorporating both the between and the within-imputation uncertainty (the square root of the "total variance").
df: degrees of freedom associated with the t reference distribution used for interval estimates.
signif: P-values for the two-tailed hypothesis tests that the estimated quantities are equal to zero.
lower: lower limits of the (100*confidence)% interval estimates.
upper: upper limits of the (100*confidence)% interval estimates.
r: estimated relative increases in variance due to nonresponse.
fminf: estimated fractions of missing information.

18. rngseed(seed)

Initializes random number generator seed. The argument “seed” should be a positive number, preferably a large integer. This function must be called at least once to set the random generator seed before the simulation or imputation functions in this package, such as “da.cat”, “mda.cat”, “dabipf.cat”, “imp.cat”, etc, can be used.

Appendix III: Nine S-PLUS Functions for Mixed Variables in Schafer's Multiple Imputation Software

1. prelim.mix(x,p)

Performs preliminary data manipulations for x—a matrix of incomplete mixed data. The continuous variables will be centered, scaled, and sorted by missingness patterns and the categorical variables will be grouped and sorted. It returns a list of objects that summarizes various features of the incomplete data matrix. The list will be used by functions em.mix, ecm.mix, da.mix, imp.mix, etc.

The arguments are:

x: data matrix containing missing values. The rows of x correspond to observational units, and the columns to variables. Missing values are denoted by NA. The categorical variables must be in the leftmost rows of x, and they must be coded with consecutive positive integers starting with 1. For example, a binary variable must be coded as 1, 2 rather than 0,1.

p: number of categorical variables in "x".

This function returns a list of twenty-nine components that summarize various features of x after the data have been collapsed, centered, scaled, and sorted by missingness patterns. Components that might be of interest to the user include:

nmis: a vector of length ncol(x) containing the number of missing values for each variable in x.

r: matrix of response indicators showing the missing data patterns in x. Observed values are indicated by 1 and missing values by 0. The row names give the number of observations in each pattern, and the columns correspond to the columns of x.

d: vector of length p indicating the number of levels for each categorical variable.

ncells: number of cells in the cross-classified contingency table, equal to `prod(d)'.

2. em.mix(s, start, prior=1, maxits=1000, showits=T)

Finds the ML estimate for incomplete mixed data under a unrestricted general location model through EM algorithm. If zero cell counts occur in the complete-data table, the maximum likelihood estimate may not be unique, and the algorithm may converge to different stationary values depending on the starting value. Also, if zero cell counts occur in the complete-data table, the MLE may lie on the boundary of the parameter space. Setting the prior counts greater than one will give a unique posterior mode in the interior of the parameter space.

The arguments are:

s: summary list of an incomplete data matrix produced by the function `prelim.mix'.

start: optional starting value of the parameter. This is a parameter list in packed storage, such as one returned by this function or by `da.mix'. If structural zeros appear in the contingency table, `start$pi' should contain zeros in those positions and ones elsewhere. If no starting value is supplied, `em.mix' regards all zeros as random zeros and chooses its own appropriate starting value.

prior: Optional vector or array of hyperparameter(s) for a Dirichlet prior distribution. By default, uses a uniform prior on the cell probabilities. EM algorithm finds the posterior mode, which under a uniform prior is the same as a maximum-likelihood estimate. If structural zeros appear in the table, prior counts for these cells should be set to one.

maxits: maximum number of iterations performed. The algorithm will stop if the parameter still has not converged after this many iterations.

showits: if `TRUE', reports the iterations of EM so the user can monitor the progress of the algorithm.

This function returns a list representing the maximum-likelihood estimates (or posterior mode) of the normal parameters. This list contains cell probabilities, cell means, and covariances. The parameter can be transformed back to the original scale and put into a more understandable format by the function ‘getparam.mix’.

3. ecm.mix(s,margins,design,start,prior=1,maxits=1000,showits=T)

Finds the ML estimate for incomplete mixed data under a unrestricted general location model through EM algorithm. If zero cell counts occur in the complete-data table, the maximum likelihood estimate may not be unique, and the algorithm may converge to different stationary values depending on the starting value. Also, if zero cell counts occur in the complete-data table, the MLE may lie on the boundary of the parameter space. Setting the prior counts greater than one will give a unique posterior mode in the interior of the parameter space.

The arguments are:

s: summary list of an incomplete data matrix `x' produced by the function `prelim.mix'.

margins: vector describing the sufficient configurations or margins in the desired loglinear model. The variables are ordered in the original order of the columns of `x', so that 1 refers to `x[,1]', 2 refers to `x[,2]', and so on. A margin is described by the factors not summed over, and margins are separated by zeros. Thus c(1,2,0,2,3,0,1,3) would indicate the (1,2), (2,3), and (1,3) margins in a three-way table; that is, the model of no

three-way association. See also the `loglin' function, which specifies margins in the same manner.

design: design matrix specifying the relationship of the continuous variables to the categorical ones. The dimension is `c(D,r)' where $D$ is the number of cells in the contingency table, and $r$ is the number of effects which must be less than or equal to $D$. The order of the rows corresponds to the storage order of the cell probabilities in the contingency table; see `getparam.mix' for details.

start: optional starting value of the parameter. This is a list such as one created by `em.mix'. If structural zeros appear in the table, `start$pi' should contain zeros in those positions and ones elsewhere. If no starting value is supplied, `em.mix' regards all zeros as random zeros and chooses its own appropriate starting value.

prior: Optional vector or array of hyperparameter(s) for a Dirichlet prior distribution. By default, uses a uniform prior on the cell probabilities. ECM finds the posterior mode, which under a uniform prior is the same as a maximum-likelihood estimate. If structural zeros appear in the table, prior counts for these cells should be set to one.

maxits: maximum number of iterations performed. The algorithm will stop if the parameter still has not converged after this many iterations.

showits: if `TRUE', reports the iterations of ECM so the user can monitor the progress of the algorithm.

This function returns a list representing the maximum likelihood estimates or posterior modes of the cell probabilities, within means and variance-covariances under a restricted general location model. This parameter can be put into a more understandable format by the function `getparam.mix'.

4. da.mix(s, theta, steps=1, prior=1.5, showits=F)

Performs data augmentation for the general location model without restrictions. Given a starting value, it simulates a random walk through the posterior distribution of the parameter. At each step, missing data are randomly imputed under the current parameter (I-step), and a new parameter value is drawn from its posterior distribution given the completed data (P-step). After a suitable number of steps are taken, the resulting value of the parameter may be regarded as a random draw from its incomplete-data posterior distribution.

For structural zeros, both the starting value and the prior counts must be set to zero. A suitable starting value is a table with zeros corresponding to structural zeros, and ones elsewhere. Suitable starting values may also be obtained from `em.mix'. The starting value should lie in the interior of the parameter space. Hence, caution should be used when using a maximum likelihood estimate (e.g., from `em.mix') as a starting value. Random zeros in the complete-data table may produce MLE's on the boundary of the parameter space. This difficulty can be overcome by applying `em.mix' with prior counts greater than one, ensuring a mode in the interior.

The arguments are:

s: summary list of an incomplete data matrix created by the function `prelim.mix'.
theta: starting value of the parameter. This is a parameter list such as one created by the function `em.mix'.
steps: number of data augmentation steps to be taken.
prior: Optional vector or array of hyperparameter(s) for a Dirichlet prior distribution. The default is a Dirichlet prior with all prior counts= .5. If structural zeros appear in the table, prior counts for these cells should be set to zero.
showits: if `TRUE', reports the iterations so the user can monitor the progress of the algorithm.

This function returns a list containing new parameter estimate. The parameter can be put into a more understandable format by the function `getparam.mix'.

5. imp.mix(s, theta, x)

Imputes missing data under the unrestricted general location model with user-supplied values of parameter. The random number generator seed must be set at least once by the function `rngseed' before this function can be used.

The arguments are:

s: summary list of an incomplete data matrix `x' created by the function `prelim.mix'.
theta: value of the parameter under which the missing data are to be randomly imputed. This is a parameter list such as one created by `em.mix' or `da.mix'.
x: the original data matrix used to create the summary list `s'. If this argument is not supplied, then the data matrix returned by this function may disagree slightly with the observed values in `x' due to rounding errors.

This function returns a matrix of the same form as `x', but with all missing values filled in with simulated values drawn from their predictive distribution given the observed data and the specified parameter.

6. dabipf.mix(s, margins, design, theta, steps=1, prior=1.5, showits=F)

Performs data augmentation/Bayesian IPF algorithm for the general location model with restrictions. Given a starting value, it simulates a random walk through the posterior distribution of the parameter. At each step, missing data are randomly imputed under the current parameter (I-step), and a new parameter value is drawn via Bayesian IPF algorithm given the completed data. After a

suitable number of steps are taken, the resulting value of the parameter may be regarded as a random draw from its incomplete-data posterior distribution.

For structural zeros, both the starting value and the prior counts must be set to zero. A suitable starting value is a table with zeros corresponding to structural zeros, and ones elsewhere. Suitable starting values may also be obtained from `ecm.mix'. The starting value should lie in the interior of the parameter space. Hence, caution should be used when using a maximum likelihood estimate (e.g., from `ecm.mix') as a starting value. Random zeros in the complete-data table may produce MLE's on the boundary of the parameter space. This difficulty can be overcome by applying `ecm.mix' with prior counts greater than one, ensuring a mode in the interior.

The arguments are:

s: summary list of an incomplete data matrix created by the function `prelim.mix'.

margins: vector describing the sufficient configurations or margins in the desired loglinear model. The variables are ordered in the original order of the columns of `x', so that 1 refers to `x[,1]', 2 refers to `x[,2]', and so on. A margin is described by the factors not summed over, and margins are separated by zeros. Thus c(1,2, 0,2,3,0,1,3) would indicate the (1,2), (2,3), and (1,3) margins in a three-way table; that is, the model of no three-way association. See also the `loglin' function, which specifies margins in the same manner.

design: design matrix specifying the relationship of the continuous variables to the categorical ones. The dimension is `c(D,r)' where $D$ is the number of cells in the contingency table, and $r$ is the number of effects which must be less than or equal to $D$. The order of the rows corresponds to the storage order of the cell probabilities in the contingency table; see `getparam.mix' for details.

theta: starting value of the parameter. This is a parameter list such as one created by the function `ecm.mix'.

steps: number of data augmentation/Bayesian IPF steps to be taken.

prior: Optional vector or array of hyperparameter(s) for a Dirichlet prior distribution. The default is a Dirichlet prior with all prior counts= .5. If structural zeros appear in the table, prior counts for these cells should be set to zero.

showits: if `TRUE', reports the iterations so the user can monitor the progress of the algorithm.

This function returns a list containing new parameter estimates. The parameter can be put into a more understandable format by the function `getparam.mix'.

7. getparam.mix(s, theta, corr=F)

Presents parameters of general location model in an understandable format

The parameters are:

s: summary list of an incomplete normal data matrix created by the function `prelim.mix'.
theta: list of parameters such as one produced by the function `em.mix'.
corr: if `FALSE', returns a list containing an array of cell probabilities, a matrix of cell means, and a variance-covariance matrix. If `TRUE', returns a list containing an array of cell probabilities, a matrix of cell means, a vector of standard deviations, and a correlation matrix.

If `corr=F', the function returns a list containing parameter estimates of cell probabilities, cell means and variance-covariances; if `corr=T', it returns a list containing parameter estimates of cell probabilities, cell means, standard deviations and correlation matrix. The list contains the following components:

pi: array of cell probabilities whose dimensions correspond to the columns of the categorical part of x. The dimension is `c(max(x[,1]),max(x[,2]),...,max(x[,p]))' where p is the number of categorical variables.
mu: Matrix of cell means. The dimension is `c(q,D)' where $q$ is the number of continuous variables in $x$, and $D$ is 'length(pi)'. The order of the rows, corresponding to the elements of `pi', is the same order we would get by vectorizing `pi', as in `as.vector(pi)'; it is the usual lexicographic order used by S and Fortran, with the subscript corresponding to `x[,1]' varying the fastest, and the subscript corresponding to `x[,p]' varying the slowest.
sigma: matrix of variances and covariances corresponding to the continuous variables in `x'.
sdv: vector of standard deviations corresponding to the continuous variables in `x'.
r: matrix of correlations corresponding to the continuous variables in `x'.

8. loglik.mix(s, theta)

Calculates loglikelihood for incomplete data under the general location model

The arguments are:

s: summary list of an incomplete data matrix `x' created by the function `prelim.mix'.
theta: parameter list, such as one produced by `ecm.mix' or `da.mix'.

This function returns the value of the loglikelihood function at `theta'.

9. rngseed(seed)

Initializes random number generator seed. The argument "seed" should be a positive number, preferably a large integer. This function must be called at least once to set the random generator seed before the simulation or imputation functions in this package, such as "da.cat", "mda.cat", "dabipf.cat", "imp.cat", etc., can be used.

This page intentionally left blank.

Appendix IV. Imputations by NCES, Schafer's Software, and Proc Impute

	Degree BS/BA	NCES Impute	Schafer #1	Schafer #2	Proc Impute	Teaching Exper.	NCES Impute	Schafer #1	Schafer #2	Proc Impute	Birth Year	NCES Impute	Schafer #1	Schafer #2	Proc Impute
Mean	66.617	66.58	66.599	66.604	66.601	10.214	10.21	10.221	10.22	10.217	42.92	42.909	42.895	42.907	42.891
Std	6.833	6.855	6.846	6.839	6.851	5.659	5.67	5.662	5.656	5.659	7.11	7.125	7.118	7.111	7.127
# Missing	142	-	-	-	-	82	-	-	-	-	114	-	-	-	-
1	.	65	69	67	66	10	10	10	10	10	43	43	43	43	43
2	.	63	67	71	72	23	23	23	23	23	41	41	41	41	41
3	.	61	65	62	62	7	7	7	7	7	39	39	39	39	39
4	.	70	79	69	71	13	13	13	13	13	48	48	48	48	48
5	.	69	71	69	70	4	4	4	4	4	47	47	47	47	47
6	.	51	56	53	50	8	8	8	8	8	29	29	29	29	29
7	.	70	67	69	70	12	12	12	12	12	46	46	46	46	46
8	.	65	67	65	67	11	11	11	11	11	43	43	43	43	43
9	.	71	74	73	73	6	6	6	6	6	49	49	49	49	49
10	.	71	72	72	76	9	9	9	9	9	49	49	49	49	49
11	.	71	74	67	72	13	13	13	13	13	49	49	49	49	49
12	.	61	60	57	59	4	4	4	4	4	39	39	39	39	39
13	.	52	51	49	50	0	0	0	0	0	30	30	30	30	30
14	.	66	69	67	68	5	5	5	5	5	44	44	44	44	44
15	.	68	67	77	67	6	6	6	6	6	46	46	46	46	46
16	.	69	66	72	70	5	5	5	5	5	47	47	47	47	47
17	.	52	54	57	74	8	8	8	8	8	30	30	30	30	30
18	.	56	58	62	61	12	12	12	12	12	34	34	34	34	34
19	.	76	75	79	78	8	8	8	8	8	54	54	54	54	54
20	.	72	72	73	71	7	7	7	7	7	50	50	50	50	50
21	.	53	52	61	56	5	5	5	5	5	31	31	31	31	31
22	.	55	61	58	53	10	10	10	10	10	33	33	33	33	33
23	.	58	61	60	59	2	2	2	2	2	36	36	36	36	36
24	.	59	62	62	60	9	9	9	9	9	37	37	37	37	37
25	.	57	55	61	56	15	15	15	15	15	35	35	35	35	35
26	.	55	59	55	68	11	11	11	11	11	33	33	33	33	33
27	.	68	68	71	67	1	1	1	1	1	46	46	46	46	46
28	.	70	72	74	69	10	10	10	10	10	48	48	48	48	48
29	.	67	65	70	70	5	5	5	5	5	45	45	45	45	45

	Degree BS/BA	NCES Impute	Schafer #1	Schafer #2	Proc Impute	Teaching Exper.	NCES Impute	Schafer #1	Schafer #2	Proc Impute	Birth Year	NCES Impute	Schafer #1	Schafer #2	Proc Impute
30	.	66	62	64	64	3	3	3	3	3	44	44	44	44	44
31	.	78	76	75	77	10	10	10	10	10	56	56	56	56	56
32	.	69	74	71	67	11	11	11	11	11	47	47	47	47	47
33	.	54	54	58	55	18	18	18	18	18	32	32	32	32	32
34	.	60	62	63	60	8	8	8	8	8	38	38	38	38	38
35	.	80	69	64	68	10	10	10	10	10	47	47	47	47	47
36	.	65	69	73	65	7	7	7	7	7	43	43	43	43	43
37	.	69	74	64	67	17	17	17	17	17	47	47	47	47	47
38	.	59	58	63	56	20	20	20	20	20	37	37	37	37	37
39	.	67	67	68	67	15	15	15	15	15	45	45	45	45	45
40	.	63	61	65	63	5	5	5	5	5	41	41	41	41	41
41	.	72	70	70	72	6	6	6	6	6	50	50	50	50	50
42	.	62	68	63	62	4	4	4	4	4	40	40	40	40	40
43	.	66	72	70	66	14	14	14	14	14	44	44	44	44	44
44	.	63	62	62	67	6	6	6	6	6	41	41	41	41	41
45	.	53	55	57	60	14	14	14	14	14	31	31	31	31	31
46	.	70	68	67	62	13	13	13	13	13	48	48	48	48	48
47	.	69	72	71	69	13	13	13	13	13	47	47	47	47	47
48	.	72	73	72	71	9	9	9	9	9	50	50	50	50	50
49	.	76	56	58	57	4	4	4	4	4	35	35	35	35	35
50	.	73	74	75	76	11	11	11	11	11	51	51	51	51	51
51	.	67	68	68	67	21	21	21	21	21	45	45	45	45	45
52	.	64	68	67	67	4	4	4	4	4	42	42	42	42	42
53	.	64	66	67	68	18	18	18	18	18	42	42	42	42	42
54	.	76	76	75	76	12	12	12	12	12	54	54	54	54	54
55	.	65	66	64	70	7	7	7	7	7	43	43	43	43	43
56	.	75	73	73	79	5	5	5	5	5	47	47	47	47	47
57	.	64	63	71	63	24	24	24	24	24	42	42	42	42	42
58	.	46	49	51	47	24	24	24	24	24	24	24	24	24	24
59	.	56	60	59	58	7	7	7	7	7	34	34	34	34	34
60	.	58	61	63	56	30	30	30	30	30	36	36	36	36	36
61	.	76	78	77	76	13	13	13	13	13	54	54	54	54	54
62	.	58	60	59	73	20	20	20	20	20	36	36	36	36	36
63	.	67	73	68	68	14	14	14	14	14	45	45	45	45	45
64	.	58	55	62	58	10	10	10	10	10	36	36	36	36	36
65	.	47	51	51	50	15	15	15	15	15	25	25	25	25	25

	Degree BS/BA	NCES Impute	Schafer #1	Schafer #2	Proc Impute	Teaching Exper.	NCES Impute	Schafer #1	Schafer #2	Proc Impute	Birth Year	NCES Impute	Schafer #1	Schafer #2	Proc Impute
66	.	75	76	78	76	3	3	3	3	3	53	53	53	53	53
67	.	59	62	67	60	4	4	4	4	4	37	37	37	37	37
68	.	76	77	73	75	14	14	14	14	14	54	54	54	54	54
69	.	74	53	61	57	13	13	13	13	13	35	35	35	35	35
70	.	69	73	68	70	10	10	10	10	10	47	47	47	47	47
71	.	59	63	62	74	22	22	22	22	22	37	37	37	37	37
72	.	62	64	64	71	5	5	5	5	5	40	40	40	40	40
73	.	63	68	68	64	9	9	9	9	9	41	41	41	41	41
74	.	49	54	56	56	3	3	3	3	3	27	27	27	27	27
75	.	69	69	69	69	15	15	15	15	15	47	47	47	47	47
76	.	71	73	62	63	10	10	10	10	10	40	40	40	40	40
77	.	69	74	68	68	13	13	13	13	13	47	47	47	47	47
78	.	56	56	57	63	6	6	6	6	6	34	34	34	34	34
79	.	62	64	59	61	3	3	3	3	3	40	40	40	40	40
80	.	68	69	69	73	9	9	9	9	9	46	46	46	46	46
81	.	66	69	71	67	4	4	4	4	4	44	44	44	44	44
82	.	53	61	58	51	0	0	0	0	0	31	31	31	31	31
83	.	61	64	66	61	7	7	7	7	7	39	39	39	39	39
84	.	68	62	68	82	9	9	9	9	9	46	46	46	46	46
85	.	49	54	52	49	13	13	13	13	13	27	27	27	27	27
86	.	79	72	83	78	7	7	7	7	7	57	57	57	57	57
87	.	60	58	65	58	15	15	15	15	15	38	38	38	38	38
88	.	66	65	67	64	21	21	21	21	21	44	44	44	44	44
89	.	65	68	69	63	17	17	17	17	17	43	43	43	43	43
90	.	67	66	64	67	6	6	6	6	6	45	45	45	45	45
91	.	72	75	71	70	9	9	9	9	9	50	50	50	50	50
92	.	72	72	78	71	7	7	7	7	7	50	50	50	50	50
93	.	69	71	74	70	3	3	3	3	3	47	47	47	47	47
94	.	63	67	64	64	10	10	10	10	10	41	41	41	41	41
95	.	51	59	58	54	12	12	12	12	12	29	29	29	29	29
96	.	69	67	70	73	12	12	12	12	12	47	47	47	47	47
97	.	69	70	71	69	10	10	10	10	10	47	47	47	47	47
98	.	62	60	64	70	21	21	21	21	21	40	40	40	40	40
99	.	68	70	72	67	3	3	3	3	3	46	46	46	46	46
100	.	56	62	61	57	17	17	17	17	17	34	34	34	34	34
101	.	64	67	62	68	7	7	7	7	7	42	42	42	42	42

	Degree BS/BA	NCES Impute	Schafer #1	Schafer #2	Proc Impute	Teaching Exper.	NCES Impute	Schafer #1	Schafer #2	Proc Impute	Birth Year	NCES Impute	Schafer #1	Schafer #2	Proc Impute
102	.	61	63	63	62	16	16	16	16	16	39	39	39	39	39
103	.	61	67	64	60	12	12	12	12	12	39	39	39	39	39
104	.	68	74	68	70	13	13	13	13	13	46	46	46	46	46
105	.	56	52	56	57	3	3	3	3	3	34	34	34	34	34
106	.	73	70	69	75	15	15	15	15	15	51	51	51	51	51
107	.	64	64	68	67	7	7	7	7	7	42	42	42	42	42
108	.	53	58	54	56	19	19	19	19	19	31	31	31	31	31
109	.	59	62	62	63	5	5	5	5	5	37	37	37	37	37
110	.	78	76	77	77	6	6	6	6	6	56	56	56	56	56
111	.	54	58	58	59	5	5	5	5	5	32	32	32	32	32
112	.	58	63	61	56	9	9	9	9	9	36	36	36	36	36
113	.	63	72	61	63	14	14	14	14	14	41	41	41	41	41
114	.	73	73	74	72	3	3	3	3	3	51	51	51	51	51
115	.	69	70	70	72	6	6	6	6	6	47	47	47	47	47
116	.	50	55	53	49	11	11	11	11	11	28	28	28	28	28
117	.	61	63	63	62	8	8	8	8	8	39	39	39	39	39
118	.	61	59	64	64	19	19	19	19	19	39	39	39	39	39
119	.	62	65	63	67	9	9	9	9	9	40	40	40	40	40
120	.	70	63	73	73	15	15	15	15	15	48	48	48	48	48
121	.	68	68	70	70	9	9	9	9	9	46	46	46	46	46
122	.	60	64	57	60	25	25	25	25	25	38	38	38	38	38
123	.	60	63	61	59	17	17	17	17	17	38	38	38	38	38
124	.	68	71	70	73	10	10	10	10	10	46	46	46	46	46
125	.	67	64	69	66	6	6	6	6	6	45	45	45	45	45
126	.	65	61	72	68	5	5	5	5	5	43	43	43	43	43
127	.	76	81	81	76	6	6	6	6	6	54	54	54	54	54
128	.	72	78	74	73	10	10	10	10	10	50	50	50	50	50
129	.	66	68	64	60	0	0	0	0	0	44	44	44	44	44
130	.	53	61	49	54	10	10	10	10	10	31	31	31	31	31
131	.	54	63	58	57	15	15	15	15	15	32	32	32	32	32
132	.	58	62	57	71	3	3	3	3	3	36	36	36	36	36
133	.	88	89	88	88	0	0	0	0	0	66	66	66	66	66
134	.	45	49	50	59	0	0	0	0	0	23	23	23	23	23
135	74	74	74	74	74	.	10	14	8	13	53	53	53	53	53
136	69	69	69	69	69	.	0	13	12	18	47	47	47	47	47
137	75	75	75	75	75	.	14	11	1	15	53	53	53	53	53

	Degree BS/BA	NCES Impute	Schafer #1	Schafer #2	Proc Impute	Teaching Exper.	NCES Impute	Schafer #1	Schafer #2	Proc Impute	Birth Year	NCES Impute	Schafer #1	Schafer #2	Proc Impute
138	69	69	69	69	69	.	3	-1	6	5	47	47	47	47	47
139	74	74	74	74	74	.	16	12	19	8	49	49	49	49	49
140	74	74	74	74	74	.	9	1	9	9	54	54	54	54	54
141	71	71	71	71	71	.	5	5	13	5	46	46	46	46	46
142	63	63	63	63	63	.	16	24	15	12	38	38	38	38	38
143	60	60	60	60	60	.	20	13	11	18	38	38	38	38	38
144	57	57	57	57	57	.	2	11	17	16	34	34	34	34	34
145	64	64	64	64	64	.	4	8	10	5	42	42	42	42	42
146	57	57	57	57	57	.	4	13	14	17	35	35	35	35	35
147	70	70	70	70	70	.	9	8	9	16	46	46	46	46	46
148	71	71	71	71	71	.	0	12	8	7	49	49	49	49	49
149	60	60	60	60	60	.	0	15	6	8	34	34	34	34	34
150	62	62	62	62	62	.	0	1	-2	4	27	27	27	27	27
151	72	72	72	72	72	.	17	4	8	7	50	50	50	50	50
152	80	80	80	80	80	.	15	6	7	7	58	58	58	58	58
153	70	70	70	70	70	.	13	11	6	12	48	48	48	48	48
154	67	67	67	67	67	.	14	20	15	5	43	43	43	43	43
155	66	66	66	66	66	.	15	16	10	18	45	45	45	45	45
156	73	73	73	73	73	.	19	11	20	17	51	51	51	51	51
157	70	70	70	70	70	.	12	17	11	10	47	47	47	47	47
158	70	70	70	70	70	.	12	1	16	14	48	48	48	48	48
159	68	68	68	68	68	.	16	7	8	11	43	43	43	43	43
160	66	66	66	66	66	.	13	13	13	20	49	49	49	49	49
161	68	68	68	68	68	.	9	5	10	9	45	45	45	45	45
162	72	72	72	72	72	.	14	10	13	12	49	49	49	49	49
163	55	55	55	55	55	.	20	10	7	3	22	22	22	22	22
164	71	71	71	71	71	.	15	16	9	19	48	48	48	48	48
165	65	65	65	65	65	.	10	9	10	14	44	44	44	44	44
166	62	62	62	62	62	.	15	18	18	10	27	27	27	27	27
167	57	57	57	57	57	.	0	7	3	7	34	34	34	34	34
168	65	65	65	65	65	.	9	14	10	14	43	43	43	43	43
169	66	66	66	66	66	.	17	6	20	17	44	44	44	44	44
170	68	68	68	68	68	.	2	20	7	10	46	46	46	46	46
171	57	57	57	57	57	.	0	11	4	0	32	32	32	32	32
172	60	60	60	60	60	.	5	20	14	18	36	36	36	36	36
173	58	58	58	58	58	.	30	15	19	19	33	33	33	33	33

	Degree BS/BA	NCES Impute	Schafer #1	Schafer #2	Proc Impute	Teaching Exper.	NCES Impute	Schafer #1	Schafer #2	Proc Impute	Birth Year	NCES Impute	Schafer #1	Schafer #2	Proc Impute
174	68	68	68	68	68	.	10	2	7	7	42	42	42	42	42
175	62	62	62	62	62	.	19	14	13	7	40	40	40	40	40
176	71	71	71	71	71	.	12	15	11	13	49	49	49	49	49
177	69	69	69	69	69	.	10	12	6	12	47	47	47	47	47
178	64	64	64	64	64	.	10	18	16	6	41	41	41	41	41
179	59	59	59	59	59	.	13	10	14	12	34	34	34	34	34
180	62	62	62	62	62	.	16	18	9	27	41	41	41	41	41
181	69	69	69	69	69	.	16	5	9	13	46	46	46	46	46
182	72	72	72	72	72	.	8	-5	13	10	47	47	47	47	47
183	80	80	80	80	80	.	14	3	11	10	57	57	57	57	57
184	59	59	59	59	59	.	19	21	21	7	32	32	32	32	32
185	63	63	63	63	63	.	0	4	8	0	40	40	40	40	40
186	63	63	63	63	63	.	21	15	11	4	37	37	37	37	37
187	58	58	58	58	58	.	12	7	5	7	29	29	29	29	29
188	69	69	69	69	69	.	5	9	5	0	47	47	47	47	47
189	69	69	69	69	69	.	0	9	11	19	47	47	47	47	47
190	68	68	68	68	68	.	8	11	19	17	47	47	47	47	47
191	71	71	71	71	71	.	10	9	15	3	49	49	49	49	49
192	72	72	72	72	72	.	10	17	17	13	46	46	46	46	46
193	59	59	59	59	59	.	12	15	16	20	37	37	37	37	37
194	71	71	71	71	71	.	9	6	11	7	47	47	47	47	47
195	58	58	58	58	58	.	6	12	14	9	35	35	35	35	35
196	78	78	78	78	78	.	0	6	10	4	54	54	54	54	54
197	59	59	59	59	59	.	2	8	15	17	37	37	37	37	37
198	67	67	67	67	67	.	11	12	19	13	41	41	41	41	41
199	68	68	68	68	68	.	17	10	4	4	45	45	45	45	45
200	52	52	52	52	52	.	16	13	7	14	22	22	22	22	22
201	59	59	59	59	59	.	5	13	11	3	35	35	35	35	35
202	57	57	57	57	57	.	25	6	19	2	24	24	24	24	24
203	57	57	57	57	57	.	0	8	14	7	35	35	35	35	35
204	60	60	60	60	60	.	0	23	4	9	38	38	38	38	38
205	73	73	73	73	73	.	7	13	16	8	51	51	51	51	51
206	70	70	70	70	70	.	11	19	9	17	36	36	36	36	36
207	71	71	71	71	71	.	11	15	14	17	43	43	43	43	43
208	72	72	72	72	72	.	12	22	7	9	50	50	50	50	50
209	69	69	69	69	69	.	6	10	12	17	47	47	47	47	47

	Degree BS/BA	NCES Impute	Schafer #1	Schafer #2	Proc Impute	Teaching Exper.	NCES Impute	Schafer #1	Schafer #2	Proc Impute	Birth Year	NCES Impute	Schafer #1	Schafer #2	Proc Impute
210	62	62	62	62	62	.	1	18	14	5	40	40	40	40	40
211	73	73	73	73	73	.	10	15	14	8	52	52	52	52	52
212	61	61	61	61	61	.	0	15	-3	7	38	38	38	38	38
213	.	60	62	64	58	.	0	9	19	5	38	38	38	38	38
214	72	72	72	72	72	7	7	7	7	7	.	55	49	45	47
215	75	75	75	75	75	5	5	5	5	5	.	52	53	56	45
216	68	68	68	68	68	5	5	5	5	5	.	48	46	49	46
217	70	70	70	70	70	4	4	4	4	4	.	48	44	44	45
218	78	78	78	78	78	6	6	6	6	6	.	56	48	51	52
219	62	62	62	62	62	13	13	13	13	13	.	38	34	40	33
220	64	64	64	64	64	6	6	6	6	6	.	43	38	43	43
221	72	72	72	72	72	4	4	4	4	4	.	52	48	47	52
222	67	67	67	67	67	20	20	20	20	20	.	36	45	48	46
223	60	60	60	60	60	21	21	21	21	21	.	27	37	38	33
224	69	69	69	69	69	5	5	5	5	5	.	42	43	42	45
225	70	70	70	70	70	8	8	8	8	8	.	41	42	43	45
226	63	63	63	63	63	4	4	4	4	4	.	40	37	42	40
227	68	68	68	68	68	11	11	11	11	11	.	46	47	41	47
228	55	55	55	55	55	10	10	10	10	10	.	33	30	30	32
229	61	61	61	61	61	15	15	15	15	15	.	39	41	39	37
230	62	62	62	62	62	10	10	10	10	10	.	40	44	39	33
231	62	62	62	62	62	6	6	6	6	6	.	40	39	39	39
232	61	61	61	61	61	8	8	8	8	8	.	39	42	38	35
233	74	74	74	74	74	15	15	15	15	15	.	51	50	46	49
234	65	65	65	65	65	16	16	16	16	16	.	45	39	42	47
235	68	68	68	68	68	11	11	11	11	11	.	46	45	44	36
236	60	60	60	60	60	6	6	6	6	6	.	38	36	38	33
237	65	65	65	65	65	18	18	18	18	18	.	42	38	47	40
238	62	62	62	62	62	6	6	6	6	6	.	37	41	37	33
239	73	73	73	73	73	12	12	12	12	12	.	33	41	51	49
240	55	55	55	55	55	20	20	20	20	20	.	33	30	36	30
241	58	58	58	58	58	4	4	4	4	4	.	36	32	38	34
242	71	71	71	71	71	15	15	15	15	15	.	51	44	49	41
243	72	72	72	72	72	8	8	8	8	8	.	54	43	51	51
244	66	66	66	66	66	15	15	15	15	15	.	45	48	44	41
245	57	57	57	57	57	4	4	4	4	4	.	35	36	36	30

	Degree BS/BA	NCES Impute	Schafer #1	Schafer #2	Proc Impute	Teaching Exper.	NCES Impute	Schafer #1	Schafer #2	Proc Impute	Birth Year	NCES Impute	Schafer #1	Schafer #2	Proc Impute
246	69	69	69	69	69	5	5	5	5	5	.	45	43	44	35
247	55	55	55	55	55	6	6	6	6	6	.	33	36	34	21
248	65	65	65	65	65	11	11	11	11	11	.	43	44	46	43
249	62	62	62	62	62	8	8	8	8	8	.	42	44	36	39
250	63	63	63	63	63	9	9	9	9	9	.	44	40	38	41
251	50	50	50	50	50	18	18	18	18	18	.	26	27	33	26
252	52	52	52	52	52	14	14	14	14	14	.	30	31	31	21
253	60	60	60	60	60	7	7	7	7	7	.	33	39	35	35
254	60	60	60	60	60	8	8	8	8	8	.	36	35	33	37
255	73	73	73	73	73	5	5	5	5	5	.	51	46	51	52
256	63	63	63	63	63	4	4	4	4	4	.	37	44	42	36
257	59	59	59	59	59	8	8	8	8	8	.	34	36	42	38
258	60	60	60	60	60	7	7	7	7	7	.	38	38	40	36
259	60	60	60	60	60	7	7	7	7	7	.	38	36	39	35
260	57	57	57	57	57	9	9	9	9	9	.	35	33	36	33
261	72	72	72	72	72	13	13	13	13	13	.	51	49	47	51
262	53	53	53	53	53	18	18	18	18	18	.	31	30	31	34
263	68	68	68	68	68	0	0	0	0	0	.	46	46	46	47
264	70	70	70	70	70	15	15	15	15	15	.	46	48	48	47
265	65	65	65	65	65	21	21	21	21	21	.	44	45	43	44
266	69	69	69	69	69	21	21	21	21	21	.	29	44	41	44
267	58	58	58	58	58	18	18	18	18	18	.	36	39	31	41
268	75	75	75	75	75	11	11	11	11	11	.	53	49	47	48
269	75	75	75	75	75	13	13	13	13	13	.	51	47	54	51
270	59	59	59	59	59	7	7	7	7	7	.	37	38	39	33
271	68	68	68	68	68	17	17	17	17	17	.	48	42	44	46
272	60	60	60	60	60	11	11	11	11	11	.	41	37	40	37
273	70	70	70	70	70	14	14	14	14	14	.	49	44	47	49
274	46	46	46	46	46	7	7	7	7	7	.	24	28	20	32
275	70	70	70	70	70	20	20	20	20	20	.	43	44	48	47
276	50	50	50	50	50	7	7	7	7	7	.	28	25	31	34
277	75	75	75	75	75	13	13	13	13	13	.	49	53	54	53
278	49	49	49	49	49	23	23	23	23	23	.	27	25	30	31
279	58	58	58	58	58	2	2	2	2	2	.	36	35	34	39
280	79	79	79	79	79	8	8	8	8	8	.	53	52	51	56
281	63	63	63	63	63	5	5	5	5	5	.	45	40	42	28

	Degree BS/BA	NCES Impute	Schafer #1	Schafer #2	Proc Impute	Teaching Exper.	NCES Impute	Schafer #1	Schafer #2	Proc Impute	Birth Year	NCES Impute	Schafer #1	Schafer #2	Proc Impute
282	73	73	73	73	73	7	7	7	7	7	.	53	44	47	51
283	70	70	70	70	70	3	3	3	3	3	.	48	48	45	47
284	82	82	82	82	82	2	2	2	2	2	.	62	58	56	48
285	61	61	61	61	61	7	7	7	7	7	.	44	36	41	39
286	57	57	57	57	57	27	27	27	27	27	.	38	31	35	28
287	58	58	58	58	58	5	5	5	5	5	.	34	31	39	41
288	64	64	64	64	64	14	14	14	14	14	.	43	39	40	40
289	69	69	69	69	69	15	15	15	15	15	.	47	47	45	29
290	58	58	58	58	58	19	19	19	19	19	.	41	37	37	38
291	73	73	73	73	73	11	11	11	11	11	.	52	46	61	55
292	60	60	60	60	60	6	6	6	6	6	.	38	31	38	38
293	65	65	65	65	65	5	5	5	5	5	.	43	47	42	39
294	76	76	76	76	76	28	28	28	28	28	.	34	52	53	44
295	72	72	72	72	72	12	12	12	12	12	.	54	44	52	51
296	67	67	67	67	67	16	16	16	16	16	.	48	44	40	40
297	51	51	51	51	51	10	10	10	10	10	.	29	23	31	27
298	77	77	77	77	77	11	11	11	11	11	.	54	48	49	54
299	65	65	65	65	65	13	13	13	13	13	.	45	43	42	42
300	62	62	62	62	62	13	13	13	13	13	.	40	45	40	42
301	53	53	53	53	53	12	12	12	12	12	.	31	31	28	25
302	70	70	70	70	70	5	5	5	5	5	.	48	43	40	48
303	57	57	57	57	57	13	13	13	13	13	.	35	36	35	25
304	69	69	69	69	69	31	31	31	31	31	.	23	51	45	45
305	73	73	73	73	73	5	5	5	5	5	.	50	51	51	48
306	65	65	65	65	65	14	14	14	14	14	.	43	36	41	44
307	72	72	72	72	72	3	3	3	3	3	.	50	50	50	50
308	70	70	70	70	70	15	15	15	15	15	.	47	45	51	47
309	55	55	55	55	55	14	14	14	14	14	.	33	32	39	31
310	61	61	61	61	61	9	9	9	9	9	.	39	39	37	33
311	72	72	72	72	72	16	16	16	16	16	.	50	54	44	43
312	77	77	77	77	77	11	11	11	11	11	.	56	55	54	48
313	71	71	71	71	71	3	3	3	3	3	.	51	50	49	48
314	59	59	59	59	59	14	14	14	14	14	.	40	35	30	33
315	50	50	50	50	50	5	5	5	5	5	.	30	23	23	28
316	72	72	72	72	72	10	10	10	10	10	.	49	44	48	47
317	67	67	67	67	67	10	10	10	10	10	.	49	48	45	40

	Degree BS/BA	NCES Impute	Schafer #1	Schafer #2	Proc Impute	Teaching Exper.	NCES Impute	Schafer #1	Schafer #2	Proc Impute	Birth Year	NCES Impute	Schafer #1	Schafer #2	Proc Impute
318	.	66	80	70	79	9	9	9	9	9	.	44	55	42	55
319	.	77	56	69	76	10	10	10	10	10	.	55	35	44	50
320	.	56	68	72	77	15	15	15	15	15	.	34	40	50	53
321	.	62	50	62	56	10	10	10	10	10	.	40	24	40	36
322	.	70	65	62	54	6	6	6	6	6	.	48	41	38	30
323	.	60	58	61	64	7	7	7	7	7	.	38	36	41	42
324	.	69	61	67	62	10	10	10	10	10	.	47	33	46	42
325	63	63	63	63	63	.	0	3	4	4	.	28	38	35	35
326	66	66	66	66	66	.	11	3	-2	10	.	44	42	38	35
327	70	70	70	70	70	.	9	2	8	8	.	52	47	41	47

Notes: (1) The three variables compared from 1990-91 SASS Administrator data file are:

BS/BA: Year received bachelor's degree

Teaching Experience: Years of elementary/secondary teaching experience before becoming principal

Birth: Year of birth

(2) Schafer's imputations are rounded up.

(3) In each five-column section, the first column represents un-imputed data, the second NCES imputed data, the third and fourth Schafer's software imputed data (set1 and set 2), the fifth Proc Impute imputed data, respectively, for the cases in which at least one of the three variables has a missing value.

(4) Mean and standard deviation are calculated for the whole sample with sample size 9054.

Listing of NCES Working Papers to Date

Working papers can be downloaded as pdf files from the NCES Electronic Catalog (http://nces.ed.gov/pubsearch/). You can also contact Sheilah Jupiter at (202) 502–7444 (sheilah_jupiter@ed.gov) if you are interested in any of the following papers.

Listing of NCES Working Papers by Program Area

No.	Title	NCES contact
Baccalaureate and Beyond (B&B)		
98–15	Development of a Prototype System for Accessing Linked NCES Data	Steven Kaufman
Beginning Postsecondary Students (BPS) Longitudinal Study		
98–11	Beginning Postsecondary Students Longitudinal Study First Follow-up (BPS:96–98) Field Test Report	Aurora D'Amico
98–15	Development of a Prototype System for Accessing Linked NCES Data	Steven Kaufman
1999–15	Projected Postsecondary Outcomes of 1992 High School Graduates	Aurora D'Amico
2001–04	Beginning Postsecondary Students Longitudinal Study: 1996–2001 (BPS:1996/2001) Field Test Methodology Report	Paula Knepper
Common Core of Data (CCD)		
95–12	Rural Education Data User's Guide	Samuel Peng
96–19	Assessment and Analysis of School-Level Expenditures	William J. Fowler, Jr.
97–15	Customer Service Survey: Common Core of Data Coordinators	Lee Hoffman
97–43	Measuring Inflation in Public School Costs	William J. Fowler, Jr.
98–15	Development of a Prototype System for Accessing Linked NCES Data	Steven Kaufman
1999–03	Evaluation of the 1996–97 Nonfiscal Common Core of Data Surveys Data Collection, Processing, and Editing Cycle	Beth Young
2000–12	Coverage Evaluation of the 1994–95 Common Core of Data: Public Elementary/Secondary School Universe Survey	Beth Young
2000–13	Non-professional Staff in the Schools and Staffing Survey (SASS) and Common Core of Data (CCD)	Kerry Gruber
2001–09	An Assessment of the Accuracy of CCD Data: A Comparison of 1988, 1989, and 1990 CCD Data with 1990–91 SASS Data	John Sietsema
Data Development		
2000–16a	Lifelong Learning NCES Task Force: Final Report Volume I	Lisa Hudson
2000–16b	Lifelong Learning NCES Task Force: Final Report Volume II	Lisa Hudson
Decennial Census School District Project		
95–12	Rural Education Data User's Guide	Samuel Peng
96–04	Census Mapping Project/School District Data Book	Tai Phan
98–07	Decennial Census School District Project Planning Report	Tai Phan
Early Childhood Longitudinal Study (ECLS)		
96–08	How Accurate are Teacher Judgments of Students' Academic Performance?	Jerry West
96–18	Assessment of Social Competence, Adaptive Behaviors, and Approaches to Learning with Young Children	Jerry West
97–24	Formulating a Design for the ECLS: A Review of Longitudinal Studies	Jerry West
97–36	Measuring the Quality of Program Environments in Head Start and Other Early Childhood Programs: A Review and Recommendations for Future Research	Jerry West
1999–01	A Birth Cohort Study: Conceptual and Design Considerations and Rationale	Jerry West
2000–04	Selected Papers on Education Surveys: Papers Presented at the 1998 and 1999 ASA and 1999 AAPOR Meetings	Dan Kasprzyk
2001–02	Measuring Father Involvement in Young Children's Lives: Recommendations for a Fatherhood Module for the ECLS-B	Jerry West
2001–03	Measures of Socio-Emotional Development in Middle Childhood	Elvira Hausken

No.	Title	NCES contact
2001–06	Papers from the Early Childhood Longitudinal Studies Program: Presented at the 2001 AERA and SRCD Meetings	Jerry West
Education Finance Statistics Center (EDFIN)		
94–05	Cost-of-Education Differentials Across the States	William J. Fowler, Jr.
96–19	Assessment and Analysis of School-Level Expenditures	William J. Fowler, Jr.
97–43	Measuring Inflation in Public School Costs	William J. Fowler, Jr.
98–04	Geographic Variations in Public Schools' Costs	William J. Fowler, Jr.
1999–16	Measuring Resources in Education: From Accounting to the Resource Cost Model Approach	William J. Fowler, Jr.
High School and Beyond (HS&B)		
95–12	Rural Education Data User's Guide	Samuel Peng
1999–05	Procedures Guide for Transcript Studies	Dawn Nelson
1999–06	1998 Revision of the Secondary School Taxonomy	Dawn Nelson
HS Transcript Studies		
1999–05	Procedures Guide for Transcript Studies	Dawn Nelson
1999–06	1998 Revision of the Secondary School Taxonomy	Dawn Nelson
International Adult Literacy Survey (IALS)		
97–33	Adult Literacy: An International Perspective	Marilyn Binkley
Integrated Postsecondary Education Data System (IPEDS)		
97–27	Pilot Test of IPEDS Finance Survey	Peter Stowe
98–15	Development of a Prototype System for Accessing Linked NCES Data	Steven Kaufman
2000–14	IPEDS Finance Data Comparisons Under the 1997 Financial Accounting Standards for Private, Not-for-Profit Institutes: A Concept Paper	Peter Stowe
National Assessment of Adult Literacy (NAAL)		
98–17	Developing the National Assessment of Adult Literacy: Recommendations from Stakeholders	Sheida White
1999–09a	1992 National Adult Literacy Survey: An Overview	Alex Sedlacek
1999–09b	1992 National Adult Literacy Survey: Sample Design	Alex Sedlacek
1999–09c	1992 National Adult Literacy Survey: Weighting and Population Estimates	Alex Sedlacek
1999–09d	1992 National Adult Literacy Survey: Development of the Survey Instruments	Alex Sedlacek
1999–09e	1992 National Adult Literacy Survey: Scaling and Proficiency Estimates	Alex Sedlacek
1999–09f	1992 National Adult Literacy Survey: Interpreting the Adult Literacy Scales and Literacy Levels	Alex Sedlacek
1999–09g	1992 National Adult Literacy Survey: Literacy Levels and the Response Probability Convention	Alex Sedlacek
2000–05	Secondary Statistical Modeling With the National Assessment of Adult Literacy: Implications for the Design of the Background Questionnaire	Sheida White
2000–06	Using Telephone and Mail Surveys as a Supplement or Alternative to Door-to-Door Surveys in the Assessment of Adult Literacy	Sheida White
2000–07	"How Much Literacy is Enough?" Issues in Defining and Reporting Performance Standards for the National Assessment of Adult Literacy	Sheida White
2000–08	Evaluation of the 1992 NALS Background Survey Questionnaire: An Analysis of Uses with Recommendations for Revisions	Sheida White
2000–09	Demographic Changes and Literacy Development in a Decade	Sheida White
2001–08	Assessing the Lexile Framework: Results of a Panel Meeting	Sheida White
National Assessment of Educational Progress (NAEP)		
95–12	Rural Education Data User's Guide	Samuel Peng
97–29	Can State Assessment Data be Used to Reduce State NAEP Sample Sizes?	Steven Gorman
97–30	ACT's NAEP Redesign Project: Assessment Design is the Key to Useful and Stable Assessment Results	Steven Gorman

No.	Title	NCES contact
97–31	NAEP Reconfigured: An Integrated Redesign of the National Assessment of Educational Progress	Steven Gorman
97–32	Innovative Solutions to Intractable Large Scale Assessment (Problem 2: Background Questionnaires)	Steven Gorman
97–37	Optimal Rating Procedures and Methodology for NAEP Open-ended Items	Steven Gorman
97–44	Development of a SASS 1993–94 School-Level Student Achievement Subfile: Using State Assessments and State NAEP, Feasibility Study	Michael Ross
98–15	Development of a Prototype System for Accessing Linked NCES Data	Steven Kaufman
1999–05	Procedures Guide for Transcript Studies	Dawn Nelson
1999–06	1998 Revision of the Secondary School Taxonomy	Dawn Nelson
2001–07	A Comparison of the National Assessment of Educational Progress (NAEP), the Third International Mathematics and Science Study Repeat (TIMSS-R), and the Programme for International Student Assessment (PISA)	Arnold Goldstein
2001–08	Assessing the Lexile Framework: Results of a Panel Meeting	Sheida White
National Education Longitudinal Study of 1988 (NELS:88)		
95–04	National Education Longitudinal Study of 1988: Second Follow-up Questionnaire Content Areas and Research Issues	Jeffrey Owings
95–05	National Education Longitudinal Study of 1988: Conducting Trend Analyses of NLS-72, HS&B, and NELS:88 Seniors	Jeffrey Owings
95–06	National Education Longitudinal Study of 1988: Conducting Cross-Cohort Comparisons Using HS&B, NAEP, and NELS:88 Academic Transcript Data	Jeffrey Owings
95–07	National Education Longitudinal Study of 1988: Conducting Trend Analyses HS&B and NELS:88 Sophomore Cohort Dropouts	Jeffrey Owings
95–12	Rural Education Data User's Guide	Samuel Peng
95–14	Empirical Evaluation of Social, Psychological, & Educational Construct Variables Used in NCES Surveys	Samuel Peng
96–03	National Education Longitudinal Study of 1988 (NELS:88) Research Framework and Issues	Jeffrey Owings
98–06	National Education Longitudinal Study of 1988 (NELS:88) Base Year through Second Follow-Up: Final Methodology Report	Ralph Lee
98–09	High School Curriculum Structure: Effects on Coursetaking and Achievement in Mathematics for High School Graduates—An Examination of Data from the National Education Longitudinal Study of 1988	Jeffrey Owings
98–15	Development of a Prototype System for Accessing Linked NCES Data	Steven Kaufman
1999–05	Procedures Guide for Transcript Studies	Dawn Nelson
1999–06	1998 Revision of the Secondary School Taxonomy	Dawn Nelson
1999–15	Projected Postsecondary Outcomes of 1992 High School Graduates	Aurora D'Amico
National Household Education Survey (NHES)		
95–12	Rural Education Data User's Guide	Samuel Peng
96–13	Estimation of Response Bias in the NHES:95 Adult Education Survey	Steven Kaufman
96–14	The 1995 National Household Education Survey: Reinterview Results for the Adult Education Component	Steven Kaufman
96–20	1991 National Household Education Survey (NHES:91) Questionnaires: Screener, Early Childhood Education, and Adult Education	Kathryn Chandler
96–21	1993 National Household Education Survey (NHES:93) Questionnaires: Screener, School Readiness, and School Safety and Discipline	Kathryn Chandler
96–22	1995 National Household Education Survey (NHES:95) Questionnaires: Screener, Early Childhood Program Participation, and Adult Education	Kathryn Chandler
96–29	Undercoverage Bias in Estimates of Characteristics of Adults and 0- to 2-Year-Olds in the 1995 National Household Education Survey (NHES:95)	Kathryn Chandler
96–30	Comparison of Estimates from the 1995 National Household Education Survey (NHES:95)	Kathryn Chandler
97–02	Telephone Coverage Bias and Recorded Interviews in the 1993 National Household Education Survey (NHES:93)	Kathryn Chandler
97–03	1991 and 1995 National Household Education Survey Questionnaires: NHES:91 Screener, NHES:91 Adult Education, NHES:95 Basic Screener, and NHES:95 Adult Education	Kathryn Chandler
97–04	Design, Data Collection, Monitoring, Interview Administration Time, and Data Editing in the 1993 National Household Education Survey (NHES:93)	Kathryn Chandler

No.	Title	NCES contact
97–05	Unit and Item Response, Weighting, and Imputation Procedures in the 1993 National Household Education Survey (NHES:93)	Kathryn Chandler
97–06	Unit and Item Response, Weighting, and Imputation Procedures in the 1995 National Household Education Survey (NHES:95)	Kathryn Chandler
97–08	Design, Data Collection, Interview Timing, and Data Editing in the 1995 National Household Education Survey	Kathryn Chandler
97–19	National Household Education Survey of 1995: Adult Education Course Coding Manual	Peter Stowe
97–20	National Household Education Survey of 1995: Adult Education Course Code Merge Files User's Guide	Peter Stowe
97–25	1996 National Household Education Survey (NHES:96) Questionnaires: Screener/Household and Library, Parent and Family Involvement in Education and Civic Involvement, Youth Civic Involvement, and Adult Civic Involvement	Kathryn Chandler
97–28	Comparison of Estimates in the 1996 National Household Education Survey	Kathryn Chandler
97–34	Comparison of Estimates from the 1993 National Household Education Survey	Kathryn Chandler
97–35	Design, Data Collection, Interview Administration Time, and Data Editing in the 1996 National Household Education Survey	Kathryn Chandler
97–38	Reinterview Results for the Parent and Youth Components of the 1996 National Household Education Survey	Kathryn Chandler
97–39	Undercoverage Bias in Estimates of Characteristics of Households and Adults in the 1996 National Household Education Survey	Kathryn Chandler
97–40	Unit and Item Response Rates, Weighting, and Imputation Procedures in the 1996 National Household Education Survey	Kathryn Chandler
98–03	Adult Education in the 1990s: A Report on the 1991 National Household Education Survey	Peter Stowe
98–10	Adult Education Participation Decisions and Barriers: Review of Conceptual Frameworks and Empirical Studies	Peter Stowe
National Longitudinal Study of the High School Class of 1972 (NLS-72)		
95–12	Rural Education Data User's Guide	Samuel Peng
National Postsecondary Student Aid Study (NPSAS)		
96–17	National Postsecondary Student Aid Study: 1996 Field Test Methodology Report	Andrew G. Malizio
2000–17	National Postsecondary Student Aid Study:2000 Field Test Methodology Report	Andrew G. Malizio
National Study of Postsecondary Faculty (NSOPF)		
97–26	Strategies for Improving Accuracy of Postsecondary Faculty Lists	Linda Zimbler
98–15	Development of a Prototype System for Accessing Linked NCES Data	Steven Kaufman
2000–01	1999 National Study of Postsecondary Faculty (NSOPF:99) Field Test Report	Linda Zimbler
Postsecondary Education Descriptive Analysis Reports (PEDAR)		
2000–11	Financial Aid Profile of Graduate Students in Science and Engineering	Aurora D'Amico
Private School Universe Survey (PSS)		
95–16	Intersurvey Consistency in NCES Private School Surveys	Steven Kaufman
95–17	Estimates of Expenditures for Private K–12 Schools	Stephen Broughman
96–16	Strategies for Collecting Finance Data from Private Schools	Stephen Broughman
96–26	Improving the Coverage of Private Elementary-Secondary Schools	Steven Kaufman
96–27	Intersurvey Consistency in NCES Private School Surveys for 1993–94	Steven Kaufman
97–07	The Determinants of Per-Pupil Expenditures in Private Elementary and Secondary Schools: An Exploratory Analysis	Stephen Broughman
97–22	Collection of Private School Finance Data: Development of a Questionnaire	Stephen Broughman
98–15	Development of a Prototype System for Accessing Linked NCES Data	Steven Kaufman
2000–04	Selected Papers on Education Surveys: Papers Presented at the 1998 and 1999 ASA and 1999 AAPOR Meetings	Dan Kasprzyk
2000–15	Feasibility Report: School-Level Finance Pretest, Private School Questionnaire	Stephen Broughman

No.	Title	NCES contact
Recent College Graduates (RCG)		
98–15	Development of a Prototype System for Accessing Linked NCES Data	Steven Kaufman
Schools and Staffing Survey (SASS)		
94–01	Schools and Staffing Survey (SASS) Papers Presented at Meetings of the American Statistical Association	Dan Kasprzyk
94–02	Generalized Variance Estimate for Schools and Staffing Survey (SASS)	Dan Kasprzyk
94–03	1991 Schools and Staffing Survey (SASS) Reinterview Response Variance Report	Dan Kasprzyk
94–04	The Accuracy of Teachers' Self-reports on their Postsecondary Education: Teacher Transcript Study, Schools and Staffing Survey	Dan Kasprzyk
94–06	Six Papers on Teachers from the 1990–91 Schools and Staffing Survey and Other Related Surveys	Dan Kasprzyk
95–01	Schools and Staffing Survey: 1994 Papers Presented at the 1994 Meeting of the American Statistical Association	Dan Kasprzyk
95–02	QED Estimates of the 1990–91 Schools and Staffing Survey: Deriving and Comparing QED School Estimates with CCD Estimates	Dan Kasprzyk
95–03	Schools and Staffing Survey: 1990–91 SASS Cross-Questionnaire Analysis	Dan Kasprzyk
95–08	CCD Adjustment to the 1990–91 SASS: A Comparison of Estimates	Dan Kasprzyk
95–09	The Results of the 1993 Teacher List Validation Study (TLVS)	Dan Kasprzyk
95–10	The Results of the 1991–92 Teacher Follow-up Survey (TFS) Reinterview and Extensive Reconciliation	Dan Kasprzyk
95–11	Measuring Instruction, Curriculum Content, and Instructional Resources: The Status of Recent Work	Sharon Bobbitt & John Ralph
95–12	Rural Education Data User's Guide	Samuel Peng
95–14	Empirical Evaluation of Social, Psychological, & Educational Construct Variables Used in NCES Surveys	Samuel Peng
95–15	Classroom Instructional Processes: A Review of Existing Measurement Approaches and Their Applicability for the Teacher Follow-up Survey	Sharon Bobbitt
95–16	Intersurvey Consistency in NCES Private School Surveys	Steven Kaufman
95–18	An Agenda for Research on Teachers and Schools: Revisiting NCES' Schools and Staffing Survey	Dan Kasprzyk
96–01	Methodological Issues in the Study of Teachers' Careers: Critical Features of a Truly Longitudinal Study	Dan Kasprzyk
96–02	Schools and Staffing Survey (SASS): 1995 Selected papers presented at the 1995 Meeting of the American Statistical Association	Dan Kasprzyk
96–05	Cognitive Research on the Teacher Listing Form for the Schools and Staffing Survey	Dan Kasprzyk
96–06	The Schools and Staffing Survey (SASS) for 1998–99: Design Recommendations to Inform Broad Education Policy	Dan Kasprzyk
96–07	Should SASS Measure Instructional Processes and Teacher Effectiveness?	Dan Kasprzyk
96–09	Making Data Relevant for Policy Discussions: Redesigning the School Administrator Questionnaire for the 1998–99 SASS	Dan Kasprzyk
96–10	1998–99 Schools and Staffing Survey: Issues Related to Survey Depth	Dan Kasprzyk
96–11	Towards an Organizational Database on America's Schools: A Proposal for the Future of SASS, with comments on School Reform, Governance, and Finance	Dan Kasprzyk
96–12	Predictors of Retention, Transfer, and Attrition of Special and General Education Teachers: Data from the 1989 Teacher Followup Survey	Dan Kasprzyk
96–15	Nested Structures: District-Level Data in the Schools and Staffing Survey	Dan Kasprzyk
96–23	Linking Student Data to SASS: Why, When, How	Dan Kasprzyk
96–24	National Assessments of Teacher Quality	Dan Kasprzyk
96–25	Measures of Inservice Professional Development: Suggested Items for the 1998–1999 Schools and Staffing Survey	Dan Kasprzyk
96–28	Student Learning, Teaching Quality, and Professional Development: Theoretical Linkages, Current Measurement, and Recommendations for Future Data Collection	Mary Rollefson
97–01	Selected Papers on Education Surveys: Papers Presented at the 1996 Meeting of the American Statistical Association	Dan Kasprzyk
97–07	The Determinants of Per-Pupil Expenditures in Private Elementary and Secondary Schools: An Exploratory Analysis	Stephen Broughman
97–09	Status of Data on Crime and Violence in Schools: Final Report	Lee Hoffman
97–10	Report of Cognitive Research on the Public and Private School Teacher Questionnaires for the Schools and Staffing Survey 1993–94 School Year	Dan Kasprzyk

No.	Title	NCES contact
97–11	International Comparisons of Inservice Professional Development	Dan Kasprzyk
97–12	Measuring School Reform: Recommendations for Future SASS Data Collection	Mary Rollefson
97–14	Optimal Choice of Periodicities for the Schools and Staffing Survey: Modeling and Analysis	Steven Kaufman
97–18	Improving the Mail Return Rates of SASS Surveys: A Review of the Literature	Steven Kaufman
97–22	Collection of Private School Finance Data: Development of a Questionnaire	Stephen Broughman
97–23	Further Cognitive Research on the Schools and Staffing Survey (SASS) Teacher Listing Form	Dan Kasprzyk
97–41	Selected Papers on the Schools and Staffing Survey: Papers Presented at the 1997 Meeting of the American Statistical Association	Steve Kaufman
97–42	Improving the Measurement of Staffing Resources at the School Level: The Development of Recommendations for NCES for the Schools and Staffing Survey (SASS)	Mary Rollefson
97–44	Development of a SASS 1993–94 School-Level Student Achievement Subfile: Using State Assessments and State NAEP, Feasibility Study	Michael Ross
98–01	Collection of Public School Expenditure Data: Development of a Questionnaire	Stephen Broughman
98–02	Response Variance in the 1993–94 Schools and Staffing Survey: A Reinterview Report	Steven Kaufman
98–04	Geographic Variations in Public Schools' Costs	William J. Fowler, Jr.
98–05	SASS Documentation: 1993–94 SASS Student Sampling Problems; Solutions for Determining the Numerators for the SASS Private School (3B) Second-Stage Factors	Steven Kaufman
98–08	The Redesign of the Schools and Staffing Survey for 1999–2000: A Position Paper	Dan Kasprzyk
98–12	A Bootstrap Variance Estimator for Systematic PPS Sampling	Steven Kaufman
98–13	Response Variance in the 1994–95 Teacher Follow-up Survey	Steven Kaufman
98–14	Variance Estimation of Imputed Survey Data	Steven Kaufman
98–15	Development of a Prototype System for Accessing Linked NCES Data	Steven Kaufman
98–16	A Feasibility Study of Longitudinal Design for Schools and Staffing Survey	Stephen Broughman
1999–02	Tracking Secondary Use of the Schools and Staffing Survey Data: Preliminary Results	Dan Kasprzyk
1999–04	Measuring Teacher Qualifications	Dan Kasprzyk
1999–07	Collection of Resource and Expenditure Data on the Schools and Staffing Survey	Stephen Broughman
1999–08	Measuring Classroom Instructional Processes: Using Survey and Case Study Fieldtest Results to Improve Item Construction	Dan Kasprzyk
1999–10	What Users Say About Schools and Staffing Survey Publications	Dan Kasprzyk
1999–12	1993–94 Schools and Staffing Survey: Data File User's Manual, Volume III: Public-Use Codebook	Kerry Gruber
1999–13	1993–94 Schools and Staffing Survey: Data File User's Manual, Volume IV: Bureau of Indian Affairs (BIA) Restricted-Use Codebook	Kerry Gruber
1999–14	1994–95 Teacher Followup Survey: Data File User's Manual, Restricted-Use Codebook	Kerry Gruber
1999–17	Secondary Use of the Schools and Staffing Survey Data	Susan Wiley
2000–04	Selected Papers on Education Surveys: Papers Presented at the 1998 and 1999 ASA and 1999 AAPOR Meetings	Dan Kasprzyk
2000–10	A Research Agenda for the 1999–2000 Schools and Staffing Survey	Dan Kasprzyk
2000–13	Non-professional Staff in the Schools and Staffing Survey (SASS) and Common Core of Data (CCD)	Kerry Gruber
2000–18	Feasibility Report: School-Level Finance Pretest, Public School District Questionnaire	Stephen Broughman
Third International Mathematics and Science Study (TIMSS)		
2001–01	Cross-National Variation in Educational Preparation for Adulthood: From Early Adolescence to Young Adulthood	Elvira Hausken
2001–05	Using TIMSS to Analyze Correlates of Performance Variation in Mathematics	Patrick Gonzales
2001–07	A Comparison of the National Assessment of Educational Progress (NAEP), the Third International Mathematics and Science Study Repeat (TIMSS-R), and the Programme for International Student Assessment (PISA)	Arnold Goldstein

Listing of NCES Working Papers by Subject

No.	Title	NCES contact
Achievement (student) - mathematics		
2001–05	Using TIMSS to Analyze Correlates of Performance Variation in Mathematics	Patrick Gonzales
Adult education		
96–14	The 1995 National Household Education Survey: Reinterview Results for the Adult Education Component	Steven Kaufman
96–20	1991 National Household Education Survey (NHES:91) Questionnaires: Screener, Early Childhood Education, and Adult Education	Kathryn Chandler
96–22	1995 National Household Education Survey (NHES:95) Questionnaires: Screener, Early Childhood Program Participation, and Adult Education	Kathryn Chandler
98–03	Adult Education in the 1990s: A Report on the 1991 National Household Education Survey	Peter Stowe
98–10	Adult Education Participation Decisions and Barriers: Review of Conceptual Frameworks and Empirical Studies	Peter Stowe
1999–11	Data Sources on Lifelong Learning Available from the National Center for Education Statistics	Lisa Hudson
2000–16a	Lifelong Learning NCES Task Force: Final Report Volume I	Lisa Hudson
2000–16b	Lifelong Learning NCES Task Force: Final Report Volume II	Lisa Hudson
Adult literacy—see Literacy of adults		
American Indian – education		
1999–13	1993–94 Schools and Staffing Survey: Data File User's Manual, Volume IV: Bureau of Indian Affairs (BIA) Restricted-Use Codebook	Kerry Gruber
Assessment/achievement		
95–12	Rural Education Data User's Guide	Samuel Peng
95–13	Assessing Students with Disabilities and Limited English Proficiency	James Houser
97–29	Can State Assessment Data be Used to Reduce State NAEP Sample Sizes?	Larry Ogle
97–30	ACT's NAEP Redesign Project: Assessment Design is the Key to Useful and Stable Assessment Results	Larry Ogle
97–31	NAEP Reconfigured: An Integrated Redesign of the National Assessment of Educational Progress	Larry Ogle
97–32	Innovative Solutions to Intractable Large Scale Assessment (Problem 2: Background Questions)	Larry Ogle
97–37	Optimal Rating Procedures and Methodology for NAEP Open-ended Items	Larry Ogle
97–44	Development of a SASS 1993–94 School-Level Student Achievement Subfile: Using State Assessments and State NAEP, Feasibility Study	Michael Ross
98–09	High School Curriculum Structure: Effects on Coursetaking and Achievement in Mathematics for High School Graduates—An Examination of Data from the National Education Longitudinal Study of 1988	Jeffrey Owings
2001–07	A Comparison of the National Assessment of Educational Progress (NAEP), the Third International Mathematics and Science Study Repeat (TIMSS-R), and the Programme for International Student Assessment (PISA)	Arnold Goldstein
Beginning students in postsecondary education		
98–11	Beginning Postsecondary Students Longitudinal Study First Follow-up (BPS:96–98) Field Test Report	Aurora D'Amico
2001–04	Beginning Postsecondary Students Longitudinal Study: 1996–2001 (BPS:1996/2001) Field Test Methodology Report	Paula Knepper
Civic participation		
97–25	1996 National Household Education Survey (NHES:96) Questionnaires: Screener/Household and Library, Parent and Family Involvement in Education and Civic Involvement, Youth Civic Involvement, and Adult Civic Involvement	Kathryn Chandler

No.	Title	NCES contact
Climate of schools		
95–14	Empirical Evaluation of Social, Psychological, & Educational Construct Variables Used in NCES Surveys	Samuel Peng
Cost of education indices		
94–05	Cost-of-Education Differentials Across the States	William J. Fowler, Jr.
Course-taking		
95–12	Rural Education Data User's Guide	Samuel Peng
98–09	High School Curriculum Structure: Effects on Coursetaking and Achievement in Mathematics for High School Graduates—An Examination of Data from the National Education Longitudinal Study of 1988	Jeffrey Owings
1999–05	Procedures Guide for Transcript Studies	Dawn Nelson
1999–06	1998 Revision of the Secondary School Taxonomy	Dawn Nelson
Crime		
97–09	Status of Data on Crime and Violence in Schools: Final Report	Lee Hoffman
Curriculum		
95–11	Measuring Instruction, Curriculum Content, and Instructional Resources: The Status of Recent Work	Sharon Bobbitt & John Ralph
98–09	High School Curriculum Structure: Effects on Coursetaking and Achievement in Mathematics for High School Graduates—An Examination of Data from the National Education Longitudinal Study of 1988	Jeffrey Owings
Customer service		
1999–10	What Users Say About Schools and Staffing Survey Publications	Dan Kasprzyk
2000–02	Coordinating NCES Surveys: Options, Issues, Challenges, and Next Steps	Valena Plisko
2000–04	Selected Papers on Education Surveys: Papers Presented at the 1998 and 1999 ASA and 1999 AAPOR Meetings	Dan Kasprzyk
Data quality		
97–13	Improving Data Quality in NCES: Database-to-Report Process	Susan Ahmed
Data warehouse		
2000–04	Selected Papers on Education Surveys: Papers Presented at the 1998 and 1999 ASA and 1999 AAPOR Meetings	Dan Kasprzyk
Design effects		
2000–03	Strengths and Limitations of Using SUDAAN, Stata, and WesVarPC for Computing Variances from NCES Data Sets	Ralph Lee
Dropout rates, high school		
95–07	National Education Longitudinal Study of 1988: Conducting Trend Analyses HS&B and NELS:88 Sophomore Cohort Dropouts	Jeffrey Owings
Early childhood education		
96–20	1991 National Household Education Survey (NHES:91) Questionnaires: Screener, Early Childhood Education, and Adult Education	Kathryn Chandler
96–22	1995 National Household Education Survey (NHES:95) Questionnaires: Screener, Early Childhood Program Participation, and Adult Education	Kathryn Chandler
97–24	Formulating a Design for the ECLS: A Review of Longitudinal Studies	Jerry West
97–36	Measuring the Quality of Program Environments in Head Start and Other Early Childhood Programs: A Review and Recommendations for Future Research	Jerry West
1999–01	A Birth Cohort Study: Conceptual and Design Considerations and Rationale	Jerry West
2001–02	Measuring Father Involvement in Young Children's Lives: Recommendations for a Fatherhood Module for the ECLS-B	Jerry West
2001–03	Measures of Socio-Emotional Development in Middle School	Elvira Hausken

No.	Title	NCES contact
2001–06	Papers from the Early Childhood Longitudinal Studies Program: Presented at the 2001 AERA and SRCD Meetings	Jerry West
Educational attainment		
98–11	Beginning Postsecondary Students Longitudinal Study First Follow-up (BPS:96–98) Field Test Report	Aurora D'Amico
Educational research		
2000–02	Coordinating NCES Surveys: Options, Issues, Challenges, and Next Steps	Valena Plisko
Eighth-graders		
2001–05	Using TIMSS to Analyze Correlates of Performance Variation in Mathematics	Patrick Gonzales
Employment		
96–03	National Education Longitudinal Study of 1988 (NELS:88) Research Framework and Issues	Jeffrey Owings
98–11	Beginning Postsecondary Students Longitudinal Study First Follow-up (BPS:96–98) Field Test Report	Aurora D'Amico
2000–16a	Lifelong Learning NCES Task Force: Final Report Volume I	Lisa Hudson
2000–16b	Lifelong Learning NCES Task Force: Final Report Volume II	Lisa Hudson
2001–01	Cross-National Variation in Educational Preparation for Adulthood: From Early Adolescence to Young Adulthood	Elvira Hausken
Engineering		
2000–11	Financial Aid Profile of Graduate Students in Science and Engineering	Aurora D'Amico
Faculty – higher education		
97–26	Strategies for Improving Accuracy of Postsecondary Faculty Lists	Linda Zimbler
2000–01	1999 National Study of Postsecondary Faculty (NSOPF:99) Field Test Report	Linda Zimbler
Fathers – role in education		
2001–02	Measuring Father Involvement in Young Children's Lives: Recommendations for a Fatherhood Module for the ECLS-B	Jerry West
Finance – elementary and secondary schools		
94–05	Cost-of-Education Differentials Across the States	William J. Fowler, Jr.
96–19	Assessment and Analysis of School-Level Expenditures	William J. Fowler, Jr.
98–01	Collection of Public School Expenditure Data: Development of a Questionnaire	Stephen Broughman
1999–07	Collection of Resource and Expenditure Data on the Schools and Staffing Survey	Stephen Broughman
1999–16	Measuring Resources in Education: From Accounting to the Resource Cost Model Approach	William J. Fowler, Jr.
2000–18	Feasibility Report: School-Level Finance Pretest, Public School District Questionnaire	Stephen Broughman
Finance – postsecondary		
97–27	Pilot Test of IPEDS Finance Survey	Peter Stowe
2000–14	IPEDS Finance Data Comparisons Under the 1997 Financial Accounting Standards for Private, Not-for-Profit Institutes: A Concept Paper	Peter Stowe
Finance – private schools		
95–17	Estimates of Expenditures for Private K–12 Schools	Stephen Broughman
96–16	Strategies for Collecting Finance Data from Private Schools	Stephen Broughman
97–07	The Determinants of Per-Pupil Expenditures in Private Elementary and Secondary Schools: An Exploratory Analysis	Stephen Broughman
97–22	Collection of Private School Finance Data: Development of a Questionnaire	Stephen Broughman
1999–07	Collection of Resource and Expenditure Data on the Schools and Staffing Survey	Stephen Broughman
2000–15	Feasibility Report: School-Level Finance Pretest, Private School Questionnaire	Stephen Broughman

No.	Title	NCES contact
Geography		
98–04	Geographic Variations in Public Schools' Costs	William J. Fowler, Jr.
Graduate students		
2000–11	Financial Aid Profile of Graduate Students in Science and Engineering	Aurora D'Amico
Imputation		
2000–04	Selected Papers on Education Surveys: Papers Presented at the 1998 and 1999 ASA and 1999 AAPOR Meeting	Dan Kasprzyk
2001–10	Comparison of Proc Impute and Schafer's Multiple Imputation Software	Sam Peng
Inflation		
97–43	Measuring Inflation in Public School Costs	William J. Fowler, Jr.
Institution data		
2000–01	1999 National Study of Postsecondary Faculty (NSOPF:99) Field Test Report	Linda Zimbler
Instructional resources and practices		
95–11	Measuring Instruction, Curriculum Content, and Instructional Resources: The Status of Recent Work	Sharon Bobbitt & John Ralph
1999–08	Measuring Classroom Instructional Processes: Using Survey and Case Study Field Test Results to Improve Item Construction	Dan Kasprzyk
International comparisons		
97–11	International Comparisons of Inservice Professional Development	Dan Kasprzyk
97–16	International Education Expenditure Comparability Study: Final Report, Volume I	Shelley Burns
97–17	International Education Expenditure Comparability Study: Final Report, Volume II, Quantitative Analysis of Expenditure Comparability	Shelley Burns
2001–01	Cross-National Variation in Educational Preparation for Adulthood: From Early Adolescence to Young Adulthood	Elvira Hausken
2001–07	A Comparison of the National Assessment of Educational Progress (NAEP), the Third International Mathematics and Science Study Repeat (TIMSS-R), and the Programme for International Student Assessment (PISA)	Arnold Goldstein
International comparisons – math and science achievement		
2001–05	Using TIMSS to Analyze Correlates of Performance Variation in Mathematics	Patrick Gonzales
Libraries		
94–07	Data Comparability and Public Policy: New Interest in Public Library Data Papers Presented at Meetings of the American Statistical Association	Carrol Kindel
97–25	1996 National Household Education Survey (NHES:96) Questionnaires: Screener/Household and Library, Parent and Family Involvement in Education and Civic Involvement, Youth Civic Involvement, and Adult Civic Involvement	Kathryn Chandler
Limited English Proficiency		
95–13	Assessing Students with Disabilities and Limited English Proficiency	James Houser
Literacy of adults		
98–17	Developing the National Assessment of Adult Literacy: Recommendations from Stakeholders	Sheida White
1999–09a	1992 National Adult Literacy Survey: An Overview	Alex Sedlacek
1999–09b	1992 National Adult Literacy Survey: Sample Design	Alex Sedlacek
1999–09c	1992 National Adult Literacy Survey: Weighting and Population Estimates	Alex Sedlacek
1999–09d	1992 National Adult Literacy Survey: Development of the Survey Instruments	Alex Sedlacek
1999–09e	1992 National Adult Literacy Survey: Scaling and Proficiency Estimates	Alex Sedlacek
1999–09f	1992 National Adult Literacy Survey: Interpreting the Adult Literacy Scales and Literacy Levels	Alex Sedlacek

No.	Title	NCES contact
1999–09g	1992 National Adult Literacy Survey: Literacy Levels and the Response Probability Convention	Alex Sedlacek
1999–11	Data Sources on Lifelong Learning Available from the National Center for Education Statistics	Lisa Hudson
2000–05	Secondary Statistical Modeling With the National Assessment of Adult Literacy: Implications for the Design of the Background Questionnaire	Sheida White
2000–06	Using Telephone and Mail Surveys as a Supplement or Alternative to Door-to-Door Surveys in the Assessment of Adult Literacy	Sheida White
2000–07	"How Much Literacy is Enough?" Issues in Defining and Reporting Performance Standards for the National Assessment of Adult Literacy	Sheida White
2000–08	Evaluation of the 1992 NALS Background Survey Questionnaire: An Analysis of Uses with Recommendations for Revisions	Sheida White
2000–09	Demographic Changes and Literacy Development in a Decade	Sheida White
2001–08	Assessing the Lexile Framework: Results of a Panel Meeting	Sheida White
Literacy of adults – international		
97–33	Adult Literacy: An International Perspective	Marilyn Binkley
Mathematics		
98–09	High School Curriculum Structure: Effects on Coursetaking and Achievement in Mathematics for High School Graduates—An Examination of Data from the National Education Longitudinal Study of 1988	Jeffrey Owings
1999–08	Measuring Classroom Instructional Processes: Using Survey and Case Study Field Test Results to Improve Item Construction	Dan Kasprzyk
2001–05	Using TIMSS to Analyze Correlates of Performance Variation in Mathematics	Patrick Gonzales
2001–07	A Comparison of the National Assessment of Educational Progress (NAEP), the Third International Mathematics and Science Study Repeat (TIMSS-R), and the Programme for International Student Assessment (PISA)	Arnold Goldstein
Parental involvement in education		
96–03	National Education Longitudinal Study of 1988 (NELS:88) Research Framework and Issues	Jeffrey Owings
97–25	1996 National Household Education Survey (NHES:96) Questionnaires: Screener/Household and Library, Parent and Family Involvement in Education and Civic Involvement, Youth Civic Involvement, and Adult Civic Involvement	Kathryn Chandler
1999–01	A Birth Cohort Study: Conceptual and Design Considerations and Rationale	Jerry West
2001–06	Papers from the Early Childhood Longitudinal Studies Program: Presented at the 2001 AERA and SRCD Meetings	Jerry West
Participation rates		
98–10	Adult Education Participation Decisions and Barriers: Review of Conceptual Frameworks and Empirical Studies	Peter Stowe
Postsecondary education		
1999–11	Data Sources on Lifelong Learning Available from the National Center for Education Statistics	Lisa Hudson
2000–16a	Lifelong Learning NCES Task Force: Final Report Volume I	Lisa Hudson
2000–16b	Lifelong Learning NCES Task Force: Final Report Volume II	Lisa Hudson
Postsecondary education – persistence and attainment		
98–11	Beginning Postsecondary Students Longitudinal Study First Follow-up (BPS:96–98) Field Test Report	Aurora D'Amico
1999–15	Projected Postsecondary Outcomes of 1992 High School Graduates	Aurora D'Amico
Postsecondary education – staff		
97–26	Strategies for Improving Accuracy of Postsecondary Faculty Lists	Linda Zimbler
2000–01	1999 National Study of Postsecondary Faculty (NSOPF:99) Field Test Report	Linda Zimbler

No.	Title	NCES contact
Principals		
2000–10	A Research Agenda for the 1999–2000 Schools and Staffing Survey	Dan Kasprzyk
Private schools		
96–16	Strategies for Collecting Finance Data from Private Schools	Stephen Broughman
97–07	The Determinants of Per-Pupil Expenditures in Private Elementary and Secondary Schools: An Exploratory Analysis	Stephen Broughman
97–22	Collection of Private School Finance Data: Development of a Questionnaire	Stephen Broughman
2000–13	Non-professional Staff in the Schools and Staffing Survey (SASS) and Common Core of Data (CCD)	Kerry Gruber
2000–15	Feasibility Report: School-Level Finance Pretest, Private School Questionnaire	Stephen Broughman
Projections of education statistics		
1999–15	Projected Postsecondary Outcomes of 1992 High School Graduates	Aurora D'Amico
Public school finance		
1999–16	Measuring Resources in Education: From Accounting to the Resource Cost Model Approach	William J. Fowler, Jr.
2000–18	Feasibility Report: School-Level Finance Pretest, Public School District Questionnaire	Stephen Broughman
Public schools		
97–43	Measuring Inflation in Public School Costs	William J. Fowler, Jr.
98–01	Collection of Public School Expenditure Data: Development of a Questionnaire	Stephen Broughman
98–04	Geographic Variations in Public Schools' Costs	William J. Fowler, Jr.
1999–02	Tracking Secondary Use of the Schools and Staffing Survey Data: Preliminary Results	Dan Kasprzyk
2000–12	Coverage Evaluation of the 1994–95 Public Elementary/Secondary School Universe Survey	Beth Young
2000–13	Non-professional Staff in the Schools and Staffing Survey (SASS) and Common Core of Data (CCD)	Kerry Gruber
Public schools – secondary		
98–09	High School Curriculum Structure: Effects on Coursetaking and Achievement in Mathematics for High School Graduates—An Examination of Data from the National Education Longitudinal Study of 1988	Jeffrey Owings
Reform, educational		
96–03	National Education Longitudinal Study of 1988 (NELS:88) Research Framework and Issues	Jeffrey Owings
Response rates		
98–02	Response Variance in the 1993–94 Schools and Staffing Survey: A Reinterview Report	Steven Kaufman
School districts		
2000–10	A Research Agenda for the 1999–2000 Schools and Staffing Survey	Dan Kasprzyk
School districts, public		
98–07	Decennial Census School District Project Planning Report	Tai Phan
1999–03	Evaluation of the 1996–97 Nonfiscal Common Core of Data Surveys Data Collection, Processing, and Editing Cycle	Beth Young
School districts, public – demographics of		
96–04	Census Mapping Project/School District Data Book	Tai Phan
Schools		
97–42	Improving the Measurement of Staffing Resources at the School Level: The Development of Recommendations for NCES for the Schools and Staffing Survey (SASS)	Mary Rollefson
98–08	The Redesign of the Schools and Staffing Survey for 1999–2000: A Position Paper	Dan Kasprzyk

No.	Title	NCES contact
1999–03	Evaluation of the 1996–97 Nonfiscal Common Core of Data Surveys Data Collection, Processing, and Editing Cycle	Beth Young
2000–10	A Research Agenda for the 1999–2000 Schools and Staffing Survey	Dan Kasprzyk
Schools – safety and discipline		
97–09	Status of Data on Crime and Violence in Schools: Final Report	Lee Hoffman
Science		
2000–11	Financial Aid Profile of Graduate Students in Science and Engineering	Aurora D'Amico
2001–07	A Comparison of the National Assessment of Educational Progress (NAEP), the Third International Mathematics and Science Study Repeat (TIMSS-R), and the Programme for International Student Assessment (PISA)	Arnold Goldstein
Software evaluation		
2000–03	Strengths and Limitations of Using SUDAAN, Stata, and WesVarPC for Computing Variances from NCES Data Sets	Ralph Lee
Staff		
97–42	Improving the Measurement of Staffing Resources at the School Level: The Development of Recommendations for NCES for the Schools and Staffing Survey (SASS)	Mary Rollefson
98–08	The Redesign of the Schools and Staffing Survey for 1999–2000: A Position Paper	Dan Kasprzyk
Staff – higher education institutions		
97–26	Strategies for Improving Accuracy of Postsecondary Faculty Lists	Linda Zimbler
Staff – nonprofessional		
2000–13	Non-professional Staff in the Schools and Staffing Survey (SASS) and Common Core of Data (CCD)	Kerry Gruber
State		
1999–03	Evaluation of the 1996–97 Nonfiscal Common Core of Data Surveys Data Collection, Processing, and Editing Cycle	Beth Young
Statistical methodology		
97–21	Statistics for Policymakers or Everything You Wanted to Know About Statistics But Thought You Could Never Understand	Susan Ahmed
Statistical standards and methodology		
2001–05	Using TIMSS to Analyze Correlates of Performance Variation in Mathematics	Patrick Gonzales
Students with disabilities		
95–13	Assessing Students with Disabilities and Limited English Proficiency	James Houser
Survey methodology		
96–17	National Postsecondary Student Aid Study: 1996 Field Test Methodology Report	Andrew G. Malizio
97–15	Customer Service Survey: Common Core of Data Coordinators	Lee Hoffman
97–35	Design, Data Collection, Interview Administration Time, and Data Editing in the 1996 National Household Education Survey	Kathryn Chandler
98–06	National Education Longitudinal Study of 1988 (NELS:88) Base Year through Second Follow-Up: Final Methodology Report	Ralph Lee
98–11	Beginning Postsecondary Students Longitudinal Study First Follow-up (BPS:96–98) Field Test Report	Aurora D'Amico
98–16	A Feasibility Study of Longitudinal Design for Schools and Staffing Survey	Stephen Broughman
1999–07	Collection of Resource and Expenditure Data on the Schools and Staffing Survey	Stephen Broughman
1999–17	Secondary Use of the Schools and Staffing Survey Data	Susan Wiley
2000–01	1999 National Study of Postsecondary Faculty (NSOPF:99) Field Test Report	Linda Zimbler
2000–02	Coordinating NCES Surveys: Options, Issues, Challenges, and Next Steps	Valena Plisko
2000–04	Selected Papers on Education Surveys: Papers Presented at the 1998 and 1999 ASA and 1999 AAPOR Meetings	Dan Kasprzyk

No.	Title	NCES contact
2000–12	Coverage Evaluation of the 1994–95 Public Elementary/Secondary School Universe Survey	Beth Young
2000–17	National Postsecondary Student Aid Study:2000 Field Test Methodology Report	Andrew G. Malizio
2001–04	Beginning Postsecondary Students Longitudinal Study: 1996–2001 (BPS:1996/2001) Field Test Methodology Report	Paula Knepper
2001–07	A Comparison of the National Assessment of Educational Progress (NAEP), the Third International Mathematics and Science Study Repeat (TIMSS-R), and the Programme for International Student Assessment (PISA)	Arnold Goldstein
2001–09	An Assessment of the Accuracy of CCD Data: A Comparison of 1988, 1989, and 1990 CCD Data with 1990–91 SASS Data	John Sietsema
Teachers		
98–13	Response Variance in the 1994–95 Teacher Follow-up Survey	Steven Kaufman
1999–14	1994–95 Teacher Followup Survey: Data File User's Manual, Restricted-Use Codebook	Kerry Gruber
2000–10	A Research Agenda for the 1999–2000 Schools and Staffing Survey	Dan Kasprzyk
Teachers – instructional practices of		
98–08	The Redesign of the Schools and Staffing Survey for 1999–2000: A Position Paper	Dan Kasprzyk
Teachers – opinions regarding safety		
98–08	The Redesign of the Schools and Staffing Survey for 1999–2000: A Position Paper	Dan Kasprzyk
Teachers – performance evaluations		
1999–04	Measuring Teacher Qualifications	Dan Kasprzyk
Teachers – qualifications of		
1999–04	Measuring Teacher Qualifications	Dan Kasprzyk
Teachers – salaries of		
94–05	Cost-of-Education Differentials Across the States	William J. Fowler, Jr.
Training		
2000–16a	Lifelong Learning NCES Task Force: Final Report Volume I	Lisa Hudson
2000–16b	Lifelong Learning NCES Task Force: Final Report Volume II	Lisa Hudson
Variance estimation		
2000–03	Strengths and Limitations of Using SUDAAN, Stata, and WesVarPC for Computing Variances from NCES Data Sets	Ralph Lee
2000–04	Selected Papers on Education Surveys: Papers Presented at the 1998 and 1999 ASA and 1999 AAPOR Meetings	Dan Kasprzyk
Violence		
97–09	Status of Data on Crime and Violence in Schools: Final Report	Lee Hoffman
Vocational education		
95–12	Rural Education Data User's Guide	Samuel Peng
1999–05	Procedures Guide for Transcript Studies	Dawn Nelson
1999–06	1998 Revision of the Secondary School Taxonomy	Dawn Nelson