UNITED NATIONS STATISTICAL COMMISSION
and
ECONOMIC COMMISSION FOR EUROPE

CONFERENCE OF EUROPEAN STATISTICIANS
STATISTICAL STANDARDS AND STUDIES - No. 44

STATISTICAL DATA EDITING

Volume No. 1

METHODS AND TECHNIQUES

UNITED NATIONS
New York and Geneva, 1994

CONTENTS

C. IMPLEMENTATION OF DATA EDITING PROCEDURES

PREFACE

Data editing methods and techniques may significantly influence the quality of statistical data as well as the cost efficiency of statistical production. The aim of this publication is to assist National Statistical Offices in their efforts to improve and economize their data editing processes.

Different methods and techniques can be used in the various stages of the data editing process; that is in survey management, data capture, data review and data adjustment.

Questions as to what methods and techniques to use for data review and data adjustment, as well as the iterative character of these two steps, are often very sensitive. The publication, therefore, particularly focuses on these issues.

Part A -- Review of Statistical Data Editing Methods and Techniques -- presents a more detailed treatment of data editing methods and considerations pertaining to each. It describes some examples and experiences achieved in statistical practice. To complete the methodological part of the presented review, the Fellegi-Holt methodology, as a most frequently used approach, is presented here in its entirety.

Part B -- Macro-editing Procedures -- is devoted to macro-editing methods as powerful tools for the rationalizing of the data review process through error detection at the aggregated level. Significant savings are achieved in some cases when implementing macro- instead of micro-editing procedures. The most advanced experiences in this respect are reported from Statistics Sweden .

Another important aspect is an appropriate computerization of the data editing process. **Part C -- Implementation of Data Editing Procedures** -- deals with some of these features. Surprisingly few commercially developed software packages provide efficient facilities for statistical data editing. Taking into account a mixed technological environment, only SAS was reported. Tailor-made software systems prevail. Some experiences with SAS applications and individually prepared systems such as GEIS, DC2 (both prepared in Statistics Canada) and BLAISE (developed in the Netherlands Central Bureau of Statistics) are presented in this part of the publication

An integral part of the publication is the **Bibliography** presented in **Part D**. The authors' intention in the preparation of this section was to spread knowledge on the already prepared and published studies and other materials dealing with statistical data editing methods and techniques.

This publication is based on contributions of the group of experts working within the project on Statistical Data Editing in the programme of work of the Conference of European Statisticians. It was compiled and edited by the Statistical Division of the United Nations Economic Commission for Europe.

This material represents an extensive voluntary effort on the parts of authors. The editors express their appreciation and thanks to all authors who contributed to the publication, as well as to all members of the UN Working Group on Statistical Data Editing whose joint efforts contributed significantly to the preparation of this publication.

AN INTRODUCTION TO THE DATA EDITING PROCESS

by Dania P. Ferguson
United States Department of Agriculture
National Agricultural Statistics Service

Abstract: A primer on the various data editing methodologies, the impact of their usage, available supporting software, and considerations when developing new software.

I. INTRODUCTION

The intention of this paper is to promote a better understanding of the various data editing methods and the impact of their usage as well as the software available to support the use of the various methodologies.

It is hoped that this paper will serve as:

- a brief overview of the most commonly used editing methodologies,
- an aid to organize further reading about data editing systems,
- a general description of the data editing process for use by designers and developers of generalized data editing systems.

II. REVIEW OF THE PROCESS

Data editing is defined as the process involving the review and adjustment of collected survey data. The purpose is to control the quality of the collected data. This process is divided into four (4) major sub-process areas. These areas are:

- **Survey Management**
- **Data Capture**
- **Data Review**
- **Data Adjustment**

The rest of this section will describe each of the four (4) sub-processes.

1. Survey Management

The survey management functions are: a) **completeness checking**, and b) **quality control** including audit trails and the gathering of cost data. These functions are administrative in nature. Completeness checking occurs at both survey and questionnaire levels.

At survey level, **completeness checking** ensures that all survey data have been collected. It is vitally important to account for all samples because sample counts are used in the data expansion procedures that take place during Summary. Therefore, changes in the sample count impact the expansion. A minimal completeness check compares the Sample count to the questionnaire count to insure that all samples are accounted for, even if no data were collected. In the case of a Census, the number of returned questionnaires are compared to the number of distributed questionnaires or to an estimated number of questionnaires expected to be returned.

Questionnaire level completeness checking insures routing instructions have been followed. Questionnaires should be coded to specify whether the respondent was inaccessible or has refused, this information can be used in verification procedures.

Survey management includes **quality control** of the data collection process and measures of the impact on the data by the data adjustment that occurs in sub-process No. 3 below. Survey management is a step in the quality control process that assures that the underlying statistical assumptions of a survey are not violated. "Long after methods of calculation are forgotten, the meaning of the principal statistical measures and the assumptions which condition their use should be maintained", (Neiswanger, 1947).

Survey management functions are not data editing functions per se but, many of the functions require accounting and auditing information to be captured during the editing process. Thus, survey management must be integrated in the design of data editing systems.

2. Data Capture

Data capture is the conversion of data to electronic media. The data may be key entered in either a **heads down** or **heads up** mode.

a. **Heads down** data entry refers to data entry with no error detection occurring at the time of entry. High-speed data - entry personnel are used to key data in a "heads down" mode. Data entered in a heads down mode is often verified by re-keying the questionnaire and comparing the two keyed copies of the same questionnaire.

b. **Heads up** data entry refers to data entry with a review at time of entry. Heads up data entry requires subject matter knowledge by the individuals entering the data. Data entry is slower, but data review/adjustment is reduced since simple inconsistencies in responses are found earlier in the survey process. This mode is specially effective when the interviewer or respondent enter data during the interview. This is known as **Computer Assisted Interviewing** which is explained in more detail below.

Data may be captured by many automated methods without traditional key entry. As technology advances, many more tools will become available for data capture. One popular tool is the touch-tone telephone key-pad with synthesized voice computer-administered interview. Optical Character Readers (OCR) may be used to scan questionnaires into electronic form.

The use of electronic calipers and other analog measuring devices for Agricultural and Industrial surveys is becoming more common place.

The choice of data-entry mode and data adjustment method have the greatest impact on the type of personnel that will be required and on their training.

3. Data Review

Data review consists of both **error detection** and **data analysis**.

a. **Manual data review** may occur prior to data entry. The data may be reviewed and prepared/corrected prior to key-entry. This procedure is more typically followed when heads-down data entry is used.

b. **Automated data review** may occur in a batch or interactive fashion. It is important to note that data entered in a heads-down fashion may later be corrected in either a batch or an interactive data review process.

- **Batch data review** occurs after data entry and consists of a review of many questionnaires in one batch. It generally results in a file of error messages. This file may be printed for use in preparing corrections. The data records may be split into two files. One containing the 'good' records and one containing data records with errors. The latter file may be corrected using an interactive process.

- **Interactive data review** involves immediate review of the questionnaire after adjustments are made. The results of the review are shown on a video display terminal and the data editor is prompted to adjust the data or override the error flag. This process continues until the questionnaire is considered acceptable by the automated review process. Then results of, the next questionnaire's review by the auto review processor are presented. A desirable feature of Interactive Data Editing Software is to only present questionnaires requiring adjustments.

  Computer-Assisted Interviewing (CAI) combines interactive data review with interactive data editing while the respondent is an available source for data adjustment. An added benefit is that data capture (key-entry) occurs at interview time. This method may be used during telephone interviewing and with portable data-entry devices for on-site data collection.

  CAI assists the interviewer in the wording of questions and tailors succeeding questions based on previous responses. It is a tool to speed the interview and assist less experienced interviewers. CAI has mainly been used in **Computer-Assisted Telephone Interviews (CATI)**, but as technological advances are made in miniaturization of personal computers, more applications will be found in **Computer Assisted Personal Interviewing (CAPI)**.

c. **Data review** (error detection) may occur at many levels.

- **Item level** - Validations at this level are generally named "**range checking**". Since items are validated based on a range. Example: age must be > 0 and < 120. In more complex range checks the range may vary by strata or some other identifier. Example: If strata = "large farm operation" the acres must be greater than 500.

- **Questionnaire level** - This level involves across item checking within a questionnaire. Example 1: If married = 'yes' then age must be greater than 14. Example 2: Sum of field acres must equal total acres in farm.

- **Hierarchical** - This level involves checking items in related sub-questionnaires. Data relationships of this type are known as "hierarchical data" and include situations such as questions about an individual within a household. In this example, the common household information is on one questionnaire and each individual's information is on a separate questionnaire. Checks are made to insure that the sum of the individual's data for an item does not exceed the total reported for the household.

d. **Across Questionnaire level** edits involve calculating valid ranges for each item from the survey data distributions or from historic data for use in outlier detection. Data analysis routines that are usually run at summary time may easily be incorporated into data review at this level. In this way, summary level errors are detected early enough to be corrected during the usual error correction procedures. The across questionnaire checks should identify the specific questionnaire that contains the questionable data. Across questionnaire level edits are generally grouped into two types: **statistical edits** and **macro edits**.

- **Statistical Edits** use the distributions of the data to detect possible errors. These procedures use current data from many or all questionnaires or historic data of the statistical unit to generate feasible limits for the current survey data. **Outliers** may be identified in reference to the **feasible limits**. Research has begun in the more complicated process of identifying inliers, (Mazur, 1990). **Inliers** are data falling with feasible limits, but identified as suspect due to a lack of change over time. A measurable degree of change is assumed in random variables. If the value is too consistent then the value might have simply been carried forward from a prior questionnaire rather than newly reported. The test therefore consists of comparison to the double root residual of a sample unit over time. If the test fails then the change is not sufficiently random and the questionnaire should be investigated. At USDA-NASS this test is applied to slaughter weight data. The assumption being that the head count of slaughtered hogs may not vary by much from week to week. But, the total weight of all slaughtered hogs is a random variable and should show a measurable degree of change each week.

- **Macro Edits** are a review of the data at an aggregate level. Inconsistencies are traced to the individual records involved. Much of the current work in this area is being carried out by Leopold Granquist (1991) of Statistics Sweden. His work is based on the belief that it is very desirable to determine the impact on the

summary by serious errors in order to avoid making adjustments that will not be of consequence at summary level.

The data review process should allow for detection of errors of different levels of severity. It should also allow for the decision to be made whether to correct an error.

4. Data Adjustment (Data Editing and Imputation)

Manual data adjustment is when the selection of a more reasonable value is done by a person. It may involve writing down, for key entry, the adjustments to be posted to the survey data file using a batch procedure. "Manual" data adjustments may also take place interactively as in the process of "heads-up" data entry or interactive data review.

Automated data adjustments occur as a result of computer actions. A desirable option in any system allowing computer actions is to allow for the overriding of those actions at some level. Batch data adjustment results in a file of corrected (edited/imputed) records with accompanying messages to report on the computer actions taken to make the adjustments.

The data may be imputed following a wide range of methodologies some being much easier to program than others. The simplest involves the completion of calculations within one questionnaire (such as, obtaining a missing sum at the bottom of a column).

Automated imputations generally fall into one of five categories.

a. **Deterministic** - where only one correct value exists, as in the missing sum at the bottom of a column of numbers. A value is thus determined from other values on the same questionnaire.

b. **Model based** - use of averages, medians, regression equations, etc. to impute a value.

c. **Deck** - A donor questionnaire is used to supply the missing value.

   Hot deck - a donor questionnaire is found from the same survey as the questionnaire with the missing item. The "**nearest neighbour**" search technique is often used to expedite the search for a donor record. In this search technique, the deck of donor questionnaires come from the same survey and shows similarities to the receiving record, where similarity is based on other data on the questionnaire that correlates to the data being donated. For example: Similar size and location of farm might be used for donation of fuel prices.

   Cold deck - same as hot deck except that the data is found in a previously conducted similar survey.

d. **Mixed** - In most systems there is usually a mixture of categories used in some fixed ranked fashion for all items. Statistics Canada's GEIS (Generalized Edit and Imputation System), for example, first uses a deterministic approach. If it is not successful, then a hot

deck approach is tried. This is followed by a model based approach. If all these approaches fail, then a manual imputation occurs through a human review process.

For more detailed explanations of methods a) through d) read the paper by Giles and Patrick, (1986).

e. **Expert Systems** - Expert systems are only recently being applied to data editing and much research is beginning in this area. "An expert system is an intelligent computer program that uses knowledge and inference procedures to solve problems that are difficult enough to require significant human expertise for their solution. Every expert system consists of two principal parts: the **knowledge base** and the **inference engine**. The knowledge base contains both factual and heuristic knowledge. Factual knowledge is items commonly agreed upon by spokesmen in a particular field. **Heuristic knowledge** is the less rigorous, more experiential and more judgmental knowledge of performance or what commonly constitutes the rules of "good judgment" or the art of "good guessing" in a field. A widely used representation for the knowledge base is the rule or IF/THEN statement. The IF part lists a set of conditions in some logical combination. Once the IF part of the rule is satisfied, the THEN part can be concluded or problem solving action taken. Expert systems with knowledge represented in rule form are called **rule based systems**", (Magnas, 1989). The inference engine makes inferences by determining which rules are satisfied by facts, ordering the satisfied rules, and executing the rule with the highest priority.

GEIS from Statistics Canada contains a rule set which is manipulated to generate the Knowledge Base. The Fellegi and Holt methodology is used as the inference engine algorithm to derive the minimum set of values to be imputed. It provides an interface between the user and Data Editing System allowing experimentation with the data before and during data editing. This data analysis provides information that is used to change rules and task specifications.

"Computer-Assisted Data Editing" is one manifestation of the use of expert systems. This application is found at the U.S. Department of Energy - Energy Information Administration (EIA). EIA has developed a computer-assisted data editing application for the PC based on the LEVEL5 inference engine. It has resulted in quality improvements and increased speed in the data adjustment process with the use of personnel having little or no subject matter knowledge. This is an specially significant accomplishment since energy data requires extensive coding for types of engines and fuels, (Magnas, 1989).

Expert data editing systems make so-called intelligent imputations. In an expert system, the computer mimics human actions. For example a subject area expert may specify a hierarchy of methods to be used in imputing an item. One item may use a deterministic followed by a hot deck approach. While another item might require a model based approach. Each item on the questionnaire would be resolved according to its own hierarchy of approaches. The next being automatically tried when the method before it has failed. SPEER (Structured Program for Economic Editing and Referrals) system from the U.S. Bureau of the Census is an application of an intelligent imputation.

Each of the above approaches can be carried out in very simple or very sophisticated ways. The more sophisticated approaches tend toward heuristic processes, where the methodology adapts as specified by the data distributions and combinations of items that are encountered.

III. CONSIDERATIONS IN IMPLEMENTATION

Data review and adjustment is a cyclical process. The adjustments require further review in order to catch any new inconsistencies introduced by the adjustments.

1. **Minimizing the number of iterations** of this cyclical process is the subject of much discussion. Of particular note are the ideas of Fellegi & Holt (1976) in Statistics Canada and the advent of interactive data editing.

In the Fellegi & Holt approach, the logic that was used for data review is analyzed by the machine and used to generate machine logic that identifies the minimum set of items on a questionnaire needing to be corrected to resolve an inconsistency. Correction of these items would ensure that no new inconsistencies would be introduced during data adjustment. Thus, reducing the iterations of the data review/adjustment cycle.

Examples of software systems based on the Fellegi & Holt methodology are:

CAN-EDIT, Statistics Canada(implementation of an early version of the Fellegi & Holt methodology)
DIA from INE, Spain (an extension of Fellegi & Holt)
SPEER from U.S. Bureau of the Census
GEIS from Statistics Canada and
AERO from CSO, Hungary
DAISY from INS, Italy.

Interactive data editing reduces the time frame needed to complete the cyclical process of review and adjustment. Although, it does not necessarily decrease the number of iterations of the cycle. The decrease in time to complete the process is achieved through the elimination of the need to order and file questionnaires between iterations. In order to be prepared for the next edit error listing in a batch system, all questionnaires must be ordered so that they may later be located for adjustment. Ordering is required because the current batch is added to previous batches which may still be unresolved as new errors were introduced in the process of solving previous ones. In an interactive data editing environment, the questionnaires are presented and resolved as they are keyed or in one adjustment session thereby eliminating the need to re-order file and re-file the questionnaires. An example of an interactive editing systems are:

BLAISE by the Netherlands, Central Bureau of Statistics (supports CATI and CAPI)
LXES by the U.S., National Agricultural Statistics Service and
IMPS, CONCOR module by the U.S., Bureau of the Census.

2. Many **different technologies** are applied in developing data editing systems:

Databases are especially well suited for editing hierarchical data and, historic or coded data that require multiple file look-ups.

Examples of systems built on database technology are:

The CAN-EDIT (based on RAPID), and GEIS (Oracle based) systems developed by Statistics Canada.

The GODAR (based on BOS and RAPID) system from Republican Statistical Office of Serbia (reported in 1991).

Statistical packages are well suited environments in which to build systems that use statistical and macro-editing methods. This also easily accommodates edits on continuous data.

Examples of these systems are:

The Survey Processing System (SPS) from the U.S. Department of Agriculture, NASS.
CASCADA from INE, Spain.

Micro-computer environments are well suited to interactive data editing. Computer assisted telephone and personal interviewing are further applications of interactive data adjustment where the respondent is the source for the "manual" adjustments.

Examples of a micro-computer based editing systems are:

CONCOR by the U.S. Bureau of the Census (A portion of IMPS)

BLAISE by the Netherlands, Central Bureau of Statistics.

The BLAISE system integrates the data entry, data editing and computer assisted interviewing functions in a micro-computer environment. This results in major savings of the time spent in re-specifying information common to these processes.

IV. CONCLUDING REMARKS

Many options are available at each stage of the data editing process. Once the desired options are identified a decision must be made whether to use existing software. When choosing an appropriate software, a List of Evaluation Criteria for Software on Data Editing (1991) prepared within the ECE/UNDP Statistical Computing Project - Phase 2 could be of valuable assistance.

Final thoughts for those designing new editing systems: The use of a machine should be to expedite and assist the human procedures. Use of human resources and impact on office work flows should be the primary consideration. The human effort required to develop and maintain data review/correction logic must be considered.

The integration of questionnaire design, computer assisted interviewing, data entry, analysis, and summarization into the edit process greatly enhances Survey Management and reduces redundancy of the data definition portions of the program code.

A REVIEW OF THE STATE OF THE ART IN AUTOMATED DATA EDITING AND IMPUTATION

by Mark Pierzchala
United States Department of Agriculture
National Agricultural Statistics Service

Abstract: This paper explores some general approaches to the automation of data editing and imputation and summarizes the progress made up until September 1988 in each of the approaches. The state of the art in four institutions, Statistics Canada, U.S. Bureau of the Census, Netherlands Central Bureau of Statistics, and National Agricultural Statistics Service (NASS) is reviewed.

I. INTRODUCTION

1. Background

The editing process in NASS is cyclic, batch oriented, and involves professional review of computer edit printouts. The process involves hand coding and correction of items in an initial edit and entry process followed by redundant hand correction and data entry of fields that violate edits. The system requires a certain amount of paper shuffling as questionnaires are filed, pulled, and refiled as many times as the corresponding records fail the edits. In some surveys, such as the June Agricultural Survey, imputation is done by the commodity specialist not only for item nonresponse but also for total nonresponse and in either case with unknown effects on the distributions of the data. In data editing, the emphasis is on within record (within-questionnaire) data validation leaving between-record analysis to post-edit analysis packages. Relatively few corrections are carried out by the computer; almost all must be made by specialists again with unknown effects on univariate and multivariate distributions. Edit failures are listed on printouts with no indication as to which of the fields is most likely to need correction. The effect of the edits, taken either individually or in combination, on the quality of the data is unknown. From a human perspective, the process can be tedious as the individual must work through hundreds of pages of computer printouts, not knowing the success or failure of corrections and imputations until the next editing cycle. The desire to eliminate or reduce these problems and also to broaden the agency's perspective on the editing process is the impetus for this study.

2. The study

The comparative study concerns four survey institutions, NASS, Statistics Canada, the Netherlands Central Bureau of Statistics, and the U.S. Bureau of the Census. These institutions operate in different environments and thus have taken different approaches to reducing editing problems. The environment of each organization includes the survey environment and a historical environment. The former includes the types of surveys conducted, the tightness of survey

deadlines, and the ways in which the populations are multivariately distributed. The latter concerns the history of how things have been done in the organization in the past.

II. TERMS OF REFERENCE

As this is a comparative study of the implementation of generalized or multipurpose editing systems involving four organizations in three different countries, terms of reference are needed.

1. Definition of the term editing

The definition of the term editing varies. For example, editing may be considered either as a validating procedure or as a statistical procedure. Both procedures aim to reduce errors in data sets but each has its strengths and weaknesses. Additionally, editing can be done at the record level or at some level of aggregation of individual records.

1.1 Editing as a validating procedure

As a validating procedure, editing is a within-record action with the emphasis on detecting inconsistencies, impossibilities, and suspicious situations and correcting them. Examples of validation include: checking to see if the sum of parts adds up to the total, checking that the number of harvested acres is less than or equal to that of planted acres, and checking if a ratio falls within certain bounds as set by a subject matter specialist based on expert knowledge in the field. Validation procedures may also be thought of as being established without reference to collected data.

1.2 Statistical edit

As a statistical procedure, checks are based on a statistical analysis of respondent data (Greenberg and Surdi, 1984). A statistical edit usually follows validation in an editing system. It may refer to a between-record checking of current survey data or to a time series procedure using historical data of one firm. As a between-record check, the emphasis is on detecting outliers of either univariate or multivariate distributions. One manifestation of between-record checking would be edit limits generated from distributions of a subset of current records. The most usual subset would be the first **n** records that enter the system. The error limits then would be applied to all records, the **n** records being run through the system again.

As a time series check, the aim is to customize edit limits for each firm, based on that firm's historical data as fitted to time series models. This approach has been used in the Energy Information Administration of the U.S. Department of Energy using spectral analysis (Dinh, 1987). Cathy Mazur of NASS is also investigating the use of a time series approach to edit daily livestock slaughter data. The use of historical time series to check data may be one way in which to detect inliers, that is, data which should greatly deviate from the mean but do not.

1.3 Macro-editing

These are edits which are run on aggregations of data, perhaps at some summary level or in some economic cell. Leopold Granquist of Statistics Sweden is developing some of these ideas (Granquist, 1987). They have also been mentioned in Statistics Canada as a future research topic (Sande, 1987). The U.S. Bureau of the Census does macro-editing though the process is not at a high level of automation. The aim of macro-editing is to edit data to find inconsistencies at the publishing level. It should be possible to trace the inconsistencies at the aggregate level to the individual records involved. Macro-editing focuses on those records in which corrections will have an impact at the particular aggregate level.

2. Priorities of development efforts

Examples of priorities in improving the editing process are: rationalizing the process, streamlining the process, and expanding the system's capabilities in handling expanded survey requirements. Amongst other things, rationalization refers to statistical defensibility, maintaining univariate and multivariate distributions, and the consistent handling of errors and missing data. Streamlining focuses on performing tasks more efficiently with more powerful tools. Expansion of capabilities means larger edit programs, more flexibility at the local level, a larger number of variables, retention of the data as it is first keyed, more kinds of edit functions, and a system which does not preclude the addition of any features.

Organizations with differing priorities will allocate research and development resources differently. For example, rationalization of the edit process requires the development of theory and algorithms, whereas streamlining requires the acquisition of new hardware and new systems development.

3. Manner of making corrections and imputations; the roles of people and machines

The manner in which corrections are made can be a very contentious issue. Implementation of new technology will at least have the effect of modifying the way in which editing tasks are done. This includes tasks performed by the subject-matter specialist (in NASS the agricultural statistician), the clerk, and the data entry operator. In the most extreme manifestation of automation, certain parts of the jobs of these people could be eliminated. The resolution of this issue may be as much a personnel management question as a statistical one. There are several ways in which corrections could be made. Some examples:

- The subject-matter specialist takes action on edit failures in a cyclic process using paper printouts in a batch processing system.

- The subject-matter specialist takes action on edit failures in an interactive computer session.

- The computer takes action on some edit failures without review. The more difficult cases are referred to the subject matter specialist to contact the respondent or otherwise deal with the matter.

- The data entry operator corrects some types of errors at time of data entry leaving other errors for the computer or the specialist to handle.

The size of the survey, the time frame of the survey, and the resources that the survey organization is willing to commit to editing will all determine which mix of computer actions and personal actions is possible (see para. 7 further on). In economic surveys, it is also unlikely that any generalized system will be able to handle all records.

The choices made concerning the roles of people and machines will affect the acquisition of hardware and software. A specialist correcting errors interactively will require some type of terminal or personal computer to do so. If only the specialist is to make corrections, then research should focus on giving the specialist more powerful tools. If the editing function of the computer is to be increased (for whatever reason), or if greater defensibility is desired, then research should be directed towards algorithms and theory.

4. The future role of CATI and CAPI

Computer Assisted Telephone Interviewing (CATI) and Computer Assisted Personal Interviewing (CAPI) are technologies which can perform record validation at the time of data collection. If an organization is collecting data primarily through these new technologies, then it may be redundant to commit large resources to the validation part of an edit program. If, on the other hand, the organization must collect data through the mail, (as must the U.S. Bureau of the Census), or otherwise continue to use paper questionnaires, then further development of the validation programs is probably justified. One consideration is the timing of the implementation of CATI and CAPI. If implementation is 10 years away, then it is more justifiable to develop a new validation system than if it is two years away. Another consideration is how much of the validation program will be transferred to CATI and CAPI. Another consideration is how much of the validation program will be transferred to CATI and CAPI. In NASS, CATI data are run through an editing program as not all edits are contained within CATI.

5. Location and dispersion of the work

One of the organizations surveyed, the Netherlands Central Bureau of Statistics, is located in one building and therefore, dissemination of new technology is aided by the proximity of resource personnel. The other organizations have many locations. The issue here is whether the same tasks are carried out in many locations and if different tasks are carried out in different locations. If the former is true, there is a problem of support, training, commitment, and consistency. In this case, the organization may need simple-to-use systems as expertise is shared by telephone or word of mouth. If different tasks are carried out in different places, then there is a problem of coordination between parts of the system. For example, the U.S. Bureau of the Census' data are key-entered in Jeffersonville, Indiana, while editing is carried out in Suitland, Maryland. In this case, the separation of functions enforces a division of labor which might preclude the implementation of some systems. In other words, greater resources may have to be committed to making the interface between many users and the system more understandable. Hardware and training costs may be more expensive in organization with many locations.

6. Specialization of survey processing tasks

Specialization impacts the processing of survey data in that all of the specialized editing tasks, both before and during the survey, must be coordinated in an overall editing system. An example of specialization is one person creating an editing program and another using it to process survey data. In a modern survey organization, survey processing may be divided into tasks performed by tens of hundreds of people. The greater the amount of specialization, the more the system will have to be constructed in modules embedded in an overall system that will coordinate the work of many different people. One effect of automation may be to improve productivity enough to allow fewer people to handle more tasks. For example, survey processing from questionnaire design to writing and testing of edits may be handled by one small group of people.

7. Time frames, sample sizes, and editing resources of surveys

The size of the survey, the time frame of the survey, and the resources that the survey organization is willing to commit to editing will all determine which mix of computer actions and personal actions is possible. For example, a great number of survey schedules to be processed in a limited time may preclude personal action on each record. As another example, an organization with tight deadlines may not be able to let specialists enter data and correct it at the same time, as the speed that comes with specialization is required. On the other hand, an organization with declining staff numbers and tightening deadlines may be forced to adopt heretofore unneeded technologies. It may have to improve the productivity of the editing personnel in their handling of each record, or it may have to allow the computer to handle more of the routine errors without review, referring only difficult cases to the specialist.

8. Statistics to be derived and analyses to be conducted vis-a-vis the effect of editing and imputation on distributions of the data

Different types of imputations have different effects on the marginal and joint distributions of the data. For example, in item nonresponse, one possibility is to impute the average of the items from good records into the incomplete records (hot-deck imputation). In the former case, the distribution will not be changed as much. Both methods will give the same estimated average (at least in the limit), but the first method will understate the magnitude of the standard error. This is an issue of whether or not distributions must be maintained. Some statistics are not sensitive to altered distributions, for example averages, totals and proportions (although their standard errors are). Other statistics, such as measures of dispersion or multivariate analyses, are sensitive to altered distributions. If distributions are to be maintained, then it may be better to leave the bulk of the editing, correction and imputation to the computer. That is, some imputation procedures, including hand imputation, may not be suitable for some statistics and analyses.

Any imputation scheme rests on (sometimes implied) assumptions about the distributions of data for use nonrespondents compared to that of respondents. These assumptions should be stated and critically examined as to their validity.

9. The degree of variability of population affects imputation

As observed by Tanks Barr, (1988, personal communication), it may be easier to impute for missing values in a gas station survey than for agricultural items because gas station prices vary much less than items in an agricultural survey.

10. Planned uses of the data and the availability of the data

This point derives from point H. If record level data must be released to other organizations, then the collecting organization is obliged to leave the multivariate distributions as intact as possible as not all future data uses are known in advance. Once the data are outside the organization, there is no way to tell how they will be utilized, that is, whether multivariate analyses will be carried out or statistics will be generated that require distributions to remain intact. For example, the Economist Research Service of the U.S. Department of Agriculture obtains record level data through NASS from the Farm Costs and Returns Survey (FCRS) survey. As NASS does not know every future use of the data, the editing procedure should maintain the multivariate distributions (NASS does not currently impute in the FCRS). At least, if many imputations are carried out, they should then be flagged and a description of imputation procedures should also be included with the data.

11. Types of surveys

The types of surveys being processed, such as economic, social, or production surveys, will reflect on the complexity of the editing programs with regards to such items as routing, type of data (categorical or continuous), the degree of inter-relation between the fields as expressed through edits, the reliability of the edits, and the type and complexity of the edit themselves. These attributes affect the degree to which the editing process can be automated.

12. Previous survey experience

Organizations have different experiences regarding the degree of noncooperation, item nonresponse, partial nonresponse, and the frequency of errors in collected data. The relative amounts of resources spent on each survey step will be different. As a result, different organizations will have different perspectives and priorities in the development of new systems. Systems which may be justified in some organizations on the basis of the tradeoff between cost and data quality may not be justified in others. The validity of the editing and imputation procedures as well as their defensibility is also at stake. An organization with high rates of nonresponse may have greater difficulty in establishing a workable and defensible system in which the computer makes the bulk of corrections. For example, it would be harder to implement hot-deck imputation in a survey with 30% nonresponse than in one with 5% nonresponse because donor records may not be available for all recipient records in the former case.

13. Hardware

Computer intensive procedures, or interactive handling of the data, may be too expensive on leased mainframes if the organization is charged according to resources used. This may result in greater reliance on hand processing or a situation in which some features are not even

considered due to the cost. On the other hand, microcomputers may not have the capacity to handle some editing functions or they may not have access to historical information.

14. Software environment and programming support

The term software environment refers to whether or not the editing system will reside in a data base environment. If editing is to be carried out in a database environment, the question is whether the data base will be shared out between locations or be centralized. In the latter case, then at least part of the edit will have to be carried out on the computer carrying the database. Programming support refers to the amount of support available to customize editing programs for each survey, to modify a generalized program for each survey, or to support a program in different environments (editing on microcomputers as opposed to a main frame for example) as well as maintaining existing programs.

15. Purposes and costs of editing

See Granquist's "On the need for generalized numeric and imputation system" in this publication, and Pullum, Harpham and Ozsever (1986), for good discussions on the purposes of editing systems. These papers address the tradeoffs between improvements in data quality and costs of editing. In the former paper, Granquist estimates that editing takes from 20 to 40 percent of survey budgets in periodic surveys in Statistics Sweden and wonders if the benefits are worth the expenditures. In the latter paper, which discusses the editing of the World Fertility Survey, it is reported that estimates derived from raw tapes in 6 countries were essentially the same as those derived from edited data tapes. In other words, the machine editing had no appreciable effect on the analysis other than delaying the production of statistics by one year. The authors of these two papers do not question the basic necessity of editing, but consider that some editing resources could be allocated to other areas to improve data quality or that editing could be done in better ways.

Pullum, et. al., cite 5 reasons why the World Fertility Survey did implement stringent editing policies. They are cited as general beliefs as to why editing is done.

- To produce a gain in the yield of the fieldwork, that is, to minimize the number of responses excluded from analysis.

- To improve the validity of the findings, that is, to remove systematic errors that may lead to bias.

- To improve the correspondence between the structure of the questionnaire and that of the responses, the net effect being the easing of further tabulation and analysis.

- Users have more confidence in data which are internally consistent because such consistency reflects on the entire process of data collection and preparation.

- The perception that editing is a hallmark of professional survey research.

In this review of Pullum's et.al. World Fertility Survey paper, Granquist maintains that only reasons 3 and 4 really benefit from editing in the way it was carried out here, that is, through a Generalized Edit System. Granquist (1984a) describes the following purposes of editing:

- To give detailed information about the quality of the survey.
- To provide basic data for the improvement of the survey.
- To tidy up the data.

Granquist further believes that Generalized Edit Systems usually apply too many checks, that editing systems do not essentially improve data quality, and that editing systems can give a false impression of data quality.

16. Productivity and costs of editing

Another way in which to consider the effects of editing costs on the manner in which automation is affected is to plot the rapidly declining costs of computing against labor costs that are either constant or climbing. Kinds of automation considered too expensive 5 to 10 years ago, (for example computationally intensive programs or interactive handling of corrections), may be less expensive now, or in the future, than remaining with a labor intensive status quo.

III. THE FELLEGI AND HOLT SCHOOL OF EDIT AUTOMATION AND IMPUTATION

The literature emanating from this school of thought is concerned primarily with the stage of editing known as data validation. This school is characterized by its foundation in set theory, borrows heavily from techniques in Operations Research, Statistics, and Computer Science (Sande, 1979), and is guided by certain principles: that each record satisfy all edits, that correction be accomplished by as few changes as possible, that editing and imputation both be part of the same process, and that any imputation procedure retain the structure of the data. Automation of editing and imputation are required because some of the above desired principles are beyond the ability of the human editors. Automation may not be cheaper than the more labor intensive methods, but the computer can apply all edits quickly and consistently (Fellegi and Holt, 1976). Emphasis is placed on the rationalization and the defensibility of the editing process. Statistics Canada (where Fellegi is Chief Statistician of Canada) and the U.S. Bureau of the Census are implementing this approach.

1. Changing as few fields as possible in correction of data

In their 1987 paper, Fellegi and Holt outline a set theoretic approach which, if applied to categorical data or to linear edits of continuous data, would lead to the identification of a *minimal set* of fields that need to be corrected in order to clean the record. The corrections, if made according to the editing rules, guarantee that the whole record will pass all edits. This result can somewhat be extended since some nonlinear edits can be rendered into a linear form (e.g. one can render a ratio edit into two linear inequalities), (Giles and Patrick, 1986). This approach requires that a *complete set* of edits be generated from the *explicit edits* written by the subject-matter specialist. The idea is that there are *implied edits* which can be generated by logical implication from the explicit edits. For example, if $1 < a/b < 2$ and $2 < b/c < 4$, are explicit edits, then $2 < a/c < 8$ is an implied edit obtained algebraically from the explicit edits. The complete set of edits is the union of the explicit edits and the implicit edits. Once the complete set of edits is determined, a minimal set of fields can be determined for every possible set of edit failures. The determination of a minimal set of fields is called *error localization*. There are still some cases involving nonlinear edits in which it is generally impossible to find minimal sets because the complete set of implied edits cannot be found. The minimal set does exist however (Greenberg, personal communication).

2. Editing and imputation as the same process

In the Fellegi and Holt automated editing process, imputation constraints, when taken together, are called a *feasible region* and are derived from the set of complete edits. Corrections or imputations falling within this feasible region are guaranteed to pass the edits. Fellegi and Holt show that for categorical data or for continuous data under linear edits, either there is a feasible region or some edits are in conflict. In practice, there are some types of nonlinear edits which are not amenable to the determination of a feasible region. In such cases, the imputation can be run through the edits again to ensure that all imputations conform to the edits. In any case, it is a precept of this school of thought that all corrections and imputations pass all edits, although this may not be strictly adhered to in practice.

3. Retaining the structure of the data

One of the major objectives of the Fellegi and Holt school is to retain the structure of the data. This means than univariate and multivariate distributions of survey data reflect as nearly as possible the distributions in the population. Statistics Canada is doing this already by the use of hot-deck imputation. The U.S. Bureau of the Census uses hot-decking for agricultural surveys, for some demographic surveys, and the decennial censuses. Hot-deck imputation seeks to find a record similar to that of the incomplete record on the current set of survey records and to impute to missing variables from the complete record to the incomplete record. This can be done one variable at a time, the aim being to preserve the univariate distributions, or all variables at once, the aim then being to preserve the multivariate distributions. Retaining structure is important if there is to be multivariate analysis, if not all uses of the data are known in advance (e.g., it is not known who will have to access to it), or if statistics which depend on the distribution (e.g., quantiles) are to be calculated.

4. Implementation

Implementation of the approach of Fellegi and Holt has proved to be a challenge for nonlinear edits and continuous data. Checking the consistency of explicit edits, the generation of implied edits and the determination of an acceptance region require Operations Research (OR) methods (Sande, 1979). In hot-deck imputation, procedures from OR are needed to minimize the search for donor records. For a minimal set of fields, a best corresponding set of matching variables must be determined. An exact match between a candidate and donor record may not be possible in the continuous case, thus a *distance function* is used to define similarity. Some numerical imputations are not guaranteed to pass edits as are categorical imputations, thus redonation may be necessary, (Giles and Patrick, 1986). A donor record may have characteristics similar to those in the candidate record, but the operation may have a different size, thus scaling is required. Continuous edit checks that are linear are amenable to known Operations Research procedures whereas non-linear edits (such as conditional checks) are not. In the words of Brian Greenberg, U.S. Bureau of the Census, "To the extent that the methods developed by Fellegi and Holt for categorical data and by Sande for continuous data under linear constraints are employed in these (editing and imputation) routines, a high level of rigor will be introduced into this system. Any success in developing procedures to systematically address the comparable criterion for conditional numerical, conditional categorical, or mixed edits will be a fine methodological advance"(Greenberg, 1987a). In some records, more than one minimal set of fields may exist. If so, some procedure is needed to determine which set should be corrected. One method is to assign weights to reflect the relative reliability (in the opinion of the subject matter expert) of each field. Thus, if multiple minimal fields are found, the least reliable set of fields is updated.

5. Manifestations

Both Statistics Canada and the U.S. Bureau of the Census have implemented this editing philosophy to a certain degree. Neither system fully automates the editing process. Since the systems are not fully automated, some records are reviewed by the specialist. These records are either too difficult to be dealt with by one machine, or are referred to the specialist according to certain pre-determined criteria such as size of firm.

5.1 United States Bureau of the Census

In the U.S. Bureau of the Census, present implementation of the philosophy of Fellegi and Holt resides in a sub-system called the SPEER System (Structure Program for Economic Editing and Referrals). SPEER handles continuous data under ratio edits, and has six main components: Edit Generation, Edit Analysis, Edit Checking, Error Localization, Imputation, and Diagnostics (Greenberg, 1987a). From survey to survey, it is the Imputation module which requires great change. In the Census experience, the Edit Generation, Edit checking, and the Error Localization modules remain virtually unchanged (Greenberg, 1987a). SPEER resides within a larger editing system. This reflects the fact that there are a number of tasks (such as SIC code assignment, GEO assignment) that SPEER is not designed to perform. Additivity checks are also handled in SPEER. Other types of checks can be handled before or after SPEER is invoked or in special satellite routines within SPEER itself. Changes made outside SPEER at times cause violations of edits within SPEER. Census has adapted the approach of Fellegi and Holt as far as possible to increase the degree of automation. Greenberg has extended the approach of Fellegi and Holt into the realm of ratio edits. However, this is done by considering the ratio edits as a set, doing

what is possible within that set and sending the result to the broader system for further processing.

Imputation modules are applied one field at a time. These imputation modules consist of a series of rules that are utilized in a sequence until one of the rules generates a value that will satisfy the edits.

These modules are easy to create and can easily be revised to accommodate new understandings about the data (Greenberg and Surdi, 1984). When the imputation modules fail, the record is output to the specialist. In the interactive process, the statistician is presented with a list of fields in error and with ranges within which the value of each field must fall. The specialist enters a value for one field at a time, and each time the computer recalculates the ranges for the remaining fields to be changed. The result of the determination of a minimal set of fields and of the calculation of feasible regions is that the cyclic process of error printouts, error correction, and more error printouts is diminished or eliminated.

Brian Greenberg, Principal Researcher in the Statistical Research Division, views the editing process in two stages: (1) automated batch runs for all records, and (2) manual review for specially targeted records. It is not desirable to remove the analyst review component of the process. The aim is to provide the analyst with more information on the review document coming out of the batch run to assist in review tasks. The analyst review tasks should be done in an interactive mode working with the computer. The objectives of the analysts' job would not fundamentally change though the mechanics and logistics might.

The Bureau of the Census has processed one large survey, the Census of Construction Industries, with their system which included the SPEER sub-system. This was done on a mainframe because of the number of records involved (several hundred thousand). For two surveys in the Economic Surveys Division, the 1987 Enterprise Summary Report, and the 1987 Auxiliary Establishment Report, the Bureau of the Census is considering employing a combination of mainframe and microcomputers. The mainframe would be used for batch processing and automated imputation and would refer difficult cases to the specialist to handle on a microcomputer. The number of cases handled on the microcomputer would depend on the referral criteria which in turn would depend on how much the editing and imputation algorithms on the mainframe were trusted. Referral criteria can include the magnitude of changes made by SPEER or the size of the firm involved. In addition, the Industry Division has developed computer programs based on the SPEER methodology, and they have been used for the 1986 Annual Survey of Manufactures and the 1987 Census of Manufactures. The Agricultural Division of the Bureau of the Census is considering using the system for Agriculture Economic and Land Ownership Survey for which data collection will start in 1989.

5.2 Statistics Canada

In the Statistics Canada survey processing system for economic surveys, two modules of this system will handle distinct parts of the editing process. The first module is the Data Collection and Capture (DC2) module, the second is the generalized Edit and Imputation System (GEIS). DC2 is in the prototype stage whereas the GEIS has recently been completed and documented. Different modules are being created for different parts of the edit process because in Canada, the response unit may be different from the statistical unit. For example, a firm might

provide data for two factories on one questionnaire. In this case, the responding unit would be the firm and the statistical units would be the factories. DC2 would customize the forms to the respondent, do some basic editing at that level, and flag questionnaires for follow-up. All document control (status codes, etc.), a substantial amount of correction and all necessary follow-up are done in this preliminary edit. GEIS is meant to handle data at the statistical unit level, that is after the data have been processed by DC2. Only unresolved cases or cases of minor impact are passed to the Generalized Edit and Imputation System as a last resort, at which point an effort is made to solve all problems by imputation (Kovar, 1990a, b).

For now, GEIS will handle data which have not been processed by DC2. In this instance, it is expected that the amount of hand editing will be held to a minimum. Hand checking will be confined primarily to making sure that numeric data are entered in numeric fields and the like, and that control data on the first page is correct. GEIS has not yet been used in a production mode as the developers are still looking for clients. It is in the GEIS system that the philosophy and techniques of the Fellegi and Holt school of editing are currently in place.

Currently, GEIS handles only those edits that are linear and data that is positive but within these constraints, most edits and data can be handled. Many nonlinear edits can be recast in a linear form and negative data values can be given as a difference of two positive numbers (e.g., profits = income - outgo). In the future, these constraints are to be relaxed. GEIS is embedded in the relational data base management system ORACLE, which facilitates the organization and the handling of data (Kovar, 1990a,b). This aids in monitoring the edit and imputation process.

GEIS as an editing program consists of four main parts: specification of edits, analysis of edits, application of edits, and outlier detection (Kovar, 1990a,b). The specification of edits is done by a subject matter specialist working together with a methodologist. Specification is typically done on a micro-computer. The system performs a syntax check and also checks that variables employed in the edits have been specified in the questionnaire.

Further checking of the edits occurs in the analysis of the edits. This can be accomplished because the edits are linear and the data is positive. The edit analysis checks the consistency of the edits. The analysis also checks that redundant edits do not further restrict the feasible region of the data values. The system then generates the acceptable ranges for all variables, the extreme points of the feasible region, and the set of implied edits (Kovar, 1990a,b and Sande, 1979). This part of the system aids the analyst in determining if the edits are meaningful. It also helps to verify whether all edits are entered correctly.

In the application of the edits, an error localization procedure is invoked to determined the minimal number of fields to be corrected. Alternatively, the same procedure can be used to find the minimally weighted set of fields to be corrected. The latter alternative utilizes additional information on the reliability of the fields as judged by the specialist. If an edit failure can be cleared up by the imputation of only one value for a variable, then that value is imputed, that is, the error localization procedure handles deterministic cases. Uncorrected or unimputed records are passed onto the imputation procedure. In the imputation procedure, two general methods are available, donor imputation and other imputation procedures. Donor imputation is implemented by hot-deck imputation. This is meant to be the primary method of imputation. Hot-deck imputation is preferred because it retains the structure of the data. Other imputation procedures include imputation of historic values (which can be trend adjusted), imputation of means, and

ratio and regression estimators. These methods are backup methods used when the hot-deck procedure fails. They will not preserve the structure of the data as effectively as the hot-deck method. GEIS also has a facility which allows a choice of imputation methods by field.

Outlier detection is in the form of a statistical edit that operates on all records at once and cannot be applied at the same time as the other edits. The module can serve two distinct purposes: to determine the edit bounds, or to identify outlying values which can be flagged for imputation or for other considerations in subsequent modules (Kovar, 1990a,b).

The mathematical procedures needed for optimization and search are written in C. GEIS is being produced in several releases, with new features available in each release. Methodologists do not feel that they have all the answers yet and would like to provide a wide selection of alternatives for planning the edit and imputation. However, offering maximum flexibility and maximum satisfaction results in a system which lacks consistency. A more unifying theoretical basis is needed (Sande, 1987). The GEIS system is not as easy to use as desired and a great deal of intervention is still necessary. Statistics Canada expects system implementation to require a substantial amount of time.

6. Future research

The following list of topics must be researched in order to fully implement the goals of the Fellegi and Holt school. This list was compiled from literature and from personal communication with people from the U.S. Bureau of the Census and Statistics Canada.

- Non-linear edits (including conditional edits) in order to generate the set of implied edits and hence the complete set of edits and to generate a minimal set for non-linear edits.

- Negative values of variables.

- Implicity defined constants.

- What to do with multiple solutions (multiple minimal sets) in error localization.

- Variance estimation in the presence of imputed data (Sande, 1987).

- Zero values versus missing values, that is, does a blank on a questionnaire represent a zero or has the item been skipped.

- More intelligent or expert systems.

- Automated macro-edits carried out on tabulations of statistics rather than micro data with comparison between cells, between variables with historic data, and with other data sources, in order to avoid embarrassing data discontinuities, identity design and estimation problems, and lead to the formulation of improved micro-edits (Sande, 1988).

- Determination of which imputation option to use in which context.

- How to edit and impute when the data from a reporting unit includes data at the location level, establishment level, and all Canada level (Sande, 1988).

- What should be done when this year's imputation must be based on last year's imputation.

- Mixed edits (Giles, 1987).

- The blend in a multipurpose system between general routines and survey-specific procedures (Greenberg, pc).

7. Problems with this approach

Following is a list of problems gleaned from literature and from personal communication:

- The theory is not worked out for all cases. It can be implemented for categorical data, continuous data under linear edits, and ratio edits but not for some kinds of nonlinear edits such as conditional or mixed edits. (Though it may be possible to render some nonlinear edits into a linear form).

- Programming complexity (Fellegi and Holt, 1976).

- Distinguishing valid zeroes from missing data (Kovar). This is a problem for Statistics Canada because some of their survey data are obtained from computerized tax files with no recourse to the tax form in determining what a blank means.

- The choice of the appropriate imputation method in various contexts.

- The determination of which minimal set to correct when multiple minimal sets are found.

- The subject matter specialists may not be happy with going from a system in which specialist action is required to a system in which the computer takes most of the actions.

8. The desirable features of the system

Following is a collection of features from several papers which were either explicitly stated or implied. Priority is not given to the features. These features may or may not be implemented at this stage.

8.1 Methodological features

- There should be an orderly framework and philosophy for the development of edit and imputation procedures.

- Each record should satisfy edits by changing the fewest possible fields. ("Keeping the maximum amount of data unchanged"), (Fellegi and Holt, 1976).

- Imputation should be based as much as possible on the good fields of any record to be imputed (Fellegi and Holt, 1976).

- The frequency structure of the data file should be maintained for joint frequencies as well as for marginal frequencies.

- Imputation rules should be derived from corresponding edit rules without explicit specification.

- Imputations should not violate edits.

- The system should supply methodologically sound modules to be assembled by the user (Kovar).

- Defensibility of methods should be a priority (may somewhat constrain the user), (Kovar).

- When imputing for a deletion due to edit failures, one should endeavour to utilize the reported value although it is incorrect (Example: the respondent answers in pounds where tons are requested).

- Feedback should be provided on the impact of the system on estimates.

- The system should detect univariate outliers (Kovar).

8.2 System-oriented features

- It should not be necessary to specify imputation rules (Fellegi and Holt, 1976).

- The edit program should logically deduce a set of implied edits.

- Each record should be edited only once.

- Edit procedures should be implemented in a generalized editing system as part of the automation of the editing process. Thus, systems development need not be delayed by specific edit specifications. (However, it still may be desirable in some cases to introduce survey specific procedures. See the last item under paragraph 6, Further Research).

- Program rigidity should be reduced. (Fellegi and Holt, 1976). That is, changes to the program should be easy to accomplish without making errors.

- The generalized editing system should be embedded in a relational data base management system (Sande, 1987). The database environment should allow:
  . easy monitoring of imputation process (Kovar).
  . measurement of impact (of editing process) on estimates (Kovar).
  . measurement of the effect of imputation on particular states of the process (Kovar).

- The software should be portable between computers of varying types (Sande, 1987).

- The system should be modular, allowing the development of distinct stages of the editing and imputation process. It should also allow ease of updating and the ability to add new modules.

- Records as captured by the system should be kept in a file separate from those being edited and corrected. This allows evaluation of the technique and also allows one to go back to the respondent with data as the respondent gave it.

8.3 Subject matter specialist oriented features

- The subject matter specialists should be an integral part of a team implementing automated editing and imputation procedures.

- Only edits should have to be specified in advance as imputation rules would be generated by the computer (Fellegi and Holt, 1976).

- The subject specialist should be able to inspect the set of implied edits derived from the explicitly specified edits in order to evaluate the explicitly specified edits.

- In any survey, different sets of edit specifications (corresponding to different parts of the questionnaire) should be able to be created concurrently by two or more specialists. (The system would take care of any reconciliation), (Fellegi and Holt, 1976).

- Respecification of edits and imputation rules should be done easily (Greenberg, and others).

- The specialist should be able to experiment with the explicit edits either before the survey is conducted or after a small subset of the records are in, thus gaining information on the explicit edits.

- The generalized edit and data correction system should allow respecification of edits without reprogramming (Fellegi and Holt, 1976).

- The edit programme should be in modules so that the subject expert can easily enter or change explicit edits, and new versions of the system can easily be installed.

- The edit programme should provide feedback on how the edits are affecting data quality.

- The system should be comprehensible to and easily modifiable by the users of the sub-system.

- The system should be flexible and satisfying to the user, (give the users what they want, Kovar).

- The system should provide the user with a menu including a choice of functions applicable to the data or subsets of the data (Sande, 1987).

- A sample of edit failures should be checked. Statistics should be collected on the frequency with which each field is involved in an edit failure, the frequency with which each edit is failed and the frequency with which each field is identified for change (Sande, 1987).

IV. STREAMLINING AND INTEGRATING THE SURVEY PROCESS; THE BLAISE SYSTEM FROM THE NETHERLANDS

In this approach, as implemented by the Netherlands Central Bureau of Statistics, the automation of the editing process, although of importance, is only one part of the automation of the overall survey process. In this approach, no new theoretical tools are implemented to rationalize the editing process. The idea is to take the current cyclic batch process performed on mainframes and to put it on microcomputers, thus creating an interactive process. Because the survey process is now handled with essentially one integrated system of modules, the data need to be specified once only. That is, the specialist does not have to write an editing program, as it is derived automatically from the specification of the questionnaire on the computer. In addition, CATI and CAPI modules are generated automatically from the BLAISE questionnaire. The emphasis is on streamlining current methods and on integrating most computerized survey functions.

The subject matter specialist plays two important parts, the specification of the questionnaire and the resolution of the data. In the resolution of the data, the specialist is either a data checker/corrector or a data typist/checker/corrector.

1. Subject matter specialist as error checker/corrector

The data are entered normally, by very fast data entry personnel. The file is passed to the subject matter specialist who uses the microcomputer to correct errors one questionnaire at a time. After corrections are made, the record is re-edited on the spot and redisplayed with new error messages, if any. Thus the batch cycle is changed to an interactive micro cycle (micro having two meanings here, microcomputer, and each record cycling by itself through the editing program until it is correct). The questionnaires are available to the specialist for reference. The specialist sees the errors on the microcomputer screen along with error messages. The questions as they appeared on the questionnaire are not presented on the screen, rather mnemonic variables

are displayed. The questions are available from a help screen if needed. The record can be routed to a holding file of difficult questionnaires, if necessary. The system does not generate a minimal set of fields to be corrected as in the Fellegi and Holt school. It does display the number of times each field is involved in an edit failure. The premise is that the fields flagged the most frequently should be the first to be corrected as they are the ones which are more likely to be causing the problems.

2. Subject-matter specialist as data typist/checker/corrector

The data are entered by the specialist, who is not as fast as the regular entry personnel. However, the record is edited as it is entered. The specialist entering the data is also qualified to correct errors. Thus, data entry and reconciliation are combined. The extra time in data entry is offset by the one time handling of the record. Codes can be entered interactively.

3. The desirable features of the Blaise

Following is a collection of features which were explicitly stated or implied in several papers. The features are not prioritized. These features may or may not be implemented at the present time.

- Forms should not need to be prepared for entry (Bethlehem, 1987).

- The cyclic nature of editing should be removed (Bethlehem, 1987).

- The work should be concentrated as much as possible in the same department (Bethlehem, 1987).

- The work should be done as much as possible on the same computer (Bethlehem, 1987).

- There should be a reduction in time needed for editing (Bethlehem, 1987).

- The system should be applicable to different surveys.

- Superfluous activities should be eliminated.

- Error checking should be an intelligent and interactive process carried out between the subject matter specialist and the computer.

- The structure of the questionnaire and the properties of the resulting data should be specified only once (Bethlehem, 1987).

- Editing automation should be part of a larger automation process. (This is really the crux of the matter, but a Blaise questionnaire that contains total information about the questionnaire be constructed, as from which all products fall out,

including edit programs, CATI and CAPI instruments, etc. The Blaise questionnaire is considered a knowledge base in an artificial intelligence context).

- Data entry should have an interactive capability.

- An interface with statistical packages should be possible without the necessity of respecification of the data.

- The system should be user friendly (Bethlehem, 1987).

- The system should be based on microcomputers such as IBM AT/XTs and compatibles (Bethlehem, 1987).

- The updating of questionnaires from one survey to another should be easy (Bethlehem, 1987).

4. Impetus of the project

The Netherlands Central Bureau of Statistics implemented a data editing research project. The objective was to assemble management data on the editing process through the analysis of the processing of four selected surveys of various types and characteristics. For example, in the initial hand editing stage of survey processing, the steps in the editing process were listed and evaluated for their contributions to improved data quality. Editing activities were classified into three types: real improvements, preparation for data entry, and superfluous activities (such as writing a minus sign for missing data). In one survey, the relative share of the time spent on these activities was 23%, 18%, and 59% respectively. These measurements were made by inspection of the questionnaires after the surveys were completed. Other quantitative and qualitative aspects of survey processing were measured, such as rates of edit failure, time needed to clean a file, as well as the ways in which personnel interacted with the computer systems. Some of the findings of the project (Bethlehem, 1987):

- Different people from different departments were involved.

- Different computer systems were involved.

- Not all activities were aimed at quality improvement.

- Manual check of complex routing structures was difficult and time consuming.

- The editing process was cyclic.

- Repeated specifications of the data were necessary. The term repeated specification refers to the practice of specifying variables, valid values for the variables, relationships between variables, and routes to be followed depending upon values of the variables in the survey stream. These items were specified for the questionnaire (on paper or as a CATI or CAPI instrument), and again in data entry software, in an editing and analysis program, and in a summary program.

This problem was compounded if the various tasks were carried out on different computers using different software. In those cases, significant resources were spent just transferring data from one location to the next. This last point led to the design of the Blaise system (Denteneer, et al., 1987).

V. NASS DEVELOPMENT EFFORT

The theme underlying NASS's Survey Processing System (SPS) is one of integration. The term integration impacts the SPS in two major ways. Firstly, this term refers to one of the impetuses of the development of the SPS, that is, the integration of NASS surveys into one coherent sequence of surveys. This was originally done under the name of the Integrated Survey Program and is now known as the Quarterly Agricultural Survey Program. Secondly, the term refers to the integration of the distinct steps of the survey process under a unifying system. As such, the SPS has modules for data capture, data validation, and statistical editing although this latter module is not fully developed or utilized. In the future, the SPS will also encompass modules for imputation, analysis, summary and reports with further connections to a public use data base and a secure agency data base. A *specifications generator* is to be developed and will serve as the unifying feature of the SPS. It is envisioned that the specifications generator will output files for further processing into paper questionnaires, CATI and CAPI instruments, and an editing system. Integration will also serve to ensure the consistency of editing procedures across all surveys.

The implementation of the Integrated Survey Program served as an impetus to the development of the SPS because the previous edit and summary system could not handle the requirements of the new program. For example, there was a need to be able to process each section of the questionnaire differently as regards completion codes and refusals in order to summarize the sections independently. In addition, the old system could not handle the large number of variables demanded by the new survey system and it would not allow data to be compared between records.

Beyond the system limitations mentioned above, a number of new capabilities are desired for statistical purposes. The term editing was expanded to include statistical edits which involve cross-record comparisons at the time of the more traditional data validation (Vogel, et al., 1985). It was also desired that the effect of imputations and non-response be known at various levels of aggregation, that procedures be consistent across all surveys, and that NASS procedures be statistically defensible. Editing and imputation of missing data are not considered as parts of the same process in the sense of Fellegi and Holt. That is, the edits are not used to define a feasible region for imputation. However, nothing in the system prevents records with imputed values from being run through the edits once again.

NASS is probably one of the few agencies in the world to have programmed its editing software in the statistical language SAS (R). The use of SAS for editing has sparked some interest in an international editing research group because of its portability, its wide use as an analysis tool, its flexibility, and its amenability to developments in statistical editing and in micro macro combination edits (Atkinson, 1988b). These advantages also apply to NASS, that is, nothing is precluded, therefore options are maintained.

1. Limitations of the old system (Generalized Edit System).

Following is a list of limitations gleaned from various NASS reports, conversations with NASS personnel, or noted from personal experience:

- An artificial limit to the number of parameter cards is often exceeded.
- Parameters are difficult to write.
- Error printouts are difficult to understand.
- Types and numbers of edit functions are limited.
- The system is not updated systematically.
- The system is not supported with training.
- The manual is written in an undefined jargon and contains few examples.
- Comparisons between records are not allowed, that is, the statistical distributions of the data were not reviewed.
- The system only points out errors; it does not correct them (Vogel, et al., 1985).
- The manual resolution of errors can vary from state to state (Vogel, et al., 1985).
- There is no built-in method to evaluate the effect of editing (Vogel, et al., 1985).
- Cross-record processing is not allowed.
- The system cannot handle the thousands of variables required by the new system of surveys.

2. Current and desired features of the Survey Processing System

2.1 Broad aims

- Procedures should be objective, repeatable, and statistically defensible (Vogel, et al., 1985).

- Procedures should be consistent across all surveys.

2.2 Impact of editing, contribution of nonresponse

- The edit system should allow a review of the number of edit actions by type and by their individual effect on the final indication (Vogel, et al., 1985).

- The edit system should allow the contribution from nonresponse to be known (Vogel, et al., 1985).

- The Board should be able to monitor how data editing, imputation for nonresponse, and adjustment for outliers affect the estimates (Vogel, et al., 1985).

- The edit system should allow the commodity statistician to monitor the relationship between raw and edited data (Vogel, et al., 1985).

- The system should retain survey data as it is reported (Vogel, et al., 1985). This would allow some measurement of how the editing process affects the estimates.

- Statistician edits should appear in separate fields from the reported data (Vogel, et al., 1985).

- Reported data should be compared with edited data (Vogel, et., 1985).

- Statistician editing should occur only after data are in computer media (Vogel, et al., 1985).

2.3 Statistical edit versus data validation

- Data validation should be distinguished from Statistical editing. Statistical editing would follow data validation (at least in the computer program), (Vogel, et al., 1985).

- Development of statistical edits at the time of edit should be carried out, (Vogel, et al., 1985 & Barr, 1984).

- Data errors identified during the statistical edit and analysis process should be resolved using statistical procedures to ensure consistency across surveys across States (Vogel, et al., 1985).

- Data analysis routines should be incorporated into the statistical edit stage of the edit (Vogel, et al., 1985).

- The system should have the ability to detect outliers and inliers (Vogel, et al., 1985).

- There should be an audit trail for adjustments from outliers (Vogel, et al., 1985 & Barr, 1984).

- Interface with analysis tools should be allowed (Ferguson, 1987).

2.4 Imputation

- Imputation procedures for both list and area frame surveys should be incorporated (Barr, 1984).

2.5 Added capacity and flexibility

- The system should allow item code validation of questionnaires by state and version.

- The system should have the ability to process states individually with their own error limits.

2.6 Ease of use

- The system should have menu driven systems that are easy to use (Vogel, et al., 1985 & Barr, 1984).

- The elimination of the hand edit is a goal (Vogel, et al., 1985).

- Data verification should be on an interactive basis (Vogel, et al., 1985).

- Customized data listings should be available (Ferguson, 1987).

- The system should provide easy and timely access to data at all levels, i.e., reporter, county, district (Barr, 1984).

- Edit checks should be easy to specify (Ferguson, 1987).

- An error description should be provided in the error printout (Ferguson, 1987).

2.7 System attributes

- The system should provide the capability to interface with the following (Barr, 1984):

    - data entry systems;

  - microcomputers/minicomputers;
  - the NASS data management system;
  - statistical packages;
  - LSF;
  - CATI and hand held data recording devices;
  - report generator.

- The system should start with the capabilities of the NASS Generalized Edit System and built from there (Barr, 1984).

- There should be an artificial limit to the number of edits (Barr, 1984).

- The system should be capable of editing and maintaining several levels of data within or between records including the ability to use previously reported data (Barr, 1984).

- There should be flexibility in data entry formats including full screen, on and off-line procedures, entry without item codes, etc., (Barr, 1984).

- The system should have survey management capabilities (Barr, 1984).

- The system should meet all agency security needs (Barr, 1984).

- A specification generator should be developed from which files for paper questionnaires, CATI and CAPI instruments, and an editing system can be generated from one specification of the survey variables.

3. Implementation

The new Survey Processing System is being written and documented. It is being used to process the Quarterly Agricultural Surveys, the Farm Costs and Return Survey, and the Prices Paid by Farmers Survey.

The emphasis so far is on handling expanded survey requirements. These include an increase in the numbers of edits and variables and the use of cross-record checks to improve data validation as it is currently handled in the agency. The system can access previously reported data and since it is in SAS, it has the capability of comparing data between records. Though data correction is still made by printout, English messages instead or error codes are printed out. It is far easier to write edits than previously and there are not limits to the number of edits that can be written. Research on some features listed above has yet to begin. This includes work on statistical edits, the automation of all or most of the corrections, and the elimination of the hand edit before data entry.

3.1 Broad aims

If the objectives of editing are objectivity, repeatability, and statistical defensibility, then they have not been fully attained in most of NASS's surveys. The current NASS editing systems primarily use within record computer checks and a subject matter specialist. In that the SPS is written in SAS, it is capable of accommodating procedures which would accomplish these goals. The attainment of these objectives is firstly a matter of definition and theory and secondly a matter of systems development.

3.2 Impact of editing, contribution of nonresponse

The Survey Processing System allows a review of the number of edit actions by type but does not allow a review of their effects on the final indications. The contribution from nonresponse for crops is made available to the Agricultural Statistics board before estimates are set. There is no provision for monitoring how data editing and adjustment for outliers affect the estimates. The system has the capability of allowing the commodity statistician to monitor the relationship between raw and edited data but as this capability has not been used, the programming code has been commented out. (That is, it is still there if anyone wants to use it). The system does not retain survey data as it is reported, nor do statistician edits appear in separate fields from the raw data. The issue of comparing reported data with edited data is problematic because CATI reports and paper reports are mixed in the same file and CATI reports are, in effect, edited at the time of data collection. Statistician edits occur both before and after the data are in computer media, not solely afterwards.

3.3 Statistical edit versus data validation

Data validation is distinguished from statistical editing in the Survey Processing System. That is, a place is reserved for a statistical editing module if the research on how best to implement a statistical edit is carried out for each survey. A statistical edit is distinguished from statistical analysis in that a statistical edit is carried out at the time of the data validation. NASS's Prices Paid by Farmers survey employs a statistical edit to search for outliers. The June Agricultural Survey edit program also does some cross-record checking at the field and tract levels but does not check for outliers. The June Agricultural Survey edit program also does some cross-record checking at the field and tract levels but does not check for outliers. An audit trail for adjustments from outliers exists for the Farm Costs and Returns Survey but not for the other surveys. An interface with analysis tools is in place.

The Concept of statistical editing in NASS remains undeveloped. NASS's analysis packages serve some of the same purposes as a statistical edit, in that the analysis packages serve to detect outliers. A major difference between the analysis packages and the statistical edit as listed above is that the analysis packages are run after data validation and imputation and not at the time of data validation. Another difference is that edit limits are not generated from the analysis packages. Statistical editing may refer to cross-record comparisons of current survey records for one or a set of variables. The concept may also refer to using historical data within a record in order to set individual error limits for each firm.

3.4 Imputation

The capability to implement procedures for both list and area frame surveys does not yet exist within the Survey Processing System. Imputations are being carried out in the Quarterly Agricultural Surveys but these imputations are not done within the SPS. They are carried out between data validation (in the SPS) and analysis. Though imputed values are not rerun through the validation edit, it is possible to see the effects of some imputations in the analysis package. If at that point there are any glaring imputation mistakes, they can be corrected.

3.5 Added capacity and flexibility

The Survey Processing System allows as an option the validation of questionnaires by state and version. The capability of states to process data with their own error limits is also available.

3.6 Ease of use

The Survey Processing System has some menu driven systems in place with which to generate parameters for editing and analysis. The elimination of the hand edit is an unrealized goal for NASS as a whole, although there are states that have eliminated it. Data validation is not on an interactive basis, but since the system is in SAS, this could be done either on the mainframe or on microcomputers. In order to install interactive editing, an editing interface would have to be written so the records could be immediately re-edited when changes are made. Customized data listings are available and this capability is being updated. The system allows easy and timely access to data at all levels because it is written in SAS. The specification of edit checks is much easier than before although not as easy as desired. However, a better specifications generator will be created in the future. Error descriptions instead of error codes are provided in the printout.

3.7 System attributes

The Survey Processing System, utilizing the many SAS capabilities, has the capability to interface with all present or anticipated NASS data handling systems. The SPS has all the capabilities of the old system and many more. There is no artificial limit to the number of edits. The system can edit and maintain several levels of data within or between records and can access previously reported data. Data can be entered in a variety of ways including by item code, full screen entry, and on and off-line procedures. The system has some survey management capabilities but these are to be improved. The system fully meets all of NASS's security needs. The specifications generator has yet to be developed beyond its rudimentary beginnings.

4. Discussion

Not all of the desired features listed above have been implemented or even researched. However, considerable progress has been made especially in the realm of system development. Much effort has been put forth in order to preserve maximum flexibility in the Survey Processing System. Thus the system has the potential to accommodate almost any mathematical or statistical procedure, to be used on many kinds of computers, and to allow more powerful tools to be put

in the hands of the editing personnel. The limiting factors are the development of theory, research, money, and the vision of management.

4.1 The role of the data editors in the field

Aside from the listing of some desired features of the system, no explicit discussion has been offered in NASS documents as to the job content of the data editors, that is the data entry personnel, the clerks, and the statisticians, as regards editing. Several scenarios have already been presented in section II.3. Though some changes have taken place from the standpoint of the field editors, the SPS has not yet impacted their job in a major way. However, future implementations of the SPS have the potential to change their job content dramatically, from reducing the amount of editing the personnel are expected to do to supplying more powerful tools to do the same job. Keeping in mind the constraints of resources and goals of NASS, such as objectivity, repeatability, and defensibility, the editors themselves should be brought into the process as regards how they are to act in the system. For example, if interactive capability is to be introduced, the people who are to use the system should have a hand in the design of the interfaces. This includes clerks as well as statisticians.

4.2 Integration of CATI and CAPI with the SPS

Even with the introduction of CATI and CAPI, some sort of editing system will be needed, as not all editing functions are carried out with these data collection technologies. Some consideration will have to be given as to how the editing functions will be divided between the SPS and CATI and CAPI. For example, it is not likely that cross-record checks could be successfully carried out in a CAPI environment if those checks were to involve records collected on another computer. On the other hand, CATI records collected on a LAN could be checked with a statistical edit conducted on a LAN.

4.3 Research and implementation

Two research projects are being conducted in the Research and Applications Division. The first, by Antoinette Tremblay and Ralph V. Matthews, is entitled "A Track of Wheat Objective Yield Raw Data to Final Summary". This study tracks data from its reception in the office to final summary. Estimates are calculated at each stage of the editing and summary process to track the effect of the editing process on the level of estimates. The aim of this report falls within the realm of collecting management information. That is, it attempts to measure the effects of current editing procedures. This report, and others like it, will impact the SPS indirectly by pointing out areas in which research should be conducted.

The second project by Cathy Mazur is entitled "Statistical Edit System for Weekly Slaughter Data". The aim of this research is to determine whether it is possible to use each firm's historical data in a time series model to edit current data. It is possible that this research will be incorporated directly into the SPS for some specialized surveys, especially for those dealing with agricultural business firms.

A statistical edit has been implemented for the Prices Paid by Farmers Survey. It remains to be seen how far this approach can be extended to other surveys as there are some differences

between this survey and the larger NASS surveys. In the Prices Paid survey, all reports are run through the system in batch. Records in each of ten regions in the country are compared on an item by item basis and regional distributions are generated for each item. A second run is used to correct data which are outliers based on current reports from the first run. A second type of statistical edit based on historical data is used in the survey. Data from the previous collection period are used to generate error limits for the next period. These error limits are manually reviewed before being used, and can be changed if necessary.

Further research remains to be done in the areas of statistical editing for NASS's large surveys; automated imputation, inspection and analysis of edits in current NASS surveys, macro-edits, statistical edits, and automation of current editing procedures, that is, interactive capability. In these areas, four groups of people must, at times, interact: systems developers, statistical researchers, users in headquarters who plan surveys and write editing programs, and data editors in the field. (See further, IV. Future Research).

4.4 Statistical defensibility, objectivity, and repeatability

It is possible that the definition of the term statistically defensible will change with different types of surveys. For example, in some commodity surveys, the sole intent may be to publish averages, totals, or rates. In these cases, the only access to the data would reside within NASS, and in theory at least, all direct uses of the data are known in advance. In the former case, imputations of averages for item nonresponse may be statistically defensible. In the latter case, this procedure probably would not be defensible, as it would have the effect of changing the structure of the data, that is, changing the marginal and joint distributions of the data. (NASS does not impute for the FCRS survey). As another example, in the Fellegi and Holt vision of data editing and imputation, imputations must agree with edits. In the NASS processing of the QAS, imputations are not run through the editing system again although implausible imputations may be detected in the analysis packages. Is this defensible? In order to attain the goal of statistical defensibility, the term will have to be defined at some point. The same will have to be done for the terms objectivity and repeatability.

4.5 Imputation

Imputation should not be performed to such an extent that major problems in data collection are masked. That is, no imputation method can overcome high nonresponse and error rates. Furthermore, if record level data are to be released to others, then imputations should be flagged, and the manner in which they were made should be documented and available to the user. To the extent that imputations are justifiable, whether they arise from correction of errors or through nonresponse, they should be made in an efficient and defensible manner.

According to Dale Atkinson (1988a), imputations for total nonresponse in the QAS are possible if one makes use of ancillary data as list frame control data or previous survey data. Imputations in the QAS are based upon extensive modelling of any previous survey or control data which re available for a particular nonresponse record. The current methods do not preserve distribution in which the primary benefit (at least for the QAS) would be in better variance estimation. (This reflects how the data are used, i.e., expansions or averages reported, only NASS uses the data, etc.). The benefits do not outweigh the costs of trying to maintain a distributional structure for each item. Methods are now based primarily upon the logic used when manually

imputing data. Atkinson suggests research to: 1)compare the NASS exact imputation procedures against alternative approaches used outside of NASS and widely discussed in the statistical literature, and 2) investigate ways to compensate for variance understatement resulting from imputation. He also states that statistical defensibility needs to be addressed.

VI. FURTHER RESEARCH

1. Describe the problem

Collect management data concerning the editing process. What are the resources being used and on which survey steps are they being used? How much cycling is there in the edit process for each survey? How tight are deadlines to be in the future? How well do enumerators follow skip pattern? What percent of the data is imputed and how does this vary by topic or item? How are estimates changed as a function of the current editing process? The answers to these questions are necessary in order to compare the magnitude of possible improvements to the costs of implementing new systems and to the costs of remaining in the current system.

Current NASS operating procedure involves performing a comprehensive hand edit on questionnaires before data entry. Thus there is no way to measure the quality or necessity of the editing process by referring to computer files. Also, it is not possible to measure changes in data between the time they are collected and after they have been processed. The only way to collect this kind of data would be to conduct a survey of questionnaires used in NASS surveys such as the Farm Costs and Returns Survey, the Objective Yield Surveys, and the Quarterly Agricultural Surveys. The ultimate sampling unit would be the questionnaire. The respondents would be NASS statisticians in the state offices. A stratified random sample of the questionnaires would be collected and each questionnaire would be rated in various ways. Data items would include measure of enumerator performance such as whether the correct skip pattern was followed. Other data items would measure office performance in terms of data quality and productivity including rating the percent of editing activity as it appears on the questionnaire as to whether it was harmful, superfluous, or good (contributed to data quality).

Other management data could be collected from headquarters by tracking certain computer files. The data to be collected here would include the extent of cycling being done for the various surveys as well as the distribution of questionnaire arrivals into the system. For a random sample of offices, every error printout file could be saved for the particular survey. After the edits for that office are clean, a matching process between files would be executed, the matching being done on the ID. It would then be possible to determine a distribution of occurrences of IDs in terms of the number of times each questionnaire passed through the computer edit.

A third type of management data that could be collected would be the time spent on various survey tasks. This would have nothing to do with time sheet recordings as used in the Administrative Records System. This measurement would be done by selected employees keeping a log of how their time was spent on each survey task. It would attempt to detect superfluous activity (e.g., filing and refiling of questionnaires) that would not appear through an inspection of questionnaires. That is, it would help to point out where new technologies could streamline the process. It would also provide baseline data for measuring productivity gains.

The purpose of collecting this management information would be to determine where research resources should be spent regarding at least two technologies. These are new editing procedures and the introduction of CAPI. For example, the rate of occurrence of enumerators following a wrong route may be less than 1% or as high as 10%. In the former case, CAPI might not be justified based on this phenomenon alone, whereas in the latter case it might be. Other vital information might be gathered in the realm of non-sampling errors, the effect of imputations whether done by hand or by machine, and the success in training enumerators. In this latter realm, one could imagine a whole different emphasis on training according to the feedback which could be gained by this agency self inspection. For example, it may be that data quality is compromised more by enumerators following incorrect paths through questionnaires than by their misunderstanding of individual questions. If so, future training schools for the survey would emphasize following the correct path through the questionnaire.

2. Classification of errors and edits

Construct a classification of edits and of errors occurring on NASS questionnaires. Determine the success of detecting the errors and the success of correcting them. List the kinds of edits being used by NASS in its surveys according to the edit classification. Determine if the edits in use by NASS would be amenable to the determination of a minimal set and feasible regions. Perhaps a subset of the edits would be amenable to such an approach and could thus be utilized to reduce the need for review by survey specialists. One other aspect of this research is to see if NASS over-edits its data. Statistics Sweden has gained a 50% reduction of error signals by inspecting and analysing edits in one survey and eliminating redundant edits in some cases and broadening bounds in others (Granquist, 1988). This reduction in noise was accomplished without a reduction in data quality and according to Granquist, a slow increase in data quality, as specialists had more time to concentrate on true problems. After every survey, NASS reviews the number of times each edit is invoked and will adjust edits accordingly. This recommendation would go beyond this routine analysis by analysing patters of edit failures. That is, are there edits which always or usually fail at the same time because they concern the same field?

3. Macro-edits

Macro-edits should be able to do two things. Edit data at the aggregate level, and trace inconsistencies at the aggregate level to individual questionnaires. Macro-edits focus the analysis on those errors which have impact on published data. These tasks are already performed to a large extent by hand referral to analysis package output. Research should be pursued along the lines of automating macro edits.

4. Statistical edits

Research on automated statistical editing, both in batch and interactive modes should be conducted. The best way to detect outliers and the best way to resolve the status of the suspicious data should be determined. The use of high resolution work stations in conjunction with interactive data analysis packages should also be explored.

5. Imputation

Conduct research on imputation, its impact on the edits and the maintaining of distributions, (see Atkinson, 1988a). Some questions which should be answered: What is the extent of item nonresponse, partial nonresponse and total nonresponse? What are the proper imputation methods for each kind of non nonresponse? How do these vary by survey topic? For example, can item nonresponse in crops and livestock be handled in the same way? By sampling frame? By survey? Is it defensible to hand impute for total nonresponse in JES tracts? Why, in the QAS, are imputed livestock records not used in the summary while imputed crops and grain stocks are fed into the summary? How do agricultural populations differ in structure from other populations and how are imputation procedures used by other organizations applicable to agency needs?

6. Statistical defensibility, objectivity and repeatability

Conduct research on a definition of statistical defensibility, objectivity, and repeatability as they apply to the editing and imputation process.

7. The Bureau of Census SPEER software

Inspect the Bureau of Census software when it becomes available. The Bureau of Census has offered to allow NASS to inspect the software they are developing for the IBM AT.

8. The Netherlands Central Bureau of Statistics Blaise software

Continue to inspect the Blaise software as it is sent from the Central Bureau of Statistics in the Netherlands. This research should be carried out regardless of the editing research, as it might be applicable to the CATI and CAPI work done in this agency. It could also stimulate research into the concept of integrating all aspects of the survey, from questionnaire design to summary.

9. Microcomputers

Determine the feasibility of using microcomputers, either solely or in LANs, to perform various editing tasks. NASS already has some microcomputer based editing and summary programs in place for special purposes, including the Peanut Stocks survey, and a system in use in Pakistan. The next logical step is to see if the Survey Processing System on the microcomputer can handle NASS's questionnaires, especially the Farm Costs and Returns Survey and the Quarterly Agricultural Surveys. Possible productivity gains could be estimated if the research in point A is carried out.

ON THE NEED FOR GENERALIZED NUMERICAND IMPUTATION SYSTEMS

by L. Granquist
Statistics Sweden

Abstract: The aim of this paper is to discuss the needs, objectives, use and achievements of numeric generalized systems for editing based on the Fellegi-Holt methodology. Findings of a study on the computerized editing of the World Fertility Survey are presented.

I. INTRODUCTION

The systematic approach to editing given in the famous paper by Fellegi and Holt (1976) has had an enormous influence on the developments of generalized software for editing. Those principles of editing are adopted as the underlying methodology in systems for editing and imputation of qualitative data (CAN-EDIT, AERO, DIA) and now also of quantitative data (NEIS, SPEER, CANADA's planned system). In the following, all those systems are denoted GS (generalized systems). The basic principles of the Fellegi-Holt methodology are:

- data in each record should satisfy all edits;
- imputation rules should be derived from the edit rules without explicit specification.

However, evaluations of the GS applications are very sparse, as are critical discussions on means and ends, which problems are actually solved and which are not, if the means are rational considering what can be accomplished and so on. The aim of this paper is to discuss and question the needs, objectives, uses and achievements especially of numeric GS. It is based on the findings of a study of the machine editing of the World Fertility Survey (WFS), as they are reported in Pullum et al (1986): The machine editing of large sample surveys: The experience of the world fertility survey. These experiences are consistent with other reported experiences of editing.

The Pullum paper discusses the machine editing of the WFS. Although GS were not used, the evaluation of the WFS editing is in fact an evaluation of the basic principles underlying GS.

The authors of the study present their findings in the form of recommendations for similar types of surveys or survey projects and it is stated that all recommendations are valid for any editing process. Some of them are given in this paper and comments are made on their implications for GS.

II. THE STUDY OF THE MACHINE EDITING OF WFS

1. WFS and its aims concerning editing

"The World Fertility Survey (WFS) conducted 42 large surveys in developing countries using complex schedules and did as much of the data processing as possible within the countries.

One of its major ambitions was to produce data of first-rate quality. This goal was sought in every phase of operation including questionnaire design, sample selection, interviewing, data processing and analysis of findings.

The aims concerning editing were:

- WFS should be a hallmark of professional survey research;

- WFS should serve as a vehicle to introduce modern editing to statistical offices in developing countries".

It means that experts on editing were involved in developing the program for every country. The checking and updating were wholly computerized. The reconciliation was performed manually, that is a knowledgeable individual examined the original questionnaire along the error printout and wrote out a correction statement.

Hence the error identification and the subsequent adjustment operation was not automated. This could have been achieved by a GS. The WFS surveys fulfils all conditions for applying the completely automated options of GS. (Evidently, WFS can furnish excellent data for testing and evaluating GS under ideal conditions for this type of editing systems).

2. The study

Six countries were selected for this study, simply on the availability of early raw data files. It is claimed in the report that "if these countries are representative with respect to data quality, it is probably because their data tend to be poorer than average. That is, the effects of editing may tend to be somewhat exaggerated with that choice of countries".

The study consisted in comparing the dirty (unedited) and clean (edited) pairs of files in:
- diagnostic marginal distributions;
- two-way tables;
- fertility rates; and
- multivariate analyses.

3. The scope of the WFS study

The study is limited to the machine editing in WFS. This editing was preceded by two kinds of edits, which were entirely manually done:

- Field edits: Undertaken during the interview by the interviewer and his supervisor with the possibility of getting new information from the respondent.

- Office edits: Carried out by clerks before the data entry operation and consisting of coding and checking of answers.

Structural editing was excluded from the study. The report says: "It will be assumed that structural checking must always be done. If it is not done, then it is difficult to interpret the files properly".

However, structural editing cannot be done in existing GS.

Maybe it is not absolutely necessary to have an automated structural editing function in a GS. In the WFS study, all records with original structural edits were matched away from the files. It resulted in surprisingly low percentages of cases lost, namely 2.32; 1.51; 6.19; 0.36; 0.00; 0.56 for the six countries selected for the study.

Thus the dirty file consisted of data, which had been edited by field and office edits only. The clean files were the finally processed files, where all cases with structural errors had been matched away.

4. Findings

The most notable weakness in the dirty estimates was that the fertility rates were too low. This deficiency was traced to a feature of one particular program.

The rather elaborate logit regression and multiple regression differed surprisingly little between the dirty and clean files.

The multivariate analyses were relatively insensitive to the editing. Specifically, inferential changes about the magnitude of effects and their statistical significance are almost always less than differences between two independent and clean samples.

The cost of the machine editing is crudely estimated to be an average delay of approximately one year. Compared to the benefits, such delays are excessive".

My summary of these findings is that (i) the machine editing has no impact at all on the analysis, and (ii) the machine editing caused a delay of every national survey for one whole year.

Furthermore, the study led to the following overall statement :

"Consistency is more desirable because of later data processing convenience than because of its value for analysis, but it should be achieved quickly and practically and never by referring back to the physical questionnaires".

The latter part of this statement is in accordance with the Fellegi-Holt methodology. What can be questioned is whether the automatic imputation methods are in accordance with the first part, which says that it does not matter which imputation methods are used.

III. OUTCOMES OF THE STUDY

1. A few concepts

To interpret and discuss the findings of the WFS study and to clarify the following comments and conclusions, a few concepts concerning errors and editing problems are reviewed. These concepts are discussed in detail in Granquist (1984).

Errors are classified either as negligence errors or as misunderstanding errors on the basis of two dimensions, number of cases and reason for the error. Misunderstanding errors are those affecting a number of records and are consequences of ignorance, poor training or misunderstanding, or are committed deliberately for some reason or other. All other errors are classified as negligence errors.

Generally, a negligence error is the result of carelessness by the respondent or by the survey process up to the editing phase. In a repetition of the survey, the same error should probably not be found for the same item of the same record (questionnaire).

In the processing phase of a survey, errors cause problems of two kinds, namely:

they may distort the quality of the data, that is, the data may not meet the quality requirements;

they cause problems in the processing of the data.

The first kind of errors is called quality errors and the second kind is called process trouble errors. A specific error may be a quality error as well as a process trouble error.

Bearing in mind that negligence errors sometimes affect the quality, it can be said that process trouble errors are likely to originate among the negligence errors and the anticipated misunderstanding errors, and that quality errors are likely to originate among the unknown misunderstanding errors.

2. The editing problem

Errors occur in every survey both in the collection phase and in the production phase. Furthermore, every particular survey has to meet some quality requirements, although these generally are not very precisely formulated. The inevitability of errors and the presence of quality

requirements is traditionally solved by microediting without first posing the question: what is the problem caused by errors in data and how should it be rationally solved?

3. The editing of WFS

The WFS study is focused on the WFS editing problem. The authors say: "The study has been motivated by conflicting reactions from users of WFS data. Some users are pleased that the data tapes are free from internal inconsistencies. Other users are frustrated by the length of time between the fieldwork and the emergence of the principal results (often more than three years and rarely less than two).

The latter consider the editing task to consist of fighting process trouble errors and that WFS spent too many resources and too much time on it (the study assessed the average delay to one year.)

Editing in the WFS sense according to the Pullum paper is intended "to detect whether the various responses are consistent with one another and with the basic format of the survey instrument and to resolve any detected inconsistencies through adjustment. Editing is not properly described as the correction of errors, conversely a good many errors, of many kinds, will not even be touched by the editing process. (The point here is that there is no way of genuinely validating any of the responses)."

This is a perfect diagnosis of the objective and the effects of modern GS. That is why the evaluation of the machine editing in WFS could be considered as an evaluation of GS applied in a special type of surveys.

GS editing should not be considered as a correction of errors and consequently, use of a GS should not be induced by quality reasons. The users of GS should be aware of this. A GS should be regarded as one facility or tool among others for solving the process trouble errors in the survey. This limitation of GS is discussed in Granquist (1983).

One very plausible explanation to the fact that there was not any improvement in the quality at all, is that the data capture operation of WFS was extremely successful. The field edits removed all such errors (and a good many of the process trouble errors). A data capture editing procedure might have been a useful tool. If the editing problem was to fight process trouble errors (due to the quality of all survey operations of WFS), then the editing task could have been excellently solved by, for example, the Dutch system BLAISE (see Bethlehem et al (1987)). A GS should not be applied in such a case.

WFS uncritically adopted the following four statements concerning editing survey data:

(i) "Editing is generally believed to produce a gain in the yield of the fieldwork"

Major components of the cost of a survey are:

- the design of the questionnaire;

- the sample;
- interviewing (the field-work).

Editing is small compared to these, it will make more cases and specific items usable and will minimize the number of responses which must be excluded from analysis."

My comments: WFS is talking about removing formal errors (process trouble errors), which make cases and specific items unusable. The cost of editing may be small compared to the cost of carrying out a new survey, but nevertheless, the costs of editing are heavy. Concerning periodic surveys, editing as it is now carried out by our offices takes 20-40% of the survey budget.

(ii) "Editing is believed to improve the validity of the findings"

Estimates on edited data will tend to be closer to the population values, which they intend to estimate. This is based on the belief that discrepancies in data tend to be systematic rather than random and will introduce a certain kind of bias."

My comments: It is essential to note that the most apparent weakness in GS and computer-assisted editing on the whole is just the handling of misunderstanding errors (systematic errors). This can be read in Hill (1978) and is discussed in Granquist (1983).

The only GS which has faced this problem seriously is the Spanish DIA. It has a solution for anticipated or known systematic formal errors, which is incorporated within the Fellegi-Holt methodology. However, it can be questioned whether this is a rational solution. The statement below concerning imputation methods in GS is valid also for this deterministic imputation method.

In WFS, there was no improvement of the quality at all, maybe because they were no systematic errors present in the dirty data files. However, there is no improvement to expect if the editing process is not focused on the detection of systematic errors (misunderstanding errors), Granquist (1983).

(iii) "Editing improves the correspondence between the structure of the questionnaire and that of responses"

Internally consistent data greatly facilitate tabulation, even though the conclusions may not be affected."

(iv) "A user has more confidence in data which are internally consistent."

Only statements (iii) and (iv) are real benefits of editing in the sense it was carried out in WFS. However, it is very important to note that this does not necessarily mean that all inconsistencies have to be removed. It is sufficient to remove the errors which obstruct the tabulation and are not otherwise detectable by a user.

An explanation of the excessive time spent on editing in WFS may be that the ambition was to remove all format errors. Everything that could be checked was checked. Please note that

this mistake is committed in every editing process. Every GS is designed to execute all possible checks in order to meet any requirement by the user, who is encouraged to use as many checks as he believes is necessary. This has caused every GS to be very complicated and very expensive to run. Subject-matter specialists cannot use the software without considerable help from EDP-specialists.

The mistake of too many checks apply even more to periodic business surveys, the scope of numeric GS. There are many other possibilities of checking, due to the existence of relations (correlations) between certain variable (items) and the possibility of doing comparisons (in general "differences" or "ratios" are used as checks) with data from earlier periods.

"When viewed in the context of sampling errors, data entry errors, and other non-sampling errors, the significance of sophisticated editing procedures appears diminished. One might even argue that meticulous editing gives the user a false sense of confidence in the data. (It is difficult, however, to assess the improvement in data quality from editing, relative to the total survey error, because we do not know the magnitude of the total survey error)".

However, the most serious problem is that numerous checks in editing processes convey an illusory confidence in the quality of data to the survey staff.

4. Is editing in the WFS or GS meaning necessary?

The importance, whether psychological or more analytic, of agreement among subtotals, etc. cannot be ignored. Especially for advanced and comparative analysis, it is indeed important that the data be clean. Data users expect them to be clean, and if they are not, users will embark on their own cleaning exercises. One cannot deny the consensus among statisticians, users and data processors that data files should be free of inconsistencies.

The study accepts that it is indeed desirable to run edit checks using software and specifications prepared in advance and to achieve consistency in the data even though this consistency may have a negligible impact on the analysis. The issue, then, is not whether the data should be internally consistent but how this condition will be achieved.

The study indicates that machine editing can and should be done with greater efficiency.

The main recommendation is that the potential difficulty of achieving consistency should be anticipated and a strategy be developed well in advance, dependent on the circumstances.

These findings seem to justify generalized software for complete automatic editing according to the Fellegi-Holt methodology.

This becomes an issue of cost-benefit analysis. The cost for using a particular GS may be calculated relatively easily. The benefit is of course more difficult to estimate. From the study, we know that the editing may not have any impact on the quality. The dirty and the edited files gave the same results. This means that any reasonable imputation method will remove inconsistencies. The only condition is that the method should be very fast and cheap.

The methods for imputations in modern GS seem to be far too sophisticated when facing the results of the WFS. This, in combination with my own conclusion from the study that they used too many checks like in all other editing processes, leads to the conclusion that such generalized software is too sophisticated to use for cleaning statistical data files from formal errors.

This paragraph can be summarized as follows:
- it is necessary to remove certain errors from the data-files;
- every check has to be justified by the error it is intended to detect, and this error has to be detectable by a user studying the published tables as it may cause trouble in the further data processing;
- imputations should be very simple to permit fast and cheap computer runnings.

5. WFS experiences applied to quantitative editing

There are three additional problems concerning the editing of periodic business surveys compared to interview surveys with mainly qualitative data, namely:
- single errors may have an impact on the estimates;
- the checks on quantitative data may only indicate suspicious data, some of which may be correct;
- it is possible to go back to the respondent for verification of suspicious data.

In WFS, the editing cost tens of thousands of dollars a year per country and above all prolonged each survey by one full year. According to the Pullum paper, the main reason is the time it took to handle the error messages by going back to the physical questionnaires.

This is only the first step to verify suspicious data in a typical periodic business survey editing process. Very often, this has to be followed by a contact with the respondent, which is very expensive. This should imply that the checking system be tuned carefully to the data to be edited. Generally, this is not done.

At Statistics Sweden and in almost every statistical office, the checking system is designed according to:

- the mass checks principle;
- the safety first principle.

This implies that the system has too many checks which identify far too many data as suspicious. Besides, most of the errors found are of no importance to the estimates.

These facts explain why the results from the WFS study may be so important. It discovered that editing did not change the estimates. Translated to a typical well-planned periodic survey, it means that there are very few errors which may have an impact on the quality (given that there are no systematic errors present in the raw data file).

Thus, if there are only a few important errors in a datafile, the editing process should be designed in a manner appropriate to those errors, and not as if there were lots of errors. Certainly,

it should not be based on possible checks. Then, it is not justified to use the automatic versions of numeric GS.

6. Drawbacks of GS in economic surveys

A GS has three features which distinguishes it from an error detecting system (ED), that is, systems with only manual reconciliation. These features are:

- the rule analyzer;
- the automatic procedure;
- the formalized checks.

The role of the rule analyzer is to guarantee that the automatic "correction" procedure always imputes values which permit the record to pass all the editing checks. This principle is essential in the Fellegi-Holt methodology, but causes the GS to be very complicated and imposes restrictions on the possibilities to detect errors other than formal errors. This is because only "domain" checks can be used. An observation on an item has to belong to the domain of the variable (validity check) and to the multivariate domain of any combination of any of the variables of the statistical unit (consistency check). In numeric GS, much work is spent on minimizing this multivariate domain of all items of a survey.

A trade-off of the rule analyzer is that it can check the editing rules against certain inconsistencies. However, this is of very limited value, because in practice, the checks are formulated with sufficient care in detail in BOC annual conference report (1987).

The automatic correction procedure determines which fields to impute and then executes the imputation. The procedure may be of value only in surveys in which the ingoing quality of data is high. Systematic errors should be sparse and generally anticipated.

The checks have to be formulated in a special way, but there is not a severe restriction for the user. Much worse is that only formal errors can be handled (because of the rule analyzer). In periodic economic surveys, it is probably necessary to point out suspicious data. Serious errors may be located within the multivariate domain of the variables.

In brief, the rule analyzer and the automatic correction procedure are not needed in well-planned surveys and impose restrictions on the detection of certain kinds of serious errors in economic surveys.

7. Macro editing, an alternative to GS

The merit of GS is that they can reduce the resources spent on editing. However, GS are not efficient tools for every kind of survey. Alternatives are needed, in particular for periodic business surveys. Such an alternative is macro editing.

Generally, error detecting programs, like the systems used in WFS, are used for the editing of economic surveys. The basic problem of all traditional record-by-record editing is to discern those observations which contain the most important errors. Such procedures exclude any possibility to assess the importance of an error just when the error messages are handled. Every

flagged observation has the same weight and claims about the same amount of resources irrespective of the importance of the suspected error. Many errors have a negligible impact on the estimates because the magnitude of the error is small or the errors cancel out.

The only methods to solve those problems are:

- to construct checks more sensitive to observations of high relative importance than to observations of low relative importance;
- to tune the whole system of checks very carefully to the data to be edited.

But even in cases where efforts have been spent to focus on potentially important errors, there are clear indications that too many resources are spent on editing in comparison with the outcome. Indeed, the endeavours have improved the editing process essentially, but are far from sufficient.

Granquist (1987) discussed this in detail in relation to a macro-editing procedure (MAEP) applied on the delivery and orderbook situation survey. This MAEP was found superior to the traditional editing system, which was as well as possible tuned to the material. The principal idea behind MAEP is to study the observations which have the greatest impact on the estimates.

MAEP has been constructed as an interactive menu programme written in APL. There are three functions to select the records to be studied, namely:

- the highest positive changes;
- the highest negative changes;
- the highest contributions.

These functions can be applied to the total (the whole industry) and each of the 38 branches. For a selected function and domain of study, the screen shows the 15 records of the indata file which have the highest value (weighted) of the selected variable, sorted top down.

Having all the top 15 records on the screen, the user can select any of the records shown with all its contents to find out if an error has been committed. If one is identified, he can up-date the record directly on the screen and immediately see the effects.

Another method which has proved to be usable is to apply an error-detecting system (EDIT-78) twice. First, checks are run on the aggregates and then on the records of the file with the suspicious aggregates.

In Hidiroglou et al (1986), a method called "statistical edits" is described. It is intended to be included in the new GS, which Statistics Canada is developing in detail in Giles (1987). A GS based mainly on such "statistical edits" (a better word for macro editing in this context) may be a good solution for focusing the editing on important errors in well-planned surveys.

IV. FINAL REMARKS

GS do not solve any quality problems. They can only clean-up the data files. However, this is done to an unnecessary extent and too sophisticated methods for imputations are generally used. The software is too complicated to develop and too difficult and expensive to use in respect to what really can be accomplished. This is very much due to an unconscious adoption of the mass checks approach to editing and to features of the Fellegi-Holt methodology. This software encourages checking of all that can be checked instead of focusing the editing process on the detection and handling of important errors. In the way such systems are used, they give the producer an illusory confidence in the quality of the data and the user of the survey a false sense of confidence in the data.

According to the experiences of WFS and other evaluation programs, such generalized software is justified if:

- it can be used by subject-matter specialists without help from EDP-specialists;
- it is cheap to run;
- it is used with common sense, that is only necessary checks are included;
- the user of the software is aware of its limitations concerning the improvement of the quality of the data. In other words, this kind of editing (that is the cleaning operation) should be considered as the first step in an editing operation focused on misunderstanding or systematic errors.

All editing processes are designed as if there are numerous errors present in the data. They should instead focus on important errors and not on possible checks.

An alternative to the discussed numeric edit and imputation approach for the cleaning of data of periodic economic surveys is to use the above outlined macro-editing methods in combination with carefully selected checks on the record level.

EVALUATION OF DATA EDITING PROCEDURES: RESULTS OF A SIMULATION APPROACH

by Emiliano Garcia Rubio and Vicente Peirats Cuesta,
National Statistical Institute of Spain

Abstract: There is a general controversy about the impact of editing on the data quality. Arguments for and against editing are widely used; however, there are not many studies to support any evidence. The present paper describes one of those studies. It has been conducted within the frame of the Data Editing Joint Group (DEJG) of the ECE/UNDP Statistical Computing Project, which had, as an important task, the production of information about the suitability of the different types of editing.

I. INTRODUCTION

The study presented here compares two editing approaches of a data file subject to errors by means of different experiments. These approaches are:

- Record Level Imputation (RLI) Based on the Fellegi and Holt methodology, RLI claims to take advantage of the existing relationship among the record variables. As variables in a questionnaire are generally correlated, the value of one variable has "some information" on the values of other variables. That is, there is some redundancy in each questionnaire. The Fellegi and Holt methodology, and other imputation strategies, try to take advantage of this redundancy, in order to edit the records; and

- Weight Adjustment at the Aggregated Level (WAL)
WAL consists in the expansion of the "good records" as an estimation of the "total records". This approach is based on the assumption that the distribution of the "Good" records is the same as the distribution of the "total records".

In this study, only "qualitative data" have been used and only "random errors" have been considered. The variables are classified in two types:

- "flow variables", that is, the variables whose values determine the flow of the questionnaire, and, in this sense, are control variables; and

- "semantic variables", that is, those irrelevant for the flow of the questionnaire.

The experiments aim to verify the hypothesis that for "qualitative data" and "random errors", assuming redundancy among the questionnaire variables, the estimates obtained with RLI are better than the estimates produced with WAL.

The paper is divided into four parts. The first part, introduces the methodology involved. The second part describes the files and the steps of the experiment. The third part presents the results of the experiment. The fourth part are concluding remarks.

II. METHODOLOGY

In an experiment about editing and errors, we must define what we consider erroneous data and what are the possible type of errors.

Erroneous data is the data rejected by any type of edit rules. We say that the data rejected contains either a "random error" or a "systematic error". Although it is not a clear cut on their definition, we say that "random errors" are those errors caused by a sum of small undetermined factors; they are well described by an error model, and slightly affect the data distributions. Systematic errors, on the other side, may be caused by well determined mechanisms and may bias the data distributions.

Here we only deal with random errors, which were added to this study data in the following way:

Let's denote A(1),...,A(i),...,A(r) the valid codes of the variable A and by N the number of records. For each field, A in our example, we chose a priori global level of error A, for instance: 2%, 5%, 10%. We also randomly chose the records to be modified. That is, for an expected number of $N.\varepsilon A$ records, we replaced at random the previous code A(i) by A(j), (i=j), or by X (generic invalid code). We applied this process to each record field or to a selected subset of the records.

Previously to adding in for building the file of "good records" was:

- to define a set of conflict rules and valid codes for each variable.

- to apply these rules to a file and to remove the records detected with erroneous data.

General observations on the experiment are:

Firstly, the way of selecting the code to modify a variable A depends on what we want to evaluate. 1) If no information is held on variable A type of errors, then we change A(i) by A(j) (j=i) or by A(X) with probability 1/r. 2) If we know something about the possible type of errors, then we choose a priori probabilities:

$$\sum_{j=i}^{r} \varepsilon i(j) + \varepsilon i(X) = \varepsilon A$$

where:

$\varepsilon i(j)$: is the probability of change the code A(i) by A(j),

$\varepsilon i(X)$: is the probability of change the code A(i) by the generic invalid X and

εA : is the error level for variable A.

Secondly, the same rules used to obtain the "good records" file will be used to check the simulated file.

Thirdly, it is clear that any editing procedure do not totally detect the "errors" added to the file. Given this fact, in the analysis of the results we can distinguish between detected and not detected errors.

To close this section, we now describe indicator Φ, used to compare the two imputation strategies.

It is a comprehensive indicator of the distance between two relative distributions for the codes of variable A:

$$\Phi = \sum_{1}^{r} |f(A(i)) - g(A(i))|$$

where f(A(i)) is the relative frequency of the code A(i) in the first distribution and g(A(i)) in the second. In this study, only the marginal distributions have been considered.

For each file, we also considered the total number of records and the relative marginal distributions of each field valid and invalid codes.

III. DESCRIPTION OF THE EXPERIMENT, FILES AND STEPS

1. The experiment files

In this study, two different surveys and files were considered:

a) - The EPA file, a file with data from the Spanish labour force survey ,"Encuesta de Poblacion Activa" (EPA). EPA is a complex survey containing a hierarchical file edited in several steps, which combines both manual and automatic procedures. A generalized software for qualitative data editing, DIA[1] is used in the automatic imputation. For this study, we selected the step in which the most important variables and the "Auxiliary variables" are edited using DIA. (The "auxiliary variables" are variables that substitute

[1] DIA is a generalized software for data editing of qualitative data developed by INE, Spain.

blocks of real variables of the questionnaire in order to edit the flow of the questionnaire).

Figure 1 presents the variables in the EPA file (Only the variables relevant for the study are described).

Figure 1

VARIABLE	CODES	DESCRIPTION (*)
AGE	16-99	Age in years
SEX	1,6	Sex (1: male, 6: female)
NAC1	1,6	Nationality (Spanish or not)
SILA	B,1,6	Occupational status
SERVM	B,1,6	
ESPEX	B,1,6	
FUNEX	B,1,6	
NOTRAB	B,1,6	
BUSCA	B,1,6	If the interviewer is looking for another job
RZBUS	B,1,5	Why is he (she) looking for another job
OFEMP	B,1,3	
CICLO1	1,6	
AUFIN	0,1,6	Auxiliary var.: block for foreigners
AUESTU	0,1	Auxiliary var.: block for studies
AUOCUP	0,1,6	Auxiliary var.: block for characteristics of the job
AUF	0,1,6	Auxiliary var.: block for people without job
AUG	0,1,2,3	Auxiliary var.: Block for people looking for a job
AUSIDI	0,1	
AUMUN1	0,1	
(*) = Blank		

To add flexibility to the study, we decided to create an artificial file, called MEPA.

b) - The MEPA file. It is a file with a subset of the EPA variables, the codification of some of them has been simplified. We built an artificial file, which initially had all the possible combinations of the variable codes (the cartesian product: 202.500 records).

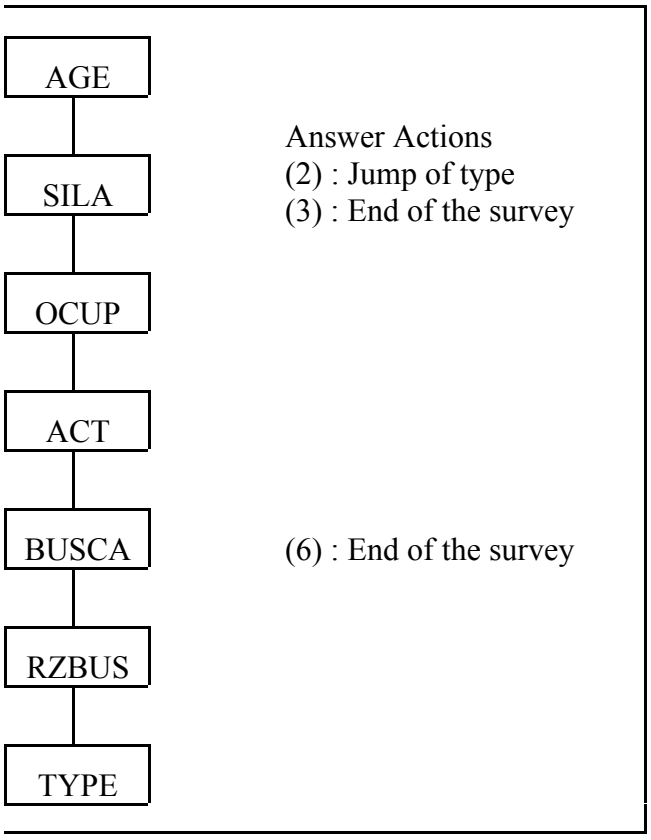
Figure 2 presents a description of the variables and codes.

Figure 2

VARIABLE	CODES	DESCRIPTION
AGE	1-9	Age in decades
SILA	1-3	Occupational status, the meaning of the codes are: -1-: working -2-: not working and looking for a job -3-: not working and not looking for a job
OCUP	B, 1-9	Occupation. The codes have no meaning. (The idea is to use it in the different experiments with more or less redundance with other variables).
ACT	B, 1-9	Branch activity of the job. The use of the codes is here similar to the variable OCUP
BUSCA	B, 1-6	If he (she) is looking for another job JOB -1-: Yes -2-: No
RZBUS	B, 1-4	Why is he (she) looking for another job -1-: to improve the current job -2-: as a complement to the current job -3-: because the current job is unstable -4-: other reasons
TYPE	B, 1-4	What type of job is he (she) looking for -1-: full-time -2-: part-time -3-: of another type -4-: any of them

Figure 3 presents the flow of the EPA questionnaire

Figure 3



2. The experiment steps

Each experiment had the following steps:

- **Step 1 - Creating the "consistent file" (FILE.GOOD)**

Initially we had an ORIGINAL.FILE. As we said before, the EPA file was a real quarterly file, and, the MEPA file, was a simulated file with all possible combinations of the codes. For each experiment, the ORIGINAL.FILE was processed against the editing rules with DIA, obtaining a file without errors. This is FILE.GOOD which has the "correct data"; that is, the ones we tried to approach to in our estimations;

- **Step 2 - Producing an "erroneous file" (FILE.ERROR)**

We introduced controlled errors in FILE.GOOD, as described in chapter II, and we obtained the FILE.ERROR. Generally, we selected the erroneous codes with proportional probabilities;

- **<u>Step 3</u> - Detection of errors in "FILE.ERROR"**

The FILE.ERROR was processed against DIA. The program produces the following two files:

- FILE.GOOD2 with the error-free records from FILE.ERROR. This file was used to obtain the marginal distributions, that we call WAL. (Weight Adjustment at Aggregated Level); and

- FILE.ERROR2 with the erroneous records from FILE.ERROR.

- **<u>Step 4</u> - Imputation of FILE.ERROR2**

FILE.ERROR2 was imputed using DIA and the file resulting from this imputation was appended to FILE.GOOD2 obtaining FILE.EDITED. This file was used to obtain the marginal distributions RLI estimates.

- **<u>Step 5</u> - Computing the Indicator Φ for WAL and RLI**

From the distributions of the variables in FILE.GOOD (step 1), and the distributions of the variables in FILE.GOOD2 and FILE.EDITED (steps 3 and 4), we computed the indicator Φ for WAL and RLI:

Let's denote by F the marginal distribution of a variable (A in our example), in the FILE.GOOD, by F1 the same in FILE.GOOD2, and by F2 in the FILE.EDITED. Then Fi is the marginal frequency of the code A(i). If we denote:

$$D1i = |Fi - F1i|$$
$$D2i = |Fi - F2i|$$

then the Φ indicators are obtained by:

$$\Phi WAL = \sum_{i=1}^{r} |Fi - F1i| = \sum_{i=1}^{r} D1i$$

$$\Phi RLI = \sum_{i=1}^{r} |Fi - F2i| = \sum_{i=1}^{r} D2i$$

Comparing the indicator values ΦWAL and ΦRLI, we say that the smaller the indicator value the better is the editing method (because when the indicator is small, the distance between the estimation and the true data is also small).

IV. RESULTS OF THE EXPERIMENTS

1. EPA File:

Step 1 Once processed, the EPA.ORIGINAL.FILE with DIA, we obtained the EPA.FILE.GOOD with 139.796 records

Step 2 After introducing errors, we had the EPA.FILE.ERROR

Step 3 We detected errors with DIA, obtaining:
EPA.FILE.GOOD2, with 136.580 records
EPA.FILE.ERROR2, with 3.216 records

Step 4 We imputed the erroneous records with DIA, and appended them to those of EPA.FILE.GOOD2. The resulting file is EPA.FILE.EDITED. We obtained the RLI estimators from this file.

The following tables present the indicators obtained for WAL and RLI in EPA:

FREQUENCIES AND INDICATORS IN EPA

VAR.1: SEX

COD	F *	F1	D1 *	F2	D2
X	0*	0	0 *	0	0
B	0*	0	0 *	0	0
1	484*	477	7 *	483	1
6	515*	522	7 *	516	1
		Φ WAL=0	* Φ RLI=6		

Where:

X : Invalid code
B : no answer (blank code)
F : Frequency (per 1000) in the -EPA.FILE.GOOD-
F1: Frequency (per 1000) of the good data in the EPA.FILE.GOOD2- (this is for WAL)
D1: Difference between -F- and -F1-
F2: Frequency (per 1000) of the edited data in the EPA.FILE.EDITED- (this is for RLI)
D2: Difference between -F- and -F2-

$\Phi WAL = \Sigma D1i$

$\Phi RLI = \Sigma D2i$

VAR.2: NAC1

```
COD    F *  F1    D1 * F2  D2
)))))))3)))))))))3)))))))))
 X      0*     0   0 *   0  0
 B      0*     0   0 *   0  0
 1    998*   998   0 * 995  3
 6      1*     1   0 *   4  3
S)))))))2)))))))))3)))))))))
            Φ WAL=0 * Φ RLI=6
```

VAR.3: CICLO1

```
COD    F *  F1    D1 * F2  D2
 )))))))3)))))))))3)))))))))
 X      0*     0   0 *   0  0
 B      0*     0   0 *   0  0
 0      0*     0   0 *   2  2
    1000* 1000   0 * 997  3
  ))))))2)))))))))3)))))))))
            Φ WAL=0 * Φ RLI=5
```

VAR.4: AUESTU

```
COD    F *  F1    D1 * F2  D2
)))))))3)))))))))3)))))))))
 X      0*     0   0 *   0  0
 B      0*     0   0 *   0  0
 1      0*     0   0 *   0  0
 6    998* 1000   2 * 999  1
)))))))2)))))))))3)))))))))
            Φ WAL=2 * Φ RLI=1
```

VAR.5: AUSIDI

```
COD    F *  F1    D1 * F2  D2
)))))))3)))))))))3)))))))))
 X      0*     0   0 *   0  0
 B      0*     0   0 *   0  0
 0     10*    10   0 *  10  0
 1    989*   989   0 * 989  0
)))))))2)))))))))3)))))))))
            ΦWAL=0  * Φ RLI=0
```

VAR.6: AUMUN1

```
COD    F *  F1    D1 * F2  D2
)))))))3)))))))))3)))))))))
 X      0*     0   0 *   0  0
 B      0*     0   0 *   0  0
 0   1000* 1000   0 * 997  3
 1      0*     0   0 *   2  2
)))))))2)))))))))3)))))))))
            ΦWAL=0 * ΦRLI=5
```

VAR.7: SERVM

```
COD    F *  F1    D1 * F2  D2
)))))))3)))))))))3)))))))))
 X      0*     0   0 *   0  0
 B    991*   991   0 * 991  0
 1      0*     0   0 *   0  0
 6      8*     8   0 *   8  0
)))))))2)))))))))3)))))))))
            ΦWAL=0 * ΦRLI=0
```

VAR.8: ESPEX

```
COD    F *  F1    D1 * F2  D2
)))))))3)))))))))3)))))))))
 X      0*     0   0 *   0  0
 B    998*   998   0 * 998  0
 1      1*     1   0 *   1  0
 6      0*     0   0 *   0  0
)))))))2)))))))))3)))))))))
            ΦWAL=0 * ΦRLI=0
```

VAR.9: FUNEX

```
COD    F *  F1    D1 * F2  D2
)))))))3)))))))))3)))))))))
 X      0*     0   0 *   0  0
 B    998*   998   0 * 998  0
 1      0*     0   0 *   0  0
 6      1*     1   0 *   1  0
```

```
))))))))2))))))))))3)))))))))
             ΦWAL=0 * ΦRLI=0
```

VAR.10: NOTRAB

```
COD    F *   F1    D1 * F2   D2
)))))))3)))))))))3)))))))
 X      0*     0    0 *    0  0
 B    985*   991    6 * 975 10
 1      0*     0    0 *    0  0
 6     13*     8    5 *   23 10
 )))))))2)))))))))3)))))))))
            ΦWAL=11 *ΦRLI=20
```

VAR.11: BUSCA

```
COD    F *   F1    D1 * F2   D2
)))))))3)))))))))3)))))))))
 X      0*     0    0 *    0  0
 B    607*   612    5 * 607  0
 1     13*    13    0 *   13  0
 6    379*   374    5 * 379  0
)))))))2)))))))))3)))))))))
            ΦWAL=10 * ΦRLI=0
```

VAR.12: AUFIN

```
COD    F *   F1    D1 * F2   D2
 )))))))3)))))))))3)))))))))
 X      0*     0    0 *    0  0
 B      0*     0    0 *    0  0
 0    998*   998    0 * 995  3
 1      1*     1    0 *    3  2
 6      0*     0    0 *    0  0
 )))))))2)))))))))3)))))))))
             ΦWAL=0 * ΦRLI=5
```

VAR.13: AUOCUP

```
COD    F *   F1    D1 * F2   D2
)))))))3)))))))))3)))))))))
 X      0*     0    0 *    0  0
 B      0*     0    0 *    0  0
 0    606*   611    5 * 606  0
 1    392*   387    5 * 392  0
 6      0*     0    0 *    0  0
 )))))))2)))))))))3)))))))))
            ΦWAL=10 * ΦRLI=0
```

VAR.14: AUF

```
COD    F *   F1    D1 * F2   D2
)))))))3)))))))))3)))))))))
 X      0*     0    0 *    0  0
 B      0*     0    0 *    0  0
 0    403*   398    5 * 403  0
 1     92*    74   18 *   93  1
 6    503*   527   24 * 503  0
)))))))2)))))))))3)))))))))
            ΦWAL=47 * ΦRLI=1
```

VAR.15: OFEMP

```
COD    F *   F1    D1 * F2   D2
)))))))3)))))))))3)))))))))
 X      0*     0    0 *    0  0
 B      3*     4    1 *    5  2
 1     22*    19    3 *   23  1
 2     79*    69   10 *   80  1
 3    894*   905   11 * 890  4
)))))))2)))))))))3)))))))))
            ΦWAL=25 * ΦRLI=8
```

VAR.16: AUG

```
COD    F *   F1    D1 * F2   D2
)))))))3)))))))))3)))))))))
 X      0*     0    0 *    0  0
 B      0*     0    0 *    0  0
 0    390*   385    5 * 390  0
 1    503*   527   24 * 503  0
 2      0*     0    0 *    0  0
 3    105*    87   18 * 105  0
)))))))2)))))))))3)))))))))
            ΦWAL=47 * ΦRLI=0
```

VAR.17: RZBUS

```
COD    F *   F1    D1 * F2   D2
)))))))3)))))))))3)))))))))
 X      0*     0    0 *    0  0
 B    986*   986    0 * 986  0
 1      3*     3    0 *    3  0
 2      5*     5    0 *    5  0
 3      0*     0    0 *    0  0
```

```
 4      2*     2    0 *    2  0
 5      0*     0    0 *    0  0
))))))))2))))))))))3))))))))))
             ΦWAL=0 * ΦRLI=0
```

VAR.18: SILA

```
COD    F *   F1    D1 * F2   D2
))))))))3))))))))))3))))))))))
 X      0*     0    0 *    0  0
 B      0*     0    0 *    0  0
 1      8*     8    0 *    8  0
 2    378*   379    1 * 368 10
 3      0*     0    0 *    0  0
 4      1*     1    0 *    1  0
 5     14*     8    6 *   24 10
 6     92*    73   19 *   92  0
 7     23*    20    3 *   19  4
 8    480*   506   26 * 483  3
))))))))2))))))))))3))))))))))
             ΦWAL=55 *ΦRLI=27
```

VAR.19: AGE

```
COD    F *   F1    D1 * F2   D2
))))))))3))))))))))3))))))))))
 X      0*     0    0 *    0  0
 B      0*     0    0 *    0  0
16     21*    21    0 *   21  0
17     22*    22    0 *   22  0
18     21*    21    0 *   21  0
19     22*    22    0 *   22  0
20     23*    21    2 *   23  0
........*..........*........
99      0*     0    0 *    0  0
))))))))2))))))))))3))))))))))
             ΦWAL=31 * ΦRLI=6
```

A summary of the indicators obtained for the most significant variables, is given hereafter:

TABLE 1

VARIABLES

	SEX			NOTRAB	BUSCA		AUG		SILA	AGE	TOTAL
ΦWAL	14	0	...	11	10	...	47	...	55	31	252
ΦRLI	2	6	...	20	0	...	0	...	27	6	86

The column TOTAL is the sum of all variable indicators. The tables show an increase in the data quality when RLI's (ΦRLI =86 < ΦWAL =252).

Nevertheless, the result is not uniform if we analyze it for each variable in the study: for some of the variables, there is an obvious increase in quality (for instance SEXO1, BUSCA, AUG); for others variables there is a decrease in quality (f.i. NOTRAB). To discover the reason for these differences in behavior, we repeated the experiment with the artificial survey: MEPA.

2. MEPA file:

Experiments with the EPA file suggested the idea that, in theory, it is possible to distinguish two kinds of variables:

- FLOW-VARIABLES. They are variables whose answer determines the questionnaire flow. A good example is the SILA variable in the MEPA questionnaire.

- NON-FLOW-VARIABLES, or semantic variables, whose answer is irrelevant to the questionnaire flow. For instance, the OCUP variable.

Our hypothesis is that the internal redundancy of the records can help improving the results of the imputations when we are treating "random errors". In this sense, an error in a "flow variable" can easily be detected and imputed thanks to the redundancy with the rest of the "routing information".

For semantic variables, the situation is not so clear. Let's imagine an error in a typical "non flow variable", detected because its code is incompatible with another variable. For such a case the incompatibility rule detects the incompatible variables but not exactly what is the erroneous one.

Given this fact, we were interested in testing the impact of imputing or not imputing the errors depending on the kind of variables. This whas another reason for designing a ficticious file; to better analyze both kinds of variables.

Two experiments were carried out with the MEPA file.

a) First experiment: with incomplete control of the routing

This first experiment was conducted with a set of edits where two of the edits needed to control the questionnaire were missing.

Edits used:

A/ SKIP PATTERN EDITS (EDITS CONTROLING THE ROUTING)

SILA(1)OCUP(B)
SILA(¬1)OCUP(¬B)
OCUP(¬B)ACT(B)
ACT(¬B) BUSCA(B)
ACT(B)BUSCA(¬B)
BUSCA(1)RZBUS(B)
BUSCA(¬B)RZBUS(¬B)
RZBUS(¬B)TYPE(B)
SILA(2)TYPE(B)

B/ SEMANTIC EDITS

AGE(1)OCUP(9)
OCUP(1)ACT(¬1,2)
OCUP(2)ACT(2)
OCUP(3-5)ACT(¬3-5)
RZBUS(2)TYPE(1)

With the initially specified set of edits, the MEPA.ORIGINAL.FILE only had 11.385 GOOD RECORDS. These records were repeated 10 times in order to produce a MEPA.FILE.GOOD with 113.850 records.

Here, errors were introduced at random with different probabilities for the different variables. After following the steps described above, we got the results to be analyzed. Table 2 and Table 3 present the related summary. For variables such as AGE whose redundancy is small and the level of error low, there was a small increase in quality with RLI (when the errors are imputed). Surprisedly enough, this was not the case for the other variables in the example, as SILA and BUSCA.

- SILA has a slight decrease in quality (7 to 5).
- BUSCA has an ever more important decrease (12 to 4).

TABLE 2

	VARIABLES		
	AGE	SILA	BUSCA
WAL	7	5	4
SRI	4	7	12

TABLE 3 shows the errors in each variable: the level, the number of errors and the number of imputations.

TABLE 3

	VARIABLES		
	ERROR LEVEL	NUMBER ERRORS	NUMBER IMPUTATIONS
AGE	4%	4681	631
SILA	6%	6867	6738
BUSCA	4%	4613	4251

The conclusion after this experiment was that the Record Level Imputation produced worst estimates than the Weight Adjustment at the Aggregated Level.

As this result was in clear contradiction with our previous hypothesis and results, we kept on experimenting to confirm, or not, the unexpected results. First we checked the process, trying to detect any error. This lead us to the discovery that two SKIP PATTERN RULES were missing in the test.

b) Second experiment: with incomplete control of the routing

After including the missing pattern skip rules, this second experiment rules were:

A/ SKIP PATTERN EDITS
SILA(1)OCUP(B)
SILA(¬1)OCUP(¬B)
OCUP(¬B)ACT(B)
OCUP(B)ACT(¬B) (missing before)
ACT(¬B)BUSCA(B)
ACT(B)BUSCA(¬B)
BUSCA(1)RZBUS(B)
BUSCA(¬1)RZBUS(¬B)
RZBUS(¬B)TYPE(B)
SILA(2)TYPE(B)
SILA(¬2)RZBUS(B)TYPE(¬B) (missing before)
B/ SEMANTIC EDITS
AGE(1)OCUP(9)
OCUP(1)ACT(¬1,2)
OCUP(2)ACT(2)
OCUP(3-5)ACT(¬3-5)
RZBUS(2)TYPE(1)

1) After introducing the new set of rules, only 7.821 records out of 202.500 were "correct records" (in the previous experiment 11.385 were correct). These correct records were repeated to produce a file with acceptable frequencies. The file so generated, MEPA.FILE.GOOD, had 81.360 records.

2) The errors were produced similarly as in the previous experiments. In several trials we changed the error level of the variables. The results of the experiment, with errors-level 8% for SILA and ACT and 2% for the other variables, were:

VAR. : AGE

COD	F *	F1	D1 *	F2	D2
X	0*	0	0 *	0	0
B	0*	0	0 *	0	0
1	95*	95	0 *	94	1
2	113*	113	0 *	112	1
3	113*	112	1 *	112	1
4	113*	113	0 *	113	0
5	113*	113	0 *	113	0
6	113*	112	1 *	112	1
7	113*	113	0 *	113	0
8	113*	113	0 *	113	0
9	113*	111	2 *	113	0

ΦWAL=4 * ΦRLI=4

VAR. : SILA

COD	F *	F1	D1 *	F2	D2
X	0*	0	0 *	0	0
B	0*	0	0 *	0	0
1	955*	959	4 *	955	0
2	26*	24	2 *	26	0
3	17*	16	1 *	17	0

ΦWAL=7 * ΦRLI=0

VAR. : OCUP

```
COD    F *  F1    D1 * F2  D2
))))))))3))))))))))3)))))))))
 X      0*    0   0 *   0  0
 B     44*   40   4 *  44  0
 1     35*   34   1 *  35  0
 2    141*  139   2 * 139  2
 3     53*   51   2 *  51  2
 4     53*   51   2 *  52  1
 5     53*   51   2 *  52  1
 6    159*  162   3 * 160  1
 7    159*  161   2 * 161  2
 8    159*  162   3 * 160  1
 9    141*  143   2 * 142  1
))))))))2))))))))))3)))))))))
            ΦWAL=23 *ΦRLI=11
```

VAR. : ACT

```
COD    F *  F1    D1 * F2  D2
))))))))3))))))))))3)))))))))
 X      0*    0   0 *   0  0
 B     44*   40   4 *  44  0
 1    104*  102   2 * 102  2
 2     86*   97  11 *  98 12
 3    139*  134   5 * 134  5
 4    139*  137   2 * 136  3
 5    139*  153  14 * 152 13
 6     86*   84   2 *  84  2
 7     86*   84   2 *  83  3
 8     86*   81   5 *  80  6
 9     86*   84   2 *  83  3
))))))))2))))))))))3)))))))))
            ΦWAL=49 *ΦRLI=49
```

VAR. : BUSCA

```
COD    F *  F1    D1 * F2  D2
))))))))3))))))))))3)))))))))
 X      0*    0   0 *   0  0
 B     44*   40   4 *  44  0
 1    896*  899   3 * 895  1
 6     59*   59   0 *  59  0
))))))))2))))))))))3)))))))))
            ΦWAL=7 * ΦRLI=1
```

TABLE 4 shows the number of errors and imputations for some variables in this example.

TABLE 4

VAR	ERROR LEVEL	ERRORS	IMPUTATIONS
AGE	2%	1651	232
SILA	8%	6504	6495
OCUP	2%	1588	1078
ACT	8%	6620	1832
BUSCA	2%	1577	1582

TABLE 5 shows a summary of the indicators.

TABLE 5

	VARIABLE				
	AGE	SILA	OCUP	ACT	BUSCA
ΦWAL	4	7	23	49	7
ΦRLI	4	0	11	49	1

Thus, we can state an improvement in data quality with RLI (that is, RLI's indicators are smaller than WAL's). This is specially clear in SILA and BUSCA, typical "flow-variables".

However, as far as OCUP and ACT are concerned, the results are not as clear. First, if we examine the rules, we see that these "non-flow-variables" are highly related between them. That is, the information necessary to detect the error and impute one of them must be taken from the other. Second, the ratio:

R = NUMBER OF IMPUTATIONS / NUMBER OF ERRORS is:

ROCUP = 0.7 (approximately) and RACT = 0.3 ; consequently OCUP has been much frequently imputed than ACT. OCUP has undergone a partial increase in quality but ACT none at all.

Some preliminary conclusions could be the following:

- when the errors occur in variables such as SILA, which is an example of strong redundancy, it is easy to improve their quality through the related variables.

- when the errors occur in variables such as ACT, only partially related with one or two other variables, the increase in quality is not so clearly guaranteed because the minimum change principle may select either an erroneous variable or a correct variable for imputation.

V. FINAL CONCLUSIONS

The results of this experiment confirms that there is an increase in data quality when applying automatic imputation (based on Fellegi and Holt methodology) to random errors in surveys as those used in these experiments.

Keeping in mind that:

- The increase in quality depends on the redundancy of the data. If there is none at all, there is no support for the imputations. In that case, in theory and more or less in practice too, the results after imputing or not, should be similar.

- The flow of the questionnaire introduces some redundancy.

- The quality of the whole editing process. If the imputations are not carefully made, that is with good tools and well specified rules, the result of imputation can be a decrease in data quality.

- The efficiency of the process is measured in terms of costs of time and resources in general. It is necessary to consider that to use the good data as an estimator is cheaper and quicker. However a well designed automatic imputation procedure can result in a a very efficient process.

After this experiment we feel the need for other studies comparing the quality impact and the cost of different types of imputation: manual versus automatic imputation, a combination of both methods, etc. The indicators and the whole procedure described here, could be a useful approach to conduct and analyze the results.

A SYSTEMATIC APPROACH TO AUTOMATIC EDIT AND IMPUTATION[2]

by I.P. Fellegi, Statistics Canada
and D. Holt, University of Southampton, United Kingdom

This article is concerned with the automatic editing and imputation of survey data using the following criteria.

1. The data in each record should be made to satisfy all edits by changing the fewest possible items of data (fields);

2. As far as possible the frequency structure of the data file should be maintained;

3. Imputation rules should be derived from the corresponding edit rules without explicit specification.

A set of procedures is developed for identifying the fewest number of fields which may be changed to make the resulting data satisfy all edits. Several methods are proposed for determining the specific values to be imputed into each field.

I. INTRODUCTION

The last stage of the processing of survey data, just prior to tabulations and data retrieval, usually involves some form of editing. By editing we mean

A. the checking of each field of every survey record (the recorded answer to every question on the questionnaire) to ascertain whether it contains a valid entry; and
B. the checking of entries in certain predetermined combinations of fields to ascertain whether the entries are consistent with one another.

Checking Type A may involve determining whether the entry of any field is an invalid blank (for some fields blank entries may be valid -- e.g., the reported income of minors -- but we mean here invalid blanks), and whether the recorded entries are among a set of valid codes for the field. Examples are: age should not be blank or negative; marital status should not be blank; number of children should be less than 20.

While edits of Type A are immediate consequences of the questionnaire and code structure, edits of Type B are usually specified on the basis of extensive knowledge of the subject

---

[2] This paper was originally published in the Journal of the American Statistical Association, March 1976, Volume 71, Number 353, pp 17-35. It is presented in this publication by courtesy of the authors.

matter of the survey. Conceptually, edits of Type B specify in one form or another sets of values for specified combinations of fields which are jointly unacceptable (or, equivalently, sets of values which are jointly acceptable). Examples are: if age is less than 15 years, then marital status should be single; the acreage reported to be under different specified crops should be equal to total reported acreage under cultivation; the total value of production reported in a given industry divided by total manhours worked should be between two predetermined constants.

This article essentially deals with computer edits. When a record fails some of the edits, we have, theoretically, five options:

1. Check the original questionnaires in the hope that the original questionnaire is correct and the edit failures are caused by coding error or error introduced during the conversion of data to machine-readable form;
2. Contact the original respondent to obtain the correct response (or verify that the reported response was correct in its original form);
3. Have clerical staff "correct" the questionnaire using certain rules which would remove the inconsistencies;
4. Use the computer to "correct" the questionnaire, also using certain rules which would remove the inconsistencies;
5. Drop all records which fail any of the edits or at least omit them from analyses using fields involved in failed edits.

Option 5 would involve an implicit assumption that the statistical inferences are unaffected by such deletions. This is equivalent to assuming that the deleted records have the same distribution as the satisfactory records. If such an assumption must be made it would seem much more sensible to make it through imputation techniques. At any rate, when population totals have to be estimated, some form of correction would still be necessary: if the correction is not made through explicit imputation, it would have to be made through some form of editing. However, implicit in weighting is the imputation of the appropriate mean to all fields of all records which are involved in failed edits. We believe that a better imputation results if we make use of the valid parts of questionnaires and impute for as few fields as possible.

Corrections of Type 1 and 2, particularly 2, are certainly worthwhile if feasible: one should, whenever possible, avoid "manufacturing" data instead of collecting it. Often, however, it is not possible to recontact the respondents, mostly due to limitations of time and money. In these cases, 1, 3, and 4 are left open to us. However, the incidence of cases where 1 is fruitful can and probably should be minimized by a rigorous application of quality controls to the coding and data conversion operations.

At any rate, whether we use 1 and 2, we usually reach a point where 3 or 4 has to be applied. Of these alternatives, we have a strong preference for 4: if the data are to be "corrected" using predetermined rules, it is usually much more preferable, although not necessarily more economical, to let the computer apply these rules rather than clerks. Contrary to clerks, the computer applies the rules consistently and fast; moreover, if editing is done by a computer but correction by clerks, usually the data have to be edited again to insure that all inconsistencies have been removed. This may lead to a long series of computer edits alternating with clerical

corrections and the interface between the two operations involves the usually relatively slow process of input and output, as well as potentially complex logistic and control operations.

The major disadvantage of a procedure involving the computer for both editing and correction is the potential programming complexity and the relative rigidity of the computer programs, i.e., the complexity of the programs is usually such that changes are time consuming, expensive and error-prone. Edit and correction rules are often more or less independently specified with no full assurance that the corrections will render the data consistent with respect to the edits. Also, in the absence of some overall philosophy of correction and in the light of the very complex rules often used, the net impact of the corrections on the data is unforeseeable and, possibly even worse, not readily capable of a systematic post-implementation evaluation.

This article proposes an approach which potentially overcomes (or at least ameliorates) these problems. With respect to complexity, the approach presented simplifies the programming task by identifying a uniform and particularly simple form for all edits, which also assists in overcoming the potential rigidity of computer programs. In fact, the "rigidity" usually arises because the edits are frequently specified in the form of logically complex and interrelated chains or networks. The article breaks these down into a series of simple and unrelated edit rules of a common form; once this form of edits is implemented it is relatively easy to add additional edit rules or remove some of the edits already specified. This ease of modifying the edits should facilitate carrying out a set of initial edits in exploratory fashion (possibly on a sample of records) and their modification, as desirable, prior to data correction.

In effect, the approach lends itself to the implementation of a generalized edit and data correction system driven by subject-matter specification, consisting of specifying inconsistent combinations of codes. The simplicity with which edit specifications can be altered without the need to reprogram provides the most important payoff of the approach: it facilitates experimentation with alternative edit specifications and permits evaluation of their impact on the "corrected" data. Should this impact be found unacceptable, the entire edit system can be respecified with ease and rerun without delay.

Data correction is often the most complex of the survey data processing operations. This article presents an approach which, optionally, altogether eliminates the necessity for a separate set of specifications for data corrections: the needed corrections are automatically deduced from the edits themselves. On one hand, this should simplify specification of correction routines and, on the other hand, insure that the corrections are always consistent with the edits. The consistency of corrections with edits becomes a particularly important issue if we want the flexibility of changing the initially specified edits. A fuller evaluation of the benefits offered by this approach is presented in Section 6.

In an earlier paper, Freund and Hartley (1967) proposed a least squares type approach to data correction in the case of quantitative variables with arithmetic edits. The approach assumed a metric underlying the data. However, for qualitative data (i.e., coded data such as sex, occupation, etc.) there is often no basis for assuming a metric, and such an approach to data correction (imputation) is not realistic. In fact, the central concern of this article is with qualitative data, although some of the theoretical results also apply to quantitative data. These will be pointed out specifically. A subsequent paper by one of the authors deals with quantitative data specifically.

There are three criteria for imputation of qualitative data which we have attempted to meet, the first of which is overridingly important in the rest of this article.[3]

1. The data in each record should be made to satisfy all edits by changing the fewest possible items of data (fields). This we believe to be in agreement with the idea of keeping the maximum amount of original data unchanged, subject to the constraints of the edits, and so manufacturing as little data as possible. At the same time, if errors are comparatively rare it seems more likely that we will identify the truly erroneous fields. This criterion appears to be reasonable, particularly for qualitative (coded) data, since it provides the only feasible measure of the extent of changes due to imputations.
2. It should not be necessary to specify imputation rules; they should derive automatically from the edit rules. This would insure that imputed values will not continue to fail edits, simplify the task of specifying edits and imputations, simplify their computer implementation, facilitate the implementation of subsequent changes to specifications, and generally lead to a more controlled operation.
3. When imputation takes place, it is desirable to maintain, as far as possible, the marginal and even preferably the joint frequency distributions of the variables, as reflected by the "correct" records, i.e., those which pass the edits.

The idea of Criterion 3 can be explained in terms of an example. Suppose that in a survey we collect data about age, sex, occupation and labor-force status. If, say, in a given record, labor-force status is identified as having been reported in error, usually we know no more about the particular record than that it represents an individual belonging to that subpopulation which has identical age, sex and occupation to that shown on the given record. In the absence of other information, the logical inference about labor-force status would be the average labor-force status of this subpopulation as measured by the error-free records; since the average of a set of codes does not make sense, we replace the average by a code value randomly selected from this subpopulation. Even in the case of quantitative variables, where the average is a sensible quantity, a suitably chosen value from the subpopulation might still be preferable, since imputing the average repeatedly would distort the distributions.

In concluding this introduction, we emphasize that the motivation for this article is not to make editing and imputation so simple and routine as to tempt survey takers and processors to replace direct observation and measurement by computer imputations. The approach to editing is predicated on the assumption that one wants to alter the original observations to the least possible extent. We provide a systematic approach and tools to achieve this. While the amount of imputation is thus minimized, it may still be unacceptably high in particular surveys, depending on the quality of data collection and conversion, relative to the particular requirements of statistical inference for which the survey was taken. Thus, while the question of the impact of automatic editing and imputation on the subsequent statistical inference is outside the scope of

---

[3] By imputation for a given record we mean changing the values in some of its fields to possible alternatives with a view to insuring that the resultant data record satisfies all edits. To the extent that an originally recorded value may be an invalid blank, the term encompasses data correction necessitated by partial non-response or by prior clerical editing.

this article, we strongly recommend that this impact be studied in the context of the particular applications.

II. THE NORMAL FORM OF EDITS

At the beginning, let us restrict ourselves to records containing only qualitative (coded) data, i.e., data which are not subject to a meaningful metric. Edits involving such data will be referred to as *logical edits*, as opposed to *quantitative arithmetic edits*, which are meaningful only in the presence of a metric.

Edits of qualitative (coded) data, as expressed by experts with a knowledge of the subject matter of the survey, are usually conveyed in the form of narratives, flow charts, or decision tables. Whatever the medium, however, an edit expresses the judgment of some experts that certain combinations of values or code values in different fields (corresponding to the different questions on the questionnaire) are unacceptable.

Let each editable record contain N fields. Denote by A_i the set of possible values or code values which may be recorded in Field i, the number of such values being n_i (not necessarily finite).

The concept that a particular record (questionnaire), say, **a**, has in Field i a code belonging to some subset of A_i, say, A_i^0, can be concisely denoted by

$$\mathbf{a}\ 0\ A_i^0 \qquad (2.1)$$

Mathematically, this notation is not entirely rigorous because **a**, being a vector of, N components, is not a member of the set of scalars A_i^0. What we really understand by (2.1) is that **a** is a member of the Cartesian product $A_1 \times A_2 \times ... \times A_{i-1} \times A_i^0 \times A_{i+1} \times ... \times A_N$ However, use will simplify the notation by using A_i^0 instead of the corresponding Cartesian product -- the context always clarifying whether A_i^0 is actually a subset of A_i, or a subset of the total code space $A_1 \times A_2 \times ... \times A_N$.

Now the concept of an edit restriction can be made more precise. The combination of code values in different fields which are unacceptable is a subset of the code space and can, in general, be expressed as

$$f(A_1^0, A_2^0, \ldots, A_N^0), \qquad (2.2)$$

where A_i^0 are subsets of the code space and the function f connects these subsets through the 1 (set intersection) and c (set union) operations. The function f thus defines a subset of the code space and becomes an edit if, through some prior knowledge, it is declared a set of unacceptable code combinations, in the sense that a record **a** is said to fail the edit specified by f whenever

$$\mathbf{a}\ 0\ f(A_1^0, A_2^0, \ldots, A_N^0). \qquad (2.3)$$

A repeated application to f of the "distributive law" connecting the set union and intersection operations will transform the right side of (2.3) to a form consisting of the union of intersections of sets given by

$$f(A_{1}^{0}, A_{2}^{0}, \ldots, A_{N}^{0}) = (A_{i1}^{1} \cup A_{i2}^{1} \cup \ldots \cup A_{im1}^{1}) \cap (A_{j1}^{2} \cup A_{j2}^{2} \cup \ldots \cup A_{jm1}^{2}) \cap \ldots \cap (A_{k1}^{r} \cup A_{k2}^{r} \cup \ldots \cup A_{km1}^{r}). \quad (2.4)$$

Now it is easy to show that $\mathbf{a} \in f$ will occur if and only if $\mathbf{a}$ is a member of any one (or more) of the sets defined by the brackets on the right side of (2.4). So the edit defined by (2.3) can be broken down into a *series* of edits of the form

$$\mathbf{a} \in \bigcap_{i \in s} A_{i^*}, \quad (2.5)$$

where s is a suitably defined index set on the fields of the questionnaire, A_{i^*} are subsets of the code space of the form, given by (2.1) and (2.5) defines a subset of the code space such that any record in it fails an edit.

What (2.5) states is perhaps intuitively obvious: any edit, whatever its original form, can be broken down into a series of statements of the form "a specified combination of code values is not permissible".

Several points should be noted.

1. Formula (2.5) indicates that instead of testing each record as to whether it falls into the kinds of edit failure sets shown on the left side of (2.4), we can test them with respect to the simpler sets on the right side of (2.5). We will write

$$\bigcap_{i \in s} A_{i^*} = F, \quad (2.6)$$

to indicate that the set on the left side is such that any record in it would be an edit failure. Moreover, clearly, (2.6) is not altered substantively if we write instead

$$\bigcap_{i=1}^{N} A_{i^*} = F, \quad (2.7)$$

where for all fields $i \notin s$ we put $A_{i^*} = A_i$ (the set of all possible code values for Field i). We say that Field i *enters* an edit of the form (2.7) *explicitly* if A_{i^*} is a proper subset of the set of codes for Field i.

2. Formulas (2.6) or (2.7) will be referred to as the *normal form of edits*. What we have shown is that any complex edit statement can be broken down into a *series* of edits, each having the normal form. Since the derivation of (2.6) involved only the repeated application of the distributive law connecting the $\cap$ and $\cup$ operators, it provides an operationally feasible procedure of recasting edit rules into a series of edits in the normal form. We will assume throughout this article that, unless otherwise stated, all edit rules are provided in the normal form. The set of such edit rules in the normal form, as specified by the subject matter experts, will be referred to as *explicit edits* (to distinguish them from logically implied edits -- a concept to be introduced later).

3. Hereafter it will be assumed that the a priori specified edits (usually provided by experts of the subject matter of the survey) are in the normal form (2.6) or (2.7). However, before proceeding, it is relevant to ask: how much extra work do we shift on the subject-matter experts by requiring them to adhere to formal syntax provided by the normal form? The language of edit specifications which they are used to contains, essentially, two types of statements.

a. Simple validation edits, stating that the permissible code values for Field i are given by the set A_i, any other value being an error (response error, data conversion error, etc.). This can be converted into the normal form very easily and automatically. For each field the set A_i can be expanded to include a single code, say, c_i, which is to represent all the invalid codes of Field i. During the loading of the unedited data into a data base (or some other equivalent operation), all invalid codes (including invalid blanks) in Field i are replaced by the code c_i. Now a series of edits of the normal form

$$\{c_i\} = F \qquad i = 1, 2, \ldots, N$$

is equivalent to other forms of specifying the simple validation edits.

b. More complex consistency edits involving a finite set of codes are typically of the form: "Whenever **a** 0 $f(A_1', A_2', \ldots, A_N')$, then it should follow that **a** 0 $g(A_1'', A_2'', \ldots, A_N'')$," i.e., whenever a record has certain n combinations of code values in some fields, it should have some other combinations of code values in some other fields. Since g has the same structure as f (both being constructed from unions or intersections of code sets), we can take the complement of the set represented by g (which will also formally have the same structure as f); then the edit statement just given is logically equivalent to: "Whenever **a** 0 f 1 $\bar{g}$, then the record fails the edit." This latter statement, however, is formally equivalent to (2.3), so it can also be converted into the normal form (2.6). Hence, whether the originally specified edits are given in the form which defines edit failures explicitly or in a form which describe conditions that must be satisfied, the edit can be converted into a series of edits in the normal form, each specifying conditions of edit failure.

Since the conversion of consistency edits into a series of edits in the normal form is an operation which can be completely specified in terms of an algorithm whose steps consist only of the elementary operations of Boolean algebra (taking complements, applying the distributive laws, etc.), a computer algorithm can easily be written to accomplish this, thereby permitting subject-matter experts to specify edits in the language they are used to. Despite the simplicity of such an algorithm, we decided not to implement it, but rather ask our subject-matter experts to adhere to the extremely simple syntax provided by the normal form. The advantages of doing so will become clear in Section 6. Suffice to say here, that the form of edit specification provided by (2.5) was, precisely due to its simplicity, fully embraced by subject-matter experts who were involved in the first application of the system based on the methodology of this article.

4. As has already been emphasized, we have to draw a sharp distinction between *logical edits* (i.e., edits involving a questionnaire, each field of which contains a finite number of codes) and *quantitative edits* (i.e., edits involving a questionnaire, each field of which is measured on a continuous scale). The normal form of edits is both a natural and operationally feasible way of

identifying edit conflicts of the logical kind (i.e., that certain code combinations are declared edit failures). Arithmetic edits, as shown shortly, can be cast into the normal form, but this is not their natural form (i.e., a series of equalities or inequalities). However, since both logical and arithmetic edits can be cast in the normal form, the theoretical results of Section 3 (which depend on the *possibility* of casting edits in the normal form) apply to both logical and arithmetic edits. The implementation strategy of Sections 4 and 5, which is mostly based on the assumption that the edits are actually stated in the normal form, is likely to be practicable as stated only for logical edits. Possible implementation strategies for arithmetic edits are specifically indicated whenever relevant, but they will be taken up in more detail in a forthcoming paper by one of the authors.

We will indicate in the following how linear arithmetic edits can be expressed in the normal form.

We postulate (although this is not entirely necessary) for the remainder of this article that all arithmetic expressions involved in the edit specifications are linear. Such expressions may involve equalities or inequalities and would take the following form. Whenever

$$0 \lessgtr M(a_1, a_2, a_3, \ldots, a_N),$$

then the record fails this edit; where $a_1, a_2, a_3, \ldots, a_N$ are variables corresponding to the fields of the questionnaire and can assume values in the sets $A_1, A_2, \ldots, A_N$; and where the function M is a linear expression. Without loss of generality, suppose that a_1 has a nonzerocoefficient.

Then one can rewrite the preceding expression as

$$a_i \lessgtr L(a_1, a_2, a_3, \ldots, a_N),$$

where L is also a linear function. Finally, this inequality can be expressed in a normal form as 8 set of edits

$$A_{10} \times A_{20} \times \times A_{30} \ldots \times A_{N0} = F,$$

where

$$A_{10} = \{a_1 : a_1 \lessgtr L(a_1, a_2, a_3, \ldots, a_N)\}$$

$$A_{20} = \{a_2\}, A_{30} = \{a_3\}, \ldots, A_{N0} = \{a_N\}$$

and there is one such set of edits corresponding to each possible choice of the values $a_2, a_3, \ldots, a_N$ such that $a_2 \in A_2, a_3 \in A_3, \ldots, a_N \in A_N$

To illustrate, suppose that in a survey of farms, three fields on a questionnaire relate, respectively, to number of cultivated acres, unimproved acres, and total acres of land. The corresponding variables are a_1, a_2 and a_3, the set of possible values of the three variables being A_1, A_2, and A_3 (each presumably the set of nonnegative numbers). An edit may be

$$a_1 + a_2 \ldots a_3,$$

i.e., that a record satisfying this relation is an edit failure. This can be transformed into a set of edits in the normal form simply as

$$\{a_3: a_3 \ \ldots a_1 + a_2\} \ 1 \ \{a_1\} \ 1 \ \{a_2\} = F, \text{ for all } a_1 \$ 0 \text{ and } a_2 \$ 0;$$

i.e., for any a_1, and a_2 a combination of the values a_1, a_2 and any a_3 which is not equal to the sum of a_1 and a_2 is an edit failure.

5. Often the set of possible values of all fields is finite. Such is the case, for example, if all the characteristics on the questionnaire involve qualitative codes (e.g., sex, industry, labor-force status). Even if some of the characteristics involve a conceptually infinite or a very large number of possible code values (such as age, income, etc.), the edits may only distinguish explicitly some finite number of ranges. For example, if income enters only two edits distinguishing between the ranges 5,000-10,000 and 7,000-12,000, respectively, then the totality of all edits requires distinguishing only between the five ranges 0-4,999; 5,000-6,999; 7,000-10,000; 10,001-12,000; 12,000+, and we may consider for purposes of editing the field "income" as assuming only five possible code values. In such cases quantitative fields can be treated for purposes of editing and imputation as if they were qualitative (coded) fields.

When the set of possible values of all fields is finite, a simple possible implementation of an edit system is provided in Section 5.

6. We accept a convention that whenever one of the fields of a record contains an invalid entry (i.e., one which is not among the set of possible values), we consider the record as failing all edits which that field enters explicitly. The code value "blank" may or may not be among the set of possible (i.e., permissible) values of a field; e.g., blank in the income field might be a possible value in a general population survey, whereas blank in the age field would not be a possible value.

Example: The following example (clearly not realistic) illustrates the procedure of converting edits into the normal form.

Suppose in a demographic survey one of the edits specified by subject-matter experts is: "If a person's age is #15 years or he (she) is an elementary school student, then relationship to head of household should not be head and marital status should be single".

This edit can be converted into the normal form in series of steps:

[(Age # 15) *or* (Elementary School)] *implies* [(*not* Head) *and* (Single)]
[(Age # 15) *or* (Elementary School)] *and not* [(*not* Head) *and* (Single)] = *Failure*
[(Age # 15) *or* (Elementary School)] *and* [(Head) *or* (*not* Single)] = *Failure*
(Age # 15) *and* (Head) = *Failure*
(Age # 15) *and* (not Single) = *Failure*
(Elementary School) *and* (Head) = *Failure*
(Elementary School) *and* (not Single) = *Failure*

The last four statements together are equivalent to the originally specified edit. They are in the normal form.

III. THE COMPLETE SET OF EDITS

A record which passes *all* the stated edits is said to be a "clean" record, not in need of any "correction." Conversely, a record which fails *any* of the edits is in need of some corrections. Unfortunately, generally one knows only which edits are failed, but not which fields are causing the edit failures. In this section we will try to build a bridge leading to an inference from the knowledge of the edits which failed to the identification of the fields which need to be changed to remove the edit failures. The main idea can be intuitively summarized as follows (a precise treatment is given subsequently). Suppose that a record fails some of the specified edits. For the sake of simplicity, suppose also that there is a *single* field, say, Field *i*, which enters all the failed edits. Then it should be possible to find a value for that field which will convert *all* the failed edits to satisfied ones (and not convert any satisfied edits to failed ones). In fact, if this could not be done, then this would mean that in the presence of the current values in the other fields, all the possible values of Field *i* result in some edit failures; in this case, so to speak, the value in Field *i* is irrelevant; the *other* fields are the cause of at least some of the edit failures. So, one might feel, there must be at least one failed edit which involves the *other* fields but not Field *i*. This would appear to contradict our initial assumption, namely that Field *i* enters all the failed edits. Indeed, this would be an *actual* contradiction if all the edits were explicitly stated.

As it is, often the initially specified edits *imply* logically some other edits. However, if we could be sure that *all* edits (explicit and implied) are explicitly known, under the simple assumptions stated it should be possible to convert all failed to satisfied ones by changing the value in a single field. Thus, while the initially specified edits are all that are necessary to identify the records which pass or fail the edits, the edits implied by them logically must also be considered if one wishes to investigate systematically the field(s) which must be changed to "correct" the record (change it in such a fashion that the changed record passes all the edits).

Some examples might further clarify the notion.

Example 1: Suppose that a questionnaire contains three fields.

Field	*Possible codes*
Age	0-14, 15+
Marital Status	Single, Married, Divorced, Widowed, Separated
Relationship to Head of Household	Head, Spouse of Head, Other

Suppose there are two edits,

I (Age = 0-14) 1 (Mar. Stat. = Ever Married) = *F*

II (Mar. Stat. = Not Now Married) 1 (Rel. to Head = Spouse) = *F*,

where Ever Married stands for the subset of marital status codes {Married, Divorced, Widowed, Separated} and Not Now Married stands for the subset of codes {Single, Divorced, Widowed, Separated}.

Suppose that a record has the values Age = 0-14, Marital Status = Married, Relationship to Head of Household = Spouse. This record fails Edit I, passes Edit II. In an attempt to try to correct this record through imputation, we may consider changing the field Marital Status. It is easy to verify that, leaving Age and Relationship to Head of Household unchanged, all possible Marital Status codes would result in a record which would fail one or the other of the two edits. One would suspect, therefore, that there must be a hidden conflict between the current values of Age and Relationship to Head of Household -- irrespective of Marital Status. In fact, in this simple example, it is intuitively clear that there is a conflict between the age being between zero and 14 years and relationship to head of household being spouse. The existence of such a conflict, as one which is implied logically by the two stated edits, can be formally established as follows (a more rigorous method is provided by the Lemma, stated later in this section).

Edits I and II can be restated (using the arrow ~ to mean "implies") as

I (Age = 0-14) Y(Marital Status = Single)

II (Marital Status = Not Now Married) Y (Rel. to Head =Not Spouse).

Since it is obviously true that

(Marital Status = Single) Y (Marital Status = Not Now Married).

we obtain, combining I and II,

(Age = 0-14) Y (Rel. to Head = Not Spouse)

Finally, the preceding statement can be recast, equivalently, as

III (Age = 0-14) 1 (Rel. to Head = Spouse) = *F*

Thus Edit III is logically implied by Edits I and II.

It can be shown, using the methods provided later in this section, that no new edits can logically be derived from these three edits. A set of edits having this property will be called a *complete set of edits*.

We can now illustrate the main result of this section as follows. The current record, as just described, fails Edits I and III. In attempting to correct this record, we will select fields which, between them, explicitly figure in all the failed edils. We say that such fields "cover off" the *failed edits*. Now in this instance, Edits I and III are covered off either by the single field Age, or by any two of the three fields, or by the three fields together. According to the main theorem of this section, any combination of fields which covers off all the failed edits can be changed so that the resulting record will pass all edits. In this instance we can change Age to 15+ (leaving the other two fields unchanged) and this will result in a conflict-free record; alternatively, we could appropriately change any two fields (leaving the third field unchanged) to obtain a conflict-free record. In such a situation one can argue that the combined evidence presented by all the fields together seems to point to the single field of Age being in error, rather than some combination of two fields being in error.

The point to note in this example is the importance of identifying, together with the initial edits (Edits I and II), the logically implied edit (Edit III). Edits I and II only suffice to identify that the current record is subject to a conflict. However, it is only after having identified the logically implied Edit III that we are in a position to determine systematically the field(s) which have to be changed to remove all inconsistencies.

Example 2: Suppose that a questionnaire has four quantitative fields, the information recorded in these fields being denoted, respectively, by the variables *a, b, c,* and *d*. Suppose that there are two arithmetic edits connecting these variables, each indicating a condition of edit failure (it is a uniform convention adopted throughout that we specify conditions of edit failure, rather than the opposite conditions of edit consistency):

$$\begin{array}{lllll} \text{I} & a + c + d \# b & \text{or, equivalently,} & a \; ! \; b + c + d \leq 0 \\ \text{II} & 2b \# a + 3c & \text{or, equivalently,} & !a + 2b \; !3c \leq 0. \end{array}$$

Let the current record contain the values: $a = 3$, $b = 4$, $c = 6$, $d = 1$. It is easy to see that the first edit passed; the second is failed. It is not immediately clear, even in this simple example, which fields (variables) need to be changed to satisfy the edits. However, if we write the logically implied edits, the situation clarifies immediately. It is easy to verify that the following three inequalities are all implied by the preceding two (they are linear combinations with positive weights of those two inequalities):

$$\begin{array}{ll} \text{III} & b \; ! \; 2c + d \# \text{O} \\ \text{IV} & a \; ! \; c + 2d \# 0 \\ \text{V} & 2a \; ! \; b + 3d \# 0 \; . \end{array}$$

Now one can verify that Edits II, III, and IV fail; I and V are satisfied. Looking for variables which, between them, enter all the failed edits, it is immediately obvious that variable *c* enters all the failed edits, or alternatively, any pair of variables also enter (between them) all the failed edits. Thus, according to the results of this section, one can remove all inconsistencies by choosing a suitable value for *c* (leaving the values of the other three variables unchanged); alternatively, one would have to change the values of at least two variables. Thus, the only single variable which can be suitably changed is *c*. In effect, changing *c* to any value between zero and 5/3 will result in a record which satisfies all the edits, e.g., $c = 1$ would be a suitable imputation.

Under more general assumptions we will identify a set of minimal number of fields which, when changed, will result in satisfying all the edits But first we need a general method enabling us to write all the edits logically implied by the explicitly stated edits.

We will first state and prove a Lemma which provides a method of deriving implied edits from a set of edits. It will be shown later (Theorem 2) that *all* implied edits can be derived by repeated applications of the Lemma.

Lemma: If e_r are edits for all $r \; O \; s$ where *s* is any index set,

$$e_r: \bigcap_{j=1}^{N} A_{jr} = F \text{ for all } r \in s.$$

Then, for an arbitrary choice of i ($1 \leq i \leq N$), the expression

$$e^*: \bigcap_{j=1}^{N} A_{j*} = F \qquad (3.1)$$

is an implied edit, provided that none of the sets A_{j*} is empty, where

$$A_{j*} = \bigcap_{r \in s} A_{jr} \quad j = 1, \ldots, N;\ j \neq i.$$

$$A_{j*} = \bigcup_{r \in s} A_{jr}.$$

The proof is presented in the Appendix.

It is relevant to emphasize that the edits $e_r (r \in s)$ in the statement of the Lemma may represent *any* subset of the set of edits. For example, if we start with, say ten edits, we may try to derive an implied edit from any two, three, four, etc. of them. Also, implied edits, once derived, can participate in the derivation of further implied edits. The edits from which a particular new implied edit is derived are called "contributing edits".

It will be shown subsequently (Theorem 2) that *all* implied edits can be generated through a repeated application of this Lemma. For reasons of presentation, we postpone the statement of Theorem 2 to the end of this section.

To illustrate the Lemma, we return to Example 1. There are two edits

e_1: (Age = 14) $\cap$ (Rel. to Head = Any code) $\cap$ (Mar.Stat. = Ever Married) = F
e_2: (Age = Any code) $\cap$ (Rel. to Head = Spouse) $\cap$ (Mar.Stat. = Not Now Married) = F.

Letting Marital Status play the role of Field i of the Lemma (the generating field), we obtain

A_{3*} = (Mar. Stat. = Ever Married) $\cup$ (Mar. Stat. = Not Now Married)
= (Mar. Stat. = Any code).

While for the other two fields,

A_{1*} = (Age = 0-14) $\cap$ (Age = Any code) = (Age = 0-14)
A_{2*} = (Rel to Head = Any code) $\cap$ (Rel to Head = Spouse) = (Rel to Head = Spouse).

Thus we obtain an implied edit,

e^* : (Age = 0-14) 1 (Rel. to Head = Spouse) 1 (Mar.Stat. = Any code) = F.

This is the same implied edit derived earlier using a more heuristic approach.

If all the sets A_{jr} are proper subsets of A_j, the complete set of code values for Field j, but

$$A_i^* = \text{A},$$

then the implied edit (3.1) is said to be an *essentially new edit*. In fact, in this case (3.1) does not involve all the fields explicitly involved in the e_r (Field i being explicitly involved in the Edit e_r but not in (3.1)). Field i is referred to as the generating field of the implied edit.

In Example 1, we used Marital Status as the generating field. The codes for this field were proper subsets of the set of all possible marital status codes in Edits e_1 and e_2. However, in the implied edit e^*, the code set corresponding to this field is the set of all possible codes of Marital Status. Thus, e^* would be called an essentially new implied edit. However, using Age as the generating field, we would obtain

(Age= Any code) 1 (Rel. to Head= Spouse) 1 (Mar.Stat.= Divorced, Widowed, Separated) = F.

This would not be an essentially new implied edit according to our terminology, because in one of the edits (e_2) the generating field (Age) is represented by the set of all possible codes for that field. Intuitively, one can also see why we would not call this edit an "essentially new implied edit": while it is an implied edit, it is simply a weaker form of e_2 and, thus, does not add to our understanding of the constraints imposed by the edits on the data.

Referring to the first paragraph of this section, we are now in a position to define unambiguously the concept of knowing all the relevant edits (explicit and implied) which the data are required to satisfy. The set of explicit (initially specified) edits, together with the set of all essentially new implied edits, is said to constitute a *complete set of edits* and is denoted by Ω. Clearly, any finite set of explicit edits corresponds to a complete set of edits.

Define Ω_K as that subset of Ω hich involves only Fields 1, 2, ... , K. Formally, Ω_K consists of those edits

$$e_r : \bigcap_{i=1}^{N} A_{ir},$$

for which

$$A_{ir} = A_j \qquad j = K+1, K+2, \ldots, N,$$

where A_j is the complete set of code values of Field j. The following theorem holds.

Theorem 1: If a_{i0} (i = 1, 2, ... , K ! 1) are, respectively, some possible values for the first K ! 1 fields, and if these values satisfy all edits in $\Omega_{K!1}$, then there exists some value a,~E such that the values a_{i0} (i= 1, 2, ... , K) satisfy all edits in Ω_K. The proof is presented in the Appendix. We just note here that it depends on the set of edits being complete with respect to the essentially new implied edits, but not necessarily all possible implied edits.

In terms of Example 1, if e_1, e_2, and e^* are a complete set of edits Ω corresponding to e_1 and e_2, then the subset Ω_2 of this set, involving only Fields 1 and 2 (i.e., Age and Relationship to Head), consists of e^* (since e^* is the only edit which does not involve the third field, Marital Status). Ω_1, the subset of edits involving only Field 1 (Age), is empty. Thus, with $K = 3$, Theorem 1 states that if a record has some code for each of Age and Relationship to Head which satisfies e^*, then there is at least one code for Marital Status which, together with the current codes for Age and Relationship to Head, would result in the record satisfying all edits in Ω_3 (e_1, e_2, and e^*).

Corollary 1: Suppose that a questionnaire has N fields, and assume that Fields 1, ... , K ! 1 have values a_{i0} (i = 1, ... , K ! 1) such that all edits in $\Omega_{K!1}$ are satisfied; then there exist values a_{i0} (i = K, ... , N) such that the values a_{i0} (i = 1, 2, ... , N) satisfy all edits. The proof follows immediately by repeated application of Theorem 1.

We are now in a position to state precisely the meaning of the intuitively stated paragraph at the beginning of this section.

Corollary 2: Suppose that a record (questionnaire) has N fields having the values a_i (i = 1, ... , N). Suppose that S is a subset of these fields having the property that at least one of the values a_i (i 0 S) appears in each failed edit, i.e., in each edit failed by the given record. Then values a_{i0} (i 0 S) exist such that the imputed record consisting of the values a_i (i ó S), together with a_{i0} (i 0 S), satisfies all edits. The proof is given in the Appendix.

We now see the full impact of having a complete set of edits. We have only to select *any* set of fields having the property that at least one of them is involved in each failed edit and it then follows that a set of values exists for these fields which, together with the unchanged values of the other fields, will result in an imputed record satisfying all edits. If the set of fields which we select is the set containing the smallest number of fields (minimal set), then Corollary 2 states that by changing the values in these fields (but keeping values in other fields unchanged) we will be able to satisfy all the edits. Thus, if we have a complete set of edits, Corollary 2 provides an operational procedure whereby, given a questionnaire, the smallest number of fields can be identified, which, if changed, will result in all edits being satisfied by the given questionnaire.

It should be emphasized that the Lemma and Theorem 1 apply to all types of edits which can be expressed in the normal form. As indicated earlier, the normal form of edits is a natural language to users in describing edit inconsistencies for logical edits, applying to coded data. Linear arithmetic edits applicable to quantitative data can be expressed in the normal form, although for these latter types of edits the normal form does not represent a natural language. Thus, as will be seen in Section 5, the Lemma provides the basis for an operationally feasible method of deriving implied edits from logical edits. While, in the case of linear arithmetic edits, the Lemma does not directly result in an operational procedure of deriving implied edits, it still holds as a theoretical result, together with Theorems 1 and 2. In Section 5 we will develop an appropriate algorithm applicable to linear arithmetic edits which can be shown to be equivalent to the procedure of the Lemma.

While the Lemma provides a method of generating essentially new implied edits, it does not guarantee that repeated application of that method will, in fact, generate all essentially new implied edits. That this is, in fact, the case is stated by Theorem 2.

Theorem 2: If

$$e_p: \underline{\underline{\sum_{i=1}^{N}}} A_{ip} = F$$

is an edit which is logically implied by the explicit edits, then it can be generated by the edit generation procedure of the Lemma. The proof is given in the Appendix.

A useful result of the method of generating implied edits is that any internal inconsistency in the explicit edits is identified. We have an intuitive notion of what inconsistent edits may mean: requirements, concerning the data, that are self-contradictory. However, as just indicated, an edit represents a restriction on the code space, and a series of edits represents a series of such restrictions. They cannot, in themselves, be contradictory. They can, however, contradict the initial field by field identification of permissible code values. Thus, a set of edits is said to be *inconsistent* if they jointly imply that there are permissible values of a single field which would automatically cause edit failures, irrespective of the values in the other fields (clearly, such values should not be in the set of possible values for the field). Thus, an inconsistent set of edits means that there is an implied edit e, of the form

$$e_r: A_{ir} = F,$$

where A_{ir} is a proper subset of the set of possible values for some Field i. However, if A_{ir} is a subset of the possible values of Field i, then this edit could not be an originally specified edit. Since the edit generating process identifies all implied edits, it follows that this edit will also be generated. It is a simple matter to computer check the complete set of edits to identify implied edits of this type and, thus, to determine whether the set of originally specified edits is inconsistent.

IV. IMPUTATION

We have developed a method which, corresponding to any current record, identifies a set of fields (in particular the smallest set) whose values can be changed in such a fashion that the resulting record would satisfy all edits. The next step is the choice of suitable values for those fields. We will consider a number of ways of performing this second step.

Primarily we will confine ourselves to the so-called "hot-deck" type imputation methods. These essentially consist of imputing for a field of the current record the value recorded in the same field of some other record which, however, passed all the relevant edits. This method attempts to maintain the distribution of the data as represented by the records which passed the edits.

Let us assume that we have a record $\{a_i : i = 1, \ldots, N\}$ of which the first K fields are to be imputed, this being the minimal set of fields. We assume throughout the sequel that we have a complete set of edits.

Method 1, Sequential Imputation: Let us begin by imputing for Field K and then systematically impute for Fields $K - 1, K - 2, \ldots, 1$.

Consider all of the M edits (if any) in which Field K is specifically involved, but not Fields $1, 2, \ldots, K - 1$.

$$e_r: \sum_{i=K}^{N} A_{ir} = F \quad r = 1, \ldots, M \tag{4.1}$$

Since the Kth Field is explicitly involved in each edit, A_{Kr} is never the whole set of possible values of Field K.

Now consider a given record. Clearly, among the edits in (4.1), we may disregard those which the given record satisfies *on account of its values in Fields* $K + 1, K + 2, \ldots, N$ since these edits will remain satisfied irrespective of what value we impute in Field K. In other words, for a given record we may disregard those edits in (4.1) for which

$$a_{i0} \notin A_{ir}, \quad \text{for at least one } i = K + 1, K + 2, \ldots, N,$$

and consider only those M' edits for which

$$a_{i0} \in A_{ir}, \quad \text{for all } i = K + 1, K + 2, \ldots, N.$$

Loosely speaking, these are all the edits among (4.1) which the given record fails or may fail, depending on its value in Field K.

Now if we want to satisfy all these edits by imputing a value to Field K, the value to be imputed must satisfy

$$a_K^* \in \bigcap_{r=1}^{M'} A_{Kr}, \qquad (4.2)$$

where A_{Kr} is the complement of A_{Kr} with respect to A_K. This is always possible, since, if the set on the right side of (4.2) was empty, it would mean, according to Theorem 1, that there is an edit which the given record fails and which involves only Fields $K + 1, K + 2, \ldots, N$. This is contrary to the choice of the Fields 1, 2, ... , K for imputation.

Having imputed a_K^*, we have all edits satisfied which involve Field K and any of the Fields $K + 1, K + 2, \ldots, N$ (but which do not involve Fields 1, 2, 3, ... , $K - 1$). We next consider all edits involving Fields $K - 1, K, \ldots, N$, but not Fields 1, 2, , $K - 2$ and will satisfy these, as before, by imputing a suitable value for Field $K - 1$ (leaving all other fields unchanged). We continue until all the Fields 1, 2, ... , K have been imputed, and hence, by the construction, all the edits satisfied.

Example 3: Suppose that a record contains five fields, each with its possible set of codes.

Sex	*Age*	*Mar. Status*	*Rel. to Head*	*Education*
Male	0-14	Single	Wife	None
Female	15-16	Married	Husband	Elementary
	17+	Divorced	Son or Daughter	Secondary
		Separated	Other	Post-secondary
		Widowed		

The following five edits apply (they are a complete set).

e_1: (Sex = Male) $\cdot$ (Rel. Head = Wife) = F

e_2: (Age = 0-14) $\cdot$ (Mar. Stat. = Ever Married) = F

e_3: (Mar. Stat. = Not Married) $\cdot$ (Rel. Head = Spouse) = F

e_4: (Age = 0-14) $\cdot$ (Rel. Head = Spouse) = F

e_5: (Age = 0-16) $\cdot$ (Educ. = Post-secondary) = F

Let the current record contain the characteristics

Field 1 = Sex = Male

Field 2 = Age = 12

Field 3 = Marital Status. = Married

Field 4 = Rel. to Head = Wife

F'ield 5 = Education = Elementary.

It is easy to verify that Edits e_1, e_2, e_4 fail. No single field covers off all these edits. There are three pairs of fields which do: Sex and Age, Age and Relationship to Head, Marital Status and Relationship to Head. Suppose a decision has been made to impute Sex and Age, leaving the

other three fields unchanged (thus $K = 2$). Now consider all edits which involve Age (the Kth variable), but do not involve Sex (variable $K - 1 = 1$): these are the edits e_2, e_4, e_5. Thus, $M = 3$. However, e_5 is satisfied on account of the values of Fields $K + 1$, $K + 2$, $K + 3$ (the three fields which are not imputed). Specifically, because Education ... Post-secondary, e_5 will be satisfied whatever Age we impute; thus, e_5 need not be considered when imputing Age (therefore, $M' = 2$). So we have to consider the sets $\}_{22}$ and $\}_{24}$ (the complements of the code sets for Age in Edits e_2 and e_4). According to (4.2), we have to choose an Age value a_{2*} such that

$$a_{2*} \in \}_{22} \cup \}_{22} = (\text{Age} = 15+) \cup (\text{Age} = 15+) = (\text{Age} = 15+).$$

Thus, we can impute any age greater than or equal to 15 years.

Having imputed Age (suppose 22 years), we next impute Sex. Again, we consider all edits involving Sex (there are no other fields to impute). Only e_1 involves Sex. We check whether e_1 is satisfied on account of the values in the fields already imputed or left unchanged -- this is not the case. Thus, now $M = M' = 1$. Hence, we have to impute a code, a_{1*}, for sex such that

$$a_{1*} \in \}_{11} = (\text{Sex} = \text{Not Male}).$$

Clearly, the only feasible imputation is Sex = Female.

It should be pointed out that this method appears to be predicated on being able to find, at each step, edits which involve only one of the fields to be imputed, together with fields which do not need to be imputed or which have already been imputed.

We may, however, reach a point where we cannot choose a field satisfying this criterion. Suppose that after $K - k$ fields have been imputed according to this procedure, there is no failed edit which involves Field k without some of the remaining Fields $k - 1$, $k - 2$, ... , 1. In this case we may simply impute any value for Field k and then continue with Field $k - 1$.

Conceptually, the actual imputation can be as follows: having determined from (4.2) the set of acceptable values, one of which must be imputed, we search (with a random starting point) among the records which passed all edits or among those which were already imputed but for which the given field was *not* imputed. We accept as the current imputed value for each field the first acceptable values (as defined by (4.2)) encountered during the search. By so doing, the various acceptable values will be chosen with probability proportional to their occurrence in the population as a whole, as long as the errors among the original records occur more or less in random order.

The procedure is subject to some obvious disadvantages.We are not taking full account of other information in the record which might be associated (correlated) with the field being imputed, except that the imputed value will be consistent with the largest possible number of originally reported fields in the record. Also, since the fields are imputed one by one, the joint distribution of values imputed to individual fields will be different from that in the population (except when the distributions involved are independent), although their marginal distribution will be the same. This last disadvantage is true of most sequential imputation methods. The method can be improved if we restrict the search to those records which have in some specified

other fields identical (or similar, according to some definition) values to those in the current record.

Method 2, Joint Imputation: Consider all the edits. Assume that Fields 1 to K are to be imputed and consider, as with Method 1, only those M'' edits which the given record can potentially fail, depending on the choice of values for imputation to Fields 1, ... , K, i.e.,

$$e_r: \bigcup_{i=1}^{N} A_{ir} = F, \quad \text{where } A_{i0} \cap A_{ir} \text{ for all } i = K+1, \ldots, N; \; r = 1, \ldots, M''.$$

Consider the sets

$$A_i^* = \bigcup_{r=1}^{M''} A_{ir} = F, \quad i = K+1, \ldots, N \tag{4.3}$$

A_i^* cannot be empty, since $a_{i0} \cap A_{ir}$, $r = 1, \ldots, M''$. If we choose any previously processed (i.e., edited and imputed) record, whose values in Fields $K+1, \ldots, N$ are within the corresponding sets (4.3), then, since that record satisfies all the edits, its values in Fields 1, 2, ... , K may be used for the current imputation and will automatically satisfy all the edits.

Note that there is now no need to calculate the possible values of the fields to be imputed, but we do have to identify the sets (4.3) of other fields. These sets identify "ranges" for each of the fields left unchanged in such a way that any other previously accepted record, whose values in Fields $K+1$ to N fall in these ranges, will provide suitable values for imputation in Fields 1 to K.

This is not likely to be as difficult as it might first appear, since the number of edits for which $a_{i0} \cap A_{ir}$ for all $i = K+1, \ldots, N$ may not be too large.

We illustrate this procedure using Example 3. There are two fields to impute, Sex and Age. We first consider all the edits which these two fields enter: e_1, e_2, e_4, e_5. Of these, e_5, will be satisfied whatever we impute for Sex and Age because the Education code of the current record is Elementary school (i.e., not Post-secondary). Thus, only e_1, e_2 and e_4 need to be considered when imputing Age and Sex $M'' = 3$. The sets A_i^* of (4.3) are constructed as follows.

$A_3^* = A_{31} \cap A_{32} \cap A_{34}$ = (Mar.Stat. = Any code)
∩ (Mar.Stat. = Ever Married)
∩ (Mar. Stat. = Any code)
= (Mar. Stat. = Ever Married)

$A_4^* = A_{41} \cap A_{42} \cap A_{44}$ = (Rel. to Head = Wife)
∩ (Rel. to Head = Any code)
∩ (Rel. to Head = Spouse)
= (Rel. to Head = Wife)

$A_5^* = A_{51} \cap A_{52} \cap A_{54}$ = (Education = Any code).

So if we search the previously processed "clean" records until we encounter one whose Marital Status is one of the codes subsumed under Ever Married, *and* whose Relationship to Head

code is Wife, *and* whose Education code is any code, we can impute for the current record the Sex and Age codes of that record.

Note that this method may also be simply extended to take account of information in the other fields which are not explicitly involved in edits with the particular fields to be imputed, but which are thought to be closely associated. In fact, if the number of fields and the number of codes for each field is not too large, one may altogether avoid the determination of the "ranges" A_i^* and simply borrow the values to be imputed from a record which passed all edits and which, in all relevant fields not to be imputed, agrees identically with the given record. Here, "relevant" implies involvement in edits which may potentially fail depending on the choice of values for imputation.

In terms of Example 3, Marital Status and Relationship to Head were relevant for the imputation of Sex and Age, but Education was not. Note that the concept of which fields are relevant in this sense may vary from one record to another. If the record in Example 3 had an Education code of Post-secondary, then Education would also have been relevant in imputing Age.

This method takes much more account of the information contained in the other data fields of the record to be imputed. The sets of values A_i^* essentially define jointly a subset of the whole population to which the record under consideration belongs and which is smaller than the whole population. The use of other fields thought to be well associated with the field to be imputed strengthens the use of the information contained within the record. On the other hand, since we are searching for a record belonging to a smaller subset of the population, it will probably take longer than with Method 1 to find a suitable record -- although, at least partly offsetting this is the fact that there is only one search instead of K.

Since the imputation is done jointly for the K fields, it is not just the values of the single fields which will now appear in the same proportions as in the population, but also the incidence of combinations of values. Thus, no longer is there a danger of imputing into the same record two reasonably common values which almost never occur together.

Clearly, the relative merits of Methods 1 and 2 depend on the complexity of the edits, the amount of information contained in some fields in a questionnaire about others, the number of fields and their possible code values, and the frequency of edit failures. Almost certainly any attempt to use Method 2 will need a default option in case no previously accepted record can be found within a reasonable search period. In fact, the system implemented at Statistics Canada uses Method 2 as the main imputation procedure, with Method 1 as a default option.

V. IMPLEMENTATION PROCEDURES

This section is devoted to an outline of a number of practical methods developed to facilitate the implementation of the previous sections. A generalized system for editing and imputation has, in fact, been developed based on the concepts of this article.[4]

5.1 The Application of Logical Edits to a Record

The entire implementation strategy of a system of automatic editing and imputation hinges on the possibility of representing logical edits (in the normal form) as well as coded data records in the form of extremely simple strings of 0's and l's ("bits," according to computer terminology). The method will be illustrated using Example 3.

The column heading of Table 1 corresponds to a description of all possible codes of all possible fields of Example 3, one column corresponds to each possible code, a set of columns corresponds to one field. Each line in the table is a representation of one edit.

Each edit in the normal form can be fully characterized in terms of the code sets with which each field enters that edit. For example, e_3 can be written as

e_3: (Sex = Any code) 1 (Age = Any code)
1 (Mar. Stat. = Not Married) 1 (Rel. Head = Spouse)
1 (Education = Any code) = *F*.

This edit is fully characterized by the five code sets corresponding to the five fields. These code sets, in turn, can be characterized by entering in Table 1 an entry of l for each member of the set, and 0 elsewhere. Thus, for e_3, we enter l's in the columns corresponding to the two possible Sex codes, the three possible Age codes, the four possible Education codes; however, of the five possible Marital Status codes, only the four columns corresponding to the description Not Married are completed with l; in the Married column we enter 0; and of the four columns corresponding to the Rel. Head codes we enter l only in the two columns of Husband and Wife (i.e., Spouse), putting 0's in the other two columns. Table l thus provides a unique representation of the five edits of Example 3. It is called the *logica1 edit matrix*.

[4] For further information about the system see Podehl W. M. (1974) and Graves R.B. (1976).

1. Representation of the five Edits and One "Current record" of Example 3

Edit	Sex		Age			Marital Status					Rel. to Head				Education			
	Male	Female	0-14	15-16	17+	Sin-gle	Mar-ried	Di-vorced	Sepa-rated	Wid-owed	Wife	Hus-band	Son, Daugh-ter	Other	None	Ele-men-tary	Sec-ond-ary	Post-sec-ond-ary
e_1	1	0	1	1	1	1	1	1	1	1	1	0	0	0	1	1	1	1
e_2	1	1	1	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1
e_3	1	1	1	1	1	1	0	1	1	1	1	1	0	0	1	1	1	1
e_4	1	1	1	0	0	1	1	1	1	1	1	1	0	0	1	1	1	1
e_5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	1
Current record	1	0	1	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0

Having represented all edits in this fashion, we can similarly represent a data record. To represent the current record of Example 3, under each field we enter 1 in one of the columns, 0's elsewhere: the column where 1 is entered corresponds to the code for that field of the current record. Comparing the last row of Table 1 with the current record of Example 3, the representation becomes almost self-explanatory.

Now it is very easy to apply to our current record each of the edits. Conceptually, we superimpose the current record on each of the rows corresponding to the edits. It is easy to verify that the current record fails an edit if and only if all the l's of the data record are overlaid on l's in the row corresponding to that edit. Put differently, if an edit is viewed as a vector of 0's and l's (in this example, a vector of 18 components) and if the current record is similarly viewed, the data record would fail an edit if and only if the scalar product of the two vectors is equal to the number of fields (five, in this example).

A faster and more effective equivalent algorithm would consist of selecting from the preceding edit matrix those columns corresponding to the code values of the current record. In the example, we would obtain for the current record the five first columns in Table 2. We add a last column, which is simply the product of the entries in each of the rows. 0 in the last column indicates that the current record passes the edit; 1 indicates that the edit is failed. Thus, we see once again that the record of Example 3 passes Edits e_3 and e_5, but it fails Edits e_1, e_2, and e_4. The validity of this algorithm is easily verified.

It can be readily seen that if a given edit is represented by a string of l's for every possible code of a given field (such as Sex in e_2), this indicates that a record will pass or fail that edit irrespective of its code in the given field.

2. Representation of Edit Algorithm

Edit	Male	0-14	Married	Wife	Elementary	Product
e_1	1	1	1	1	1	1
e_2	1	1	1	1	1	1
e_3	1	1	0	1	1	0
e_4	1	1	1	1	1	1
e_5	1	1	1	1	0	0

Thus, Edit e_2 will pass or fail depending on the code for Age and for Marital Status -- but irrespective of the code for Sex, Relationship to Head, or Education. No edit can be represented by a set of 0's for all code values in a field, since this would imply that the edit could not be failed by any record.

Single-field edits can also be included in the representation illustrated by Table 1. All we need do is expand the code set for every field by the inclusion of an extra code labeled "Invalid." Then a single field validation edit, say, for Sex, would be represented by 1 in the column corresponding to the Invalid code of the field Sex, the other columns of Sex being filled out with 0's, and all other columns of all other fields being filled out with l's (indicating that for this particular edit the fields other than Sex are irrelevant, and as far as Sex is concerned, the only code which would fail this edit would be a code Invalid).

Note that, as far as editing only is concerned, implied edits need not be considered -- if the initially stated explicit edits all pass, no implied edit can fail. Thus, e_4 would not need to be considered since it has been shown to be implied by e_2 and e_3. Nevertheless, when we come to imputation, the implied edits become relevant.

5.2 The Derivation of a Complete Set of Logical Edits

Having represented logical edits in the form of the logical edit matrix, it is easy to devise a procedure implementing the Lemma for generating new edits.

To illustrate, refer to Table 1. Let us derive an implied edit form e_2 and e_3, using Marital Status as the generating field. The construction of the new edit from e_2 and e_3 (represented as in Table 1) can be graphically illustrated.

Edit	*Sex*		*Age*			*Mar. Stat.*					*Rel. to Head*				*Education*			
e_2	1	1	1	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1
e_2	1	1	1	1	1	1	0	1	1	1	1	1	0	0	1	1	1	1
	and		and						or		and				and			
Implied edit	1	1	1	0	0	1	1	1	1	1	1	1	0	0	1	1	1	1

In the representation of the implied edit for each field, except the generating field, we enter 0 for a code if any of the contributing edits has 0 there -- entering 1 only if all the contributing edits have 1 in that code. For the generating field this is reversed: we enter 1 for a

code if any of the contributing edits has l there -- entering 0 only if all the contributing edits have 0 in that code.

The new edit produced is a valid essentially new implied edit unless

1. one of the fields contains all 0's;
2. the generating field does not contain all 1's;
3. the new edit is already contained in an existing edit.

This last condition is easily checked, since, for the new edit to be redundant there must be an edit already identified which has a 1 in every location where the new edit has a 1. Note that the new edit is already in the normal form (2.7) and can be added to the list of edits.

This "implied edit" satisfies both of the first two criteria just listed. In fact, it is Edit e_4 of Table 1. Thus, because of Condition 3, it would not be added to the list of edits (e_4 is already part of that list).

In very simple cases one could combine all edits in pairs, triples and higher combinations, using each field as the generating field in turn, until all combinations have been exhausted. It is clear that for all but the simplest of situations this would be tedious and a screening operation is required for fruitless combinations. The following rules could form the basis of a suitable algorithm. The proof of each rule has either been given previously or is straightforward .

1. For any set of edits to produce an essentially new edit, they must have a field in common which is specifically involved in each of them (through a proper subset of its values).
2. No essentially new edit will be produced from an implied edit and a subset of the edits from which it was itself generated.
3. A combination of edits using a particular Field i as the generating field need not be considered if some subset of the proposed combination using the same Field i has already resulted in an essentially new implied edit.

Remember that any new edit which involves just one field indicates that the original explicit edits are themselves inconsistent. Remember also, that having derived all implied edits, it is necessary to add the single field valid code checks to form the complete set of logical edits.

With these simple conditions it is possible to develop an algorithm to generate all essentially new implied edits and at the same time insure that the explicit edits are mutually consistent. In fact, the general system for editing and imputation, mentioned earlier, includes a module for the derivation of all essentially new implied edits. Note that the consistency of the edits is validated at the edit generation stage before data processing begins, when corrections are easily made, rather than at the data processing stage when, through a chance combination of data values, the belated discovery of the existence of inconsistencies could cause extensive dislocation of processing schedules, together, possibly, with the need to reformulate the edits and reprocess the data.

5.3 The Derivation of a Complete Set of Arithmetic Edits

It is easy to derive formal rules which correspond, in the case of arithmetic edits involving linear expressions, to a repeated application of the Lemma to generate essentially new implied edits. To save space, we consider only the case where the fields clearly divide into two classes: those involved in only logical edits and those involved in only arithmetic edits. In such a case, a complete set of edits can separately be derived for the logical edits (using the method of Section 5.1) and for the arithmetic edits (as outlined shortly). We also restrict ourselves to linear arithmetic edits.

A linear edit, expressing an edit failure takes the form

$$f(a_1, \ldots, a_N) \$ 0,$$

where the inequality will be strict according to the original edit specification, and where f is a linear function of the variables $a_1, a_2, \ldots, a_N$. Any record for which $f \$ 0$ fails this particular edit. It is easy to see that as well as edit failures specifically of this type, this form also includes the two most common other types of arithmetic edits where a linear identity must be satisfied and where a linear expression must be within a constant range (both of which can be translated into two inequalities representing edit failures). Similarly, a rational expression of the fields within a constant range can also be translated into edits of this form. Clearly, by multiplying by plus or minus one the inequality can always be standardized in the direction just shown.

Since we have limited ourselves to linear edits, each expression is completely described by $N + 1$ coefficients and an indicator of the type of inequality (strict or otherwise).

Edit	*Constant*	*Field 1*	*Field 2*	.....	*Field N*	*Indicator*
e_r	α_{0r}	α_{1r}	α_{2r}		α_{3r}	δ^r

For any Field i not involved in the edit e_r, α_{ir} is zero. For purposes of standardization the first nonzero coefficient should be ±1 (this can always be achieved by dividing through by its absolute value). $\delta^r = 1$ in the case of strict inequality, zero in the case of weak inequality. The derivation of a complete set of arithmetic edits is now achieved as follows

Theorem 3: An essentially new implied edit e_i is generated from edits e_r and e_s using Field e_i as a generating field if and only if α_{ir} and α_{is} are both nonzero and of opposite sign. The coefficients of the new edit, α_{kt}, are given by

$$\alpha_{kt} = \alpha_{ks}\, \alpha_{ir} \; ! \; \alpha_{kr}\, \alpha_{is} \; ; \; k = 0, 1, \ldots, N$$

where r and s are so chosen that $\alpha_{ir} > 0$ and $\alpha_{is} < 0$ and $\delta^t = \delta^r\, \delta^s$. These coefficients of the new edit need to be standardized by making the first nonzero coefficient equal to ±l.

In effect, the theorem simply states that from two linear inequalities where the inequality signs are in the same direction, a variable can be eliminated by taking their linear combinations if and only if the variable has coefficients in the two inequalities which are of the opposite sign. The proof is contained in the Appendix. It can also be shown that, in the case of arithmetic edits, repeated application of Theorem 3 will derive all essentially new implied edits. The proof follows directly from that of Theorem 2, but it is rather cumbersome and will be omitted.

Thus, in this most common of situations, when simple inequalities express edit failures, obtaining a complete set of edits is quite straightforward. By simply combining the suitable edits using as generating fields all possible fields, a complete set of edits will be achieved.

Note that the set of coefficients of the arithmetic edits e_r can be used in a fashion analogous to the method of Section 5.1 to determine which edits are failed by a given record. As in Section 5.1, we start with the initial explicit edits; if the record passes all of these, the implied edits need not be considered. As before, every record which has an invalid code in a field is considered as failing all edits which that field enters explicitly.

5.4 Identification of the Minimal Set of Fields for Imputation

Whether the edits be logical, arithmetic or a mixture, once a complete set has been developed we are in a position to process the records. If any edits are failed by a particular record, then the minimal set of fields for imputation needs to be identified.

Consider an (R' x N) matrix, R' being the number of failed edits in the complete set (including the single field edits) and N the number of fields on the questionnaire. Let the (r, n)th cell of the matrix be zero unless field n is specifically involved in edit r (the edit being failed by the data values of the record under consideration).

For example, the following matrix indicates that Edit 1 is satisfied and, thus, does not appear in the matrix, Edit 2 is failed and Fields 1, 2 are involved in it, etc. Edit R is failed and Fields 2, 3, ... , N are involved in it.

	Field							
Edit	*1*	*2*	*3*	.	.	.	.	*N*
e_2	1	1	0	.	.	.	.	0
.	.							.
.	.							.
.	.							.
e_R	0	1	1	.	.	.	.	1

This matrix, which we term the failed edit matrix, can be generated by computer.

The problem of identifying the smallest number of fields to be changed to make all edits satisfied is now reduced to the problem of choosing the smallest set of fields (columns) which together have at least 1 in each failed edit (row).

Clearly, in simple cases we could examine each field (column) separately to see if it entered all the failed edits (rows), and then each pair of fields, and then triples, and so on. If we expect most records to be corrected by the imputation of very few fields, this may be satisfactory, but in the case of a large questionnaire this could be very slow.

A possible alternative procedure is as follows.

1. All satisfied edits may be ignored (rows containing zeros only).
2. Identify all fields failing single field edits (validity, checks, incorrect blanks, etc.). By definition these must be included in the minimal set. All failed edits which explicitly involve these fields will now be covered off, so we can generate a modified failed edit matrix by deleting all edits in which the chosen fields appear explicitly. If no edits remain, the minimal set has been identified.
3. If Step 2 did not eliminate all edits, identify the edit involving the fewest fields (select an edit arbitrarily if there is a tie). At least one of the fields involved in this edit must be in the minimal set.
4. For each such field generate a modified failed edit matrix as in Step 2. Keep a record of the combination of fields so far selected.
5. Repeat Steps 3 and 4 until the first modified failed edit matrix vanishes.

To illustrate the algorithm, we return to Example 3. Suppose a current record (different from Table 1) is as follows

Sex = Male
Age = 0-14
Mar. Stat. = Divorced
Rel. to Head = Wife
Education = Post-secondary

A look at the edit representation part of Table 1 will show that all five edits are failed, so we get the following failed edit matrix for the given current record.

Edits	*Sex*	*Age*	*Mar.Stat.*	*Rel. to Head*	*Education*
e_1	I	0	0	I	0
e_2	0	1	1	0	0
e_3	0	0	1	1	0
e_4	0	1	0	1	0
e_5	0	I	0	0	1

A 1 corresponding to, say, Sex and Rel. to Head in e_1 indicates that these fields explicitly enter e_1; similarly, the other edits. The structure of this tabulation is immediately derived from Table 1 and the current record.

Now we start a cycle as just indicated.

Cycle 1: Select an edit involving the fewest number of fields. In the present case all edits involve two fields, so we arbitrarily select e_1. Fields in minimal set after Cycle 1: Sex, or Rel. to Head.

Next, eliminate all the edits which involve either Sex or Rel. to Head. We obtain two modified failed edit matrices, corresponding to the choice of Sex or Rel. to Head for imputation.

Cycle 2: The following tabulation shows the modified failed edit matrix if Sex is to be imputed from Cycle 1.

Edits	*Age*	*Mar.Stat.*	*Rel.to Head*	*Education*
e_2	1	1	0	0
e_3	0	1	1	0
e_4	1	0	1	0
e_5	1	0	0	1

Selecting e_2 (arbitrarily, since all edits involve the same number of fields) we could impute Age or Mar. Stat..

Possible fields in minimal set:

Sex and Age
Sex and Mar. Stat.

If we select Rel. to Head in Cycle 1, we obtain the modified failed edit matrix shown in the following tabulation.

Edits	*Sex*	*Age*	*Mar.Stat.*	*Education*
e_2	0	1	1	0
e_5	0	1	0	1

Selecting e_5 arbitrarily, we would get the following possible fields in minimal set:

Rel. to Head and Age
Rel. to Head and Education.

Corresponding to the four possible sets of fields in the minimal set after Cycle 2, we get the following four modified failed edit matrices.

If Sex and Age is imputed from Cycle 2, we obtain

Edits	*Mar. Stat*	*Rel.to Head*	*Education*
e_3	1	1	0

If Sex and Mar. Stat. is imputed from Cycle 2, we obtain

Edits	*Age*	*Rel.to Head*	*Education*
e_4	1	1	0
e_5	1	0	1

If Rel. to Head and Age is imputed from Cycle 2, we obtain

Empty.

Finally, if Rel. to Head and Education is imputed from Cycle 2, the modified failed edit matrix becomes

Edits	*Sex*	*Age*	*Mar.Stat.*
e_2	0	1	1

Since, at the end of Cycle 2 we obtained an empty modified failed edit matrix, we do not need to continue: the fields in the minimal set are Rel. to Head and Age.

Of course, in this simple example the result could be directly verified from the first failed edit matrix: Rel. to Head and Age between them cover off all failed edits, but no other pair of fields does.

Other approaches may be developed to the problem which could well be more efficient. The proposed procedure is a relatively simple iterative one for identifying all possible minimal sets for imputation purposes.

5.5 Imputation

We will now translate the two methods of imputation proposed in Section 4 into the representation developed in this section and suggest algorithms for both methods. This will be done in terms of Example 3 of Section 4. The same example is represented by Table 1 of Section 5.1. Thus, both edits and current record are as shown in Table 1 (the current record of Section 5.4 was different!).

As in Section 4, we have decided to impute for the current record the fields Sex and Age. This could have been an outcome of the algorithm outlined in Section 5.4. (For purposes of illustration, another current record was used.)

Method 1: Suppose we impute first Age (Field 2), then Sex (Field 1).

1. Identify the edits which are satisfied on account of the values which the current record has in the other three fields, irrespective of what we may impute for Age and Sex. A look at the column of Table 1 corresponding to the current value of Mar. Stat. (= Married) indicates that Edit ... will be satisfied irrespective of the current values of any of the other fields. The current value of Rel. to Head (= Wife) does not result in the automatic satisfaction of any edit, but the current value of Education (= Elementary) results in e_5 being satisfied irrespective of the other fields.

Thus only e_1, e_2 and e_4 can possibly impose constraints on the imputation of Age and Sex.

2. Consider those edits among e_1, e_2, and e_4 which *involve* Age but *not* Sex. Again, a look at Table 1 shows that e_1 does not involve Age. The edits e_2 and e_4, both involve Age and neither involves Sex. So we have to impute for Age in such a way that these two edits are satisfied. Theorem 1 insures that this can alwavs be done. Looking at the Age portion of e_2 and e_4, we get from Table 1

Edit	*0-14*	*15-16*	*17+*
e_2	1	0	0
e_4	1	0	0

If we combine column by column the codes in such a way that the resulting row contains 1 whenever any of the rows contained 1, otherwise it contains 0, we obtain in this simple case the representation of possible imputations for Age.

0-14	*15-16*	*17+*
1	0	0

Now if we impute for Age any code where the preceding representation contains 0, we will have both Edits e_2 and e_4 satisfied. So the acceptable imputations for Age are 15-16 or 17+.

3. We search (with a random starting point) the previously accepted records until we find one with one of the Age codes just listed. We impute the value of Age from that record.

4. In this simple example we now only have Sex to impute (more generally, we would proceed sequentially field by field repeating steps 1, 2, and 3). The edits among e_1, e_2, and e_4 which involve Sex are: e_1 alone. The only possible value of Sex which results in e_1, being satisfied is Female (the only 0 entry for Sex in e_1). So the acceptable imputation for Sex is Female.

5. Repeat step 3 for Sex.

Method 2:

1. This step is identical with that of Method 1. We need to concern ourselves only with e_1, e_2, and e_4, since these are the only edits which may possibly fail, depending on the imputed values for Age and Sex -- thus, it is these we have to be careful to satisfy.

2. Consider the fields *not* to be imputed: Mar. Stat., Rel. to Head, Education. We want to consider the possible constraining effect of the current codes of these fields on the imputations which are to be made. It is easy to verify, on the basis of the material in Section 5.1, that this procedure will result in an imputation which will pass all the edits.

   For each field *not* to be imputed, identify those codes (columns of Table 1) which contain 1 in each of the relevant edits: e_1, e_2, e_4. This corresponds to the logical AND or set intersection operation for these code sets.

   We obtain

Mar. Stat.:	Married, Divorced, Separated, Widowed
Rel. to Head:	Wife
Education:	Any code.

3. Search (with a random starting point) the previously accepted records until one is found which in the fields *not* to be imputed has any combination of the codes just listed. Impute from that record its values in those fields for which the current record must be imputed. For example, if the first such record found has the following values in the fields not to be imputed: Mar. Stat. = Separated, Rel. to Head = Wife, Education = Secondary; and if for that record Age = 22, Sex = Female, then we impute for the current record Age = 22, Sex = Female.

It is a simple matter to modify these methods to utilize other information in the questionnaire which is not linked explicitly through the edits to the fields requiring imputation. The problem is simply one of limiting the search of previously processed records to those that have particular values in particular fields, determining the acceptable values or ranges of values from the record requiring imputation. Thus, for example, one might as an additional constraint only impute income from records which have the same sex, age and occupation, even though there may be no explicit edit linking income with these fields.

To illustrate this point in terms of the preceding example, the imputation for Age and Sex of the particular current record was not actually constrained by Education, in the sense that the record from which these two fields would be imputed could have any code in its Education field. However, we could, as an additional constraint, insist that whenever Age is imputed but Education is not, the record from which we impute should have the same Education status as that of the Current record. As an extreme, we could insist that the record from which we impute should have identical codes to those of the current record. Of course, the more constraints we put on the record from which we impute (in addition to the constraints imposed by the logic of the edits), the longer, generally, the search will be through the previously processed records until we find a suitable one from which to impute.

VI. THE BENEFITS OF THE PROPOSED METHOD

In summary, it may be useful to reiterate the advantages of the approach proposed here. These can be grouped under three different headings: methodological, systems, and subject matter benefits.

6.1 Methodological Benefits

1. The approach provides an orderly framework and philosophy for the development of edit and imputation procedures for surveys, in the sense that all procedures follow consistently and, largely, predictably from the edit specifications provided by subject-matter experts.
2. The procedure preserves the maximum amount of reported data, in that it minimizes the amount of imputation.
3. The procedure guarantees that records which have already been corrected will satisfy all the edit constraints.
4. Imputations are based on full information, i.e., the imputation of any field of a current record takes advantage of all the reported information of that record which is not subject to imputation (and which is logically related, through the edits, to the fields to be imputed).
5. For each questionnaire a log can be kept of all edits failed. This facilitates subsequent evaluation and the tracing of specific edit and imputation actions.
6. For each edit statement a log can be kept of all questionnaires failing it.
7. The uncorrected data can be preserved and cross-tabulated with the corrected data.

The major significance of these advantages lies in their evaluation possibilities. For example, a sample of questionnaires can be edited and imputed before the processing of the bulk of the survey data begins. If it turns out that some particular edits are failed by an inordinately large proportion of records, one would become suspicious of either the validity of the particular edit, the field procedures followed, or the ability of respondents to answer some particular questions. Similarly, crosstabulation of uncorrected data with corrected data can provide revealing early feedback on the likely gross and net effect of imputations.

6.2 Systems-Oriented Benefits

1. Systems development need not be constrained by potential delays in obtaining particular subject-matter edit specifications. A generalized edit and imputation system can be developed which is independent of any particular subject-matter application. As mentioned earlier, such a generalized system has, in fact, been developed at Statistics Canada. Specific subject-matter edit specifications become input to an already implemented system.
2. The edit and imputation problem is completely defined (at least in terms of system specifications) and is therefore relatively straightforward to implement.
3. The approach lends itself, throughout, to modularity of systems. It separates clearly the following distinct stages: analysis of edits (in the form of derivation of implied edits); editing (this itself is modular, due to the nature of the normal form of edits);

and, finally, imputation. Such a modular approach to systems implementation facilitates changes in specifications.

6.3 Subject-Matter-Oriented Benefits

1. Given the availability of a generalized edit and imputation system, subject-matter experts can readily implement a variety of experimental edit specifications whose impact can therefore be evaluated without extra effort involving systems development. This is particularly important given the generally heuristic nature of edit specifications.
2. Specifications need not be presented in the form of integrated flowcharts or decision tables, but in that of completely independent statements of the variety of conditions which should be considered as edit failures. Thus, the development of edit specifications can proceed simultaneously by a variety of subject-matter experts. This is an important consideration in the case of multisubject surveys, such as a census.
3. Only the edits have to be specified in advance, since the imputations are derived from the edits themselves for each current record. This represents a major simplification for subject-matter experts in terms of the workload of specifying a complete edit and imputation sytem.
4. Feedback is available throughout the application of the system. The first feedback, in fact, can take place prior to the availability of any data. When edits are specified by subject-matter experts, these are first analysed to derive all essentially new implied edits. The implied edits are available for review. They have been found to be very useful in practice. Clearly, whenever an implied edit is judged inappropriate, at least one of the originally specified edits must also be inappropriate. Since it is easy to identify those initially-specified edits from which a given implied edit has been derived (this is, in fact, part of the feedback from the system), such a review represents a potentially useful device to screen the originally specified edits. Other feedback features of the system include a variety of optional and default tabulations showing the number of records failing edits, the number of times each edit has been failed, etc.
5. Since each edit statement (in the normal form) is completely self-contained, and because the imputations are automatically derived from the edits, the total complex of edit and imputation system can be re-specified with complete ease by dropping some edit statements previously specified or by adding new edit statements to those previously specified. This feature, together with the possibility of early feedback throughout the operation of editing and imputation, represents a powerful combination, if used with appropriate care.

VII. FURTHER CONSIDERATIONS

Nonuniqueness of the smallest number of fields to be imputed: There is no reason that the method of obtaining the smallest set of fields, described in Section 3, should lead to a unique solution. In fact, it is quite possible that more than one minimal set of fields can be identified.

One method of dealing with this problem, particularly if Method 2 is used for imputation, is to identify all the minimal sets of fields which could be imputed and to accept that set which can be imputed soonest in the search. For the cost of this more complicated computer search technique, one will likely get the benefit of finding a suitable record after a shorter search than when looking for a single set of fields to impute. Of course, one could much more simply arbitrarily or randomly decide between the alternatives.

A priori wieghting of fields for reliability: It often happens that one has an a priori belief that some fields are less likely to be in error than others. If one field is the product of a complicated coding operation and the other is a self-coded response, then, intuitively, one is inclined to believe that the simple self-coded response is more likely to be error-free. By and large we have not tried to quantify this, believing that the data alone, by identifying the smallest possible set of fields to be changed, would more readily indicate the fields most likely to be in error. However, where there is more than one minimal set of fields, an *a priori* weight of the reliability of each field could easily be used to select among the minimal sets of fields the one to be imputed. For example, we could simply take the set with the lowest product of *a priori* weights.

Note that one could carry further the notion of *a priori* weights to the extent of choosing not the smallest number of fields which would convert all failed edits to satisfied ones, but rather the set with the smallest product of weights. The method of selecting the minimal set (see Section 5) can easily be modified to accommodate this change.

Instead of using *a priori* weights, one can edit the records in one pass through the computer and determine for each field the proportion of edits entered by the fields which are failed. This may provide a less subjective measure of the relative reliability of different fields.

APPENDIX

Proof of Lemma

If the sets A_j^* are not empty, then (3.1) is formally a valid edit. In fact, it is easy to verify that any set of values on the left side of (3.1) is included in the left side of one of the edits $e_{r\cdot}$ and is, thus, an edit failure according to the edit $e_{r\cdot}$. This completes the proof.

Proof of Theorem 1

Assume that the theorem is false. Then there are values for the first K ! 1 fields, say, a_{i0} (i = 1, 2, ... , K ! 1), which satisfy all the edits in Ω_{K-1} but which have the property that, for every possible value of Field K, one of the edits in Ω_K is violated.

Identify one such failed edit in Ω_K corresponding to each possible value a_K of Field K,

$$e_r: \underset{i=1}{\overset{K}{\_}} A_{ir} = F \quad (r = 1, 2, \ldots, R), \qquad \text{(A.l)}$$

where

$$a_{i0} \in A_{ir} \ (i = 1, \ldots, K-1)$$

$$a_K \in A_{Kr}$$

Note that some proper subset A_K of the complete set of values A_K of Field K must enter each edit in (A.l), i.e., A_{Kr} cannot be equal to A_K since, if this were so, there would be an edit in Ω_{K-1}, which is failed by $(a_{i0}, a_{i0}, \ldots, a_{K-1,0})$ -- contradicting our original assumption.

Consider the following edit, implied by the edits (A.l), and formed using the Lemma from the preceding edits e_r, with Field K as the generating field.

$$\underset{i=1}{\overset{K-1}{\_}} \{ \underset{r=1}{\overset{R}{\_}} A_{ir} \} \cup \{ \overset{R}{\underset{r=1}{\wedge}} A_{Kr} \} = F. \qquad \text{(A.2)}$$

Since $a_{i0} \in A_{ir}$ $(i = 1, 2, \ldots, K-1)$ for all r, the intersections $\underset{i=1}{\overset{R}{\_}} A_{ir}$ are not empty. Hence, (A.2) is the expression for a valid edit. Also, according to our assumption, every possible value a_K of Field K is included in one of the sets A_{Kr}, so

$$\overset{R}{\underset{r=1}{\wedge}} A_{Kr} = A_K.$$

Therefore, (A.2) reduces to the form

$$\underset{i=1}{\overset{K-1}{\_}} \{ \underset{r=1}{\overset{R}{\_}} A_{ir} \} = F. \qquad \text{(A.3)}$$

Since this edit is in Ω_{K-1} and it rules out $(a_{10}, a_{20}, \ldots, a_{K-1,0})$ as an invalid combination, we have a contradiction with the way the values a_{i0} were chosen. It must follow, therefore, that the assumption was false and that, in fact, there is at least one value of Field K, say, a_{K0}, which, together with a_{i0} $(i = 1, \ldots, K-1)$, satisfies all edits in Ω_K.

Clearly, the indexing of the fields was arbitrary and Theorem 1 holds for any set of K fields.

Note that we implicitly assumed that (A.3), an implied edit, is a member of the set of edits. Note also that (A.3) is an essentially new implied edit according to the definition following the Lemma. Thus, the validity of the theorem depends on the set of edits being complete with respect to essentially new implied edits, but not necessarily all possible edits.

Proof of Corollary 2

Suppose, without loss of generality, that S consists of the Fields $K, K+1, \ldots, N$. Thus, at least one of Fields $K, \ldots, N$ appears in each failed edit. Then there are no failed edits involving only the Fields $1, \ldots, K-1$, so that the values a_i $(i = 1, 2, \ldots, K-1)$ satisfy all edits in Ω_{K-1}. Thus, from Corollary 1, there exist values $a_{i\rho}$ $(i = K, \ldots, N)$ such that a_i $(i = 1, \ldots, K-1)$, together with a_{i0} $(i = K, \ldots, N)$, satisfy all the edits.

Proof of Theorem 2

The statement that the edit $\bigcap_{i=1}^{N} A_{ip} = F$ is logically implied by the explicit edits means that every record which fails this edit will also fail at least one of the explicit edits. Consider some values

$$a_{i0} \in A_{ip} \quad i = 1, \ldots, N. \tag{A.4}$$

(This is possible, since the sets A_{ip} are not empty.)

It follows from (A.4) that the record having the values a_{i0} $(i = 1, \ldots, N)$ must fail at least one of the explicit edits. Select one explicit failed edit corresponding to each possible value of $a_N \in A_{Np}$. Suppose there are R_N such edits corresponding to all the possible values of $a_N \in A_{Np}$

$$e_{rN}: \bigcap_{i=1}^{N} A_{irN} = F; \quad r_N = 1, \ldots, R_N .$$

Now consider the following edit e^* implied by the edits e_{rN}, (using the Lemma), with Field N as the generating field.

$$e^*: \bigcap_{i=1}^{N} A_{iq} = F,$$

where

$$A_{iq} = \bigcap_{rN=1}^{R_N} A_{irN}; \quad i = 1, \ldots, N-1$$

and

$$A_{Nq} = \bigcap_{rN=1}^{R_N} A_{NrN} .$$

This is an implied edit, since none of the intersections is empty (A_{iq} includes at least the value a_{i0}; $i = 1, 2, \ldots, N-1$).

Now $A_{N}q$ includes the set $A_{N}p$, since the edits were so constructed as to insure that every a_N 0 $A_{N}p$ was included in one of the R_N edits and, thus, in one of the sets $A_{N}{}^{r}{}_{N}$.

Since the value $a_{N\,!\,1}{}_{0}$ was arbitrarily chosen from the set of values $A_{N\,!\,1}p$, an edit like e^{*} could be derived for every value of $a_{N\text{-}1}$ 0 $A_{N\text{-}1}p$. Consider one such edit generated for each value of $a_{N\text{-}1}$ 0 $A_{N\text{-}1}p$,

$$e_{rN}{}^{*}: \_\,{}^{N}_{i=1} A_{i}{}^{r}{}_{N\,!\,1} = F;\ r_{N\,!\,1} = 1, \ldots, R_{N\,!\,1}\ . \tag{A.5}$$

where, according to the construction of the edits e~, we have

$$A_{N}{}^{r}{}_{N\,!\,1}\ \text{e}\ A_{N}p;\ r_{N\,!\,1} = 1, \ldots, R_{N\,!\,1} \tag{A.6}$$

i.e., the right side is a subset of the left side. Consider the edit implied by the edits of (A.5), using Field N ! 1 as the generating field.

$$e^{**}: \_\,{}^{N}_{i=1} A_{i}s,$$

where

$$A_{i}s = \_\,{}^{R_{N\,!\,1}}_{r_{N\,!\,1}=1} A_{i}{}^{r}{}_{N}\ ;\quad i = 1, \ldots, N\,!\,2, N;$$

$$A_{N\,!\,1}s = \ {}^{\wedge}{}^{R_{N\,!\,1}}_{r_{N\,!\,1}=1} A_{N\,!\,1}{}^{r}{}_{N\,!\,1}\ .$$

Due to the construction of this edit we have

$$A_{N\,!\,1}s\ \text{e}\ A_{N\,!\,1}p,$$

and due to (A.6) we have

$$A_{N}s\ \text{e}\ A_{N}p.$$

Continuing in this way with N ! 2, N ! 3, ... , 1, we can generate an edit

$$\_\,{}^{N}_{j=1} A_{j}x = F,$$

where $A_{jx} \in A_{jp}$, $j = 1, \ldots, N$. Clearly, the preceding edit is failed by every record which fails the edit in the statement of Theorem 2. Hence, the edit of Theorem 2 can indeed be generated from the explicit edits by the procedure of the Lemma.

Proof of Theorem 3

To prove Theorem 3, write Edits e_r and e_s as

$$e_r: f(a_1, a_2, \ldots, a_N) \geq 0$$
$$e_s: g(a_1, a_2, \ldots, a_N) \geq 0.$$

A record which satisfies the first (second) inequality fails the edit e_r (e_s).

We can express the variable corresponding to the generating field which, without loss of generality, we assume to be Field N. Observing that $\alpha_{Nr} > 0$ and $\alpha_{Ns} < 0$, we obtain

$$a_N \geq f'(a_1, a_2, \ldots, a_{N-1})$$
$$a_N \leq g'(a_1, a_2, \ldots, a_{N-1}),$$

where the coefficients β_{Nr} and β_{Ns} of f' and g' are given by

$$\beta_{kr} = \alpha_{kr} / \alpha_{Nr}$$
$$\beta_{ks} = \alpha_{ks} / \alpha_{Ns}.$$

Now suppose that a given record has values $a_{10}, a_{20}, \ldots, a_{N-10}$ such that

$$g'(a_{10}, a_{20}, \ldots, a_{N-10}) \geq f'(a_{10}, a_{20}, \ldots, a_{N-10}). \qquad (A.7)$$

In this case, whatever the value of the record in the Nth Field, it will fail at least one of the edits e_r or e_s. Indeed, in this case we will either have $a_N \geq g'$ or $a_N \leq f'$ or $g' > a_N > f'$. In any of these cases at least one e_r or e_s, will fail. For example, if $a_N \geq g'$, then, because of (A.6), we also have $a_N \geq f'$, so e_r fails.

Thus, (A.7) is an implied edit (actually an essentially new implied edit) which can be rewritten as

$$h(a_1, a_2, \ldots, a_{N-1}) \geq 0.$$

The coefficients α_k of h are clearly given by

$$\alpha_k = \alpha_{ks} / \alpha_{Ns} \quad ! \quad \alpha_{kr} / \alpha_{Nr}. \qquad (A.8)$$

Multiplying (A.8) by $\alpha_{ks}\alpha_{Nr}$, we obtain Theorem 3. It is easy to verify that the strict inequality applies only when both e_r and e_s involve strict inequalities.

It remains to show that if α_{Nr} and α_{Ns} are of the same sign (but both different from zero), then no essentially new edit can be implied by e_r and e_s using this procedure. Indeed, suppose that α_N, and α_{Ns} are of the same sign (say, both positive). We obtain, instead of f' and g',

$$e_r\colon a_N \$ f'(a_1, a_2, \ldots, a_{N-1})$$
$$e_s\colon a_N \$ g'(a_1, a_2, \ldots, a_{N-1}).$$

Suppose that

$$e_t\colon h(a_1, a_2, \ldots, a_{N-1}) \$ 0$$

is an essentially new edit implied by e_r and e_s,. Select some values

$a_{10}, a_{20}, \ldots, a_{N-10}$ which fail e_t , i.e., for which

$$h(a_{10}, a_{20}, \ldots, a_{N-10}) \$ 0.$$

If e_r and e_s are different edits, these values can also be so chosen that the corresponding values of f' and g' are not equal, say $f' > g'$. Now choose a value a_{N0} so that

$$f' > g' > a_{N0}.$$

Clearly, the record $a_{10}, a_{20}, \ldots, a_{N-10}, a_{N0}$ will fail e_t , yet it will satisfy both e_r and e_s. So e_t cannot be an edit implied by e_r and e_s (using Field N as the generating field) if a_{Nr} and a_{Ns} are of the same sign.

MACRO-EDITING -- A REVIEW OF SOME METHODS FOR RATIONALIZING THE EDITING OF SURVEY DATA

by Leopold Granquist
Statistics Sweden

Abstract: The paper presents descriptions, studies, results and conclusions (including recommendations) on: The Aggregate Method, The Hidiroglou-Berthelot Method (Statistical Edits), The Top-Down Method, The Box-Plot Method and the graphical Box Method. By simulations on survey data in a production environment it has been found that these methods can reduce the manual verifying work of suspected data by 35 - 80 % as compared to corresponding traditional micro-editing methods without any loss in quality. They are easy to understand and can easily be implemented in existing computer-assisted error detecting systems by just adding programs to the existing system. The particular features of the methods are all compared and the main finding is that the methods are very similar and that they all aim at finding efficient boundaries to generally used micro edits. Boundaries should only be based on statistics on the weighted keyed-in data.

I. INTRODUCTION

The paper is mainly devoted to an overview of studies on macro-editing methods. Emphasis is given to the rational aspects of macro-editing as compared to micro-editing by presenting the results of some studies and by discussing the problems connected with microediting. The methods are described in a brief and schematic form to make them easy to understand. It may serve as a basis for considering macro-editing methods when designing an editing system for a survey with quantitative data.

Detailed descriptions of the methods and studies are found in the references given in the text and in the reference list. Stress is laid on the specific features of every method in order to facilitate a choice.

The paper concludes with a summing-up discussion on macro-editing versus micro-editing methods.

II. WHY AND WHEN MACRO-EDITING

The essential problem of traditionally applied micro-editing procedures might be formulated as follows: Too many checks with too narrow bounds produce too many error messages which have to be verified manually by clerks. The clerks are not able to assess the importance of a suspected error. Every flagged item has the same weight and claims the same amount of resources, but many errors have a negligible impact on the estimates as they are small or cancel out. Generally, the bounds of the checks are subjectively set on the principle "safety

first" which means that only those data are accepted for which there are no reasons to suspect any error. For example, a very generally used check in business surveys at Statistics Sweden is to flag every item which indicates that the relative change since the previous survey exceeds ± 10 per cent. A considerable amount of over-editing is a general consequence of such micro editing procedures.

This paper deals with such checks of quantitative data which flag "suspicious" data for a manual review. This type of checks may be considered as opposite to validating checks, which indicate data that are erroneous. In Ferguson's article "An Introduction to the Data Editing" in this publication, they are called "Statistical Edits". Such procedures use distributions of current data from many or all questionnaires or historic data to generate feasible limits for the current survey data.

In this paper, macro-editing means a procedure for pointing out suspicious data by applying statistical checks/edits based on weighted keyed-in data. The upper and lower limits of a macro-editing check (macro edit) should be based only on:

i) the data to be edited, and
ii) the importance of the item on the total level.

The studies on the methods reported below have been simulation studies on real survey data. The results have been compared with the results of the micro-editing methods applied when the survey was processed. The changes made as a result of the micro-editing process were entered to a change file and the study consisted in investigating (by calculating a few measures) which data in the change file were flagged by the macro-editing method and which were not. The rationalizing effect was measured as the reduction of the number of flagged data, and the "quality loss" as the impact of the remaining errors (the errors found by the micro-editing of the survey, but not flagged by the macro-editing method under study).

III. THE EXPERIMENTAL DATA

The studies were carried out at Statistics Sweden and used data from the Survey on Employment and Wages in Mining, Quarrying and Manufacturing (SEW) and the Survey of Delivery and Orderbook Situation (DOS).

1. The Survey on Employment and Wages in Mining, Quarrying and Manufacturing

SEW is a monthly sample survey on employment and wages in the Swedish industry. The number of reporting units (establishments) is about 3000. The main variables are: number of workers, number of working hours, the payroll and wages/hour.

A traditional editing procedure is applied, but with inter-active up-dating, which includes checking against the edits. This editing procedure was recently revised by extending the acceptance limits of the ratio checks, co-ordinating all the checks and tuning the checks. The verifying work was then reduced by 50 % without any drop in quality. It should be noted that the rationalizing effect of the macro-editing methods studied here is compared to the revised procedure.

2. The Survey on Delivery and Orderbook Situation

DOS is a monthly sample survey of enterprises. There are about 2000 reporting units (kind-of-activity units) in a sample drawn once a year from the Swedish Register of Enterprises and Establishments. DOS estimates changes in the deliveries and the orderbook situation (changes and stocks) for both the domestic and the foreign market (six variables) for the total Swedish manufacturing industry and for 38 divisions (classified according to the Swedish Standard Industrial Classification of all Economic Activities).

The questionnaire is computer printed. The entry of the questionnaire data is carried out in three batches every production cycle by professional data-entry staff. The Top-Down macro-editing method is applied.

IV. MACRO-EDITING METHODS

Descriptions, studies and results are given on **The Aggregate Method**, **The Hidiroglou-Berthelot Method (Statistical Edits)** and **The Top-Down Method**.

The Box-Plot Method and **The Box-Method** are also reviewed, but as modifications or developments of the Aggregate Method and the Top-Down Method respectively. The Box-Method is under development and is here treated as an idea.

The Cascade of Tables Method developed by Ordinas (1988) for the editing of the Survey of Manufactures in Spain is only mentioned. This is because it is somewhat different from the type of macro-editing methods treated here. We classify it as an out-put editing method.

1. The Aggregate Method

The Aggregate Method is described in detail in Granquist's article in the latter part of this publication. It was developed in SAS as a prototype of a complete editing system for the SEW, to be run on the main-frame. A modified version of the method is reported in detail in Lindström's article later in this publication. It was developed in PC-SAS as a prototype for the SEW to be run on a personal computer.

1.1 Description of the method

The aggregate method can be defined as editing on an aggregate level followed by editing on a micro-level of the flagged aggregates. The basic idea is to carry out checks first on aggregates and then on the individual records of the suspicious (flagged) aggregates. All records belonging to a flagged aggregate of any of the variables form the error file. The checks on the individual level are then carried out on that error file.

The acceptance bounds are set manually on the basis of the distributions of the check functions. The most essential feature of the aggregate method is that the acceptance limits are set manually by reviewing lists of sorted observations of the check functions. Only the "s" largest observations and the "m" smallest observations of the check functions are printed on the lists.

Both the check function A on the aggregate level and the check function F on the individual level have to be functions of the weighted (according to the sample design) value(s) of the keyed-in data of the variable to be edited. By using the weighted values in function A, the checks on the aggregate level can be calculated in the same way as is done in traditional micro-editing. The macro-editing process can then be run as smoothly as a micro-editing process.

The lists of the sorted observations can be used either directly as a basis for reviewing observations manually (if identifiers are printed out together with the observations) or indirectly as a basis for setting acceptance limits for an error detecting program, which then can produce error messages of suspected data for manual reviewing. The advantage of using the error-detecting program is that to the reviewer the process can be made to look identical to the old one. To implement the Aggregate Method, only programs for printing the lists of the sorted observations are needed. Such programs can easily be added to the old system.

Improvements by providing the lists with statistics or graphs. Obvious improvements are to provide the lists of the distribution tails with such statistics as the median, the quartiles, the range, interquartile range or with graphs. Here, Box-plots (in detail in Ordinas, (1988)) are recommended.

1.2 The mainframe application on the SEW

The checks used in this application of the Aggregate Method consist of a ratio and a difference check. Both checks have to be rejected to indicate a value as suspicious.

The aggregates used were the 89 four-digit SSIC-groups (SSIC = The Swedish Standard Industrial Classification). The questionnaires were processed in lots on arrival at the office. There were four or more editing runs every month.

The macro-editing process was preceded by a micro-editing process in order to
- fight those errors which cannot be detected by the macro-editing method,
- make it possible to carry out the macro-editing.

The studies made on mainframe applications are described below.

An experimental study

The first study was a pure experiment to find out how methods, computer programs and so on should be constructed. The editing procedure was applied to the whole body of the edited data for a selected month. We then detected two serious errors, which had not been detected when the survey was processed. This was an indication that micro-editing does not always detect even serious errors.

Study No. 1

The next study was done on a survey round a few months after it was processed. This time the records of the data file were divided into lots exactly as when the round was originally processed.

Results:

Number of records: 2951
Number of flagged records: 274
Number of errors found: 76

When the round was originally processed the number of flagged records was 435 and the number of errors found was 205. Thus, we got a reduction by 161 flagged records = 34 per cent.

The impact of the remaining errors was calculated for each variable, for all the 89 groups for which SEW data are published. See Table 1 in the annex.

Study No. 2

The simulation was run in parallel with the regular processing in order to eliminate the risk of the results of the regular editing influencing the bounds for the checks of the aggregate method.

Results:

Number of records: 2996
Number of flagged records: 225
Number of errors found: 50

When the round was regularly processed, the number of flagged records was 389. The number of errors found was 110. Thus, we got a reduction by 164 flagged records = 42 per cent. The impact of the remaining errors was calculated for each variable, for all the 89 groups for which SEW data are published. See Table 2 in the annex.

This study was carefully analyzed. The most important findings were that the acceptance intervals of the checks should be wider and should not be symmetric around 1 for the ratios and around zero for the differences. Furthermore, the best strategy is to set the limits as close as possible to the first outlier on both sides.

If the limits had been changed according to the findings, the outcome would have been 134 flagged data would have been 134 which means a reduction of 66 % of the verifying work. For the corresponding quality table on the impact of the remaining errors on the estimates, see Table 2 in the annex.

1.3 The PC-application on the SEW (Study No 3)

A prototype of the same editing process was developed in PC-SAS for micro computers. It was evaluated on the SEW data from August, 1989. However, the realization of the Aggregate Method was somewhat modified. Instead of forming an error file from the aggregate check consisting of all questionnaires of all aggregates which had failed at least one edit, the records of a flagged aggregate were given a specific signal, telling which of the four variables that did not pass the edit. The check on the micro level was then applied only to those questionnaires which belonged to the aggregates which had failed the check for that variable. This version of the method may imply a small reduction in the number of flagged data, as fewer records are checked on the micro level than in the realization described previously.

The result of this study was a reduction in the number of flagged data by nearly 80 per cent.

However, the loss in quality was slightly higher than in the preceding studies (see Table 3 in the annex) due to the modification, the unusually large number of questionnaires in the second processing round and/or to the wider acceptance interval. It was found that this loss in quality was caused by a few large errors, which did not cause the aggregates to be flagged. These errors were very easy to detect. One interesting method of detecting those errors was to let all data pass a ratio check with very wide acceptance intervals. This could be done in the micro-editing part of the procedure or as a final check in detail in Lindström (1990b).

The aggregate checks method as such was then questioned. The only advantage of the method is that they can save storage or computer time. However, when there are no problems with either the storage capacity or the computer cost, the aggregate checks can be skipped.

The method is still a macro-editing method but the term "Aggregate Method" may not be adequate. When the tails of the distribution of the check function is provided with Box-Plots (which is recommended), this method is called the Box-Plot Method.

1.4 Conclusion

The simulations show that the Aggregate Method can form the main part of the editing process in the production processing of a survey. The method reduces the verifying work by 35 - 80 per cent without losses in quality or timeliness. The macro-editing concept is a realistic alternative or complement to micro-editing methods, and can be applied during the processing of the data under the same conditions as computer-assisted micro-editing methods, which reduces the manual verifying work to a considerable extent.

For small surveys the Box-Plot Method should be considered.

2. The Top-Down Method

The Top-Down Method is described in Granquist's article in the latter part of this publication and in Lindblom (1990). The method has been implemented in the mainframe application of "The Survey of Delivery and Orderbook Situation" (DOS). The production system

is written in APL. A prototype written in PC-SAS for running on a micro computer has been developed and is reported in Lindblom (1990).

2.1 Description of the method

The idea behind the method is to sort the values of the check functions (which are functions of the weighted keyed-in values) and start the manual review from the top or from the bottom of the list and continue until there is no noticeable effect on the estimates.

The method is described as it is applied in the DOS production system. The generalization is obvious.

The procedure is governed by an inter-active menu program, written in APL. The in-data file is successively built up by the records of the three batches from the data-entry stage. There are three functions to select the records to be studied, i.e.

i) the 15 highest positive changes
ii) the 15 highest negative changes
iii) the 15 greatest contributions

which for every variable can be applied to the total and to the 38 branches. For a selected function and domain of study, the screen shows the following list for the 15 records of the in-data file sorted top-down:

IDENTITY	DATA	WEIGHT	WEIGHTED VALUE	TOTAL
...	...	...	...	...
...	...	...	...	...

The operator selects a record and immediately gets the entire content of the record on the screen. If an error is identified he can up-date the record on the screen and at once see the effects. The record usually looses its position on the top 15 list and the total is changed. The operator goes on until further editing does not change the total.

2.2 Experiences

A study expressly designed to compare the Top-Down Method with a corresponding micro-editing method has not yet been carried out. However, the implementation of the Top-Down Method in the DOS processing system has made such an evaluation study unnecessary.

In 1985 a micro-editing procedure was developed with the intention that it should be the editing procedure for DOS. When the system was run for the first time it produced so many error messages that the subject matter specialists realized that they had neither resources nor energy to handle them. Especially as they knew by experience that only a small percentage of the flagged data really were marred by detectable errors. They had constructed the checks on basis of all the experience and subject matter knowledge they had gained during years of work with the survey.

The procedure was flexible, user-friendly and easy to fit to the data to be edited. Yet the procedure did not work.

The Top-Down procedure, which had been developed as a complementary or reserve procedure, had to be taken in use at once. The staff is very satisfied with it. It is continuously providing the staff with new information on the subject matter and problems of the survey.

Since the first processing with the macro-editing method, the number of records for manual review has decreased slowly. The subject-matter statisticians have become convinced that there is no need for editing on the industry level. The Top-Down lists are now only produced at the total manufacturing industry level. Though there still seems to be a certain amount of over-editing it is doubtless the most rational editing procedure at Statistics Sweden.

According to Anderson (1989a) the method is also considered as the most efficient out-put editing method in use at the Australian Bureau of Statistics.

2.3 Conclusion

We have shown that the Top-Down Method can be used as an editing method during the processing of a survey without losses in quality and timeliness. The method can reduce the verifying work by 50-75 per cent. The subject-matter clerks are very satisfied because they feel in control of the editing task and can see the effects of their work.

The method should not be applied to more than ten variables of a survey at the same time.

3. The Hidiroglou-Berthelot Method (Statistical Edits)

The Hidiroglou-Berthelot Method (the HB-Method) is described as a micro-editing method in more detail in Hidiroglou et al. (1986). The method is a ratio check inspired by Tukey's Explorative Data Analysis (EDA) methods in more detail Tukey (1977), and in the paper it is considered as a solution to some problems connected with the traditional ratio-check method. It is in use at Statistics Canada and there known as "Statistical Edits".

As a macro-editing method it is reported in Davila (1989). At Statistics Sweden the HB-Method has been studied on both DOS and SEW data. Only the DOS study has been reported in English in Davila (1989).

3.1 Description of the method

The HB-Method is a ratio method, for which the bounds are automatically calculated from the data to be edited. The method uses the robust parameters median, quartiles (Qi) and interquartile ranges (DrQi) instead of the mean and standard deviation to prevent the bounds from being influenced by single outliers. Then the lower (l) and the upper (u) bounds should be:

$l = R_{MEDIAN} - k * D_{rQ1}$

$u = R_{MEDIAN} - k * D_{rQ3}$

However, such a straightforward application of the ratio method has two drawbacks,

i) the outliers on the left tail may be difficult to detect
ii) the method does not take into account that the variability of ratios for small businesses is larger than the variability for large businesses

The HB-Method solves these drawbacks by a symmetric transformation followed by a size transformation.

The symmetric transformation

$$S_i = \begin{cases} 1 - R_{MEDIAN} / R_i \ , & 0 < R_i < R_{MEDIAN} \\ R_i / R_{MEDIAN} - 1 \ , & R_i \ \$ \ R_{MEDIAN} \end{cases}$$

The size transformation

$E_i = S_i * (MAX (X_i (t) \ , \ X_i (t+1)))^{U}$
$0 \# U \# 1$

E_{Q1} , E_{Q3} are the first and third quartiles of the transformation E

$D_{Q1} = MAX (E_{MEDIAN} - E_{Q1} \ , \ * A * E_{MEDIAN} *)$

$D_{Q3} = MAX (E_{Q3} - E_{MEDIAN} \ , \ * A * E_{MEDIAN} *)$

which gives the lower and upper limits of the checks:

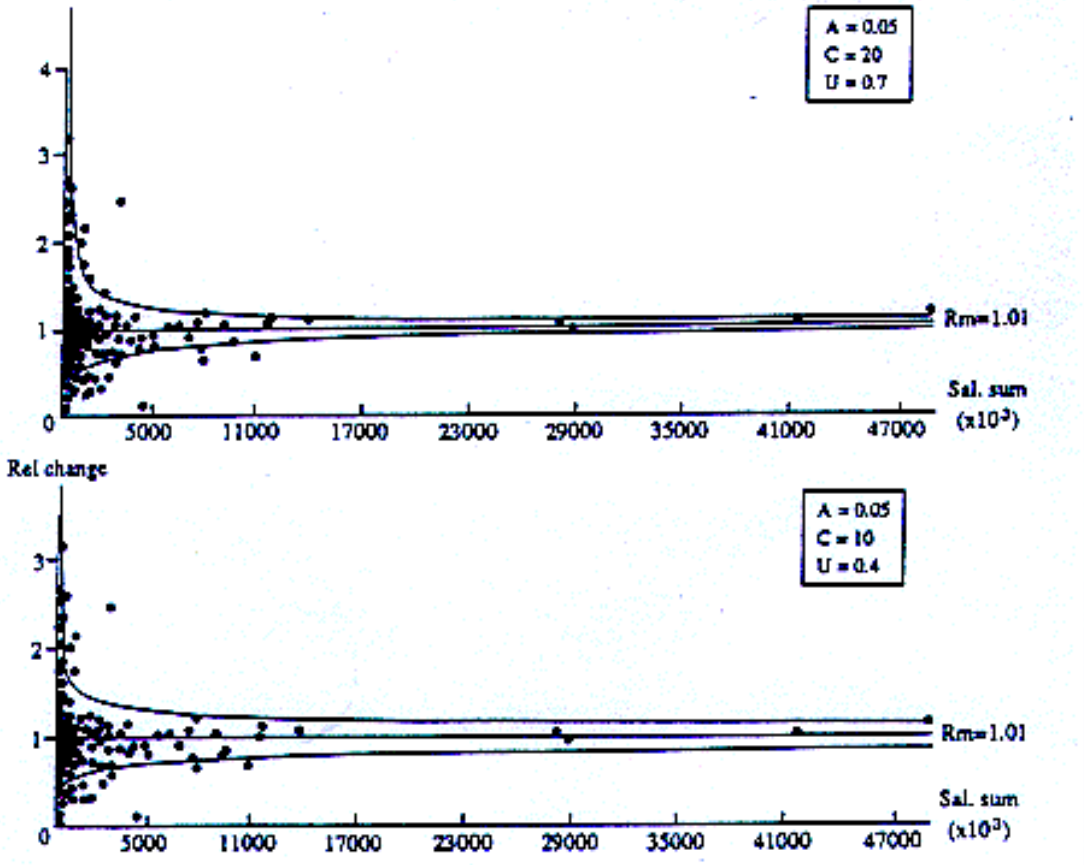
$l = E_{MEDIAN} - C * D_{Q1}$

$u = E_{MEDIAN} + C * D_{Q3}$

A is considered a constant, equal to 0.05, which means that there are only two parameters, U and C, which have to be set in advance to get the method to run. The real reason behind the symmetric transformation is to get rid of one parameter. We have found that the parameters are not very sensitive. The same values can be used for many variables of a survey.

Figure 1 may explain the transformations and the method. The figure has been produced by the prototype of the Box Method for the SEW data. The acceptance limits have been calculated for a few values of the parameters U and C. This method can be applied when implementing the HB-Method forming an operation in the production process. That way the acceptance bounds would be completely determined by the data to be edited.

FIGURE 1 HIDIROGLOU-BERTHELOT



To make the method a macro-editing method we only had to inflate the keyed-in values of the variable X, in principle by the inverted sample fraction.

3.2 The study on the Survey of Delivery and Orderbook Situation (DOS)

In the study the keyed-in data of January-88 were used together with the final data of December-87. A change file was formed by the January-88 data edited with the Top-Down Method and the original keyed-in data of January-88. The H-B Method was tuned and evaluated against the errors (changes) found (the data on the change file) by the current editing procedure, the Top-Down procedure. The change file is shown in Table 4 in the annex.

Table 5 in the appendix shows the impact of the changes found by the HB-Method.

Table 6 in the appendix shows the impact on the other variables edited by the HB-Method with the values of the parameters C and U which were used for the variable "deliveries to the domestic market".

3.3 The study on the SEW

A study designed almost exactly as the one reported has been carried out by Statistics Sweden on data of the SEW. Then the method was compared with the current editing procedure (a traditional one) and with the Aggregate Method which not yet is in use for that survey. The HB-Method was found to be superior to the current procedure and equal to or better than the Aggregate Method. This study is reported in Swedish only.

4. The Box-Plot Method

In the Box-Plot method distributions of the check functions of the weighted keyed-in values are displayed as box-plots. Then the acceptance intervals for the checks are set by the staff on the basis of these graphs and put into the regular error-detecting program. By Anderson (1989a) and the studies on the Aggregate Method reported in this paper, the definitions of extreme outliers may serve as guidelines for efficient limits. The only difference to the method reported Anderson (1986) is that the values should be weighted by (approximately) the inflation factor (according to the sample design).

The concept of Box-Plots was introduced by Tukey (1977). The Box-Plot Method as a micro-editing method is reported in Anderson (1989b). It is suggested as a macro-editing method in the discussion on the Aggregate method.

Anderson (1989a) reports an experiment carried out on data from the Australian Bureau of Statistics (ABS) survey "Average Weekly Earnings" (AWE). ABS extended the bounds of every ratio check used in that survey to (Q1-3*IQR, Q3+3*IQR), where Q1, Q3 and IQR are respectively the first and the third quartile and the interquartile range.

The study indicates that 75 per cent of the resources for the manual reviewing of flagged data from the whole editing process could be saved by limiting the manual review to "extreme outliers". In Anderson (1989a) the same method for evaluation as in the studies related above is used. The remaining errors in data had no significance at all on the estimates.

In the latter version of the report, Anderson (1989b), Anderson suggests that the lower and upper bounds of the checks should be set on the basis of a manual analysis of box-plots of the check functions. Then the bounds can be modified by taking into account outliers near the bounds for extreme outliers. Above all, the survey staff will get full control of responsibility for the data editing.

5. The Box Method

The Box Method is a graphical macro-editing method developed at Statistics Sweden.

The basic principles are to utilize computer graphics to visualize the distribution of the check function of the weighted data and the interactivity of a computer to get indications of when to stop the manual verifying work. The method may be considered as a combination of a generalized Box-Plot method and the Top-Down Method.

The keyed-in data are weighted and then put into the check function. Any mathematical expression may be used as a check function. The values of the function are plotted on the screen

and acceptance regions of any shape can also be provided. The reviewer draws a box around the observations he wants to review. On the screen, the data then appear on in advance selected items of the records belonging to the data points inside the box. For every check function the user can select the items of the records to be displayed. A change is entered inter-actively and some data (statistics) on the impact of the change will be displayed.

The method may also be used as a tool to find appropriate values of acceptance regions for other editing methods (e.g. the HB-Method).

IV. SUMMARY

1. The choice of the methods

The basic principle of all the reported macro-editing methods is that the acceptance regions are determined solely by the distributions of the received observations of the check functions. The keyed-in values of the variable to be checked are weighted by the inflation factor before it is put into the check function.

In the Aggregate, Box-Plot, HB and Top-Down methods all the values of the check function are sorted by size.

The tails of the distributions are displayed and analyzed in the Aggregate and the Box-Plot methods in order to set the acceptance bounds for the checks. In the HB-Method this setting of the acceptance limits is done automatically.

In the Top-Down and the Box methods the effects of the detected errors on the estimates "determine" how far the manual review should go. In the Top-Down Method the manual review work starts with the extreme values and goes towards the median value, while in the Box Method the records for manual reviewing are selected by the reviewer from a graphical display of the values of the check function. This selection may be supported by guide-lines (acceptance regions) displayed in the same graph.

The choice of method should be based on the number of variables to be edited by the macro-editing method and on how the staff wishes to work.

2. Macro-editing versus micro-editing methods

Macro-editing is not a new concept. It has always been used in data editing, but only as a final checking. Another term is out-put checking.

What is new is that such out-put check methods can be used in data editing procedures in the same way as micro-editing methods and that they have proved to be much more efficient than traditionally applied micro-editing procedures. Here reported studies reduce the work by 35 - 80 per cent.

Macro-editing methods of the type described in this paper may be considered as a statistical way of providing micro-editing checks with efficient acceptance limits. The limits are

based only on the data to be edited. The methods bring a kind of priority thinking to the verifying work and data are edited according to their impact on the estimates. The macro-editing methods solve the general problem inherent in micro-editing methods, i.e. that they produce too many error signals without giving any guidance as to how the resources of the verifying work are to be allocated. We have seen that with micro-editing procedures even very large errors are not always detected, due to the large number of flagged data.

However, there are no limits to the number of cases that can be selected for a manual review. The reviewer can select all the cases he deems necessary. The difference to micro-editing methods is that the cases are selected in priority order, i.e. according to the impact they may have on the estimates. The selection is mainly done by the reviewer, which means that he governs and has the full responsibility for the whole scrutinizing work. In micro-editing procedures, the reviewer is dominated by the computer and cannot see the effects of his work.

Both procedures focus on randomly appearing negligence errors and utilize the same principle to point out outliers or extreme observations. Micro-editing procedures flag data by criteria fixed in advance, based on historical data, while macro-editing procedures focus on data, which at that very moment and relative to the estimates are the most extreme ones.

Systematic errors, e.g. that many respondents misunderstand a question in the same way or deliberately give wrong answers cannot (in principle) be detected by either macro-editing or micro-editing methods. This kind of errors have to be detected by other types of methods. Below a few references are given to methods which focus on misunderstanding errors.

In Mazur (1990), the Response Analysis Survey approach is presented, which implies that in an on-going survey, a special survey on how the respondents answer certain questions is conducted by a self-response touch-tone technique. By this technique, errors are detected which cannot be found by traditional editing methods, which reduces the bias in the estimates.

In Werking et al (1988), a method to find so called in-liers in survey data is presented. In repetitive surveys there are respondents which for every period report the same figures also to questions which certainly require the answers to change between periods. They are termed in-liers, because they always lie between the bounds of traditional edits. In these experiments it is found that these in-liers may cause considerable bias in the estimates.

What we have proved is that the kind of macro-editing methods outlined here certainly provide a more efficient way than traditional micro-editing methods of reaching the same "quality" standard and that they may release resources for editing misunderstanding errors.

ANNEX

THE AGGREGATE METHOD

Study No. 1 on the SEW data TABLE 1 Number of aggregates by the total relative difference of the estimates in percent.

DIFFERENCE	WORKERS	HOURS	PAY-ROLL	WAGES/HOUR
0 < - < 0.05	4	1	6	6
0.1 - 0.4	4	5	5	8
0.5 - 0.9	1	2	4	2
2.5	1	0	0	0

Study No. 2 on the SEW data TABLE 2 Number of aggregates by the total relative difference in per cent of the estimates. The figures within parentheses show the outcome of the first experiment of the same study.

DIFFERENCE	WORKERS	HOURS	PAY-ROLL	WAGES/HOUR
0 < - < 0.05	4 (4)	3 (2)	13 (12)	10 (8)
0.1 - 0.4	3 (3)	6 (6)	4 (4)	8 (8)
0.5 - 0.9	1 (1)	1 (1)	6 (5)	3 (3)
1.0 - 1.9	1 (1)	2 (1)		1 (1)
2.0 - 2.9				
3.0 - 3.9	(1)	(1)		
4.0 - 4.9			(1)	

Study No. 3 on the SEW data TABLE 3 Number of aggregates by the total relative difference in per cent of the estimates. The figures within parentheses show the outcome of study no 2 (see TABLE 2)

DIFFERENCE	WORKERS	HOURS	PAY-ROLL	WAGES/HOUR
0	72 (78)	45 (77)	38 (66)	36 (68)
0.0 - 0.1	1 (4)	6 (2)	6 (12)	15 (8)
0.1 - 0.4	10 (3)	10 (6)	14 (4)	18 (8)
0.5 - 0.9	3 (1)	6 (1)	6 (5)	6 (3)
1.0 - 1.9	0 (1)	9 (1)	15	7 (1)
2.0 - 2.9	1	3	2	3
3.0 - 3.9	1 (1)	2 (1)	1	1
4.0 - 4.9	0	2	2 (1)	1
> 4.9	0	5	4	1

THE HIDIROGLOU-BERTHELOT METHOD

The Study on the DOS Data

TABLE 4 Change-file of the variable "domestic deliveries" (SEK 1000's)

OBS	DECEMBER	JANUARY		
		OLD	NEW	CHANGE
1	14648285	7403131	7403	-7395728
2	1560	1605000	1605	-1603395
3	1032	973693	974	-972719
4	1500	925000	925	-924075
5	1302	902805	903	-901902
6	962	593636	594	-592942
7	1926	501675	502	-501173
8	8473	38010	25328	-12682
9	110893	107891	101520	-12682
10	31745	42201	37201	-5000
11	5000	8300	5200	-3100
12	21930	19013	16730	-2283
13	3465	3084	1007	-2077
14	2879	5495	3650	-1845
15	22839	22400	20977	-1423
16	7344	11758	10840	-918
17	1651	1601	1076	-525
18	4072	4061	3691	-370
19	127	262	114	-148
20	8839	1200	1130	-70
21	434	480	462	-18
22	831	664	649	-15
23	11990	9790	9780	-10
24	4834	4404	4398	-6
25	301	116	111	-5
26	1882	4995	4994	-1
27	6245	5658	5659	1
28	274190	271455	271456	1
29	47500	38509	38539	30
30	148	35	80	45
31	1046	1860	1936	76
32	14551	14000	14169	169
33	5383	463	1032	569
34	20400	2500	3600	1100
35	173860	165000	167261	2261
36	6180	1702	4131	2429
37	6000	6000	14300	8300
38	294600	54	54000	53946

The sum of all changes: 12 997 728 (SEK 1000's)

TABLE 5 Impact of changes found by the HB-Method.

Number of changes found	Accumulated sum of changes found	Accumulated sum relative to the sum of all changes
7	5 550 152	42.7 %
8	5 562 834	42.8 %
9	12 958 565	99.7 %
10	12 959 665	99.71 %
11	12 960 234	99.71 %
12	12 960 304	99.71 %
13	12 960 305	99.71 %
14	12 962 566	99.73 %
15	12 964 995	99.75 %
16	12 966 840	99.76 %

TABLE 6 impact on the other variables edited by the HB-Method with the values on the parameters C and U which were used for the variable "deliveries to the domestic market".

VARIABLE	DELIVERIES		0RDERBOOK		STOCKS	
	Domestic	Foreign	Domestic	Foreign	Domestic	Foreign
Number of changes	38	19	109	72	31	30
Number of HB changes	9	4	6	6	4	4
Number of flags	28	25	30	35	12	10
Sum of all changes	12 997728	1 157332	9 080144	825626	4 316568	6 096551
Sum of HB changes	12 958562	1 029825	8 701388	485480	4 209380	5 856540
Relative the sum of all changes	99.7 %	88.98 %	95.82 %	58.80 %	97.52 %	96.06 %

MACROEDITING - THE HIDIROGLOU-BERTHELOT METHOD (STATISTICAL EDITS)

by Eiwor Hoglund Davila
Statistics Sweden

Abstract: The intention of this report is to evaluate a method presented by Hidiroglou and Berthelot in 1986. This heuristic method has been studied using data from the Swedish Delivery and Orderbook Survey. Although this method is rather complicated, it proved to be an efficient tool for detecting divergent observations in large periodical surveys.

I. INTRODUCTION

The Hidiroglou-Berthelot method has been studied using data from the Swedish Delivery and Orderbook Survey. During 1987 and 1988, the current method of editing was subjected to a study in which both the edited and unedited data were kept. This allowed the parameters of the Hidiroglou-Berthelot method to be estimated with the help of statistical methodology. The Hidiroglou-Berthelot could not be adapted so that all observations that had been changed in the regular editing process were found. However, the Hidiroglou-Berthelot method was shown to be able to identify the most important of these observations.

A general problem in all surveys is the appearance of incorrect or divergent values among the observations. There are many different reasons for such values e.g. a misunderstanding of questions or the purpose of the questionnaire, faults committed at data entry, inaccurate or deliberately erroneous answers. The reason might even be that it is a correct value which is differing noticeably from the rest of the observations. Consequently, it is desirable to find good methods for detecting divergent observations without spending too much effort and time. The rectification of erroneous values improves the quality of the data and of course strengthens the credibility of any estimate produced. The majority of the business-surveys at Statistics Sweden are of periodical type as for example the monthly Delivery and Orderbook Survey. The data is usually collected through mail questionnaires and then different test procedures for detecting errors and diverging observations are applied, such as matchings with registers or checks on relationships between different variables and periods. Unfortunately, this consumes a large part of the survey budget - apart from the fact that the data often is overedited. Efforts to find and implement more efficient methods for statistical editing are therefore justified. The intention of this report is to evaluate a method presented by Hidiroglou and Berthelot (1986). In Canada, the Hidiroglou-Berthelot method, below referred to as the H&B method, has been adapted to several surveys, e.g. the Delivery, Stock and Order Survey described by Lalande (1988), the Current Shipment, Inventories and Orders survey described by Tambay (1986) and the Wholesale-Retail survey described by Berthelot (1985).

II. THE HIDIROGLOU-BERTHELOT EDIT

Numerous methods have been proposed for detecting divergent observations in large periodical surveys. Some suggest that the problem should be treated as a hypothesis-testing problem, either with or without the assumption of a certain distribution for the data. Other methods use ratios of current period data to previous period data and set upper and lower bounds according to some rule which may depend on the distribution of the ratios of the data. Observations outside the bounds are considered to be divergent. Usually, the methods try to make use of the distribution of the ratios in the construction of bounds around the mean or the median value either using the standard deviation or the quartiles. Most of the suggested methods have some drawbacks, mainly because the variables in business-surveys usually exhibit very skewed distributions.

The Hidiroglou-Berthelot edit examined in this paper is a heuristic method that has been developed using parts of several other methods. By using information provided by the data itself, the intention is to identify all big changes regardless of whether it is an increase or a decrease. Given data for a variable from two consecutive periods,

$(X_i(t) , X_i(t+1))$ $\quad i = 1,2,...,n$

the individual relative change for each element is defined as:

$R_i = X_i(t+1)/X_i(t)$

H&B claim that to be able to find and treat both increases and decreases in the same way, R_i has to be transformed in the following manner:

$$S_i = \begin{cases} 1 - R_{median}/R_i , & 0 < R_i < R_{median} \\ R_i/R_{median} - 1 , & R_i >= R_{median} \end{cases}$$

where R_{median} is the median of the R_i ratios.

Half the number of S_i's are less than zero and the other half greater than zero.

However, according to H&B, the transformation ensures an equally good detection of divergent observations in both tails of the distribution. The transformation does not provide a symmetric distribution of the observations. This is perhaps more obvious if S_i is rewritten as:

$$S_i = \begin{cases} (R_i - R_{median})/R_i , & 0 < R_i < R_{median} \\ (R_i - R_{median})/R_{median} , & R_i >= R_{median} \end{cases}$$

To make use of the magnitude of the observations, a second transformation is performed:

$E_i = S_i * \{ MAX(x_i(t), x_i(t+1)) \}^U$

U is by H&B proposed to be a value between 0 and 1.

The E transformation is a rescaling of the S's that keeps the order and sign of the elements. It makes it possible to put more importance on a relatively small change in a "large" element than on a relatively large change in a "small" element. The choice of the U-value governs the importance associated with the magnitude of the data. U=1 gives full importance to the size of the element x_i and U=0 gives no importance at all to its size. In the latter case, $E_i=S_i$. The effects of different choices of U is illustrated in example 1.

EXAMPLE 1

In this example, R_{median}= 1.25 and A=0.05

$x_i(t)$	$x_i(t+1)$	R_i	S_i	E_i(U=0.1)	E_i=(U=0.4)	E_i=(U=0.9)
1	5	5	3	3.52	5.71	12.77
10	5	0.5	-1.5	-1.89	-3.77	-11.91
100	5	0.05	-24	-38.04	-151.43	-1514.30
1000	5	0.005	-249	-496.82	-3946.38	-124795.62
10000	5	0.0005	-2499	-6277.20	-99486.98	-9948698.19
100000	5	0.00005	-24999	-79053.78	-2499900.00	-790537792.47
5	1	0.2	-5.25	-6.17	-9.99	-22.35
5	10	2	0.6	0.76	1.51	4.77
5	100	20	15	23.77	94.64	946.44
5	1000	200	159	317.25	2519.98	79688.77
5	10000	2000	1599	4016.51	63657.34	6365733.66
5	100000	20000	15999	50593.28	1599900.00	505932802.98

The example shows that negative E_i's (and S_i's) indicate a decrease and positive E_i's (and S_i's) an increase. The greater U becomes, the greater the dispersion of E will be. The E_i's are distributed around zero and those E_i's that are too small/big are considered as possible outliers. In this context, the H&B definition of an outlier is an observation "whose trend for the current period to a previous period, for a given variable of the element vector $\underline{X}(t)$, differs significantly from the corresponding overall trend of other observations belonging to the same subset of the population". H&B construct upper and lower limits for an interval around E by means of the following quantities:

$D_{Q1}=MAX\{E_{median} - E_{Q1} ,\ A*E_{median} | \}$

$Q_3=MAX\{E_{Q3} - E_{median}, A*E_{median} | \}$

A is an arbitrary value suggested by H&B to be 0.05. E_{median}, E_{Q1}, E_{Q3} are the median and the first and third quartiles of the transformation E. The $A*E_{median}$ term is a protection against detecting too many non-outliers when the E_i's are clustered around a single value with only a few deviations. That is, when $E_{median} - E_{Q1}$ or $E_{Q3} - E_{median}$ are less than $A*E_{median}$. In example 1 above, with U=0.4 and A=0.05, E_{median}=1.13 which means that if E_{Q1} is in the interval [-1.1865,-1.13], then $E_{median} - E_{Q1} \leq A*E_{median} = 0.0565$ and thus $D_{Q1} = 0.0565$. All values outside the interval $\{E_{median} - C*D_{Q1} , E_{median} + C * D_{Q3}\}$
where C is a constant, are treated as outliers. Increasing the value of C gives a wider interval and a lower number of outliers. The point in using the quartiles instead of standard-deviations is to avoid too much influence from the outliers.

EXAMPLE 2

Using the values of Example 1 again, with U=0.4 and A=0.05 gives

$E_{median} = (-3.77+1.51)/2 = -1.13$

$E_{Q1} = -3946.38$

$E_{Q3} = 2519.98$

$D_{Q1} = -1.13-(3946.38) = 3945.25$

$D_{Q3} = 2519.98-(-1.13) = 2521.11$

C=10 gives the interval
$(-1.13 - 10 * 3945.25 , -1.13 + 10 * 2521.11) = (-39453.63 , 25209.97)$.

In Example 1, two observations in each tail would be classified as an outliers. If C=30, the interval is (-118358.63, 75632.17) and only one observation in each tail would be identified as a possible outlier.

III. APPLICATION OF H&B APPROACH

In the application of the H&B method, three parameters, A, U and C, had to be estimated. H&B do not give any indications about the choice of the parameters other than that A should be very small and O<=U<=1. A large number of combinations of parameter values had to be tried in order to see what happened to the material. The material used for the study contained both the raw-data set and the final data set edited according to the currently used method. Each element whose value had been changed was considered to be an outlier. Note that the meaning of the word outlier here is not exactly the same as in the definition given by H&B. The data could thus be separated in two populations, one population of outliers and one population of non-outliers. This was exploited when estimating the parameters. In fact, the problem was treated as a classification problem by Anderson, T.W. (1958). The attention was focused on finding some combination of the parameters A, U and C, that would minimize the probability of misclassification of an element. For that purpose, the following points were defined:

(i) As many outliers as possible should be correctly classified.
(ii) As few non-outliers as possible should be misclassified.

These two points contradict each other and a loss function would have been useful. Unfortunately, this was not possible since no information about the consequences of misclassification was available at the time of writing this report. A third aspect of interest was that (iii) the identified outliers ought to have both a large impact on the estimates and a monthly change that clearly diverge from previous month. The method of this paper consider (i) and (ii) in the first stage and in a second stage point (iii). A grid of different combinations of parameters U and C was run through. Parameter A was fixed at 0.05 (see section 3.1 and 4.1). For each combination, the edit identified a number of observations as possible outliers, (# outliers correctly identified by H&B)/(total # outliers) gave a measure that was used when evaluating the method. This measure can be regarded as the empirical probability of "good" classification = 1 - Pr (misclassification). After all the combinations of U and C had been tried, one was chosen for the examination of the effect on the other variables.

IV. DATA AND RESULTS

The basic idea in the current editing method for the Delivery and Orderbook Survey at Statistics Sweden is to keep on checking and changing values for as long as there is an effect on the estimates. To be more precise : the values are compared to their corresponding values of the previous month by means of the ratio. The 15 largest relative changes in each tail are picked out to be checked and possibly corrected. Finally the estimates are calculated. This procedure is repeated over and over until the estimates are considered to be stable. The method is described in more details in Granquist, 1987-04-09: Macro-editing - The Top-Down Method.

The data used for estimating the parameters of the H&B edit came from the monthly Delivery and Orderbook survey mentioned earlier. There were six variables of interest included in the raw datafile : Domestic Delivery, Export Delivery, Domestic Order, Export Order, Domestic Stock and Export Stock. The variable chosen for the adoption of the parameters, Domestic Delivery, was the one having most non-zero observations left after excluding missing values since such observations have to be taken care of separately. The data used in the H&B method were the original values reported in the month of January 1988 and the final values of December 1987 where the latter had been checked with the current edit. Comparison between the January 1988 data edited with the currently used method and corresponding raw data of the same month identified the population of outliers. For the variable "Domestic Delivery", the population of outliers consisted of the 38 observations in Table 1 and the population of non-outliers of 1511 observations (see figure 1).

TABLE 1: THE POPULATION OF OUTLIERS (DOMESTIC DELIVERY)

OBS	DECEMBER	JANUARY		CHANGE	OBS	DECEMBER	JANUARY		CHANGE
		OLD VALUE	NEW VALUE				OLD VALUE	NEW VALUE	
1	14648285	7403131	7403	-7395728	20	8839	1200	1130	-70
2	1560	1605000	1605	-1603395	21	434	480	462	-18
3	1032	973693	974	-972719	22	831	664	649	-15
4	1500	925000	925	-924075	23	11990	9790	9780	-10
5	1302	902805	903	-901902	24	4834	4404	4398	-6
6	972	593536	594	-592942	25	301	116	111	-5
7	1926	501675	502	-501173	26	1882	4995	4994	-1
8	8473	38010	25328	-12682	27	6245	5658	5659	+1
9	110893	107891	101520	-6371	28	274190	271455	271456	+1
10	31745	42201	37201	-5000	29	47500	38509	38539	+30
11	5000	8300	5200	-3100	30	148	35	80	+45
12	21930	19013	16730	-2283	31	1046	1860	1936	+76
13	3465	3084	1007	-2077	32	14551	14000	14169	+169
14	2879	5495	3650	-1845	33	5383	463	1032	+569
15	22839	22400	20977	-1423	34	20400	2500	3600	+1100
16	7344	11758	10840	-918	35	173860	165000	167261	+2261
17	1651	1601	1076	-525	36	6180	1702	4131	+2429
18	4072	4061	3691	-370	37	6000	6000	14300	+8300
19	127	262	114	-148	38	294600	54	54000	+53946

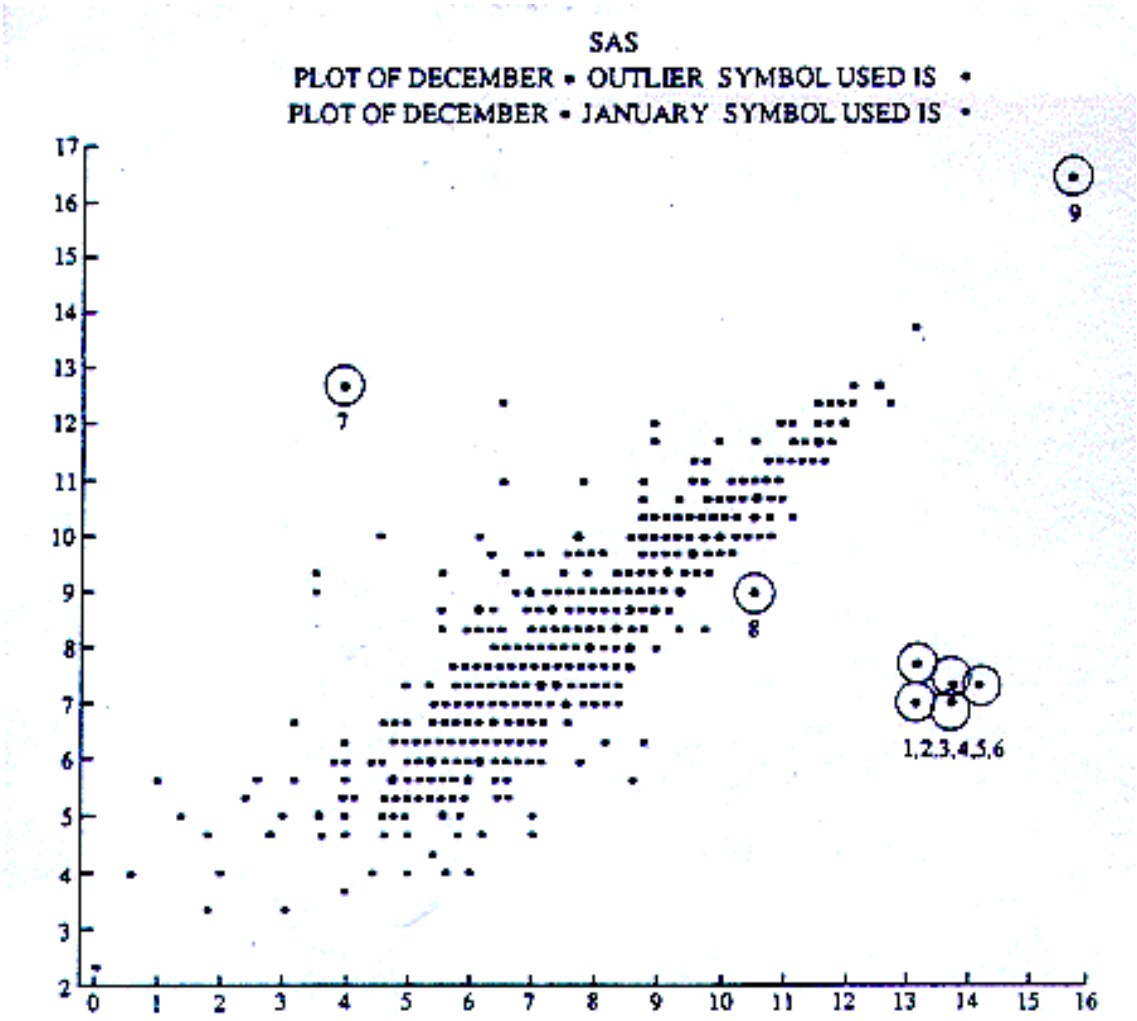


FIGURE 1:

Plot of the logarithmed values of the variable Domestic Delivery for the two successive periods. The outliers are marked with `*' and the outliers found by the H&B edit (U=0.4, C=41) are marked `*'.

1. Estimation of the parameters

According to the method described in section 3, several combinations of different values of the parameters A, U and C were tried. Changing A for some fixed value of U and C had no impact at all on the probability of misclassification. In this particular case,

$E_{median} - E_{Q1} > | A*E_{median} |$ and

$E_{Q3} - E_{median} > | A*E_{median} |$ for small values of A.

Therefore, A was set to 0.05, as suggested by H&B, which simplified the further computations. The remaining parameters U and C were then systematically varied to each other. The results are shown in Table 2.

TABLE 2: RESULTS OF THE VARIATION OF U AND C (number of correctly identified outliers)/(total number of observations classified as outliers by H&B method).

C											
47	7/21	7/20	7/20	7/22	7/23	9/30	9/32	10/37	10/43	10/51	11/62
45	7/21	7/20	7/20	7/22	7/24	9/31	9/33	10/38	10/43	10/54	11/63
43	7/23	7/21	7/22	7/22	9/28	9/33	9/34	10/39	10/46	10/56	11/64
41	7/24	7/23	7/22	7/25	9/28	9/33	9/34	10/40	10/47	10/58	11/64
39	7/26	7/23	7/22	7/25	9/29	9/33	10/37	10/43	10/47	11/61	11/67
37	7/28	7/25	7/23	7/26	9/30	9/35	10/38	10/43	19/48	11/61	11/73
35	7/28	7/25	7/24	7/27	9/33	9/35	10/39	10/44	10/52	11/64	12/79
33	7/29	7/25	7/26	8/29	9/33	10/38	10/41	10/44	10/53	12/67	13/84
31	7/31	7/27	7/29	8/29	9/36	10/39	10/42	10/47	11/58	13/73	13/85
29	7/31	7/31	7/31	8/31	10/38	10/40	10/43	11/49	11/61	13/75	14/92
27	7/32	7/31	8/32	8/31	10/39	10/41	11/45	11/52	12/67	13/82	15/96
25	7/35	7/34	8/32	8/33	11/45	11/45	11/48	11/57	13/72	13/84	15/98
23	7/37	7/38	8/33	10/41	11/45	11/47	11/56	12/63	13/77	13/89	15/105
21	7/39	9/42	9/38	10/43	11/46	11/51	12/58	12/70	13/81	14/97	16/108
19	8/43	9/43	10/42	10/45	11/50	12/56	12/62	12/76	13/86	15/102	16/119
17	9/48	9/44	10/46	11/49	12/54	12/62	12/70	12/82	14/98	15/110	16/131
15	9/52	10/49	10/52	12/55	12/62	12/67	12/81	13/92	14/105	16/124	16/139
	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0 U

As can be seen, no combination of the parameters could identify more than about 40% of the 38 outliers. An additional number of combinations of U and C, for U>1 and C>47, were also tried out without any diverging results.

2. The outliers

What kind of outliers were then found? The sum of the absolute values of the changes shown in table 1 equalled 12 997 728. This was taken as a base for comparison with the results of section 4.1.

TABLE 3: CHANGES REPRESENTED BY THE OUTLIERS IDENTIFIED BY THE H&B EDIT

Sum of all 38 changes = 12 997 728.

# of outliers found by H&B	Sum of changes they represent	% of all changes
7	5 550 152	100* (5 550 152/ 12 997 728) = 42.70
8	5 562 834	100* (5 562 834/ 12 997 728) = 42.40
9	12 958 565	100* (12 958 565/ 12 997 728) = 99.70
10	12 959 665	100* (12 959 665/ 12 997 728) = 99.71
11	12 960 234	100* (12 960 234/ 12 997 728) = 99.71
12	12 960 304	100* (12 960 304/ 12 997 728) = 99.71
13	12 960 305	100* (12 960 305/ 12 997 728) = 99.71
14	12 962 566	100* (12 962 566/ 12 997 728) = 99.73
15	12 964 995	100* (12 964 995/ 12 997 728) = 99.75
16	12 966 840	100* (12 966 840/ 12 997 7280 = 99.76

Considering the three aspects outlined in section 3.2 together with a restriction on the amount of observations appointed by H&B for a check, the number of combinations diminished considerably. For convenience, combinations given by U=0.4 or 0.5 and a C between 15 and 43 would be preferable. Another point of interest is what the observations flagged by the H&B method looked like.

TABLE 4: THE OBSERVATIONS FLAGGED BY H&B METHOD USING U=0.4 AND C=41

OBS	DECEMBER	JANUARY	OBS	DECEMBER	JANUARY
1*	14648285	7403131	15	22773	103
2*	1560	1605000	16	11434	789
3*	1032	973693	17	16166	1100
4*	1500	925000	18	61536	6040
5*	1302	902805	19	105933	8310
6*	972	593536	20	156455	7568
7*	1926	501675	21	11363	293
8*	8473	38010	22	19849	483
9*	294600	54	23	8412	34
10	657	7063	24	248	3
11	315	4985	25	66730	723
12	5500	250	26	264900	700
13	10396	34	27	18006	1478
14	5382	274	28	15074	663

All of the observations flagged by the H&B edit have been exposed to a relatively large change. With the currently used method, the first 9 of these observations, marked with a `*', as well as 29 others were checked and changed, see Table 1.

3. Effects on the other survey variables

Using one of the best combinations, U=0.4 and C=41 as a check of the effect on the other variables, the results of Table 5 were produced.

TABLE 5 : EFFECTS ON THE OTHER VARIABLES, U= 0.4 AND C=41

VARIABLE	DOMESTIC DELIVERY	EXPORT DELIVERY	DOMESTIC ORDER	EXPORT ORDER	DOMESTIC STOCK	EXPORT STOCK
# OUTLIERS	38	19	109	72	31	30
# OUTLIERS FOUND BY H&B METHOD	9	4	6	6	4	4
# OBS FLAGGED Y H&B METHOD	28	25	30	35	12	10
SUM ALL CHANGES	12 997 728	1 157 332	9 080 144	825 626	4 316 568	6 096 551
SUM CHANGES FOUND BY H&B METHOD	12 958 562	1 029 825	8 701 388	485 480	4 209 380	5 856 540
% OF ALL CHANGES	99.70	88.98	95.82	58.80	97.52	96.06

Looking at the percentages of changes represented by the outliers that were found, the results are with one exception in agreement with the current method. Obviously, there are a few observations in each population of outliers that are responsible for the largest changes.

V. CONCLUDING COMMENTS

The purpose of this study was to adapt the Hidiroglou-Berthelot edit to a Swedish material. From a theoretical point of view, this method is difficult to understand and rather complicated. This is a disadvantage above all when the method has to be adapted to local conditions or perhaps developed. However, the Hidiroglou-Berthelot edit succeeds in its objective of finding observations that have been submitted to a relatively large change from one period to a succeeding one. Some points that deserve mentioning are :

1) The edit has to be performed only once to identify all suspect observations.
2) The material itself supports the information needed.
3) There are two possibilities, parameters U and C, to control the number of observations to be flagged.
4) Little time and efforts are spent on checking observations of minor importance.

Points 2 and 3 show the periodical adaption and flexibility that exist within the limits of the edit.

MACRO-EDITING - THE AGGREGATE METHOD

by Leopold Granquist
Statistics Sweden

Abstract: The purpose of the paper is to describe and present some results of a macro-editing method which has been tested on data of the Swedish Monthly Survey on Employment and Wages (MSEW) in mining, quarrying and manufacturing. It is shown that the aggregate macro-editing method in this case considerably reduces manual work without decreasing quality or influencing timeliness.

I. INTRODUCTION

The study was carried out in the form of simulation of a macro-editing method with prototype of a complete editing process for the MSEW. Thus the prototype may be considered as an alternative editing procedure to the regular one. The main point of this prototype simulation method is to show that macro-editing methods can furnish the main part of the editing in the production of a survey. Usually, macro-editing is applied at the final stage of the whole survey operation to assure that there is no single error which may have violated an estimate. It is shown that the aggregate macro-editing method in this case reduces the amount of manual work to a considerable extent, without losses in quality or timeliness.

II. MSEW SURVEY AND ITS TRADITIONAL EDITING

The MSEW is a sample survey on employment, absences, discharges and wages in mining, quarrying and manufacturing. Data are collected every month from about 3000 establishments belonging to about 2500 enterprises. They report the number of workers on a normal day during the end of the month, the number of working hours for a chosen period, the payroll for those working hours, the number of working days in the month and data about absences, newly recruited employees, discharges and overtime.

The Swedish standard industrial classification of all economic activities (SNI) is applied. It is identical with the 1968 ISIC up to and including the four digit-level and has in addition two-digit national subclassification.

In MSEW processing a traditional micro-editing procedure is used, which means that a computer programme is run in batch with checks prepared in advance pointing out data as "suspicious" or not consistent with other data. The reconciliation is completely manual, but as data entry is carried out interactively, changed data are shown directly on the screen and checked against the editing programme.

This computer assisted editing is preceded by a manual pre-edit process, aimed at finding out if the questionnaire is possible to run and to facilitate the data entry. However, this pre-editing work is much more comprehensive than necessary.

The data entry is carried out in another department, without any editing at all. The idea of integrating the pre-edit and the data entry operations has not yet been accepted.

The questionnaires are processed in lots on arrival at the office. The last questionnaires are processed interactively. There are four or more editing runs in batch every month. The records which have been up-dated are processed together with new questionnaires from later runs.

Within the production system there is also a macro-editing process. However, this process, the cell-list editing , is only applied when the time schedule permits it and as a final editing of the micro-edited data.

In this cell-list editing, the estimate, ratio and difference to the estimate of the preceding month for every domain of study are printed out on a so called cell-list, one for every main variable. The lists are then scrutinized manually. If there is a "suspicious" estimate, a search is made for "deviating" records to find out whether there are errors in those records.

III. REVISION OF THE EDITING PROCESS

In our study, we have made revisions of both micro- and macro-editing process within the original systems.

1. The micro-editing

Concerning the micro-editing, we have reduced the number of flagged data by about 50%, without any drop in quality.

For this revision we used our interactive editing programme GRUS. We studied every check separately and how it interacted with any other check in the system.

We found that some checks were unnecessary in the sense that they did not discover any errors. It was noted that they were not redundant in the "Fellegi-Holt" meaning.

Another finding was that almost every check had too narrow bounds. They had been set according to the safety first principle. In this case only deviation of up to ±10 % from the values of the previous month were accepted.

By studying the impact of every single check and all combinations of single checks on data from an MSEW survey we constructed a new, rather well-tuned system. The main difference to the old system was much wider acceptance limits in the checks. In fact they could have had still wider limits, but at the time, the subject matter specialists were not willing to accept any greater changes.

The immediate result of using the new system was a slow rise in quality, as measured by the fact that more errors were found, due to twice the time to reconcile every flagged data. We can now state that the time for manual reconciliation work has been reduced approximately 50%. With regards to quality, we can only state that it is at the same level as before.

2. The aggregate method

The idea behind the aggregate method was very simple, that is, to use an error-detecting system (in our case EDIT-78) twice. Checks are run first on aggregates and then on the records of the file with the suspicious (flagged) aggregates.

One important feature of the procedure is that the acceptance limits are based on the distribution of the check functions of the data to be edited. This is done by manual analysis of printouts of the tails of the check variable distribution.

These printouts can also be considered as an alternative realization of our editing method because identifiers are printed out as well. The reason why we use the EDIT-78 programme is that it contains an error message procedure which prints out in the same message every found inconsistency and suspicious data of the same record, which facilitates the manual reconciliation procedure. Above all, we use EDIT-78 because it contains an up-dating procedure.

3. The first experiment

The procedure was applied to the whole file of the edited and published data of the MSEW for a selected month.

The methods were applied to the following variables: Number of working hours, pay-roll, wages per hour, number of workers. The checks for each variable consisted of a combined ratio and difference check against the values of the preceding month, i.e. both the ratio and the differences had to be rejected to indicate the present value of the variable as suspicious.

This experiment aimed at finding out how methods and computer programmes for testing checking methods and procedures should be constructed . Besides, we also happened to detect two serious errors which had not been found in the processing of the data of that month. This was an indication that micro-editing may not even always detect serious errors.

IV. EVALUATIONS BY SIMULATION STUDIES

1. Simulation on processed data

The simulation study was carried out on unedited data and the evaluation was done against the edited data for the selected month. Thus this evaluation cannot indicate any improvement of the quality. We can only find out whether the new procedure will give equal or lower quality and not differences in the manual reconciliation work.

The editing process of the MSEW surveys cannot consist of this macro-editing method only. There is a need for an additional cleaning of the data for the following reasons:

- to fight those errors which cannot be detected by the macro-editing method.
- to make it possible to carry out the macro-editing.

In our simulation study, this additional cleaning operation consisted of a manual reconciliation of errors found by an error detection programme dedicated to find certain types of errors in certain variables. Missing values and errors in the variable "number of working days" were handled by this special programme, which proceeded the macro-editing procedure.

In our study, the number of totally flagged records from this cleaning and preparatory programme was 161. The number of records in the survey was 2951. The flagged values were reconciled against the edited data file.

In order to test the macro-editing procedure under realistic conditions, the records of the data file were divided into lots exactly as when that MSEW survey was processed. However, the records from the last runs were put together into one editing round (instead of three).

All data were inflated by the inflation factors used in the normal estimating procedure. The records were then sorted and aggregated into four-digit SNI-groups. There are 89 such groups in the MSEW.

For every editing round, the procedure indicated in III.2 was executed. The reconciliation consisted in checking with the edited data. Flagged values from this macro-editing procedure were considered as errors found and revised if the micro-editing had changed the original value. By matching the macro-edited file with the micro-edited file we found out those errors in the production process which had not been detected by our procedure.

The simulation study resulted in 274 flagged data out of the 2951 records. 76 errors were found. (The number of error messages was 190 out of which 57 were corrected). These results should be compared with 435 flagged data, out of which 205 errors were found in the micro-editing process when the MSEW of that month was processed. Thus we got a reduction by 161 flagged values (34 %).

But, did this simulated macro-editing detect the most serious errors? This question may be answered by Table 1, in which each variable on the four-digit level shows the number of aggregates (the columns) by the absolute difference in % in the estimates (the rows) due to the errors not found by the aggregate method.

Table 1

Difference (%)	Workers	Working hours	Pay-roll	Wages/hour
0.05	4	1	6	6
0.1 - 0.4	4	5	5	8
0.5 - 0.9	1	2	4	2
2.5	1	0	0	0

2. Simulation during current processing

This simulation study was carried out almost exactly as the one described above. The only difference was that this one was run parallel with the normal processing of the survey. Thus we eliminated any possibility of being influenced by knowing the results of the micro-editing process when we set the boundaries for the checks of our macro-editing procedure.

However, the same evaluation method had to be applied. Of course, we should have preferred to evaluate the macro-editing method by processing the data entirely concurrently with the normal processing. This might be done in the future if the subject-matter specialists are interested in the procedure.

The cleaning operation resulted in 260 flagged records out of the 2996 totally processed records of that month.

The macro-editing procedure flagged 225 data out of which 50 errors were found. (The number of error messages was 139 out of which 45 records were revised). The micro-editing of the same variables when these data were processed flagged 389 data out of which 110 were revised. Thus the reduction of flagged data compared to the usual process was 164 or 42%.

However, the macro-editing procedure seemed to be less successful than in the previous simulation in detecting the most serious errors found in the normal processing. The results are given in Table 2.

Table 2

Number of aggregates by the absolute difference in % in the estimates due to the errors not found by the aggregate method.

Difference (%)	Workers	Working hours	Pay-roll	Wages/hour
0.05	4	2	12	8
0.1 - 0.4	3	6	4	8
0.5 - 0.9	1	1	5	3
1.0 - 1.9	1	1	0	1
2.0 - 2.9	0	0	0	0
3.0 - 3.9	1	1	0	0
4.0 - 4.9	0	0	1	0

This simulation study was carefully analyzed. First, we investigated those errors with an impact on the aggregates of more than 2% which were discovered by the usual editing but not by our simulated process. There were two errors which were accepted by the ratio checks but not by the difference checks and which were at the border of the acceptance regions of the ratio checks. A slight change upwards of the lower boundary of the ratio check should have caused these errors to be discovered by our macro-editing.

Then the impact of the boundaries of the checks on the efficiency of the error detection was analyzed. The finding was that the acceptance intervals of the checks should be wider. The strategy had been to set the limits as close to the main body of the data as possible. A better strategy seemed to be the setting of the limits as close as possible to the first outlier.

The findings of the analysis can be summed up as follows. If the simulation had been carried out with a considerably higher upper bound of the acceptance intervals of the ratio and the difference checks, then we should have got the following results:

The number of flagged data of the macro-editing checks had been reduced to 134, which means a 66 % reduction of flagged values compared with the corresponding micro-editing of these data.

The quality had not been affected as shown in Table 3. The figures within parentheses are the corresponding figures from Table 2.

Table 3

Number of aggregates by the absolute difference in % in the estimate due to the errors not found by the aggregate method.

Difference (%)	Workers	Working hours	Pay-roll	Wages/hour
0.05	4(4)	3(2)	13(12)	10(8)
0.1 - 0.4	3(3)	6(6)	4(4)	8(8)
0.5 - 0.9	1(1)	1(1)	6(5)	3(3)
1.0 - 1.9	1(1)	2(1)	0(0)	1(1)
2.0 - 2.9	0(0)	0(0)	0(0)	0(0)
3.0 - 3.9	0(1)	0(1)	0(0)	0(0)
4.0 - 4.9	0(0)	0(0)	0(1)	0(0)

V. CONCLUSIONS

On the basis of the two simulations we can state that the macro-editing procedure will lead to a substantial reduction in reconciliation work.

There is no notable loss in quality. It is true that all errors detected by the micro-editing process are not detected by the macro-editing procedure. However, these errors are small and many of them are not found by the ratio checks of the micro-editing system either. They are found by scrutinizing flagged values of other items and checks.

It is quite clear that it is important to learn how to tune the macro-editing procedure and to fit it properly to the micro-editing procedure to get an efficient and well coordinated editing system. This can only be done by gaining experience through using the system in the processing of the MSEW. We have learned that the acceptance intervals of the ratio checks should not necessarily be symmetrical and that they should be rather wide. (The "safety first" principle when defining acceptance regions does not apply to macro-editing methods either.)

There are no timeliness problems with this kind of macro-editing method. It can be applied during the processing of the data under the same conditions as computer assisted micro-editing methods.

The macro-editing concept is a realistic alternative or complement to micro-editing methods, which reduces the amount of manual reconciliation work to a considerable extent, without losses in quality or timeliness.

MACRO-EDITING - THE TOP-DOWN METHOD

by Leopold Granquist
Statistics Sweden

Abstract: The paper gives a brief description of the Delivery and Orderbook Situation survey of the Swedish manufacturing industry and of its editing and imputation. It points out the main features of the applied macro-editing procedure and tries to explain why this procedure is superior to the micro-editing procedure.

I. INTRODUCTION

The purpose of this paper is to give an example of a macro-editing procedure for error detection in individual observations. It focuses on its advantages over the traditional micro-editing procedure. Both procedures are applicable within the production of the monthly survey on the Delivery and Orderbook Situation of the Swedish manufacturing industry (DOS). The paper argues that the two different methods of detecting negligence errors are based on the same assumptions and that the macro-editing procedure (MAEP) is superior to the micro-editing procedure (MIEP).

II. THE DELIVERY AND ORDERBOOK SITUATION SURVEY

The DOS is a monthly sample survey of enterprises. There are about 2000 reporting units, kind of activity units, sampled once a year from the Swedish register of enterprises and establishments. It estimates changes in deliveries and the orderbook situation (changes and stocks) every month for both the domestic and the foreign market, for the total Swedish manufacturing industry and the 38 branches (classified according to the Swedish standard industrial classification of all activities).

The design of the questionnaire is computer printed. The reported values for the two previous months are preprinted for the six variables. The questionnaire thus contains three columns. If the respondent cannot give data for the calendar month, he has to indicate data about the period to which the reported data refer.

The entry of the questionnaire data is carried out in three batches every production cycle by professional data-entry staff. The last questionnaires are entered directly interactively and then checked by the editing program, which contains checks of formal errors on the micro level.

The whole production process has thoroughly been revised recently. The new system replaced the old one (from 1970) in 1986.

Within the new system, there are two editing procedures: the automatic adjusting procedure and the automatic imputation procedure. Since correction processes are identical irrespective of the choice of the editing procedure, they will be described first.

III. THE AUTOMATIC CORRECTION PROCEDURES

Data are adjusted automatically if the report period differs from the present calendar month. This is performed before editing and thus adjustments will be checked. In the old system, these adjustments were carried out manually.

Imputations are only made for non-response (unit as well as partial non-response). The imputation procedure utilizes data reported by the unit for the six previous months, which are adjusted by means and trends from the branch to which the unit belongs. Imputations form part of the estimation procedure and are neither reported back to the respondents nor used in imputations for succeeding months. They are carried out after editing, and thus they are not checked.

IV. THE MICRO-EDITING PROCEDURE

The MIEP was supposed to be the main editing procedure. However, it was only used for the first few months and will therefore be described very briefly here.

The MIEP is a traditional record-checking procedure. Such procedures are used in practically all the surveys carried out by Statistics Sweden. Data classified as erroneous or suspect according to specified checks are printed on error lists or displayed on the screen. Error messages are then scrutinized manually. Very often they involve telephone calls to the respondents. Corrections are entered interactively and then again checked by the editing rules. However, the number of corrections (detected errors) is relatively small.

The system has an "Okay"-function (key) which allows the approval of original data or changes found to be correct, even though they had not passed all checks.

Validity checks, consistency checks and ratio checks are used (against the previous month). The acceptance regions are intervals, which can be up-dated any time during the editing process.

When the new production system was run for the first time, the MIEP was not very successful. It produced so many error messages that the subject-matter specialists realized that they had neither resources nor energy to handle all error messages, especially since they knew from the experience gained from the old system that only a small percentage of the flagged data indicated detectable errors. They had built up the checks on the basis of all the experience and subject-matter knowledge they had gained during years of work with the survey. The procedure was flexible, user-friendly and easy to fit to the data to be edited. In spite of all that, the procedure did not work. Of course this was a great disappointment.

A basic problem of all traditional record-by-record editing is to identify the observations which contain the most important errors. The MIEP, as all automatic editing programmes built

on the same or similar editing principles, does not provide any possibility to assess the importance of an error when the error messages are handled. Every flagged observation has the same weight and claims about the same amount of resources irrespective of the importance of the suspected error. Many errors have a negligible impact on the estimates, because their magnitude is small or they cancel out. But when handling the error messages, the importance of a specific item is not known. There is a need for thumb rules to provide priorities. However, it is very difficult, maybe even impossible, to construct practical priority rules. The only way to solve problems in such a procedure is:

- to construct checks more sensitive to observations of high relative importance than to observations of low relative importance, and
- to fit the whole system of checks very carefully to the data to be edited.

But even in cases where efforts have been spent to focus the editing on potentially important errors, there are clear indications that too many resources are spent on editing in comparison with the outcome. Indeed, the editing process has been considerably improved, but this is far from sufficient.

V. THE MACRO-EDITING PROCEDURE

The basic idea behind the MAEP is to study the observations which have the greatest impact on estimates.

The MAEP is an interactive menu programme, written in APL. The in-data file is successively built up by the records from the three batches from the data-entry stage and the last received records, which are entered interactively. There are three functions to select the records to be studied, namely:

- the highest positive changes,
- the highest negative changes,
- the greatest contributions.

These functions can be applied to the total (the whole manufacturing industry) and to every one of the 38 branches. For a selected function and domain of study, the screen shows the 15 records of the in-data file which have the highest value (weighted) of the selected variable, sorted top-down. The fifteen rows of records have the following columns:

- Identity,
- observation,
- expansion factor (weight),
- inflated value (="observation" x "weight"), and
- sum.

Having all the "top 15" records on the screen, the operator can select any of the records to be shown with its entire content and then examine it to see whether an error is committed. If an error is identified, he can update the record directly on the screen and immediately see the effects.

The record usually loses its position on the "top 15" list, another record enters the list and the sum value is changed.

Scrutinizing of the "top 15" list goes on until the operator can see that further editing will only have negligible effects. All records can theoretically be inspected manually in this manner. In practice, only a few records in each of the three batches entered are entirely scrutinized.

VI. PROS AND CONS OF THE TWO EDITING PROCEDURES

It should be noted that both procedures (MIEP and MAEP) focus on the same type of errors, namely random negligence errors, and use the same principle of detection namely by pointing out "outliers" (extreme observations). The difference is that the MIEP flags suspicious observations by criteria fixed in advance based on historical data, whilst the MAEP concentrates on the observations which, at that very moment, and relative to the estimates, are the most extreme ones. The MIEP always flags a certain percentage of the correct observations, which then also have to be inspected. All flagged observations have to be scrutinized regardless of their impact on the estimates. With the MAEP, it is possible to stop the investigations as soon as the effects on estimates appear to be negligible.

If the selection criteria are equally good, the MAEP offers the advantage of just flagging observations with important errors.

The staff are very satisfied with the new editing methods. They are continuously being trained on the subject-matter and on the production and presentation problems of the survey. The user, with his growing knowledge and skill, focuses his work on the most important errors in the data of the studied period. In traditional editing programs, the user is dominated by these errors and distributions of earlier periods. With the MAEP, the user needs not handle a great number of unnecessarily flagged data, and he has the satisfaction of immediately seeing the impact of his work.

DATA EDITING IN A MIXED DMBS ENVIRONMENT

by N.W.P. Cox and D.A. Croot
Statistics Canada

Abstract: The article presents two data editing facilities: GEIS for edit and imputation functions and DC2 for data collection and capture functions. GEIS and DC2 are components of the project on Generalized Survey Function Development (GSFD) which was developed by Statistics Canada. Adaptability and functionality of these tools in a mainframe and microcomputer database management environment is a particularly highlighted issue.

I. INTRODUCTION

The 1960s saw the beginning of significant computerization of statistical surveys. The period was characterized by computer systems for surveys being built independently of one another. Systems analysis was done in isolation for each survey and there was no use of structured methodologies. A system was built for a given target computer and often consisted of large programs that read and wrote master files. System design was driven by design of these complex master files. Modularity, when introduced as a system concept, was implemented through definition of subroutines within the master program.

The introduction of statistical subject-matter specialists (methodologists) put emphasis on the development of theories, then statistically sound methods, for all phases of surveys. Specifications to systems would normally cover discrete stages of the survey, hence promoting modularity of survey functions. This led to specialization of mainline programs to some extent. Communication was still carried out through a series of master files and reference files which, in a production environment, were usually tape files.

A statistical organization typically had distinct computer environments for major survey processing areas and systems were built for a specific hardware and software configuration. As separate programs were introduced for specialized survey functions, different computer hardware environments might be targeted for different programs,although any one program would only run in its design environment. The resulting mixed computer environments were plagued by data file conversion routines as one moved data from one machine to another.

This period (60's & 70's) saw computer developers concentrate primarily on function, with data management being almost an afterthought, relegated to the realm of keeping an inventory of master files. By the 80's, one saw more and more value placed on data as a valuable resource. The issue became one of linking all the data stored in separate master files in order to maximize information retrieval.

Progressively, data base technology has provided powerful data management facilities isolating data from programs and permitting data sets to be easily and dynamically linked to one

another. The informatics area at Statistics Canada has now deemed data base technology to be sufficiently mature that its role integrating data both within and between survey processing functions can be actively encouraged. A notable and recent example at Statistics Canada of the integrating role subsumed by data base technology is the General Survey Function Development Project.

II. FORMS OF EDITING

Editing has always been a significant part of the survey taking process in national statistical offices. This has been universally perceived as an important step that improves the quality of published information. From the advent of cheaper and more plentiful computing power in the early sixties, editing has tended to rely progressively less on the instantaneous application of personal knowledge in the editing process.

The explosion of editing opportunities afforded by technological advance encourages examination of the fundamentals of editing:

- What is editing? And what principle should drive its application?

- When should editing be done to most enhance reliability without disturbing operational work flows?

- How do we use emerging technologies in an appropriate and cost effective manner for the benefit of both respondents and the statistical office?

The editing process usually comprises several different activities forming a general sequence of basic operations:

- An anomalous condition is detected, which is caused by error, misunderstanding or misinterpretation.

- A diagnosis is made to identify a data item or items that is the most likely cause of the anomaly.

- An action is performed to initiate modification to the identified data item or items.

- The record, or response, is re-edited to confirm acceptability, or the original condition indicated as acceptable.

There is considerable scope for discretion by the survey designer determining the nature of edits to be done and the stage in the survey process at which it is appropriate to do them. However, it is axiomatic that the earlier in the process an error can be detected the less the cost of its correction is likely to be.

1. Data Entry Editing

Two significant characteristics of data editing at the time of data entry are, first, that we are essentially limited to those edits that can be performed on a single record. Second, depending on the circumstances of data entry, we may well be in direct contact with the respondent, in which case many errors can be referred immediately to the respondent for revised input. The presence or absence of the respondent affects the design of the data correction process, but does not change the situation for the detection of errors, which for data entry usually is confined to intra record edits.

The types of edit appropriate for data entry time are considered to be as follows:

i) The validation of fields, for example, the numeric fields containing numeric values, date fields containing dates.

- Type: integer value for an integer variable.

- Range: upper and lower bounds may be expressed for numeric variables.

- Format: some types of data are subject to a predefinable pattern; examples of this include, postal codes, telephone numbers, addresses, social insurance numbers etc..

- Identifier validation. It is usually particularly important to establish quickly the identity of a respondent for a survey, as a result almost all surveys assign a unique identifier to each respondent. It is preferable to do this in as economic and unambiguous form as possible (eg. in a machine readable form such as a bar code) but in any event it is almost always beneficial to apply certain checks, for example range checking, checking against a valid list of all identifiers, check-digit verification etc..

ii) Fitting a data model

A set of edit rules is a model of reality that we can use to structure a survey response set and suppress irrelevant detail. At this stage, we are likely to be dealing also with edits that are inter-field. This model may prevent logical impossibilities, for example someone starting his current employment before his date of birth, or merely detect implausible situations, such as a person 21 years of age earning more than a $1 million a year.

Whilst the former category can often impose rules emanating from outside the survey, for example, the laws of nature, or of accounting practice; the latter can be quite controversial. Although we may be almost certain that a 21 year old person will not be earning more than $1 million a year, and if the mode of capture is such that access to the respondent is possible, there is a natural desire to correct the "error" at this early stage.

2. Statistical Macro Editing

Since this is the subject of another paper to the JGDE, it will not be dealt with at length here. Statistical or macro editing, viewed according to the 4 stages of editing described in the heading of the section, differs from record level editing first in the detection of the anomalous condition. Here, it is based upon analysis of an aggregate rather than an individual record. The diagnosis following may be based upon the identification of the particular record, or records, causing the anomaly. The modification action done may apply to individual records, or may be applied to the aggregate level. The final step of re-editing to confirm acceptability may again apply to individual records if the modification action was so applied. If modification and re-editing is done only at the aggregate level, the subsequent utility of the micro data base is subject to obvious limits.

Most examples of macro editing to date could be characterized as macro error detection in order to localize the diagnostic action, with, ultimately, correction at the record level (micro correction).

3. Automatic Edit & Imputation

With data entry editing, the possibility always exists to contact the respondent to resolve an edit anomaly: by telephone or using a follow-up form. The threshold between data entry editing and automatic edit and imputation has been defined at Statistics Canada to be that point in the survey process when there will be no further contact with the respondent.

Automatic edit and imputation has been slow to play an important role in survey editing. It has been surmised that a mistrust of automation has been a fact leading to over-editing; hence an inordinately large proportion of response records are rejected, and subjected to a costly, time-consuming and potentially subjective review. The outcome of this review often is that most of the records are not in error. A 1985 Statistics Canada survey of users did find a general willingness of subject matter divisions to adopt more sophisticated approaches to imputation, if the software were available.

For an automatic edit and imputation system to become a reality, one first had to have a generalized methodology addressing the question. Since such a methodology would be imbued with the use of complex algorithms and extensive data search strategies, it was also necessary to await developments in computer systems technology. A landmark methodological development was the 1976 paper by Fellegi and Holt, (Fellegi and Holt, 1976). This paper espoused a methodology adapted to making minimum change to a data record. All efforts at Statistics Canada to develop generalized edit and imputation systems have incorporated this underlying philosophy of the Fellegi and Holt paper.

The core methodology since 1976 is:

- to specify a set of edits that defines a region[5] in a multi-dimensional space within which all valid records would fall;

---

[5] This is strictly true for edits involving continuous quantitative variables, but with qualitative variables 'region' has to be interpreted as an acceptance set of discrete values.

- the set of edits is applied to each data record and if the record does not fall within the defined region, it becomes a candidate[6] to have one or more fields imputed;
- an error localization algorithm is applied to determine the fields that are to be imputed;
- a choice from several possible techniques is made to supply new values for the fields to be imputed.

A major technique for supplying imputed field values is donor imputation. In this technique, one takes data values from a donor record chosen to be as similar to the recipient record as possible. A consequent implementation issue is the choice of an algorithm for searching for a suitable donor. A simple sequential search may function well for small data sets, but with the more usual application of automatic edit and imputation methods to large data sets, a more efficient search strategy is required.

For qualitative variables, "sequential hot-decking" can be used. In this approach, the recipients and donors are merged. The file is sorted randomly, or randomly within each imputation class. The records are processed one at a time. When a recipient is encountered, the selected donor is the most recent donor that has exact matching values. This can require considerable storage, depending on the number of matching variables and the number of valid values for each one.

For quantitative data, one searches for the closest donor in terms of distance using the structure of a "k-d tree."[3] The tree is built with the set of donor records. Each recipient record traverses the tree. The use of a tree data structure greatly reduces the number of donor records to be considered.

An imputation methodology has to ensure that imputed values satisfy the edits. One possible method (and that employed in the Generalized Edit and Imputation System (GEIS)) states that instead of selecting the "most similar" donor record, one chooses several "most similar" records. If the first record is not capable of donating values that pass the edits, then one considers the second, and so on.

A final issue to be addressed in any automatic edit and imputation methodology is how to handle recipient records for whom imputed value(s) cannot be found. One approach is to have these records reviewed manually.

The algorithms and information processing techniques necessary to:

- analyze the set of edits to verify that a valid region in a multidimensional space is defined;
- determine the fields to be imputed;

- build a donor search tree;

---

[6] Such a record is referred to as a recipient record in the imputation literature.

[3] The k-d tree is a generalization of the simple binary tree used for sorting and searching. The k-d tree structure provides an efficient mechanism for examining only those records closest to the query record, thereby greatly reducing the computation required to find the best matches [Friedman et al (1977)].

- retrieve an acceptable donor;

are demanding in terms of machine resources required and required the advent of powerful processors, large memory, and comprehensive Relational Data Base Management Systems (RDBMS) before practicable cost beneficial production systems could be built.

III. GENERALIZED SURVEY FUNCTION DEVELOPMENT PROJECT

1. Concepts

The General Survey Function Development Project (GSFD) was initiated in 1986 in response to the following developments:

- A decision to redesign the whole complex of business surveys was taken in order to improve quality through better integration of business statistics and to achieve significant savings in the process;

- The many technical developments that have characterized the 80's opened up possibilities for developing general survey processing tools in a comprehensive and integrated fashion;

The first step was to form a team of methodologists and computer specialists, encouraging advances at the same time on both the methodological and system development fronts.

Although initially oriented to business surveys, GSFD products are being developed so that they can be adapted or extended for application to social surveys. The major goals are to provide high leverage to the process of survey design; to reduce the cost of survey and systems design; to reduce the lead time to implement a survey; and to help standardization of survey methodology.

Note that the latter goal leads to an interesting discussion over the benefits of standardization and flexibility. Although it is desirable to use the general survey software as a motivating tool for standardization, this could have the side effect of causing the need for certain survey specific modifications to be forced outside the GSFD framework. It is therefore important to develop the system such that the required flexibility is available to the designer without jeopardizing the promotion of standards.

Specific goals are:

- Increase the standardization of survey development and operation, without unduly limiting the scope of applicable methods;

- Facilitate experimentation with methods during the survey design stage;

- Encourage repeated use of successful methods and systems wherever applicable;

- Provide a user-friendly standard interface for each function and for each method of collection;

- Integrate survey functions into a single control environment based on a single logical data base;

- Provide performance and quality measures across survey functions;

- Provide portability across several computer architectures in order to foster the accessibility of all functions from a variety of Statistics Canada clients.

2. Industry Standards

The design objectives of GSFD are such that, as a minimum, GSFD products must be available (or easily made available) on the following industry standard operating systems[4]: MVS/XA, UNIX[5], DOS-OS/2. Furthermore, its underlying software must conform to International and Government of Canada standards: for example, ISO standard SQL/RDBMS for data management, and C-language for 3GL purposes.

3. Components

The GSFD System includes eight major parts (see Figure 1):

- the Frame, which defines the universe and from which the sample is drawn;

- the Relational Data Base, which serves all operational functions and thereby forms the central control mechanism of the design;

- the Design and Specification function, which a survey statistician uses to design the survey;

- the Sampling function;

- the Collection and Capture function;

- the Estimation function;

- the Retrieval and Dissemination function;

- the Utilities function.

---

[4] An exception exists with the Collection and Capture function. In its first implementation this function will operate only in a UNIX environment since UNIX has been chosen as the operating system to support collection and capture operations both in the Statistics Canada's Regional Offices and at Headquarters.

[5] UNIX is a propriety operating system of AT&T Corporation, but in the context of this paper it represents any UNIX type of operating system.

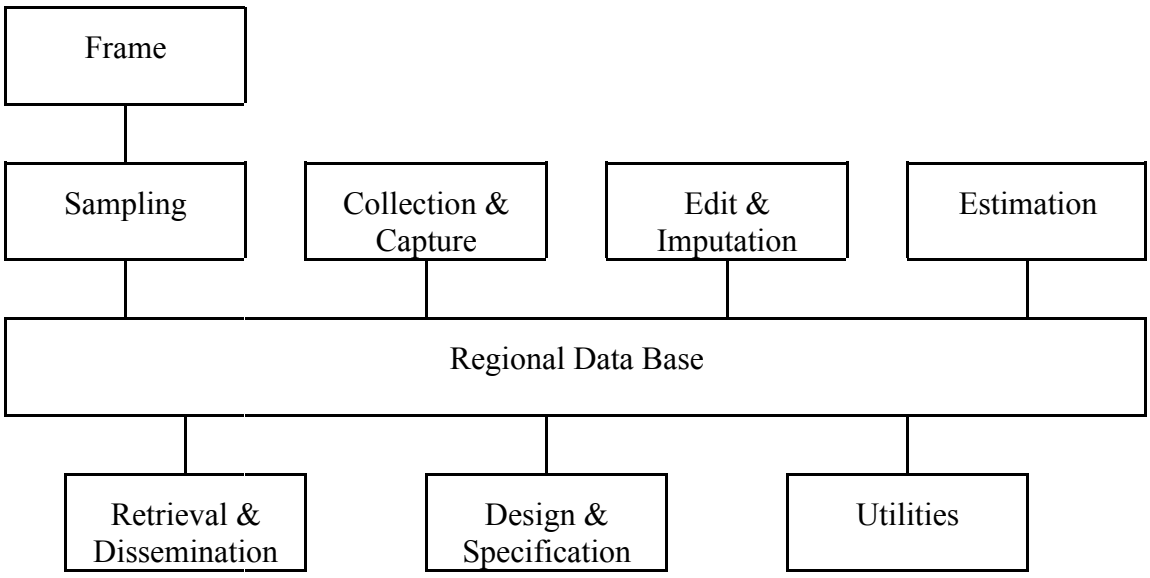


Figure 1 Major components of the GSFD

The power and simplicity of the Relational Data Base Management System (RDBMS) that manages the data in the Relational Data Base are key to the GSFD system and it is probable that this ambitious initiative would not have been attempted if this generation of data management software had not been available. The commercial RDBMS product ORACLE was selected in 1987 for the project, partly because of its ability to operate in a wide variety of computing environments.

Survey data editing facilities are provided in two GSFD functions: the Edit and Imputation function that is known by the acronym GEIS; and the Data Collection and Capture function (known as DC2).

GEIS and DC2 are described further in the following two sections.

IV. DC2 FUNCTION

1. Hardware/Software Environment

DC2 is the GSFD component that handles questionnaire preparation and provides corresponding data collection/editing facilities. The decisions made about the practical realization of the DC2 architecture are closely in accordance with the tenets of the GSFD project.

The DC2 software is being developed to run on UNIX[6] platforms in a windowing terminal environment. The Unix platforms will be configured to provide a client/server network running

[6] The goal is to stay within the mainstream of UNIX evolution to avoid the problems associated with becoming too closely tied with one or a small number of vendors.

on an Ethernet LAN. The major part of the system will be written in the programming language C. This language's main feature of interest, other than its portability and availability, is the balance between low and high level features that allow systems to be efficient and yet portable. Since ORACLE has been chosen to be the relational database management system that provides SQL support at Statistics Canada, it will be used within DC2 although its use will be constrained in the following ways:

- standard SQL will be used as the Data Definition and Data Manipulation Language wherever possible;

- ORACLE-specific features will be used only when they provide significant value, and even then these usages will be isolated.

PROLOG, which has the advantage of being a well known and relatively standard AI language, has been chosen as the means of specifying procedural edits to DC2.

2. Editing Role

The philosophy employed within the GSFD project is reflected in the DC2 product. DC2 automates all editing processes now employed in the data collection and capture phase. All manual scanning of documents done before capture will be eliminated, through edits and routing instructions defined for the application and applied by the DC2 system. Traditional field-level edits (type, range, format, etc.) is supported, also comparison of historical data to current data (a form of plausibility verification). Plausibility and accounting edits involving more than one data item will be applied as soon as all the necessary values are available.

Release 2 introduces the functionality of CATI to the system. In particular, this entails the complexities of navigation through a script that is not only acquiring data, but determining the actual structure of the instance. Although still restricted to intra-record edits, the interacting requirements of branching/routing and editing, together with the undesirability of repetitious questioning, make this a significant challenge.

The software was being developed with expandability in mind and therefore, in the long-term, more sophisticated edits could be introduced. For example, there could be instances where some form of inter-record processing may be of benefit in capture editing. However, the methodological and operational problems first caused by partial availability of data from other records require resolution.

V. GEIS FUNCTION

This section describes the GEIS (Generalized Edit and Imputation System) implementation.

GEIS is a system of generalized programs to edit and impute survey response data. These programs are packaged into six sub-systems (see Figure 2) that are used to support the operations of specifying, analysing, testing, and applying sets of survey specific edit and imputation criteria. Because the methodology underlying GEIS is based on the assumptions of linearity of edits and non-negativity of data, the implementation can rely heavily on linear programming techniques.

GEIS has been developed in a UNIX environment and is available to users in two operating system environments: MVS on the IBM mainframe, and DOS on an IBM-AT compatible microcomputer. The system is comprised of programs written in the C programming language, a menu system using the ORACLE SQL*FORMS product, and reports written using the ORACLE SQL*REPORT PRODUCT. The C programs interface to the ORACLE database using the ORACLE PRO*C product. All the facilities of GEIS are accessible through the menu system that is implemented as a hierarchical arrangement of forms. One can use this menu system to develop and test one or more sets of edit and imputation requirements for one or more questionnaires.

Using the GEIS menu system is very much the same on DOS and MVS but obtaining access to the menu system, or running GEIS programs directly, is operating system dependent. Under control of DOS all programs are executed online with program output being displayed on the screen. Under control of MVS most programs are executed in batch.

1. GEIS COMPONENTS

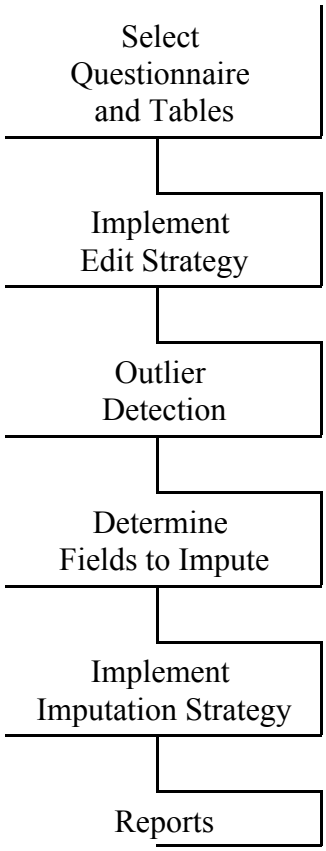


Figure 2 The six principle components of the GEIS.

An overview of the main components of GEIS is presented below.

(a) Select Questionnaire and Tables.

The first step in GEIS is to identify the survey questionnaire and specify the Oracle tables that contain survey data for that questionnaire. For further information on the use of ORACLE tables in GEIS, see under heading **Data Model** later in this Section.

(b) Implement Edit Strategy

A survey specific set of linear edits form the cornerstone of GEIS and since all GEIS components depend on them, the quality of the imputed data will only be as good as the quality of these edits. This component provides options to specify edits and to evaluate their quality.

Edits are specified interactively. The user identifies the edit, provides the linear equation, including variable names, coefficients and constant, together with an indication of whether the edit is interpreted as pass or fail. There is also a facility to update the edits, associate comments, and maintain date-added and date-updated values. During the edit specification process, syntax verification of the edits takes place. The verified edits are stored on the database.

An analysis option of this component provides information that aids the user establish a consistent (i.e., not self-contradictory) set of edits. Once a consistent set of edits has been established redundant edits[7] can be identified. In addition, the lowest and highest values possible for each variable are found given the set of edits as constraints. This analysis option employs linear programming algorithms that require the assumptions of linearity of the edits and non-negativity of the data. The algorithms used are: the Revised Simplex Method, Product Form Inverse and Given Transformation.

This component generates two reports which help the user to prepare the edits. The first lists the extremal points that are generated from the polyhedron defined by a consistent set of linear edits. These points can provide information about the validity of the edit set, since each point can be interpreted as the worst that a record can be whilst remaining acceptable to the edits. An examination of the extremal points may cause the user to make certain edits more restrictive while relaxing others. The second report lists the implied edits that are generated by taking linear combinations of members of the edit set. The implied edits uncover conditions that are being imposed on the fields but which were not directly specified. Some of these conditions may be undesirable and require modification of the original set of edits.

The last option of the Implement Edit Strategy component applies an edit set to a survey data table and supplies the user with results about edits and records passed, missed, or failed. These results serve as a diagnostic aid permitting a user to further assess the nature of the edits and data.

Using the information gathered from employing the options of this component, a user can delete or modify edits to eliminate inconsistency, delete or modify redundant edits, or add edits

---

[7] Redundant edits are those which do not further restrict the feasible region of data values in the presence of the other edits.

which further restrict the bounds (upper, lower, or both). By removing the redundant edits, the user creates a minimal set of edits. This set defines the same region as the original set but is more efficient to use since it has a smaller number of edits.

(c) Outlier Detection

This component considers all the data records concurrently, not individually. It determines upper and lower acceptance bounds for each user-specified variable by observation of the current data or, for ratios (the variable's current to previous values), by observation of the current and previous values. This serves two distinct purposes: first, it determines upper and lower bounds that could be specified as quantitative edits; second, it identifies outlying values that can be flagged for imputation or excluded from being donated in the Donor Imputation component.

When information for the Outlier Detection component is specified the user can restrict the records to be used in the outlier detection by specifying data record exclusion statements, in SQL format, on either or both of the current and historical data tables.

(d) Determine Fields to Impute

When a record fails one or more quantitative edits, there might be several combinations of fields that could be imputed so that the record would pass the set of edits. Given the underlying premise of changing as little data as possible, this GEIS component finds all those combinations that will minimize the number of fields to be changed. In the event that multiple solutions are found, all solutions are printed on an Error Localization report, but only one solution is chosen randomly and stored on the database. The user also has the option of minimizing a weighted number of fields to be imputed so that fields of greater reliability have less chance of being changed.

The error localization problem is formulated as a cardinality constrained linear program and is solved using Chernikova's algorithm. For each data record, the program formulation that determines which fields are to be imputed incorporates the edit set and the minimum sum of weights rule. The formulation includes missing and negative data values and accounts for the fields identified for imputation by Outlier Detection.

(e) Implement Imputation Strategy

Having determined which fields of which records require imputation, this component provides the options of deterministic, donor, and estimator imputation to supply valid values.

Deterministic imputation finds out which fields of a record requiring imputation can be imputed to one value considering the edits applied, other data values present in the record, and other fields selected for imputation.

This option reads the ORACLE database table identifying records containing fields to impute. Each data record containing at least one field to impute is fetched and examined by applying the set of edits also retrieved from the database. If a deterministic value is found then it replaces the original field value in the data record and the associated imputation status is

updated to reflect that a deterministic value was supplied. Deterministic imputation usually precedes donor and estimator imputation.

In donor and estimator imputation, the objective is to supply new values to ensure that the resulting data record will pass all the required edits whilst preserving the underlying distributional structure of the data. In other words, the objective is not to reproduce the true micro-data values, but rather to establish internally consistent data records that will yield good aggregate estimates. This objective can be approximated to with donor imputation but usually cannot be met if one uses the estimators.

Donor imputation uses the nearest neighbour approach. The flagged fields to impute are supplied with values from a similar, clean record (donor) where similarity is based on the reported, non-missing values. This procedure operates on a set of variables defined by an edit set and tends to preserve the structure of the data, since all variables in one edit set are imputed at the same time. That is, not only are the values imputed, but so is their interrelationship. To ensure that the edits are satisfied, several nearest neighbours are found, and the closest one that produces a record which satisfies the edits is used to impute for the record, if such a donor exists.

In the first step of donor imputation, the user specifies any fields (must match fields) that are to be mandatory in establishing a match between donor and recipient records. Next the set of all matching fields are determined for all records. For each recipient record, the matching fields are determined by an analysis of the edits with the fields to impute for that recipient whilst including the pre-specified must match fields.

The matching field values are transformed onto the interval (0,1) and the new values stored in a database table. These transformed values are used in the calculation of the distance measure between a recipient and a possible donor record in the nearest neighbour search performed in the next processing step.

In the final step of donor imputation a k-d search tree is constructed using the potential donor records and the set of all matching fields for all recipient records. Certain donor records can be optionally excluded from being added to the search tree by use of an exclusion statement in SQL format. Furthermore, a minimum number or percentage of donors that must exist for the process to continue can be specified. The nearest neighbour search for a suitable donor is performed on the search tree. A potential donor is a suitable donor for a recipient record if the recipient record passes the post-imputation edits after all pertinent fields have been imputed using the corresponding fields of the potential donor. GEIS gives the user the flexibility to specify less restrictive edits in the post-imputation edit set. Note that only the raw field values are transferred between donor and recipient records, and that if a user requires scaling of the values, then this must be done outside of the GEIS system.

Estimator imputation allows a user to replace missing and invalid values using a predetermined method. There are six available methods:

- the field value from a previous time period for the same respondent is used;

- the mean field value based on all records from the previous time period is imputed;

- the mean field value based on all records from the current time period is imputed;

- a ratio estimate, using values from the current time period is imputed;

- the value from the previous time period for the same respondent, with a trend adjustment calculated from an auxiliary variable, is imputed;

- the value from the previous time period for the same respondent, with a trend adjustment calculated from the change in reported values, is imputed.

Note that since these methods cannot preserve the structure of the data as well as does donor imputation, they are primarily intended to serve backup purposes.

The user specifies the choice of estimator[8] to an imputation specifications table, and this is used to drive the estimation imputation programs. Each program, one per algorithm, accesses the imputation specifications table to determine which fields of a questionnaire are to be subjected to the algorithm. The set of records requiring fields to be imputed is retrieved, and using the specification table, appropriate imputation action performed and the records stored with their new values. No post-imputation editing is available automatically, although sub-sets of imputed records may be selected and re-submitted to the Determine Fields to Impute GEIS component to verify if imputed records pass the original edits.

(f) Reports

Various reports are generated to ease the monitoring of the imputation process. These include tabulations of fields to be imputed by reason for imputation; match field status; distribution of donors used to impute recipients; method of imputation used by field; completion of imputation; etc..

2. Data Model for GEIS

The data model for GEIS contains three primary entities: questionnaire specification; survey data for a given time period; and linear edits. A given questionnaire identifier can be associated with many survey data sets, each storing the data for a different data collection period. Only up to two of these data sets can be used in a production run: one being identified to GEIS as the current survey data to be edited and imputed; an optional second data set representing a previous time period to be used in Outlier Detection and in certain of the Estimator Imputation methods. The current survey data set is linked for a processing run to two sets of linear edits: an initial edit set and a post-imputation edit set (if different).

The data model is set up to enable subsets of edits to be run against subsets of the survey data, thus allowing for tuning the edits for certain partitions of the data.

---

[8] The user has several options available to control the use of Estimator Imputation, for example: records may be prevented from contributing to the means used in Estimator Imputation by an exclusion statement in SQL format; also, a minimum number or percentage of usable records may be imposed.

GEIS stores the data of its entities, together with all its reference specifications, intermediate data sets and status information, as tables in an ORACLE data base. For example, data corresponding to a particular time period of a questionnaire is kept in one ORACLE table, where each row of this table contains all the data for one respondent. With so many tables used, the data management problem would have been very onerous without the facilities provided by the ORACLE data base management system.

VI. USE OF DBMS

Since 1987, the strategic direction for informatics at Statistics Canada has focused, among other things, on the use of a RDBMS together with the ISO standard interface language SQL. The GSFD suite of software modules, including the GEIS editing and imputation software, has been developed since the informatics strategic direction was put in place and are thus based on a SQL/RDBMS architecture.

The nature of the contribution to editing of the particular RDBMS employed at Statistics Canada (ORACLE), in the context of the GSFD development, is enumerated below:

(a) ORACLE has provided one of the means of developing a portable system that can be run on both small and large computer architectures (the other main facilitator of portability has been the use of the C language). ORACLE itself exists in identical form on many platforms. At Statistics Canada, RDBMS provides an identical environment on the IBM compatible mainframe under MVS, under Unix on Sun equipment, under Xenix on the Intel 386 architecture, and under DOS on the Intel 386 architecture. Since all data input and output in GEIS is handled by ORACLE, the system is truly isolated from any operating system dependency.

On porting to several platforms the following difficulties arose. ORACLE itself consisted of several component products, e.g., SQLPLUS and SQLFORMS, and it was not always possible to maintain the same version number of these products across different platforms. This led to implementation difficulties. This issue of working with different versions of products on different platforms has also shown up with the C compilers used with the Oracle product.

(b) GEIS employs many ORACLE data tables. ORACLE stores the specifications of these tables in its data dictionary. Several GEIS components reference this data dictionary to verify table and field names. Not having to build this dictionary speeded up development and greatly enhanced the generality of the system.

The underlying approach of GSFD, of which GEIS is just one component, implies that ORACLE data tables are used throughout the survey processing cycle. Then only one data dictionary is required to handle all the different functions of survey processing.

(c) The use of the DBMS has given the system data independence. Data tables are described outside of the programming system and reference to data variables can be made without reference to a table layout and without knowledge of irrelevant tables/variables. This data independence means a table specification in the Oracle data base can be changed, e.g., new variables added, without affecting existing GEIS programs.

(d) The data base allows convenient sharing of data between users with no administrative problems.

(e) Though no security features are directly implemented in GEIS, a user can invoke the security features of Oracle, such as granting access privileges, to ensure that data base tables are not accessed or updated incorrectly.

(f) The ORACLE RDBMS provides a forms product that is effectively identical on all operating platforms and this allows the user to access the GEIS system in the same manner on each machine. This has obvious benefits in terms of minimizing training needed. At present the forms interface is used on each operating platform, but by using networking it is feasible to employ a client/server architecture at the RDBMS level such that the forms interface operates on one machine, a micro, say, while the computational processing operates on another more powerful machine.

(g) The non-procedural SQL language allows one to concentrate on the nature of the editing problem and minimizes the need for data manipulation tricks. The problem environment for editing involves checking sets of data against rule sets. This type of processing maps well to a relational model and thus is very conveniently carried out in SQL. Furthermore one can employ a fully normalized design for data tables which corresponds closely with the conceptual model of the system. Data can be selected from several tables at once by using the join facility of SQL, thus avoiding the need to navigate procedurally through these tables in order to build up composite items of data. The cleanliness of a SQL implementation has simplified the development and testing of the system together with enhancing its maintainability. SQL is also a convenient tool for the user to query GEIS tables from outside of the system.

The degree of normalization employed in setting up GEIS tables is equivalent to providing a columnar view of the data. This is valid for most processes but it does lead to a performance penalty when a record oriented approach is necessary for certain operations.

(h) Because SQL is a powerful 4GL, any given SQL statement embodies the results of much analysis. In reality, one has to be careful during the development and maintenance stage to respect the power of SQL and revisit the underlying analysis behind a SQL statement as a precursor to making any changes to it. Instead of rethinking SQL statements during any maintenance activity, an easy trap to fall into is to add sections of procedural code thus compromising the simplicity obtained by using SQL.

(i) The use of the data base software has provided transparent restart facilities, making the processing immune to any system failure and since, all data is held in one repository, easing system backup.

(j) The use of a RDBMS is considered to have a performance penalty. This is particularly true on general purpose mainframes where the overhead of supporting many types of processing make CPU cycles comparatively expensive, but it has to be evaluated in terms of such matters as reduction in development and maintenance costs and the total costs for all the survey processing including manual intervention (which is reduced through the use of an editing package such as GEIS). With a RDBMS, one can achieve certain performance improvements by manually optimizing SQL code but there usually are automatic optimization features provided by the

RDBMS supplier. With GEIS performance, issues were initially wrongly attributed to the use of a RDBMS whereas they were due to other causes such as the use of a complex linear programming (LP) algorithm (Chernikova's algorithm).

VII. IMPLEMENTATION STRATEGIES

1. Phased Development

The traditional methodology of systems development assumes that the development team has access to a set of detailed specifications covering all aspects of the system. This is not the contemporary approach. With GEIS, even more than with most developments, it was necessary to construct a prototype to prove the methodological framework. An early prototype was also necessary to establish the viability of an MS-DOS micro as a platform for such complex software. The succeeding development comprised several releases of the product: from an initial version just handling the edit specifications and edit analysis elements of the system, to the current version that is functionally complete and represents software that has received considerable tuning.

The phased approach, with each release representing a useable product, encouraged feedback from both subject matter and methodology users. This helped refine the specifications, identified performance issues, and was of immense help in development of the user interface.

2. User Environments

At Statistics Canada, survey processing applications range in size from those which could fit on a micro-computer, to those requiring the processing and storage of a mainframe or larger minicomputer. The portability of GEIS enables editing and imputation on any compatible computer architecture. One possibility likely to occur with introducing GEIS processing into an existing survey is that the current system may be incompatible with the GEIS architecture; for example, the non edit and imputation portions of the survey processing may never reside on an Oracle data base. It has been accepted that the advantages of a standard edit and imputation methodology can be worth the cost of introducing front-end and back-end interfaces between existing survey systems and the GEIS system.

Strategies of editing and imputation and cost allocation between functions differ between surveys and have to be accommodated by a generalized system. GEIS is unique in that it is a production engine, and also a test bed for methodology research. The latter mode gives the methodologist the ability to test different approaches and sequences of edit and imputation operations as part of establishing a survey edit and imputation strategy. Both research and production can coexist in the same GEIS survey database.

A generalized system such as GEIS partitions into human interface elements suitable for interactive environments, e.g., specification and analysis of edits, and elements requiring high computational power, e.g., the error localization task. We are now close to the time when a cooperative network can be set up for running the GEIS system. Its architecture lends itself to having the front-end user interface elements running on a micro based workstation that is

networked to a larger processor specialized to supply the high floating point throughput needed for the linear programming algorithms.

VIII. ISSUES

The development of generalized software products, such as GEIS, under the umbrella of the GSFD project has involved systems, methodology and subject matter specialists. The significance of the effort involved, the newness (to Statistics Canada) of the approaches taken, and potential for long term impact on survey processing applications have caused certain issues to be raised and debated.

1. Technical Issues

There is the expectation that arises from the rapid implementation of preliminary releases of a generalized system: the expectation that the initial rapid development pace can be maintained until completion of the system. There are several factors that can slow development in its later stages. When using new technologies there usually is a significant learning curve to follow before one can move from a functioning system to an efficiently performing one. A further corollary to using leading edge technologies is that new versions often arrive while development is continuing and time is required to adapt the software systems to these new versions.

The use of a non-procedural language such as SQL has to be approached with care. Its power can be a two-edged sword in that, not knowing the underlying processing activities invoked, one can find that subtle variations of the formulation of a high level non-procedural SQL statement can have very large performance consequences.

The management of survey database environments being introduced into survey processing through the generalized products of the GSFD project has to be addressed. For example, a position of survey data base administrator becomes mandatory to manage the data tables in an Oracle data base and to be the focal point for data security requirements. Through the control of indexes on certain GEIS data tables, a DBA has the power to improve the runtime performance of the system. The DBA also has to understand that heavy updating of a database table (as happens in imputation processing) can lead to the physical fragmentation of the table with its concomitant performance penalty. This requires table maintenance to rectify.

2. Managerial Issues

The high degree of automation described in this paper requires powerful DBMS facilities, with a robust, extensible and open data structure, together with protection of concurrent update and data security. These features incur a cost in memory size and CPU consumption, therefore, the advent of computers with larger, cheaper memories and, cheaper CPU cycles was an enabling technology for the GSFD developments. Nevertheless, for a single survey, the use of a general system such as GSFD will cost more with a usage based computing price it would with a custom developed system. The benefits of more advanced methodology, particularly the savings in human operating costs, and the reduced development expenses, are often less visible. The corporate savings accruing from investment and use of standard, powerful general software, should not place individual survey managers, methodologists or system analysts in the dilemma of rationalising extra local costs, without strategic support.

The use of standards, inhibiting unnecessary arbitrary decisions, is valuable in reducing confusion in inter-personal communications, raising productivity in development, and enhancing the utility of survey data. Their adoption, for instance edit methods that preserve the underlying distribution, imposed through standard software reduces the risk, speeds the development cycle and simplifies periodic tuning of the survey. However, standardization can be viewed as the antithesis of flexibility, and all methodologists will attempt to optimize the design of a survey, which will frequently incorporate survey specific features. One has to be careful with general systems not to make them a straight jacket, rather a facilitating tool that encourages the balance of the adoption of standards, whilst permitting needed flexibility where and when required.

The data processing industry has made great strides towards the adoption of a client server architecture, where an assumption is made that all or most computers will be linked to networks of autonomous, cooperative computers, rather than having a large monolithic central processor. The growth of the latter was primarily due to the economies of scale, as expressed by Grosh's law, and this law no longer applies. The specializations of CPU's for particular tasks together with the potential economies in commercial software licences suggest that the client server model offers more promises for the future of computing in national statistical offices. To take an obvious metaphor, the telephone as an instrument is rather unimpressive, however the power and reach of worldwide voice communications is very impressive.

All functions of GSFD will contain a specification facility, in the case of GEIS with extensive support for research to arrive at the eventual specifications, which is best served by a high degree of user interaction. The functions also contain a production engine where the emphasis shifts to greater computational intensity. In the future, we expect both types of work to move further towards their extremes, both more computational intensity, and, even more, more attention to user friendly interfaces. The decoupling of work types supported by the client server model offers the computing environment for the future of the GSFD developments.

BLAISE - A NEW APPROACH TO COMPUTER-ASSISTED SURVEY PROCESSING

by D. Denteneer, J.G. Bethlehem, A.J. Hundepool and M.S. Schuerhoff
Netherlands Central Bureau of Statistics

Abstract: The article presents BLAISE, a system for survey data processing developed by the Netherlands Central Bureau of Statistics. The basis of the BLAISE is a powerful, structured language which describes the questionnaire and its editing procedures. With the thus specified questionnaire as an input, the system automatically generates software modules for data processing.

I. INTRODUCTION

Data collected by the Netherlands Central Bureau of Statistics have to go through a comprehensive process of editing. The consistency of data is checked and detected errors are corrected. Traditionally, this is partly carried out manually and partly by computer. In the process, different departments and different computer systems are involved. Research for a more efficient data editing process (Bethlehem (1987)) has lead to the development of a new system in which all work is carried out on one computer system (a network of microcomputers) and by one department.

The basis of the BLAISE system is a powerful, structured language (also called BLAISE) which describes the questionnaire: questions, possible answers, routing, range checks and consistency checks. With the thus specified questionnaire as input, the system automatically generates software modules for data processing, and these modules are not restricted for use in data editing only.

One of the most important modules is the so-called CADI-machine (Computer Assisted Data Input). This program provides an intelligent and interactive environment for data entry and data editing, for data collected by means of questionnaires on forms. Experiments with CADI have shown a considerable reduction in processing time, thereby yielding high quality data.

Another important module is the CATI-machine (Computer Assisted Telephone Interviewing). The BLAISE system automatically produces the software needed for telephone interviewing, including call management. A module which resembles the CATI-machine is the CAPI-machine (Computer Assisted Personal Interviewing). In the near future, BLAISE will be able to generate the software for this type of face-to-face interviewing with hand-held computers.

BLAISE also automatically generates setups and system files for the statistical package SPSS. Interfaces to other statistical packages (e.g. TAU, PARADOX, STATA) are scheduled in the near future.

II. THE BLAISE SYSTEM

BLAISE is the name of a new system for survey data processing. BLAISE controls a large part of the survey process: design of the questionnaire, data collection, data editing and data analysis. Basis of the BLAISE system is the BLAISE language, a language for the specification of the structure and contents of questionnaire. BLAISE derives its name from the famous French theologian BLAISE Pascal (1623-1662). Pascal is not only the name of the Frenchman but also of the well-known programming language, and the BLAISE language has its roots, for a large part, in this programming language.

In the BLAISE philosophy, the first step in carrying out a survey is to design a questionnaire in the BLAISE language. Since this specification forms the basis of every next step in the survey process, the language should be simple and readable. Users without much computer experience should be able to construct and read BLAISE questionnaires.

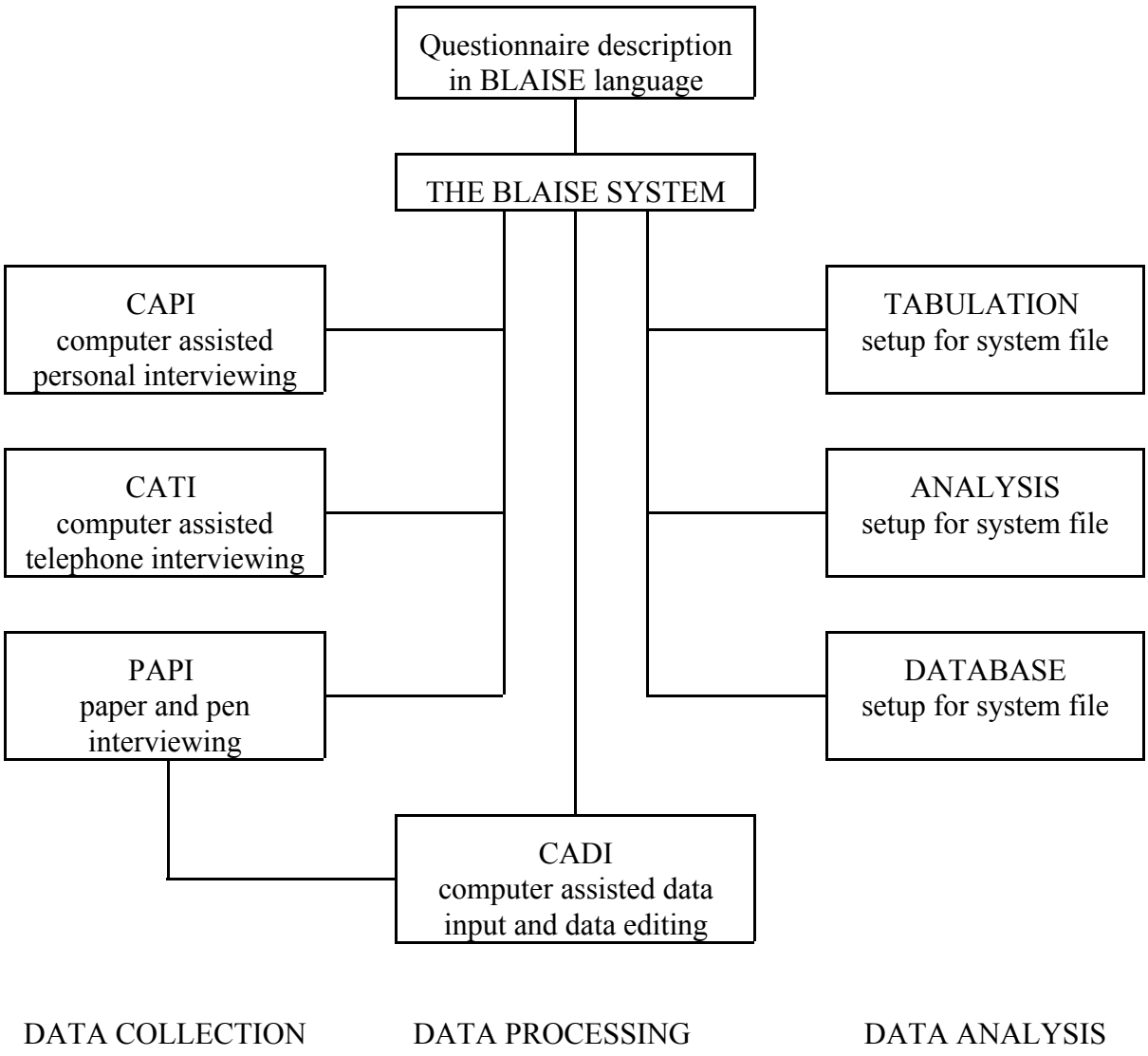
The BLAISE language supports structured questionnaire design. Traditional questionnaires are controlled by *goto's*. This jumping makes the questionnaire quite unreadable. It is hard to check whether the questionnaire is correct, i.e. whether everyone answers the proper questions. BLAISE does not have this kind of jumping. Instead, routing through the questionnaire is controlled by IF-THEN-ELSE structures. This makes the structure very clear. In almost one glance, one is able to distinguish in which cases which questions are asked.

The structured approach to questionnaire design simplifies the job of maintenance and updating of questionnaires. For surveys which are repeated regularly, minor modifications are often sufficient to produce an updated questionnaire. A group of questions about a particular subject can be kept in a single block (sub-questionnaire) e.g. a household block or an income block. If these blocks are stored in a library or database, it is easy to use the same block in different questionnaires. The design of a new questionnaire may reduce to putting together a number of blocks. Use of standardized blocks furthermore improves the exchangeability of results from different surveys.

The Blaise language is one part of the BLAISE system. The BLAISE machine is another part. The BLAISE machine is a software system which takes the BLAISE questionnaire as input, and produces software modules as output. The system is summarized in figure 1.

The user may choose between a number of options for output, thereby establishing the way in which the survey is carried out. For data collection, the user may choose between CATI (computer assisted telephone interviewing), CAPI (computer assisted personal interviewing with hand-held computers), or PAPI (a printed questionnaire for traditional paper and pencil interviewing). For CADI (computer assisted data input), the user has two options: he can either enter and edit forms in an interactive session behind the computer, or he can interactively edit records which are entered beforehand by data typists without error checking. For further processing of the "clean" data file, the user can choose between a number of interfaces which automatically generate setups and system files for software like SPSS.

FIGURE 1: THE BLAISE SYSTEM



III. THE BLAISE LANGUAGE

The input of the BLAISE system is a description of the questionnaire in the BLAISE language. This language is a clear, structured language, which can also be read by non-experienced computer users. Figure 2 gives an example of a simple questionnaire in BLAISE. A BLAISE questionnaire is like the preparation of an apple pie. The preparation contains three parts:

Part 1: Gathering the ingredients

For the preparation of an apple pie, all ingredients must first be ready. Likewise, in a BLAISE questionnaire, all questions must first be specified. This takes place in the **quest-paragraph**. A question consists of an identifying name, the text of the question and a specification of valid answers. Three types of answers can be distinguished in the example. *Sex, Marstat, PosHH and Job* have user defined answer categories. Note that such answers consist

of a short name, used for checking and routing purposes in other parts of the questionnaire, and an extended description, used for explication purposes in generated software modules. The question Age expects an integer answer which is restricted to the range 0, 1, 2, ..., 120. The question Kindjob expects an open answer. Any text may be entered, as long as the length of the text does not exceed 80 characters. The specification of valid answers at the same time specified range checks on the answers.

FIGURE 2: A SIMPLE BLAISE QUESTIONNAIRE

```
QUESTIONNAIRE Dmo;
  QUEST
  Seq (key) "Sequence number of interview":
          1..10000;
  Sex       "What is the sex of the respondent":
          (Male       "Male"
           Female     "Female");
  Age       "What is the age of the respondent":
          0..120
  MarStat   "What is the marital status of the
          respondent"
          (Married  "Married
           NotMar   "Not married");
  Job       "Does the respondent have a job?":
          (Yes   "Yes, has a job"
           No    "No, does not have a job");
  Kindjob   "What kind of job does respondent have?"
          STRING [80]

  ROUTE
      Seq; Sex; Age; MarStat; PlaceHH;
      IF Age > 15 THEN
        Job;
        IF Job = Yes THEN
          KindJob;
        ENDIF;
      ENDIF;
CHECK
  IF AGE < 15 THEN  MarStat = NotMar ENDIF;
  IF (PosHH = Single) OR (PosHH=Child) THEN
     MarStat = NotMar
   ENDIF;
  ENDQUESTIONNAIRE
```

Part 2: Mixing the ingredients

In the preparation phase, all ingredients are mixed in the proper order. The analogue for the BLAISE questionnaire is the **route-paragraph**. It describes in which situations which questions must be answered and the order in which the question must be answered. The route-paragraph clearly shows the structured approach of BLAISE. Instead of goto's, control structures like IF-THEN-ELSE-are used. In the route-paragraph, the short names of the questions are used.

Specification of the name of a question at a certain point means that the question must be asked at that point. In this example, it is obvious that the questions Sex, Age, MarStat, and PlaceHH are asked from every respondent. Only persons older than 15 are asked whether they have a job. The question KindJob is only asked if the respondent is older than 15 **and** has a job. In CATI or CAPI applications, the route-paragraph guides the interviewer through the questionnaire. In a CADI application, the route-paragraph generates checks on the routing in the completed questionnaire.

Part 3: Testing the result

After finishing the pie, a small piece of it is tried to see whether the result is tasteful. Likewise, a questionnaire should contain consistency checks. In a BLAISE questionnaire, checks are specified in the check-paragraph. In particular, consistency checks are specified (range checks are automatically generated from the quest-paragraph and route checks are implied from the route-paragraph). The interpretation of this check-paragraph is very simple: someone under 15 may not be married, and also single persons and unmarried children may not be married.

The approach sketched above would do for small applications, but not for huge ones containing thousands of questions. The BLAISE solution to these huge questions is the block concept. These larger questionnaires can always be seen as a collection of several subquestionnaires, each treating distinct subjects. Each subquestionnaire can be described in a block, with question definitions, route statements, and check statements, and such a block can be included as a "super question" in the main questionnaire. Thus a BLAISE questionnaire can be built from several blocks, where each block will treat a different subject. Moreover, blocks can be nested, so subquestionnaires can be divided into smaller parts. It is always possible to refer to questions in previously defined blocks.

The block concept also provides numerous advantages to questionnaire design. When using blocks, it is possible to design block libraries to store blocks that can simply be included in any questionnaire description. Such libraries may promote standardization of questionnaire parts that are frequently used, such as a household box or a set of income questions. Particularly, blocks are important for the design of hierarchical questionnaires. In a household survey, for example, a block can contain the questions asked to every member of the household. The whole questionnaire is therefore simply defined as an array of blocks.

The paragraph structure and the block concept are the backbone of the BLAISE language. In this general discussion, we have ignored the possibility of local variables, types to be defined in a type-paragraph, external files, and variable texts, to name just a few. Details on these matters can be found in the BLAISE reference manual (Bethlehem et. al, 1989c).

Of course, the BLAISE language can be used after the designing of the questionnaire. However, BLAISE can also be of great importance during the design. The block structure of BLAISE makes it easy to separate the subjects in a questionnaire and to distribute these among the questionnaire designers. They can execute and test their part independently. Integration of the block is easily done by including all of them into a main questionnaire.

IV. THE CADI-MACHINE

One of the data processing applications which can be produced by the BLAISE system is the CADI-machine. The CADI machine is an intelligent and interactive system for data input and data editing. Data can be processed in two ways. In the first approach, the subject-matter specialist sits behind a microcomputer with a pile of forms and processes the forms one by one. He enters the data where appropriate, and after completion of the form, he activates the check option to test routing and consistency (range errors are checked during data input). Detected errors are highlighted and explained on the screen. Errors can be corrected by consulting the form or calling the supplier of the information. After elimination of all errors, a clean record is written to file. If the specialist does not succeed in producing a clean record, a special indication is added to the record, indicating the problems. These hard cases can be dealt with later on, also with the CADI-machine.

In the second approach, the data have already been entered beforehand by data typists, without error checking. With the CADI machine, the records of the file can be edited. One by one, records are read from the file, the check option is activated and detected errors can be inspected and corrected. Again, the system keeps track of the status of the records.

The screen layout of the CADI-machine is the result of an attempt to reconstruct a paper form on a computer screen. However, redundant information is removed: questions are denoted by sequence number and name, rather than the full question text. The full information is always available via help windows.

The routing statements from the route paragraph are interpreted as a series of checks. This implies that routing errors are treated in the same way as errors arising from checks. This also implies that the CADI machine does not force the user to answer certain questions or skip other questions. Rather, the user may answer any question and has complete freedom to page through a questionnaire. This is done to encourage a user to copy the form in the exact manner.

On the other hand, this implies that the user will need to skip questions manually, which costs some extra time. To improve speed, the CADI machine supports keys to skip pages backwards and forward. The additional (psychological) advantage is that a user may turn pages, just as if he was working with a paper form.

During data entry, the user is not bothered by routing or consistency checks. This allows for fast data entry and, again, encourages a user to copy his form in the exact manner. Checks are performed when the user presses a check-key or at the end of the form. Thereafter, the screen changes slightly. In the top right hand corner, it is shown whether errors do occur. The names of questions that are involved in certain errors are followed by error counts. A user may jump to such a question (keys are available to jump to questions that are involved in errors) and look through an error window to the error messages. Such an error message consists of a statement of the error, and a list of questions that are involved in this particular error.

A large statistical office spends a lot of time on manually coded open answers, that is: translating text strings into hierarchically ordered numerical codes. One way to ease this time consuming task is to automatically translate the text strings to the numerical codes. With these methods, approximately 70% of the answers can be translated correctly. The remaining 30% has to be dealt with by coding specialists. This is at variance with the BLAISE philosophy that states that in principle, a record should be correct after one cycle of data-entry and editing. Another

approach to this problem is therefore used in BLAISE. Questions of the coding type are dealt with interactively, as for other questions. During the entry of the answer, two tools are available to facilitate coding. Firstly, there is online information about the descriptions of the next possible digits and secondly, during every stage of entry, an alphabetical sorted list of descriptions and synonyms can be consulted. The list of synonyms can also be used for non-hierarchical codes to facilitate, for instance, the numbering of municipalities. Providing that the list of synonyms used is of good quality, the major part of the questions can be coded directly.

We do not yet have extensive results about the performance of this approach, but several tests show that the method is very promising. It will be used in full scale production for coding goods in the household budget survey.

V. THE CATI-MACHINE AND THE CAPI-MACHINE

The CADI-machine concentrates on processing data which have previously been collected on paper forms. However, the BLAISE system can also be used for data collection. The system is able to produce the software required to carry out CATI-interviewing or CAPI-interviewing.

CATI-programs have been around for a long time, and it does not seem necessary to include a full discussion of CATI programs, (Nicholls and Groves (1985)). The main aspect of the BLAISE CATI-machine is that it is now fully integrated with a larger system for survey data processing, thus eliminating another language to specify a questionnaire. A number of aspects of CATI are still worth mentioning. That is, a user cannot skip questions that must be answered, nor can he answer questions that must be skipped. Thus, dynamic routing implies that an interviewer cannot violate the routing structure of the questionnaire. Note the difference with the static routing as supported by the CADI-machine. This difference arises because of the completely different use of CADI and CATI. With CADI, we want to copy a form and check what has gone wrong; with CATI, we want to avoid that anything goes wrong.

Just as with the CADI-machine, a user can move backwards through a questionnaire, turning pages. Keys are implemented to move backwards just one question, a complete page, or to go to the beginning of the interview. Moving backwards through the questionnaire is, of course, restrained by the routing structure of the questionnaire: a user can only move backwards to questions that are on the routing path. The interviewer can always change previously answered questions. This implies that the routing can be changed at anytime during the interview. Previously entered answers will reappear if the routing is restored.

The CATI-machine applies dynamic error checking. As soon as a consistency error is encountered, an error message is displayed on the screen, together with a list of all questions involved. The user may select the question to correct the error.

The BLAISE language provides for special blocks to describe a non-response conversation or an appointment conversation, including complex routing and check-paragraph. These may be invoked in the routing paragraph of the questionnaire and by means of a function key.

The CAPI-machine is a program to be used on a hand-held computer. CAPI (computer assisted personal interviewing) is a recent and promising development in the area of the data collection. CAPI combines the advantages of face-to-face interviewing with those of CATI

interviewing. The program in the hand-held computer is in control of the interview, i.e. it determines the routing and checks consistency of answers. Since checking takes place during the interview, errors can be corrected on the spot. After successful completion of the interview, a clean record is stored in the hand-held computer. Later on, data can be transmitted to the office by a modem and telephone, or by mail, stored on a diskette.

The emergence of powerful hand-held computers, running under MS-DOS, has greatly facilitated the task of generating programs for CAPI. In fact, a CAPI-program is a complete CAPI-program, be it that the call management module is stripped off.

VI. CONCLUSION

Experiments carried out by the Netherlands Central Bureau of Statistics with the predecessor of the BLAISE system (the interview language QUEST on a hand-held microcomputer running under CP/M (Bemelmans-Spork and Sikkel, 1985a and 1985b) showed that interviewers were able to work with the new CAPI technology, and that use of microcomputers in a face-to-face interview gave the respondents reason for anxiety. Due to the success of these experiments, the Netherlands Central Bureau of Statistics started in 1987 with full scale use of hand-held computers in a regular survey, see Van Bastelaer et al (1987). The data of the Labour Force Survey is now collected by means of hand-helds. Starting in the spring of 1987, ultimately 300 interviewers equipped with hand-helds, will visit 10 000 addresses each month. After a day of interviewing, the batteries of the hand-held computer are recharged at night, and the information collected that day is transmitted by telephone to the central office. The next morning, the interviewer will find a recharged machine with a clean workspace, ready for new interviews.

The BLAISE system has been tested since the middle of 1986, in its various stages of development. The BLAISE environment, the CADI-machine, and the SPSS setup facility have been used successfully for a substantial number of surveys. The other options are available as prototypes and serious testing is scheduled for 1988. The BLAISE system is implemented to run on (a network of) microcomputers under MS-DOS. To use the BLAISE environment, one needs a microcomputer with 640 Kb of RAM, and a hard disk. To run the generated applications, one needs a microcomputer with 640 Kb of memory. A hard disk is not necessary, but eases processing of larger applications in the CADI or CATI approach.

The generation of setups to serve packages for statistical analysis and tabulation clearly is an open ended part of the BLAISE system. Ideally, one would like the BLAISE system to provide setups for any package that could fruitfully be used in survey data processing. No attempt has yet been made however to solve this problem in such generality. In fact, the BLAISE system is currently generating setups for the SPSS-statistical package only (for several of its versions). In the future, a standard interface will be developed that allows for an easy adaptation of many packages.

The BLAISE system has been carefully documented (in English also) (Bethlehem et al., 1987a, 1989d). There are extensive descriptions of the use of the BLAISE environment, and of the CADI-machine. The language is described in two documents. There is a BLAISE tutorial that gives an easy-to-read introduction to the BLAISE language, and there is a BLAISE reference

manual that gives a minute and formal description of the BLAISE language. Experience has shown that knowledge of these documents suffices to use the BLAISE system successfully, even for those who did not have previous experience with computers.

SAS USAGE IN DATA EDITING

by Dania P. Ferguson
United States Department of Agriculture,
National Agricultural Statistics Service

Abstract: The article describes experiences using SAS for data editing application development in survey processing. The usage of SAS reduces the necessary development and maintenance resources for data editing applications. The Survey Processing System (SPS) is presented as an example of a generalized system written in SAS. The SPS was developed by the U.S. Department of Agriculture's National Agricultural Statistics Service.

I. INTRODUCTION

SAS is a statistical analysis system written by SAS Institute in Cary, North Carolina, United States of America. It is a statistical analysis language that allows data users to analyze their data without need of extensive EDP training. SAS has the power of a 4th Generation programming language. There are interfaces from SAS to various data bases including DB2 and Oracle. "Many relational databases allow the definition of views on the tables managed by the database. SAS/ACCESS software can be used to process these views and retrieve the DBMS data for use in the SAS system" (in detail in SAS Institute INC. (1990)). SAS runs on a wide range of operating systems (UNIX, MVS, CMS, VMS, PRIMOS, AOS/VS, OS/2, MS-DOS, PC DOS, VSE, and Macintosh - JMP interactive statistical analysis only).

The most obvious applications of statistical analysis packages to data editing are for statistical and macro editing as well as outlier detection. SAS is well suited for these purposes. SAS is easy to use for simple within record (Atkinson (1991)) editing yet, powerful enough to use as a programming language for building generalized data editing systems.

II. SAS USAGE (MANIFESTATIONS)

SAS is used for data editing by many Statistical offices. The degree to which it is used, however, varies greatly. Some examples of SAS usage in National Statistical Offices are presented below:

Canada

Statistics Canada uses both mainframe and PC SAS. Subject matter specialists use SAS to write application specific programs whilst computer specialists use SAS to write generalized routines to perform intensive calculations for the Oracle based Generalized Edit and Imputation System (GEIS) (Cox (1991)).

France

France I.N.S.E.E. uses SAS for imputation, analysis, and camera copy. There is a generalized SAS module for survey outlier computations. The rest are application specific routines. SAS is also used to create quick quality control check tables.

Hungary

In the Hungarian Statistical Office, the statisticians use SAS for analysis. They start with a clean file from the mainframe. Both PC and mainframe SAS are used to edit and create a final summary.

Spain

Spain has written a generalized system in SAS and dBase III for macro editing requiring level by level disaggregation to localize errors (Ordinas (1988)).

Sweden

Statistics Sweden has used both PC and mainframe SAS to perform macro editing of many surveys. The aggregate method was used for interactive error detection on the PC (see Lindstrom "A macro-editing application developed in PC-SAS" in this publication). The input calculations are application specific, but aggregation and error detection is generalized. Statistics Sweden is hopeful that the SAS graphics now available for the Macintosh (SAS/JMP) will support the box-plot method of macro editing (see Granquist "Macro-editing -- a review of methods for nationalizing the editing of survey data" in this publication). Marking a point on a graph to see its identifier and other associated data is a requirement for the box-plot method.

Statistics Sweden has also developed a prototype macro editing procedure using a top-down method on the PC (Lindblom (1990)). SAS/AF was used to build the menu system and SAS/FSP to edit records.

United States of America

(a) U. S. Department of Agriculture, National Agricultural Statistics Service (USDA, NASS)

USDA-NASS uses SAS on the mainframe and micros. The State Statistical Offices use PC-SAS for data editing and summarization. Some applications use macro editing of the top-down and aggregate methods 1) for outlier detection and 2) to check aggregate fit against a time series trend line.

Headquarters developed a generalized data editing system for processing National surveys in SAS. The system runs batch on the mainframe. Parameters are entered and validated on the micro using SAS/AF and SAS/FSP.

The data editing system supports survey management functions by providing an audit trail, error counts, and a status report.

The data editing system supports macro editing of the aggregate, top-down, and statistical methods. Charts are provided with the statistical edits along with a detailed listing of identifiers and data associated with outlying points. Deterministic imputation and the substitution of previous period data is supported by the edit system . An imputation procedure is written in SAS as a general application for the Crop Surveys to substitute weighted averages within each stratum.

Sample select, Optimal Sample Allocation, Analysis, Data Listing, and Summary modules have also been written in SAS. Together they form a generalized Survey Processing System with batch links to a mainframe ADABAS data base.

(b) U.S. Department of Commerce:

The various Bureau's within the Department process independently. In fact, the centers within the Bureaus often develop software for their individual use. While SAS is not used for data editing, it is used for research by some statisticians. SAS is also used for some summaries.

Edward Cary Bean, Jr. at the U.S. Bureau of the Census, proposed the Vanguard System (Bean (1988)) for information processing. The system was to be written using SAS as the programming language. Development was delayed due to budget. Only the file structures and a heads down key entry module have been implemented. The design included a generalized data editing module.

III. SAS FEATURES

SAS is a user-friendly system developed to be used by statisticians to manipulate their data. SAS provides a series of procedures (PROCs) for use in statistical manipulations. These include such procedures as:

MEANS, FREQency, MULTIVARIATE, SUMMARY, MATRIX, CLUSTER, REGression and TRANSPOSE.

The procedures work on SAS datasets created using the SAS procedures: FSEDIT, CONVERT, DBASE; or the DATA step. The FSEDIT procedure is a full screen editor for key entry. Many conversion procedures are available to convert existing files to SAS format. The DATA step affords the most flexibility in file conversion and data manipulation. In the DATA step, one is not concerned with read/write loops or the definition of working storage. SAS performs these operations automatically, although the defaults may be overridden, if one so desires.

SAS supports many reporting functions with its PRINT, CHART, FORMS, PLOT, GRAPH, and CALENDAR procedures.

SAS provides an FSCALC spreadsheet procedure for the mainframe. The SAS/AF (Applications Facility) allows users to program menu driven control of the execution of the various SAS procedures. SAS/AF can be used to build an application quickly as well as to develop computer based training.

SAS provides for interactive editing using the FSEDIT procedure. With FSEDIT, range checks can be made on fields within a record. Fields can be validated based on value lists. Consistency checks can be done between fields on the record. The FSEDIT procedure can be used again after aggregation using the SAS SUMMARY procedure or a DATA step to calculate the aggregate.

If the features of FSEDIT are insufficient for interactive editing, one can write an enhanced editing procedure. This is easily accomplished with a combination of other SAS system features, such as:

SCL - Screen Control Language, SAS/AF, SAS DATA steps, and many SAS procedures.

The DATA step can be used to merge data sets and to edit hierarchical data. SAS statistical procedures can be used to calculate edit limits based on the distribution of the data. These limits can then be used as parameters to a DATA step editing routine.

SAS has many features that make it well suited for data editing and general survey processing use. SAS is 1) easy to use by the statisticians and 2) a very powerful programming language for use by computer specialists. SAS supports enough programming tools that it can be used for generalized systems development. It is a tremendous advantage to have the programmer and statisticians speaking to the computer in the same language. It improves communications and thus, quality and productivity.

The following section presents a generalized system that was written in SAS.

IV. SURVEY PROCESSING SYSTEM (SPS)

1. Overview

The Survey Processing System was developed by the U.S. Department of Agriculture's National Agricultural Statistics Service (USDA-NASS) using SAS as the programming language. SAS was chosen mainly because NASS statisticians were already familiar with SAS and it could easily be used as both the programming and parameter specification language. It afforded the unique ability of the users and programmers to speak to the computer in the same language. While, allowing each to work at their own level of computer expertise.

On one annual survey, the use of this system reduced the parameter coding and maintenance from 1 statistical staff year to 1 staff month. The EDP staff support was reduced from 3 staff months to 1 staff week. These figures are more interesting when one considers that a very powerful data editing system was already in use. The gain was due mostly to the fact that the system design took advantage of SAS as a common communication tool.

SPS lets the statisticians write their parameters in SAS. The system merely provides tools to shelter the statistician from the operating system and file maintenance. The design sought to simplify the path between the statisticians and their data. It got the programmer out of the communication path between the statisticians and the computer. The design forces edit

specifications to be written in a modular, reusable form and automatically stores them in a shared library.

The Survey Processing System consists of the following sub-systems:

Specifications Interpreter,
Sample Select,
Optimal Sample Allocation,
Data Validation,
Data Listings,
Data Analysis, and
Summary.

2. Data Validation Sub-system

The Data Validation Sub-system is a batch data editing system. It is used to process large national surveys. Data is keyed and submitted by the State Statistical Offices (SSO's) to a mainframe. The parameters are compiled in Headquarters for most surveys. For some larger surveys the SSO's submit lists of upper and lower limits by commodity for their State.

The following is a list of the steps in the data validation process:

(a) Keyed data is validated as follows:
- Identifier information is checked against the sample master.
- State and Questionnaire Version are used to validate entries allowed on a questionnaire.

(b) The keyed data is posted to an edit data file.
Optionally creates an audit trail file of records before and after update with a date/time stamp.

(c) (Optional)
Consistency checks and critical range checks are applied to the edit data file.

(d) (Optional)
Ranges are generated by calculating statistics of each specified field on the edit data file. Up to now, the mean and standard error. Ranges are usually calculated by region or stratum for each field. In a Price survey, the average price for each item (gasoline, nails) is calculated for each Farm Production region. The average price of each type of fertilizer is calculated for each Fertilizer region. Then the standard error is used to calculate limits to flag the 5% tails for each price.

(e) (Optional)
Range checks are applied to the value of each entry.
- Default ranges can be specified for each questionnaire entry on each questionnaire version.
- State specific ranges override the default ranges set by Headquarters for each entry.

Ranges can be specified by parameter, or calculated using data from previous or the current survey (step (d)).

(f) A status report is generated showing the run totals for each state.

These include:

- a count of errors by (parameter assigned) error number and description.
- a count of records with critical errors. (Critical errors must be resolved before summarizing.)

Each State receives their own status and Headquarters receives a one page status of all states on a daily basis.

(g) (Optional)

A report is generated listing samples for which a questionnaire has not been submitted (missing report listing).

This step is run on every batch starting at the midpoint of a survey.

V. CONCLUSION

SAS is used to write quick edit logic for a one-time survey, as well as, to write generalized editing systems. It is used to write complex statistical subroutines for data base systems. SAS has many features that make it well suited for data editing and general survey processing use. SAS is 1) easy to use by the statisticians and 2) a very powerful programming language for use by computer specialists.

SAS is easy to learn by non-EDP personnel. This gives subject matter specialists the opportunity to edit and analyze their data without EDP staff. Thus, providing the freedom and flexibility required for pilot surveys and small or one-time surveys.

SAS reads and writes many file structures. Allowing SAS usage for routines to enhance existing systems. SAS should be thought of as a replacement 3rd Generation Language (3GL). Using SAS for data editing and other purposes other will soon result in increased programming efficiency. SAS procedures provide many utilities that would need to be coded in 3GLs. This frees the programming staff to concentrate on more serious development tasks.

As machine costs decrease and computer specialist salaries increase we need to find ways to quickly and easily write reusable, maintainable systems. SAS is a solution that is especially well suited to Survey Processing needs. SAS gives subject matter specialists tools to process their own data. Most importantly, SAS provides a language common to computer specialists and subject matter specialists.

A MACRO-EDITING APPLICATION DEVELOPED IN PC-SAS

by Klas Lindstrom
Statistics Sweden

Abstract: The article presents a system for the macro-editing of the Survey of Employment and Wages in Mining, Quarrying and Manufacturing, developed in Statistics Sweden. The system runs on a PC using SAS software.

I. INTRODUCTION

The Swedish monthly survey on employment and wages in Mining, Quarrying, and Manufacturing is a sample survey. Data are collected every month from about 3000 establishments belonging to about 2500 enterprises. The main variables collected are: the Number of operation days for the period, the Number of operation days for the month, the Number of working hours for the period, the Sum of wages for the period and the Number of employed.

II. THE MACRO-EDITING METHOD

The macroediting method used was the aggregate method. The idea behind this method is to use an error-detecting system twice. Checks are first made on aggregates and then on records belonging to suspicious (flagged) aggregates. The acceptance limits for the checks are based on the distribution function of the check variables.

The edits used are based on computing the following variables at branch level. The weights for the previous period are used in computing these variables.

T1=100, 0* WORK HOUR (August) / WORK HOUR (June)
S1=WORK_ HOUR (August) -WORK _HOUR (June)

T3=100, 0*SUM WAGE (August) / SUM WAGE (June)
S3=SUM_WAGE (August) -SUM_WAGE (June)

T4=100, 0*HOURLY_WAGE (August) /HOURLY WAGE (June)
S4=HOURLY _WAGE (August) - HOURLY _WAGE (June)

T5=100, 0*EMPLOYED (August) /EMPLOYED (June)
S5=EMPLOYED (August) -EMPLOYED (June)

These ratios and differences at branch level are listed in ascending order. On the basis of these lists the acceptance limits for the checks at branch level are determined. The checks which are used at branch level are of the following type:

If (Tx<Lower limit_Tx & Sx<lower limit_Sx) |
(Tx>upper limit_Tx & Sx>upper limit_Sx) then flag the branch.

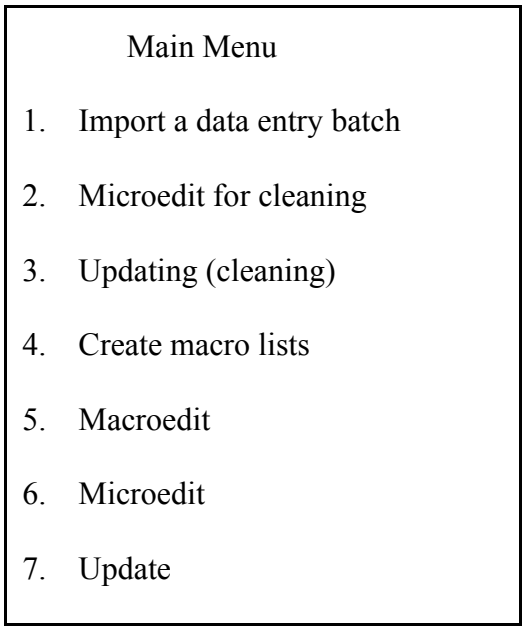
III. THE SAS-APPLICATION

For the test a simple prototype was created using SAS on the PC. The prototype was built using the SAS/AF software. It contains a number of menus to be filled in by the user.

Figure 1 shows the main menu. The editing is supposed to be done for a number of batches. Each batch is composed of the number of forms which have been data-entered at the same time. This reflects the production system where the forms arriving up to a specified date are data-entered at the same time. To edit a batch the user passes through the different alternatives on the menu.

The second and third alternatives on the menu were not used in the evaluation. The "cleaning" of the records was moved to alternative 6. The purpose of the "cleaning" is to handle records with item nonresponse.

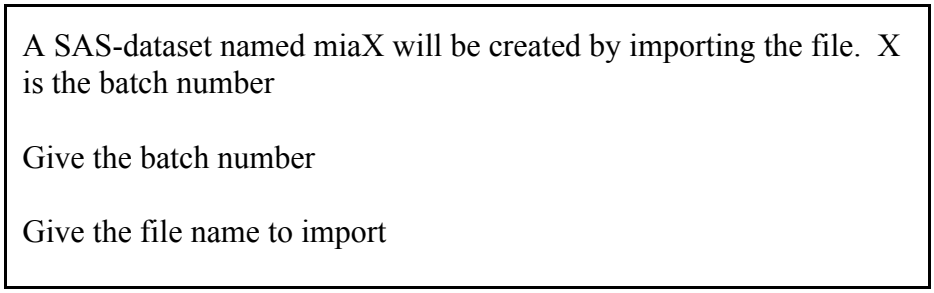
Figure 1:



The first thing to do is to import the data entry file from a diskette into the SAS-system. At the same time two of the variables are imputed, namely Number of operation days for the month and Number of operation days for the period when they are missing. This is done by choosing alternative 4 on the Main menu and then entering the batch number and the file name on the menu display in figure 2.

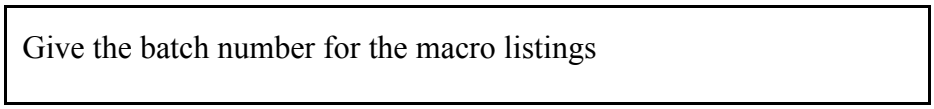
This will create a SAS-job which will import the data to a SAS-dataset and impute the variables Number of operation days for the period and Number of operation days for the month. As a result a list of the input records and a SAS-dataset will be produced.

Figure 2:



Once the batch has been imported, the next step is to create the macro lists which shall be used to determine the acceptance limits for the macro editing. This is done by choosing alternative 4 on the Main Menu and then entering the batch number on the menu displayed in figure 3. As a result a SAS-job is created that will produce the desired lists.

Figure 3:



On the basis of the lists it is possible to determine the acceptance limits for the macroediting. Choose alternative 5 on the Main Menu and then enter the acceptance limits together with the batch number on the menu displayed, see figure 4. This will create a SAS-job that will produce a list of the flagged branches and a SAS-dataset.

The SAS dataset will contain all the variables from the input and new variables indicating whether the record belongs to a flagged branch and in that case which check has caused the flagging.

Lists on the establishment level are also produced here. The lists are the same as on branch level but the differences and the ratios are calculated on establishment level. Only establishments belonging to a flagged branch for that variable appear on the list.

Figure 4:

Give the batch number for macroediting				
Give the limits for the checking				
Check	Ratio lower limit	Check upper limit	Difference lower limit	Check upper limit
1 WORK HOUR	----	----	----	----
2 SUM WAGE	----	----	----	----
3 HOURLY WAGE	----	----	----	----
4 EMPLOYED	----	----	----	----

These lists are used for determining the acceptance limits on the micro level. When the limits have been determined you choose alternative 6 on the Main Menu and the Main Menu in figure 5 will be displayed.

Here you enter the number of the batch together with the acceptance limits for the checks on micro level. As a result a SAS-job will be created that produces a new SAS-dataset containing all previously existing variables together with flags from the micro editing. The job will also produce a list of all flagged records.

Figure 5:

Give the batch number for microediting				
Give the limits for the checking				
Check	Ratio lower limit	Check upper limit	Difference lower limit	Check upper limit
1 WORK HOUR	----	----	----	----
2 SUM WAGE	----	----	----	----
3 HOURLY WAGE	----	----	----	----
4 EMPLOYED	----	----	----	----

On the basis of the error list a decision about which records to correct can be made. Then you choose alternative 7 on the Main Menu. This will result in the menu shown in figure 6 being displayed. On this menu you enter the batch number and thereafter the flagged records will be displayed one by one on the screen. Here it is possible to correct the record before continuing to the next flagged record.

V. THE RESULTS

A simulation of the editing was done on the survey for August 1989. The data was checked against the June 1989 survey.

The data for macroediting was divided into two batches, the first one with 1090 records and the second one with 1961 records.

Before the data was edited the variables Number of operation days for the period and Number of operation days for the month were imputed. When those variables had no value they received a value based on the value for the previous month. In the first batch 25 records were imputed and in the second batch 26 records were imputed.

After the imputation the macro lists were created. These were used for determining the acceptance limits for editing rules at macro level.

Eight different lists were produced. The lists contain the difference between the present and the previous survey at branch level, and the ratio between the present and previous survey. These lists were produced for the variables Number of worked hours, Sum of wages, Hourly wage and Number of employed. This gives two lists for every variable, one sorted in ascending order by the value of the difference, and one sorted in ascending order by the value of the ratio.

On the basis of the lists the limits for the acceptance interval on macro level were determined. The limits are presented in Table 1. The results of using these limits are presented in Table 2.

Table 1

Acceptance limits for the macroediting on branch level

	Batch 1		Batch 2	
	ratio	difference	ratio	difference
Number of worked hours				
lower limit	87%	-4000	84%	-14000
upper limit	120%	4000	140%	25000
Sum of wages				
lower limit	87%	-30000	89%	-300000
upper limit	113%	150000	150%	1000000
Hourly wage				
lower limit	95%	-3.0	95%	-4.0
upper limit	105%	3.0	110%	6.0
Number of employed				
lower limit	95%	-40	90%	-100
upper limit	105%	40	105%	100

Table 2

Number of branches and flagged branches per checking rule and batch at the macroediting on branch level.

	Batch 1		Batch 2	
	Number of Branches	Flagged Branches	Number of Branches	Flagged Branches
Number of worked hours	78	13	86	13
Sum of wages	78	15	86	13
Hourly wage	78	19	86	7
Number of employed	78	17	86	9
All checking rules	78	40	86	24

For all the flagged branches, lists were produced of the same type as earlier but now on establishment level instead of branch level. The lists contained all establishments belonging to

a flagged branched for the variable. The number of establishments belonging to a flagged branch is presented in Table 3.

Table 3

Number of establishments belonging to a flagged branch per checking rule.

	Batch 1	Batch 2
Number of worked hours	175	302
Sum of wages	89	308
Hourly wage	249	194
Number of employed	204	159

On the basis of these lists the acceptance limits for the editing on establishment level were determined. The limits are presented in Table 4.

Table 4

Acceptance limits for the editing on establishment leve.

	Batch 1		Batch 2	
	ratio	difference	ratio	difference
Number of worked hours				
lower limit	85%	-10000	70%	-14000
upper limit	150%	10000	130%	8000
Sum of wages				
lower limit	75%	-500000	76%	-700000
upper limit	40%	400000	150%	700000
Hourly wage				
lower limit	85%	-10.0	80%	10.0
upper limit	115%	10.0	115%	10.0
Number of employed				
lower limit	85%	-40	83%	-40
upper limit	115%	50	110%	20

These rules were applied to the material together with a number of validity checks. The result of this editing is presented in Table 5.

An establishment may have been flagged by more than one checking rule. This means that the total number of flagged establishments is less than the sum of the different checking rules.

V. COMPARING THE MACRO-EDITING AND THE PROCUTION EDITING

The flagged establishments in the macroediting were imputed with the edited values from production. When the record had been corrected in the production the same corrections were applied to the macroedited record. The number of corrected establishments is presented in Table 5.

Table 5

Number of flagged and corrected establishments after editing on establishment level.

	Batch 1		Batch 2	
	Flagged	Corrected	Flagged	Corrected
Number of worked hours	13	23		
Sum of wages	20	22		
Hourly wage	16	15	21	31
Number of employed	18	26		
Validity checks	22	21	53	50
All	80	36	11	881

After corrections had been applied, estimates were made for the Number of worked hours, the Sum of wages, the Hourly wages and the Number of employed on branch level. The absolute percentage deviation was then computed between the estimates of the production and the estimates of experiment. These results are presented in Table 6.

Table 6

Number of deviations between data edited in production and in the experiment as a function of the absolute percentage deviation for these estimates on branch level.
The results from a previous study are given in parentheses.

Absolute percentage deviation	Number of employed	Work hours	Sum of wage	Hourly wage	Sum
0	72(78)	45(77)	36(66)	36(68)	191(289)
0,0-0,1	1(4)	6(2)	6(12)	15(8)	28(26)
0,1-0,4	10(3)	10(6)	14(4)	18(8)	52(21)
0,5-0,9	3(1)	6(1)	6(5)	6(3)	21(10)
1,0-1,9	0(1)	9(1)	15	7(1)	31(3)
2,0-2,9	1	3	2	3	9(0)
3,0-3,9	1(1)	2(1)	1	1	5(2)
4,0-4,9	0	2	2(1)	1	5(1)
>4,9	0	5	4	1	10(0)

When comparing the results from this study with the results from the previous study it can be seen that the deviations are larger in this study.

A closer look at the branches with a deviation greater than 5% shows that there are 5 branches which cause those deviations. For those branches it should be sufficient to correct one record per branch. Then the deviations should be less than 5%.

The reason why these records were not flagged at the macroediting is that the difference and ratio for the branch is inside the acceptance interval. This means that the branch will not be flagged and the establishments for that branch will not at all be checked in the microediting.

VI. FURTHER EDITING

To try to find the records behind the big deviations the data was microedited once more. At this microediting ratio checks were applied to all records which did not belong to a flagged branch. The following ratios were used in the checks:

T1=100,0*WORK_HOUR (August)/WORK_HOUR (June)

T3=100,0*SUM_WAGE (August)/SUM_WAGE (June)

T4=100,0*HOURLY_WAGE (August)/HOURLY_WAGE (June)

T5=100,0*EMPLOYED (August)/EMPLOYED (June)

The ratio checks were the following type:

If (Tx<lower limit Tx OR Tx> upper limit_Tx) then flag the establishment. Tests were made with different limits for the ratio checks. The results of these tests are presented in Table 7.

Table 7

Number of flagged and thereof corrected establishments at different limits for the ratio checks.

Upper limit	Lower limit	Number of flagged			Number of corrected		
		Batch 1	Batch 2	Total	Batch 1	Batch 2	Total
500	20	0	15	15	0	12	12
400	25	3	19	22	2	15	17
300	33	9	25	36	4	16	20
200	50	45	68	113	17	29	46

The changes of the estimates which these corrections led to are presented in Tables 8,9,10, and 11.

Table 8

Number of deviations between data edited in production and in the experiment as a function of the absolute percentage deviation for the estimates on branch level after further microediting with ratio<20 or ratio>500 for not flagged branches.

Absolute percentage deviation	Number of employed	Work hours	Sum of wage	Hourly wage	Sum
0	72	47	40	39	198
0,0-0,1	1	5	6	16	28
0,1-0,4	10	12	16	20	58
0,5-0,9	3	9	8	5	25
1,0-1,9	0	10	14	6	30
2,0-2,9	1	3	2	1	7
3,0-3,9	1	1	0	0	2
4,0-4,9	0	0	1	1	2
>4,9	0	1	1	0	2

Table 9

Number of deviations between data edited in production and in the experiment as a function of the absolute percentage deviation for the estimates on branch level after further microediting with ratio<25 or ratio>400 for not flagged branches.

Absolute percentage deviation	Number of employed	Work hours	Sum of wage	Hourly wage	Sum
0	72	48	41	40	201
0,0-0,1	1	6	6	18	31
0,1-0,4	10	12	16	18	56
0,5-0,9	3	8	8	4	23
1,0-1,9	0	10	14	6	30
2,0-2,9	1	3	2	1	7
3,0-3,9	1	1	0	0	2
4,0-4,9	0	0	1	1	2
>4,9	0	0	0	0	0

Table 10

Number of deviations between data edited in production and in the experiment as a function of the absolute percentage deviation for the estimates on branch level after further microediting with ratio<33 or ratio>300 for not flagged branches.

Absolute percentage deviation	Number of employed	Work hours	Sum of wage	Hourly wage	Sum
0	72	48	41	40	201
0,0-0,1	1	7	6	16	30
0,1-0,4	10	12	17	20	59
0,5-0,9	3	8	8	4	23
1,0-1,9	0	8	12	6	26
2,0-2,9	1	4	3	1	9
3,0-3,9	1	1	0	0	2
4,0-4,9	0	0	1	1	2
>4,9	0	0	0	0	0

Table 11

Number of deviations between data edited in production and in the experiment as a function of the absolute percentage deviation for the estimate on branch level after further microediting with ratio<50 or ratio>200 for not flagged branches.

Absolute percentage deviation	Number of employed	Work hours	Sum of wage	Hourly wage	Sum
0	73	51	43	42	209
0,0-0,1	1	7	11	16	35
0,1-0,4	9	14	17	20	60
0,5-0,9	3	7	7	4	29
1,0-1,9	0	5	8	5	18
2,0-2,9	1	3	2	1	7
3,0-3,9	1	1	0	0	2
4,0-4,9	0	0	0	0	0
>4,9	0	0	0	0	0

As can be seen from the tables, records which caused the largest deviations are flagged when the acceptance limits of the ratio checks are <25 and >400. A continuation of the checking with smaller acceptance intervals lessens the deviations but at the same time the number of flagged records will increase.

The selected strategy combining the macroediting with ratio checks with very wide acceptance intervals for records belonging to a non-flagged branch will give the flags and corrections presented in Table 12.

Table 12

Number of flagged and corrected establishments after batch and type of flag. For the macroedited data with ratio check for ratio<25 or ratio>400.

	Number of flagged establishments			Number of corrected establishments		
	Batch 1	Batch 2	Total	Batch 1	Batch 2	Total
Number of worked hours	13	23	36			
Sum wage	20	22	42	15	31	46
Hourly wage	16	21	37			
Number of employed	18	26	44			
Valitidy checks	22	53	75	21	50	71
Ratio checks	3	19	22	2	13	17
Total	83	137	220	38	96	134

A comparison between the results in the production and the experiment is presented in table 13.

Table 13

Number of edited, flagged and corrected establishments at production and at the experiment.

	Number edited	Flagged		Corrected	
		number	%	number	%
Production	3051	1087	36	306	28
Experiment	3051	220	7	134	61

Macroediting caused a reduction of the number of flagged establishments of 80 percent; 61 percent of the flagged establishments were corrected compared with 28 percent at production.

The reduction in the number of flags is probably slightly greater than if the method were to be used in ordinary production since in that case, the editing would be divided into more batches and this would lead to increasing the number of flags from macroediting.