STATISTICAL COMMISSION and
ECONOMIC COMMISSION FOR EUROPE

Working Paper No. 18
English only

CONFERENCE OF EUROPEAN STATISTICIANS

UN/ECE Work Session on Statistical Data Editing
(Cardiff, United Kingdom, 18-20 October 2000)

Topic III: New techniques and tools for editing imputation

NEW DEVELOPMENTS IN AUTOMATIC EDIT AND IMPUTATION AT STATISTICS NETHERLANDS

Submitted by Statistics Netherlands[1]

Invited paper

Abstract: In close collaboration with several other institutes Statistics Netherlands has recently developed new algorithms and prototype computer programmes for automatic edit and imputation. This paper reports on these developments. The paper begins with the author's view on the history of automatic data editing,starting with Fourier in the early 19th century and, for the present, ending with the algorithm for automatic editing that has been developed at Statistics Netherlands in collaboration with the Eindhoven University of Technology. The method developed by Fourier, called Fourier elimination, for checking the consistency of a set of inequalities is considered to be the starting point of automatic editing. Two well-known methods for automatic editing, the Fellegi-Holt method and the method based on Chernikova's algorithm, are seen as an extension and as a so-called dual version of Fourier elimination, respectively. The recently developed algorithm for automatic editing, which can be applied to a mix of continuous and discrete data, is briefly described. It basically consists of generating small sets of implied edits a large number of times.

The new prototype software for automatic imputation has been developed by an international consortium of institutes that was partly funded by Eurostat. This computer programem is based on weighted automatic interaction detection, which is used to fit a tree-based model to data. The terminal nodes of such a tree form clusters that are as homogeneous as possible with respect to a response variable of interest. These homogeneous clusters can be used to select a donor record, for instance. The developed software can be applied to impute for missing values of both discrete and continuous data.

I. INTRODUCTION

1. In this paper we describe new algorithms and prototype computer programmes for automatic edit and imputation that have recently been developed at Statistics Netherlands. The development of these algorithms and software has not been undertaken solely by Statistics Netherlands. A new algorithm for automatic editing has been developed in close collaboration with the Eindhoven University of Technology. A new algorithm and software for automatic imputation has been developed by a consortium of institutes that was partly funded by Eurostat.

2. The idea for the new algorithm for automatic editing arose while studying the literature on automatic editing and related subjects, such as with Fourier elimination. Section II therefore describes the history of automatic editing, as seen by the author. The new algorithm for automatic editing itself is the subject of Section III. This algorithm can be used to identify suspicious values in a data set. These

[1] Prepared by Ton de Waal.

suspicious values are considered erroneous and are set to missing. Subsequently, the missing values, either missing in the original data set or missing due to the editing process, have to be imputed. Section IV briefly describes the new algorithm for automatic imputation. During the imputation phase the edits are not taken into account. An imputed record may therefore still fail some edits. Recently, research has been undertaken on optimally adjusting imputed values such that all edits become satisfied. Some ideas for this research are mentioned in Section V. Section VI concludes the paper with a description of the role of automatic editing and imputation in the entire editing process.

II. THE ORIGINS OF AUTOMATIC DATA EDITING

3. The well-known French mathematician and Egyptologist Baron (Jean Baptiste) Joseph Fourier led a fascinating life. Fourier was born the son of a tailor at Auxerre in 1768. He attended the local military school, where he soon showed his great talent for mathematics. After he had finished his studies there, he became a teacher of mathematics at the same school. As a strong supporter of the French Revolution, his life was in danger several times because of his political ideals. In 1795 he became a teacher at the École Normale, and within a year he became a teacher at the École Polytechnique.

4. In 1798 Fourier accompanied Napoleon on his expedition to Egypt, and became governor of Lower Egypt. He stayed in Egypt until 1801. During this period Fourier was engaged in research on Egyptian antiquities. After his return to France, Fourier was appointed prefect of Grenoble. He was given the task of publishing the Egyptian materials that had been collected by the French. In Grenoble he also began his experiments on the conduction of heat. He succeeded in determining a differential equation of fundamental importance for the understanding of heat conduction. In his book "*Théorie analytique de la chaleur*" he showed that any function of a variable can be expanded in a series of sines of multiples of the variable, the so-called Fourier series. These Fourier series and their continuous counterparts, the Fourier integrals, have become very important tools for mathematicians and mathematical physicists.

5. In 1809 Napoleon made Fourier a baron. In 1815, after Napoleon's fall from power, Fourier was appointed director of the Statistical Bureau of the Seine. For his work in Egyptology he was elected in 1826 to the Académie Francaise and the Académie de Médecine. Fourier died in Paris on 16 May 1830.

6. This is only a very brief sketch of Fourier's life. Much more information on Fourier's life can be found on the Internet (e.g. in Encyclopaedia Britannica at http://www.britannica.com, and in the MacTutor History of Mathematics archive at http://www-groups.dcs.st-andrews.ac.uk/history/Mathematicians). Below we will describe how Fourier is related to automatic editing.

7. In the early 19th century Fourier became interested in systems of linear inequalities. In particular, Fourier was interested in the problem whether a feasible solution to a specified set of linear inequalities exists. Fourier developed an interesting method to solve this problem based on successively eliminating variables from the system of inequalities. This method, which was later called Fourier elimination, was published for the first time in 1826 (see Fourier, 1826 and Kohler, 1973). Fourier himself was not too impressed by his discovery. In his book on determinate equations, "*Analyse des equations déterminées*", that was published after his death in 1831 this method for eliminating variables was omitted.

8. Although Fourier developed his elimination method for linear inequalities, it can also be used for systems of linear inequalities and equalities, of course. One way to do this is to simply express each equality as two inequalities, and then use Fourier elimination on the resultant system of linear inequalities.

9. To some degree, Fourier's discovery marked the beginning of automatic editing, although automatic editing was, of course, not even known at that time. In modern terminology one could say that Fourier gave an answer to the question whether a set of numerical variables can be imputed in such a way that a specified set of edits, given by linear inequalities or equalities, can be satisfied. This is one of the

most important questions that need to be answered when one wants to edit numerical data automatically: which variables can be imputed consistently given the edits and a particular record?

10. Fourier's method eliminates variables from a system of linear inequalities and equalities. The number of linear inequalities and equalities may change while eliminating variables. This may be a problem for large systems. For large systems the number of linear (in)equalities often becomes so high that Fourier elimination becomes impractical, i.e. too much computing time and computer memory is required to apply Fourier elimination.

11. There is a dual version of Fourier elimination. Here linear (in)equalities are eliminated rather than variables. The number of variables may vary while eliminating constraints. The dual version of Fourier elimination suffers from the same problem as the primal version: for large systems the number of variables often becomes so high that the method is impractical.

12. For many years Fourier's method was forgotten. In the 20th century it was rediscovered several times. For instance, Motzkin (1936) rediscovered the method. For this reason the method is sometimes also referred to as Motzkin elimination, or more frequently as Fourier-Motzkin elimination. Other people who have rediscovered the method are Dines (1919) and Chernikova (1964, 1965). Chernikova in fact rediscovered the dual version of Fourier elimination.

13. In 1976 Fellegi and Holt again rediscovered Fourier elimination. Fellegi and Holt, however, not only rediscovered Fourier elimination for numerical data, they also extended the method to categorical (discrete) variables. This was a major extension of Fourier's original method, although it has never been recognised as such by specialists on Fourier elimination.

14. Fellegi and Holt (1976) also described several important principles for automatic data editing and imputation. For instance, they proposed a paradigm for identifying the erroneous fields in a record. According to this paradigm the data should be made to satisfy all edits by changing the fewest possible number of fields. Because it was soon realised that not all variables are equally trustworthy, the Fellegi-Holt paradigm was later generalised to: the data should be made to satisfy all edits by changing the fields with the smallest possible sum of confidence weights. Here a confidence weight is a positive number corresponding to a variable that expresses the confidence one has in the correctness of the values of this variable.

15. The (generalised) Fellegi-Holt paradigm is the basic assumption that is used in many algorithms and computer programs for automatic edit and imputation. Examples of such computer programmes are DAISY (Barcaroli and Venturi, 1997), GEIS (Kovar and Whitridge, 1990), AGGIES (Todaro, 1999), SPEER (Winkler and Draper, 1997), DISCRETE (Winkler and Petkunas, 1997) and CherryPi (De Waal, 1996).

16. The extension of Fourier elimination to categorical data was used by Fellegi and Holt to construct an algorithm for solving the error localisation problem. In principle, this algorithm can be applied to solve the error localisation problem in both categorical and numerical data. In practice, however, this algorithm appears to be much more applicable to categorical data than to numerical data. DAISY, SPEER and DISCRETE are based on the method proposed by Fellegi and Holt for solving the error localisation problem. GEIS, AGGIES and CherryPi, however, are based on modified versions of Chernikova's algorithm (see e.g. Sande, 1978, and Schiopu-Kratina and Kovar, 1989).

III. NEW ALGORITHMS FOR AUTOMATIC DATA EDITING

17. In 1999 and 2000 Statistics Netherlands has been very actively involved in research on new algorithms for automatic editing. Several algorithms have been developed. It started when Ronan Quere, a post-graduate student from the Eindhoven University of Technology, developed a new algorithm for

automatic editing of numerical data. This algorithm is partly based on ideas from his supervisor, Cor Heurkens, from the same university. It uses Fourier elimination to determine all optimal solutions to the error localisation problem. The ideas for the algorithm arose while studying the litterature on Fourier elimination.

18. Quere (2000) reports results from simulation experiments. In these simulation experiments he compared the results of the new algorithm, and corresponding prototype software, with CherryPi. The new algorithm turned out to be substantially faster. An additional advantage of the new algorithm is that it is easier to understand and implement.

19. The new algorithm for automatic editing of numerical data can be modified in such a way that it becomes suitable for automatic editing of categorical data. That work was done by Jacco Daalmans, a student from Tilburg University. Basically, he uses the same algorithm as Quere, but replaces Fourier elimination by the Fellegi-Holt method to generate implied edits for categorical data. The Fellegi-Holt method to generate implied edits using a certain generating variable is used here as a method to eliminate that categorical variable.

20. Finally, Ronan Quere and Ton de Waal devised a method to handle mixed data and edits, i.e. data and edits involving both categorical and numerical edits. Whenever we refer to *the* new algorithm for automatic editing in this paper we will mean this algorithm. The algorithm combines Fourier elimination to eliminate numerical variables with the Fellegi-Holt method to eliminate categorical variables. The basic version of this algorithm is given below.

21. For categorical data we denote the domain, i.e. set of the possible values, of variable i by D_i. For simplicity, we assume that every edit E^j (j=1,...,J) is written in the following form: a record $(v_1,\ldots,v_m,x_{m+1},\ldots,x_{m+n})$ satisfies edit E^j if and only if the following statement holds true:

$$\text{IF } v_i \in F_i^j \text{ for } i=1,\ldots,m \text{ THEN } (x_{m+1},\ldots,x_{m+n}) \in \{\mathbf{x} \mid a_{1j}x_{m+1}+\ldots+a_{nj}x_{m+n}+b_j \geq 0\}. \qquad (1)$$

22. The part following the IF-statement is called the IF-condition; the part following the THEN-statement the THEN-condition. If the IF-condition is not satisfied, the edit is always considered satisfied irrespective of the values of the numerical variables. If the THEN-condition is the empty set, then we are dealing in fact with a categorical edit. This edit says that the combinations of categorical values that satisfy the IF-condition are not allowed.

23. The format of the edits in Quere and De Waal (2000) is a bit more general than (1), because in that paper equalities are also considered. De Waal (2000) considers even more general edits. The reason for this is hinted at in the paragraph below.

24. For simplicity we will assume in the present paper that no values are missing in the data to be edited. The actual algorithm can handle missing data, however. For more information on handling missing data we refer to Quere and De Waal (2000) and De Waal (2000).

25. The basic idea of the algorithm is that a binary tree is constructed. In each node of this tree a variable is selected that has not yet been selected in any predecessor node. We select a categorical variable only after all numerical variables have been selected. This is done in order to make sure that all edits that are generated by the algorithm have format (1). In De Waal (2000) the numerical variables do not need to be selected before a categorical variable might be selected. The price that has to be paid for this flexibility is that more complicated edits than (1) have to be considered.

26. After a variable has been selected, two branches are constructed: in one branch the selected

variable is fixed to its original value, in the other branch the selected variable is eliminated from the set of edits. In each branch the set of edits is updated. Updating the set of edits is the most important step in the algorithm. How the set of edits has to be updated depends on whether the selected variable was fixed or eliminated, and also on whether this variable was categorical or numerical.

27. Fixing a variable, either numerical or categorical, to its value is easy. We simply substitute this value in all current edits. As a result some edits may always be satisfied, e.g. when a categorical value is fixed to a value such that the IF-condition of an edit can never become true anymore. These edits may be discarded from the new set of edits. Conversely, some edits may become violated. In such a case this branch of the binary tree can never result in a solution to the error localisation problem.

28. Eliminating a variable is a complicated process. If a numerical variable is eliminated, we basically apply Fourier elimination (see Duffin, 1974; Quere, 2000; Quere and De Waal, 2000; De Waal, 2000) to eliminate that variable from the current set of edits. Some care has to be taken in order to ensure that the IF-conditions of the resulting edits are correctly defined. In particular, if we want to eliminate a numerical variable from the current set of edits, we start by copying all edits not involving the selected numerical variable from the current set of edits into the new set of edits. Next, we consider all edits involving the selected numerical variable pair-wise. Suppose we consider the pair of edits *s* and *t*. We start by checking whether the intersections $F_i^s \cap F_i^t$ are not empty for all i=1,…,m. If any of these intersections is empty, we do not have to consider this pair of edits anymore. If all intersections are non-empty, we check whether we can apply Fourier elimination to eliminate the selected numerical variable from the two THEN-conditions. If we cannot, we do not consider this pair of edits anymore. Otherwise, we eliminate the selected numerical variable from the pair of edits, and obtain a new THEN-condition. The intersections $F_i^s \cap F_i^t$ form the IF-condition of a new edit. Combining the new IF-condition with the new THEN-condition, we obtain the new edit.

29. Eliminating a categorical variable is a bit less complicated than eliminating a numerical variable. The reason is that first we have fixed or eliminated all numerical variables. As a result no numerical variables are left in any of the current edits - that is, the edits are purely categorical ones. To eliminate a categorical variable from a set of categorical edits, we basically apply the Fellegi-Holt technique (see Fellegi-Holt, 1976; Daalmans, 2000; Quere and De Waal, 2000; De Waal 2000) to eliminate that variable from the current set of edits.

30. Once all numerical and categorical variables have either been fixed or eliminated, i.e. once we are in a terminal node of the tree, we are left with a set of inequalities involving only numbers. This set of inequalities may either be consistent or inconsistent. In case the set of inequalities is consistent, the variables that have been eliminated in order to reach that terminal node, can be imputed in such a way that their new values, together with the original values of the variables that have been fixed, satisfy the original set of edits. Conversely, if the set of inequalities is inconsistent, the variables that have been eliminated cannot be imputed in such a way that their new values, together with the original values of the variables that have been fixed, satisfy the original set of edits. De Waal (2000) gives proof of the above statement.

31. Hence, in a terminal node of the tree we can immediately see whether we can impute a certain set of variables, the variables that have been eliminated to reach a certain terminal node, such that all (original) edits can be satisfied or not. Because we, in principle, have to eliminate all possible subsets of variables to reach all terminal nodes of the tree, our tree in fact determines all sets of variables that can be imputed in such a way that the (original) set of edits can be satisfied. From these sets of variables we select those with the lowest sum of confidence weights, and obtain all solutions to the error localisation problem.

32. One might raise the objection that far too many different subsets of variables have to be eliminated, i.e. that there are far too many terminal nodes in the tree, even for moderately sized problems.

The number of subsets that have to be considered can indeed be large, but the situation is not nearly as bad as it may seem at first sight. Firstly, we notice that if the minimum number of fields that have to be changed is too large, the record should not be corrected automatically anyway. In such a case, the quality of the information in the record is simply too low. Such a record should be corrected manually, or the record should be completely discarded. By demanding that the minimum number of changes should be less than, say, 5, the number of terminal nodes in the tree is reduced drastically.

33. Secondly, some branches of the tree do not have to be considered. Often it is already clear in an intermediate node that none of the terminal nodes that can be generated from this node corresponds to an optimal solution to the error localisation problem. Again, this observation leads to a drastic reduction of the number of terminal nodes that have to be considered.

34. The developed algorithm is planned to become part of SLICE, the general software package for editing and imputation that is currently being developed at Statistics Netherlands (see De Waal, 2000). The algorithm for automatic editing only sets a number of suspicious values to missing. After termination of the algorithm the missing data remain to be imputed. In the next section, we discuss recent developments in the field of automatic imputation at Statistics Netherlands.

IV. A NEW COMPUTER PROGRAM FOR AUTOMATIC IMPUTATION

35. In 1999 the so-called AUTIMP project started. AUTIMP is a European project that is partly funded by the 4th Framework Programme of the European Commission. It is planned to last till November 2000. Institutes participating in the project are University of Southampton, Office for National Statistics (UK), Statistics Finland, Instituto Nacional de Estatística de Portugal and Statistics Netherlands. The aims of AUTIMP are the evaluation of software that can be used to impute for missing data (see e.g. Mesa, Tsai and Chambers, 2000; Hoogland and Pannekoek, 2000), and the development of advanced prototype imputation software.

36. The imputation software that has been developed is based on automatic interaction detection (AID) trees, cf. Sonquist, Baker and Morgan (1971). Because the developed algorithm gives lower weights to outliers while constructing the tree, the technique is referred to as weighted automatic interaction detection (WAID). This algorithm and the corresponding software to construct WAID-trees have been developed by University of Southampton.

37. A WAID-tree, or generally a tree-based model, classifies the data in terms of the values of a set of categorical predictor variables. It is a binary tree that is generated by successively splitting a "training" data set into smaller subsets. These subsets are increasingly more homogeneous with respect to a selected response variable. This response variable may be either categorical or numerical. The process of recursively splitting the data set into two subsets continues until a stopping criterion is met. The terminal nodes in this tree form homogeneous clusters.

38. The developed imputation software can impute for missing values of both categorical and numerical variables. To use the WAID methodology to impute for missing values in a data set, the software first determines missing data patterns in this data set. For each of these missing data patterns, or parts of missing data patterns, the user has to select a set of categorical predictor variables.

39. Next, for each (part of a) missing data pattern, WAID-trees are grown using a complete "training" data set. This "training" data set may be the subset of complete records of the data set to be imputed, but it may also be a different data set. The terminal nodes of the generated WAID-trees form clusters of records that are as homogeneous as possible with respect to the variables involved in this (part of a) missing data pattern. These homogeneous clusters themselves are, however, not used by the computer program. Only the classification rules that define these homogenous clusters are used. In this way, we

can use a “training” data set to determine the classification rules, and later use another data set with donor records to actually impute for missing values.

40. After generation of the WAID-trees, and hence generation of the classification rules for constructing homogeneous clusters of records, the data set with missing values is again supplied to the computer program. We also supply a data set with donor records to the computer programme. This data set may be the same as the data set to be imputed, but it may also be a different data set. In the data set with donor records we apply the generated classification rules to construct homogeneous clusters of donor records.

41. To impute for missing values in a certain record, we determine which WAID-trees correspond to the missing data pattern of this record. More than one WAID-tree, and hence more than one homogeneous cluster, may correspond to a particular record, because separate WAID-trees may have been generated for different parts of its missing data pattern. Subsequently, we determine the homogeneous clusters corresponding to this record by using the classification rules to classify the values of the predictor variables. The records in the data set with donor records corresponding to those clusters are used to impute for the missing data in the record under consideration.

42. The imputation software supports several imputation methods, e.g., the nearest neighbour or a random donor record may be selected from a homogeneous cluster. The developed prototype imputation software is a stand-alone programme, and can be used under Windows 95/98 and Windows NT. At Statistics Netherlands the developed software is planned to become a module in SLICE (see De Waal, 2000).

V. COMBINING AUTOMATIC EDITING AND IMPUTATION

43. As we have already mentioned, the developed algorithm for automatic editing only sets a number of suspicious values to missing. Subsequently, the missing values, either originally missing or missing due to the editing process, are imputed. During the imputation phase the edit rules are not taken into account. As a result, after having solved the error localisation problem and having imputed the missing values, a resulting record may still violate the edits. This clearly is an undesirable situation because after having solved the error localisation problem, we know for sure that all edits can be satisfied by imputing the missing values in an appropriate manner.

44. A simple procedure to ensure that imputed values satisfy the edits is proposed by Fellegi and Holt (1976). The underlying idea is to eliminate all variables for which a value should be imputed from the set of edits. Provided these variables can indeed be imputed in a consistent manner, which is guaranteed if we first solve the error localisation problem, we can one by one determine feasible values for the fields to be imputed. In particular, suppose k fields have to be imputed. Of these k variables we then eliminate the first k-1 from the set of (explicit) edits. This gives a set of feasible values for the k-th field. One of these values is selected and imputed for the k-th field. For the remaining k-1 fields that have to be imputed we apply the same procedure. Fellegi and Holt (1976) show that this procedure always leads to a consistent record, provided the variables to be imputed can indeed be imputed in a consistent manner. A drawback of this procedure is that the statistical properties of the actual data may not be preserved. The procedure proposed by Fellegi and Holt may serve as a benchmark for an alternative algorithm, however.

45. At the time of writing this report, Widya Kartika, a post-graduate student from the Delft University of Technology just started developing such an alternative algorithm. We assume that first the error localisation problem has been solved to determine which fields have to be imputed and that, subsequently, these fields have been imputed before the algorithm will be applied. The algorithm that is being developed aims to modify the imputed values in such a way that the resultant record resembles the imputed record as close as possible while satisfying the edits. We aim to keep the values of the final

record close to the values of the imputed record, because we assume that the missing values have been imputed in such a way that the statistical properties of the actual data are mimicked as well as possible.

VI. AUTOMATIC EDIT AND IMPUTATION IN THE EDITING PROCESS

46. From the above discussion one might get the impression that Statistics Netherlands intends to rely solely on automatic edit and imputation for correction of its statistical data. In this final section we want to take the opportunity to make clear that this is definitely not the case. We consider automatic edit and imputation as a step in the editing process, not as the entire editing process (see also De Waal, Renssen and Van de Pol, 2000).

47. In the author's view, the ideal edit strategy is a combination of selective editing, automatic editing and (graphical) macro-editing. After data entry, simple checks and simple (semi-automatic) corrections should be applied. Examples of simple checks are range checks, examples of records to which simple corrections can be applied are cases in which it is clear that a respondent filled in a financial figure in guilders instead of the requested thousands of guilders. After that phase, selective editing should be applied to split the data into two streams: the critical stream and the non-critical stream. The critical stream consists of those records that are the most likely ones to contain influential errors; the non-critical stream consists of records that are unlikely to contain influential errors. The records in the critical stream, the critical records, are edited in a traditional computer-assisted manner. The records in the non-critical stream, the non-critical records, are not edited or are edited automatically.

48. The latter approach, editing the records in the non-critical stream automatically, clearly has our preference. The sum of the errors in the non-critical records may have an influential impact on the publication figures, although each error itself is non-influential. Moreover, many non-critical records will be internally inconsistent. This may lead to problems when publication figures are calculated. Automatic editing helps to reduce the errors in the data, and makes sure that the records become internally consistent.

49. Macro-editing is viewed as a final check just before publication. It cannot be missed, because it reveals errors that will go unnoticed with selective editing or automatic editing.

50. In the author's opinion the combined use of selective editing, automatic editing and (graphical) macro-editing is an efficient and effective way of cleaning data. In comparison with the traditional computer-assisted approach, the quality of the data can be maintained, while the resources needed to clean the data are substantially reduced and the timeliness of publication of the statistical data is clearly improved.

References

Barcaroli, G. and M. Venturi, 1997, DAISY (Design, Analysis and Imputation System): structure, methodology and first applications. *Statistical Data Editing (Volume 2)*; *Methods and Techniques*. United Nations.

Chernikova, 1964, Algorithm for finding a general formula for the non-negative solutions of a system of linear equations. *USSR Computational Mathematics and Mathematical Physics*, **4**, 151-158.

Chernikova, 1965, Algorithm for finding a general formula for the non-negative solutions of a system of linear inequalities. *USSR Computational Mathematics and Mathematical Physics*, **5**, 228-233.

Daalmans, J., 2000, Automatic error localisation of categorical data (working title). Report, Statistics Netherlands, Voorburg.

De Waal, T., 1996, CherryPi: A computer program for automatic edit and imputation. Paper presented at the UN Work Session on Statistical Data Editing, 4-7 November 1996, Voorburg.

De Waal, T., 2000, SLICE: Generalised software for statistical data editing and imputation. Report, Statistics Netherlands, Voorburg.

De Waal, T., 2000, An optimality proof of Statistics Netherlands' new algorithm for automatic editing of mixed data (working title). Report, Statistics Netherlands, Voorburg.

De Waal, T., R. Renssen and F. Van de Pol, 2000, Graphical macro-editing: possibilities and pitfalls. Paper presented at the Second International Conference on Establishment Surveys, 17-21 June 2000, Buffalo.

Dines, L.L., 1919, Systems of linear inequalities. *Annals of Mathematics*, **20**, 191-199.

Duffin, R.J., 1974, On Fourier's analysis of linear inequality systems. *Mathematical Programming Studies*.

Fellegi, I.P. and D. Holt, 1976, A systematic approach to automatic edit and imputation. *Journal of the American Statistical Association*, **71**, 17-35.

Fourier, J.B.J., 1826, Solution d'une question particulière du calcul des inégalités. In: *Oeuvres II*, Paris.

Hoogland, J. and J. Pannekoek, 2000, Evaluation of SPSS Missing Value Analysis 7.5. Report, Statistics Netherlands, Voorburg.

Kohler, D.A., 1973, Translation of a report by Fourier on his work on linear inequalities. *Opsearch*, **10**, 38-42.

Kovar, J. and P. Whitridge, 1990, Generalized Edit and Imputation System: overview and applications. *Revista Brasileira de Estadistica*, **51**, 85-100.

Mesa, D.M., P. Tsai and R.L. Chambers, 2000, Using tree-based models for missing data imputation: an evaluation using UK census data.

Motzkin, T.S., 1936, Beiträge zur Theorie der linearen Ungleichungen. Dissertation, University of Basel.

Quere, R., 2000, Automatic editing of numerical data. Report, Statistics Netherlands, Voorburg.

Quere, R. and T. De Waal, 2000, Error localization in mixed data sets, Report, Statistics Netherlands, Voorburg.

Sande, G., 1978, An algorithm for the fields to impute problems of numerical and coded data. Technical report, Statistics Canada.

Schiopu-Kratina, I. and J.G. Kovar, 1989, Use of Chernikova's algorithm in the generalized edit and imputation system. Methodology Branch Working Paper BSMD 89-001E, Statistics Canada.

Sonquist, J.N., E.L. Baker and J.A. Morgan, 1971, Searching for structure. Institute for Social Research, University of Michigan.

Todaro, T.A., 1999, Overview and evaluation of the AGGIES automated edit and imputation system. Paper presented at the UN/ECE Work Session on Statistical Data Editing, 2-4 June 1999, Rome.

Winkler, W.E. and L.A. Draper, 1997, The SPEER edit system. *Statistical Data Editing (Volume 2)*; *Methods and Techniques*. United Nations.

Winkler, W.E. and T.F. Petkunas, 1997, The DISCRETE edit system. *Statistical Data Editing (Volume 2)*; *Methods and Techniques*. United Nations.