Statistics Netherlands

Discussion Paper

Error Localisation using General Edit Operations

2014 | 14

Sander Scholtus

16-5-2014

Summary: The aim of automatic editing is to use a computer to detect and amend erroneous values in a data set, without human intervention. Most automatic editing methods that are currently used in official statistics are based on the seminal work of Fellegi and Holt (1976). In the Fellegi-Holt approach to automatic editing, the amendment of individual values plays a central role. On the other hand, human editors frequently perform more complex edit operations that involve simultaneous changes in several values; for example, they might interchange or transfer reported amounts between variables. It is difficult – in some cases even impossible – to model these complex operations under the Fellegi-Holt paradigm. Consequently, applications in practice have shown systematic differences between data that are edited manually and automatically.

In this paper, a generalisation of the Fellegi-Holt paradigm is proposed that can incorporate various complex edit operations in a natural way. In addition, an algorithm is outlined that may be used to solve the resulting generalised error localisation problem, at least in theory. It is hoped that this generalisation may be used to increase the suitability of automatic editing in practice, and hence to improve the efficiency of data editing processes.

1 Introduction

Data that have been collected for the production of statistics inevitably contain errors. A data editing process is needed to detect and amend these errors, at least in so far as they have an appreciable impact on the quality of the statistical output (Granquist and Kovar, 1997; De Waal et al., 2011). Traditionally, data editing has been a manual task, performed by human editors with extensive subject-matter knowledge. To improve the efficiency and timeliness of editing, attempts have been made to automate parts of this process. This has resulted in, on the one hand, deductive correction methods for systematic errors (e.g., unit of measurement errors, where values are reported in the wrong unit base) and, on the other hand, error localisation algorithms for random errors (De Waal et al., 2011). In this paper, I will focus mainly on the latter type of automatic editing. This usually involves minimally adjusting each record of data, according to some pre-specified optimisation criterion, so that it becomes consistent with a given set of rules known as *edit rules* (or *edits* for short). Depending on the effectiveness of the optimisation criterion and the strength of the edit rules, automatic editing may be used as a partial alternative to traditional manual editing.

Most automatic editing methods that are currently used in official statistics are based on the paradigm of Fellegi and Holt (1976). According to this paradigm, the error localisation problem is solved by finding, for each record, the smallest subset of variables that can be imputed so that the record becomes consistent with the edits. A slight generalisation is obtained by assigning so-called *confidence weights* to the variables and minimising the total weight of the imputed variables; variables with higher confidence weights are then assumed less likely to contain errors. Having obtained a solution to the error localisation problem, one needs to find suitable values to impute for the variables that have been identified as erroneous. This is a separate problem, known as the consistent imputation problem [see, e.g., De Waal et al. (2011) and their references]. In this paper, I will focus on the error localisation problem.

In practice, one usually applies automatic editing only to records that are expected to contain errors with a small to moderate influence on the target estimates. Influential errors are still treated manually (selective editing). Moreover, the outcome of manual editing is usually taken as the "gold standard" for assessing the quality of automatic editing. A critical evaluation of this assumption is beyond the scope of the present paper. Here I simply note that, by improving the ability of automatic editing methods to mimic the results of manual editing, their usefulness in practice may be increased. In turn, this means that the share of automatic editing may be increased to improve the efficiency of the data editing process (Pannekoek et al., 2013).

Some years ago, Statistics Netherlands conducted a series of evaluation studies in which data sets from the Dutch Structural Business Statistics (SBS) were edited both automatically and manually. When the results of the two editing efforts were compared, a number of systematic differences were found. Many of these differences could be explained by the fact that human editors performed certain types of adjustments that do not fit well within the constraints of the Fellegi-Holt paradigm. For instance, Bikker (2003b) mentioned cases where editors interchanged the value of *costs of type A* with that of *revenues of type A*. This type of adjustment corresponds to one underlying error: the respondent mixed up the answers to two related questions. However, under the Fellegi-Holt paradigm, this adjustment requires two independent imputations. In addition, the error of interchanging costs and revenues of the

same type often caused an edit failure that could also be solved by adjusting the value of *balance of type A*, the relevant edit rule being:

balance of type A = revenues of type A − costs of type A.

During automatic editing, this alternative solution was usually preferred because it requires only one imputation. Subject-matter specialists preferred interchanging the costs and revenues, based on their knowledge of respondent behaviour.

As another example, editors sometimes transferred (parts of) reported amounts between variables. For instance, Daalmans et al. (2011) found that editors of the SBS on wholesale sometimes transferred a part of the reported *turnover from retail trade* to *turnover from wholesale* when the original amounts did not match the reported stocks of retail goods and wholesale goods. Again, this is a complex type of adjustment involving at least two variables that nonetheless is considered as a single correction by the editors.

To some extent, systematic differences between automatic and manual editing can be prevented by a clever choice of confidence weights. For example, Bikker (2003a) derived new sets of weights that could be used to improve the quality of automatic editing for the Dutch SBS in some particular cases. In general, however, it is difficult to predict the effects that a certain modification of the confidence weights would have on the results of automatic editing. Moreover, if editors apply many different complex adjustments, it might be impossible to model all of them under the Fellegi-Holt paradigm using a single set of confidence weights.

Another option is to try to catch errors for which the Fellegi-Holt paradigm is known to provide an unsatisfactory solution at an earlier stage, during deductive correction of systematic errors through automatic correction rules (De Waal et al., 2011). Ideally, this would ensure that Fellegi-Holt-based editing is applied only to those errors for which it is suited. There are some practical limitations to this approach, however. Properly designing a large collection of automatic correction rules, and maintaining such a collection over time, can be a difficult task. For instance, the same set of rules applied to the same record may produce a different outcome depending on the way the rules are ordered. Moreover, it is not self-evident that appropriate correction rules can be found for all errors that do not fit within the Fellegi-Holt paradigm.

In this paper, a different approach is suggested. A new definition of the error localisation problem is proposed which allows the possibility that errors affect more than one variable at a time. It is shown that this problem contains error localisation under the original Fellegi-Holt paradigm as a special case. Informally speaking, under the new paradigm errors are located by minimising the number of so-called *edit operations*. Imputing a new value for one variable at a time is an example of an edit operation. However, more general edit operations can also be allowed that involve changes to multiple variables. For now, I restrict attention to numerical data and linear edits.

The rest of this paper is organised as follows. In Section 2, the concept of an edit operation as it will be used here is formally introduced and illustrated. The new error localisation problem is formulated in Section 3. Section 4 reviews some existing results that are often applied to solve the original Fellegi-Holt-based error localisation problem. In Sections 5 and 6, these results are extended and a possible algorithm for solving the new problem is outlined. Section 7 contains a small example to illustrate the algorithm. On a side note, Section 8 discusses the statistical interpretation of the error localisation problem. A small simulation study is discussed in Section 9. Finally, some conclusions and questions for further research follow in Section 10.

2 Edit operations

Let $\boldsymbol{x} = (x_1, \ldots, x_p)' \in \mathbb{R}^p$ be a record of p numerical variables. Suppose that this record has to satisfy k edit rules, in the form of the following system of linear (in)equalities:

$$\boldsymbol{A}\boldsymbol{x} + \boldsymbol{b} \odot \boldsymbol{0}, \tag{1}$$

where $\boldsymbol{A} = (a_{rj})$ is a $k \times p$ matrix of coefficients and $\boldsymbol{b} = (b_1, \ldots, b_k)'$ is a k vector of constants. Throughout this paper, $\boldsymbol{0}$ will be used to represent a vector of zeros of appropriate length; similarly, $\odot$ will represent a symbolic vector of operators from the set $\{\geq, =\}$.

I now define an edit operation g to be a linear function of $\mathbb{R}^p$ to itself, having the general form

$$g(\boldsymbol{x}) = \boldsymbol{T}\boldsymbol{x} + \boldsymbol{h}, \tag{2}$$

where $\boldsymbol{T} = (t_{ij})$ denotes a known $p \times p$ coefficient matrix and $\boldsymbol{h} = (h_1, \ldots, h_p)'$ denotes a p vector. Importantly, the elements h_i may be either known constants or linear functions of free parameters. The unique free parameters (if any) that occur in $\boldsymbol{h}$ are denoted by $\alpha_1, \ldots, \alpha_m$, or by α if there is only one. In some cases, it may be useful to impose one or several linear constraints on these parameters:

$$\boldsymbol{R}\boldsymbol{\alpha} + \boldsymbol{d} \odot \boldsymbol{0}, \tag{3}$$

with $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_m)'$, $\boldsymbol{R}$ a known matrix, and $\boldsymbol{d}$ a known vector of constants.

In automatic editing based on the Fellegi-Holt paradigm, a central role is attached to the replacement of one original value by an arbitrary new value (imputation). This is in fact a particular edit operation of the form (2) which I will call an *FH operation*. To find the FH operation that imputes the variable x_j, one takes $\boldsymbol{T}$ to be a diagonal matrix with $t_{jj} = 0$ and all other diagonal elements equal to one; in addition, all elements of $\boldsymbol{h}$ except h_j are zero, while $h_j = \alpha$, with α an unrestricted parameter. The resulting FH operation yields:

$$g\left((x_1, \ldots, x_{j-1}, x_j, x_{j+1}, \ldots, x_p)'\right) = (x_1, \ldots, x_{j-1}, \alpha, x_{j+1}, \ldots, x_p)', \tag{4}$$

with $\alpha \in \mathbb{R}$ representing the imputed value. It should be noted that for a record of p variables, p distinct FH operations can be defined. No restrictions of the form (3) are imposed directly on α; however, in the context of the error localisation problem, an FH operation is of interest only if there exists a value for α that can help to make the record consistent with the edits (1).

Example. To illustrate the concept of an edit operation, some further examples will now be given. For notational convenience, I restrict attention to the case $p = 3$.

- An edit operation that changes the sign of one of the variables:
$$g\left(\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}\right) = \begin{pmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} -x_1 \\ x_2 \\ x_3 \end{pmatrix}.$$
- An edit operation that interchanges the values of two adjacent items:
$$g\left(\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}\right) = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} x_2 \\ x_1 \\ x_3 \end{pmatrix}.$$
- An edit operation that transfers an amount between two items, where the amount transferred may equal at most K units in either direction:
$$g\left(\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}\right) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} \alpha \\ 0 \\ -\alpha \end{pmatrix} = \begin{pmatrix} x_1 + \alpha \\ x_2 \\ x_3 - \alpha \end{pmatrix},$$

with the constraint that $-K \leq \alpha \leq K$.

– An edit operation that computes the value of a total from the values of its parts:

$$g\left(\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}\right) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \\ x_1 + x_2 \end{pmatrix}.$$

– An edit operation that imputes two variables simultaneously using a fixed ratio:

$$g\left(\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}\right) = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} + \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ 0 \end{pmatrix} = \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ x_3 \end{pmatrix},$$

with the constraint that $\boldsymbol{\alpha} = (\alpha_1, \alpha_2)'$ satisfies $10\alpha_1 - \alpha_2 = 0$.

□

These examples illustrate in particular that free parameters can be a powerful tool to avoid the need to construct a separate edit operation for each possible action. For instance, an FH operation does not specify the exact value that is imputed. Similarly, in the third example above, the exact amount that is transferred between x_1 and x_3 is not specified.

Intuitively, an edit operation is supposed to "reverse the effects" of a particular type of error[1)] that occurred in the observed data. That is to say, if the error associated with edit operation g actually occurred in the observed record $\boldsymbol{x}$, then $g(\boldsymbol{x})$ is the record that would have been observed if that error had not occurred. Somewhat more formally, it is assumed here that errors occurring in the data can be modeled by a stochastic "error generating process" $\mathcal{E}$, and that each edit operation acts as a "corrector" for one particular error that can occur under $\mathcal{E}$. (A more precise definition of $\mathcal{E}$ will be given in Section 8.) Furthermore, if the edit operation g contains free parameters, the record $g(\boldsymbol{x})$ might not be determined uniquely even when the restrictions (1) and (3) are taken into account. As mentioned above, for an FH operation the value of α corresponds exactly to an imputed value for one of the variables in $\boldsymbol{x}$. More generally, one may have to "impute" values for the free parameters that occur in an edit operation, which in turn means that some of the variables in $\boldsymbol{x}$ are imputed via the linear transformation given by (2). As in traditional Fellegi-Holt-based editing, finding appropriate imputations for the free parameters will not be considered part of the error localisation problem here. On the other hand, if g does not contain any free parameters, the imputed values in $g(\boldsymbol{x})$ follow directly from the edit operation itself and the distinction between error localisation and imputation is blurred.

It should be clear that one could construct an abundance of edit operations of the form (2). In any particular application, only a small subset of these potential operations would have a substantively meaningful interpretation (in the sense that the associated types of errors are known to occur in that application). In what follows, I assume that a finite set of specific edit operations of the form (2) has been identified as relevant for a particular application. This will be called the set of *allowed edit operations* for that application. Some suggestions on how to construct this set will be given in Section 10.

[1)] Note that an error is defined in this paper as any type of disturbance that may occur in the observed data. This disturbance may be multivariate; for instance, interchanging the values of two variables can be considered an error. Hence, an error is not the same thing as an erroneous value; it is a more general concept. It is also important to distinguish between errors and edit failures. The latter can be *caused* by errors – and hence used to recognise that errors occurred in the observed data – but they are not errors as such.

3 A generalised error localisation problem

Consider a set of allowed edit operations for a given application of automatic editing. Informally, I propose to generalise the error localisation problem of Fellegi and Holt (1976) by replacing "the smallest subset of variables that can be imputed to make the record consistent" with "the shortest sequence of allowed edit operations that can be applied to make the record consistent". To give a formal definition of this generalised error localisation problem, some new notation and concepts need to be introduced first.

Let $\mathcal{G}$ be a finite set of allowed edit operations. To each edit operation $g \in \mathcal{G}$, a weight $w_g > 0$ can be associated that expresses the costs of applying edit operation g. These weights may be seen as a generalisation of the confidence weights that were mentioned in Section 1, by identifying the weight of the FH operation that imputes a new value for x_j with the confidence weight of that variable.

Consider a sequence of points $\boldsymbol{x} = \boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_t = \boldsymbol{y}$ in $\mathbb{R}^p$. A *path* from $\boldsymbol{x}$ to $\boldsymbol{y}$ is defined as a sequence of *distinct* edit operations $g_1, \ldots, g_t \in \mathcal{G}$ such that $\boldsymbol{x}_n = g_n(\boldsymbol{x}_{n-1})$ for all $n \in \{1, \ldots, t\}$. Note that it is not allowed to use the same edit operation twice on the same path, not even with a different choice of parameter(s). A path is denoted by $P = [g_1, \ldots, g_t]$. The set of all possible paths from $\boldsymbol{x}$ to $\boldsymbol{y}$ is denoted by $\mathcal{P}(\boldsymbol{x}, \boldsymbol{y})$. This set may be empty.

The *length* of a path $P = [g_1, \ldots, g_t]$ is defined as the sum of the weights of its constituent edit operations:

$$\ell(P) = \sum_{n=1}^{t} w_{g_n},$$

where, by convention, the empty path has length zero. Note that two paths have the same length if they consist of the same edit operations $\{g_1, \ldots, g_t\} \subseteq \mathcal{G}$, regardless of the order.

Next, the *distance* from $\boldsymbol{x}$ to $\boldsymbol{y}$ is defined as the length of the shortest path that connects $\boldsymbol{x}$ to $\boldsymbol{y}$. If no such path exists, then this distance is considered to be infinitely large:

$$d(\boldsymbol{x}, \boldsymbol{y}) = \begin{cases} \min\{\ell(P) \mid P \in \mathcal{P}(\boldsymbol{x}, \boldsymbol{y})\} & \text{if } \mathcal{P}(\boldsymbol{x}, \boldsymbol{y}) \neq \emptyset, \\ \infty & \text{otherwise.} \end{cases}$$

It is not difficult to check that $d(\boldsymbol{x}, \boldsymbol{y})$ satisfies the standard axioms of a metric *except* that it need not be symmetric: in general, it can happen that $d(\boldsymbol{x}, \boldsymbol{y}) \neq d(\boldsymbol{y}, \boldsymbol{x})$. This depends on the choice of $\mathcal{G}$. In fact, some edit operations can be considered symmetric in the sense that $g(\boldsymbol{x}) = \boldsymbol{y}$ if, and only if, $g(\boldsymbol{y}) = \boldsymbol{x}$.[2] Examples include the FH operation and the first three examples in Section 2. That this property is not shared by all edit operations may be seen from the last two examples in Section 2. Now, in the special case that $\mathcal{G}$ contains only symmetric edit operations, it follows automatically that $d(\boldsymbol{x}, \boldsymbol{y}) = d(\boldsymbol{y}, \boldsymbol{x})$ and hence that d is a true metric. When $\mathcal{G}$ also contains edit operations that are not symmetric, it need not hold that $d(\boldsymbol{x}, \boldsymbol{y}) = d(\boldsymbol{y}, \boldsymbol{x})$ for all $\boldsymbol{x}$ and $\boldsymbol{y}$. In that case, d is a so-called "quasimetric". Accordingly, I will refer to $d(\boldsymbol{x}, \boldsymbol{y})$ as "the distance from $\boldsymbol{x}$ to $\boldsymbol{y}$" rather than "the distance between $\boldsymbol{x}$ and $\boldsymbol{y}$".

[2] In the case that g contains free parameters, one should take "$g(\boldsymbol{x}) = \boldsymbol{y}$" to mean "there exist feasible parameter values for which g maps $\boldsymbol{x}$ to $\boldsymbol{y}$".

Let D be a closed, non-empty subset of $\mathbb{R}^p$. The distance from any point $\boldsymbol{x}$ to D is defined to be the distance from $\boldsymbol{x}$ to the nearest point $\boldsymbol{y} \in D$:

$$d(\boldsymbol{x}, D) = \min\{d(\boldsymbol{x}, \boldsymbol{y}) \mid \boldsymbol{y} \in D\}.$$

One subset of $\mathbb{R}^p$ that is of particular interest here is the set of all points that satisfy the edits (1); denote this set by D_0. Provided that the system of edits is feasible (as it should be in practice), it holds that $D_0 \neq \emptyset$. Moreover, D_0 is closed because (1) does not contain any strict inequalities.

I can now formulate the generalised error localisation problem. Consider a given set of consistent records D_0 [defined implicitly by a system of edits (1)], a given set of allowed edit operations $\mathcal{G}$, and a given record $\boldsymbol{x}$. If $d(\boldsymbol{x}, D_0) = \infty$, then the error localisation problem for $\boldsymbol{x}$ is infeasible. Otherwise, any record $\boldsymbol{y} \in D_0$ such that $d(\boldsymbol{x}, \boldsymbol{y}) < \infty$ is called a *feasible solution* to the error localisation problem for $\boldsymbol{x}$. A feasible solution $\boldsymbol{x}^*$ is called *optimal* if it holds that

$$d(\boldsymbol{x}, \boldsymbol{x}^*) = d(\boldsymbol{x}, D_0). \tag{5}$$

Formally, then, the generalised error localisation problem consists of finding an $\boldsymbol{x}^* \in D_0$ that satisfies expression (5). Note that this problem is trivial for records that are already consistent with the edits: if $\boldsymbol{x} \in D_0$, the unique optimal solution is given by $\boldsymbol{x}^* = \boldsymbol{x}$.

It should be noted that if $\boldsymbol{x}^*$ is an optimal solution to the error localisation problem for $\boldsymbol{x}$, any other record in D_0 that can be reached by the same path of edit operations is also an optimal solution. To solve the error localisation problem, it is sufficient to find an optimal path of edit operations. Constructing an associated record $\boldsymbol{x}^* \in D_0$ may then be regarded as a generalisation of the consistent imputation problem; cf. the discussion on imputation at the end of Section 2. Consequently, in the rest of this paper I will consider the error localisation problem mainly in terms of paths of edit operations. This has the advantage that it reduces the size of the search space from the – possibly uncountable – number of distinct consistent records to the – finite – number of distinct paths.

So far, no assumption has been made about the set of allowed edit operations, other than that this set should be finite. In general, the above error localisation problem might be infeasible for some records $\boldsymbol{x}$. This would happen whenever $\boldsymbol{x}$ cannot be mapped onto D_0 by any combination of distinct edit operations in $\mathcal{G}$. To avoid this situation, $\mathcal{G}$ should be sufficiently large so that $d(\boldsymbol{x}, D_0) < \infty$ for all $\boldsymbol{x} \in \mathbb{R}^p$. It can be shown that this property holds in particular for any $\mathcal{G}$ that contains all p distinct FH operations. In fact, it is possible to connect any point in $\mathbb{R}^p$ to any other point in $\mathbb{R}^p$ by a path that concatenates the FH operations associated with the coordinates on which the two points differ. For simplicity, I assume in the rest of this paper that $\mathcal{G}$ contains all FH operations. In principle, though, one could also construct other sets of allowed edit operations for which the error localisation problem is always feasible.

The special case that $\mathcal{G}$ consists *only* of the p distinct FH operations is of some interest. It is not difficult to see that the above error localisation problem then reduces to the original problem of Fellegi and Holt (1976), albeit with confidence weights. The next section reviews some existing results for this special case.

4 Error localisation using only FH operations

If FH operations are the only ones allowed, the error localisation problem of Section 3 amounts to finding the minimum of the expression

$$\sum_{j=1}^{p} w_j \delta_j, \tag{6}$$

with w_j the confidence weight of variable x_j and $\delta_j \in \{0, 1\}$, under the condition that the original record can be made consistent with the edits by imputing only the variables x_j for which $\delta_j = 1$. This yields the error localisation problem of Fellegi and Holt (1976) in a more familiar form; cf. De Waal et al. (2011).

To solve this problem, various dedicated algorithms have been developed by, among others, Fellegi and Holt (1976), Schaffer (1987), Garfinkel et al. (1988), Kovar and Whitridge (1990), Ragsdale and McKeown (1996), De Waal (2003), De Waal and Quere (2003), Riera-Ledesma and Salazar-González (2003), Bruni (2004), and De Jonge and Van der Loo (2014). See also De Waal et al. (2011) for an overview. Many of these algorithms make use of a mathematical technique called *Fourier-Motzkin elimination* (FM elimination). FM elimination transforms a system of linear constraints having p variables into a system of *implied* linear constraints having at most $p - 1$ variables; thus, at least one of the original variables is eliminated from the constraints. For mathematical details, see Appendix I.

FM elimination has the following fundamental property: the system of implied constraints is satisfied by the values of the non-eliminated variables if, and only if, there exists a value for the eliminated variable that, together with the other values, satisfies the original system of constraints. In error localisation under the Fellegi-Holt paradigm, this property can be used to assess whether a given combination of variables can be imputed to obtain a record that is consistent with the edits. Clear illustrations of this use of FM elimination are provided by the error localisation algorithms of De Waal and Quere (2003) and Fellegi and Holt (1976) themselves. It should be noted that FM elimination also has other applications outside the context of error localisation (Williams, 1986).

Example. Consider three numerical variables that should satisfy the following linear edit rules:

$$x_1 + x_2 + x_3 = 20, \tag{7}$$
$$x_1 - x_2 \geq 3, \tag{8}$$
$$-x_1 + x_2 \geq -6, \tag{9}$$
$$-x_1 + x_3 \geq 5, \tag{10}$$
$$x_1 - x_3 \geq -10, \tag{11}$$
$$x_1 \geq 0, \tag{12}$$
$$x_2 \geq 0, \tag{13}$$
$$x_3 \geq 0. \tag{14}$$

The record $\boldsymbol{x} = (x_1, x_2, x_3)' = (10, 1, -3)'$ fails some of these rules and hence requires editing. Suppose that one would like to solve the error localisation problem for this record using only the three FH operations. As an additional constraint, suppose that the maximal number of variables

to impute is $M = 2$. Using any of the relevant algorithms discussed in De Waal et al. (2011), it can be shown that this error localisation problem has just one feasible solution: impute x_1 and x_3. For the sake of brevity, I will not give a full derivation of this result here. However, I will show that imputing x_1 and x_3 is indeed a feasible solution for this record by using FM elimination.

First of all, since x_2 is not imputed here, the original value $x_2 = 1$ may be substituted in (7)–(14) to obtain a reduced system of restrictions for x_1 and x_3:

$$x_1 + x_3 = 19, \tag{15}$$
$$x_1 \geq 4, \tag{16}$$
$$-x_1 \geq -7, \tag{17}$$
$$-x_1 + x_3 \geq 5, \tag{18}$$
$$x_1 - x_3 \geq -10, \tag{19}$$
$$x_3 \geq 0. \tag{20}$$

FM elimination of x_1 from these edits yields, after some simplification, the following implied constraints for x_3:

$$x_3 \geq 12, \tag{21}$$
$$-2x_3 \geq -29. \tag{22}$$

FM elimination of x_3 from (21) and (22) yields the trivial true statement $0 \geq -5$. The fact that this statement is true implies, by the fundamental property of FM elimination, that there exists a value for x_3 that satisfies the edits (21) and (22). (It is not difficult to see that this is indeed the case.) By another application of the fundamental property, this in turn implies that there exist values for x_1 and x_3 that satisfy the edits (15)–(20). Hence, it follows that imputing x_1 and x_3 is a feasible solution to the error localisation problem here.

For future reference, it is interesting to consider the values that may be imputed for x_1 and x_3 to obtain a consistent record. Denote the imputed values by x_1^* and x_3^* and suppose that $x_1^* = 7 - \beta$ for some, as yet unspecified, parameter β. By equation (15), $x_3^* = 12 + \beta$. Furthermore, by (21) and (22) it has to hold that $0 \leq \beta \leq \frac{5}{2}$. In summary, the potential records after imputation in this example are $\boldsymbol{x}^* = (x_1^*, x_2, x_3^*)' = (7 - \beta, 1, 12 + \beta)'$, for $0 \leq \beta \leq \frac{5}{2}$. □

In the above example, a restriction was placed on the maximal number of variables to impute. In practical applications of error localisation in official statistics, records of over 100 variables are commonly encountered. In these situations, specifying an upper bound M on the number of variables that may be imputed in a single record is necessary to obtain an error localisation problem that is computationally feasible; typical choices are, for instance, $M = 12$ or $M = 15$. De Waal and Coutinho (2005) argued that the introduction of such an upper bound is reasonable, because a record that requires more than, say, fifteen imputations should be considered unfit for automatic editing anyway.

5 Implied edits under linear edit operations

In this section, I will derive a result that establishes whether a certain combination of edit operations of the form (2) can be used to make a given record consistent with a given system of edit rules. This may be seen as a generalisation of the fundamental property of FM elimination.

5.1 One edit operation

First, consider an edit operation g of the form (2) that does *not* contain any free parameters. Let $\boldsymbol{x}$ be any record and let $\boldsymbol{y}$ be the record that is obtained by applying g to $\boldsymbol{x}$; that is, $\boldsymbol{y} = g(\boldsymbol{x}) = \boldsymbol{T}\boldsymbol{x} + \boldsymbol{h}$. By definition, $\boldsymbol{y}$ satisfies the edits (1) if, and only if, $\boldsymbol{A}\,(\boldsymbol{T}\boldsymbol{x} + \boldsymbol{h}) + \boldsymbol{b} \odot \boldsymbol{0}$, which is equivalent to:

$$(\boldsymbol{A}\boldsymbol{T})\,\boldsymbol{x} + (\boldsymbol{A}\boldsymbol{h} + \boldsymbol{b}) \odot \boldsymbol{0}. \tag{23}$$

Expression (23) may be interpreted as follows: the record $\boldsymbol{y} = g(\boldsymbol{x})$ is consistent with the original edits (1) if, and only if, the record $\boldsymbol{x}$ satisfies a similar system of linear edits, with $\boldsymbol{A}\boldsymbol{T}$ as coefficient matrix and $\boldsymbol{A}\boldsymbol{h} + \boldsymbol{b}$ as vector of constants. That is to say, applying the edit operation g to $\boldsymbol{x}$ yields a consistent record if, and only if, $\boldsymbol{x}$ satisfies (23).

Example. Consider the following edits for (x_1, x_2):

$$x_1 \geq 0, \tag{24}$$

$$x_2 \geq 0, \tag{25}$$

$$x_1 + x_2 \leq 5. \tag{26}$$

Let g be the edit operation that changes the sign of x_1: $g((x_1, x_2)') = (-x_1, x_2)'$. Under this edit operation, the above edits are transformed into the following system:

$$-x_1 \geq 0, \tag{27}$$

$$x_2 \geq 0, \tag{28}$$

$$-x_1 + x_2 \leq 5. \tag{29}$$

[These edits may be obtained from (23) after some tedious algebra. They are obtained more quickly by replacing x_1 by $-x_1$ in the original edits.]

The example record $(x_1, x_2)' = (-2, 3)'$ is inconsistent with the original edit rules (24)–(26). On the other hand, it does satisfy the transformed edit rules (27)–(29). This implies that the record can be made consistent with the original edits by changing the sign of x_1. It is easily verified that the resulting record $(x_1, x_2)' = (2, 3)'$ indeed satisfies (24)–(26). □

Next, consider the case that g involves at least one free parameter $\boldsymbol{\alpha}$. One still obtains (23) but now as a system of constraints on the original record $\boldsymbol{x}$ *and* the parameters in $\boldsymbol{\alpha}$. By the same reasoning as above, a consistent record can be obtained by applying g to $\boldsymbol{x}$ with a certain choice of $\boldsymbol{\alpha}$ if, and only if, $\boldsymbol{x}$ and $\boldsymbol{\alpha}$ satisfy (23) and (if relevant) the additional restrictions (3). Interestingly, (23) and (3) constitute a system of *linear* restrictions of the form (1) for the extended record $(\boldsymbol{x}', \boldsymbol{\alpha}')'$. Therefore, FM elimination may be used to remove all free parameters from (23) and (3). This yields a system of implied linear restrictions for $\boldsymbol{x}$. Moreover,

the fundamental property of FM elimination states that $\boldsymbol{x}$ satisfies this system of implied edits if, and only if, there exist parameter values for $\boldsymbol{\alpha}$ that, together with $\boldsymbol{x}$, satisfy (23) and (3). Hence, it follows that applying the edit operation g to $\boldsymbol{x}$ can lead to a consistent record (for some choice of parameter values) if, and only if, $\boldsymbol{x}$ satisfies the system of implied edits obtained by eliminating $\boldsymbol{\alpha}$ from (23) [and, if relevant, (3)].

Example. Consider the previous example with the edits (24)–(26). This time, suppose g is the edit operation that transfers an amount of at most four units between x_1 and x_2, in either direction: $g((x_1, x_2)') = (x_1 + \alpha, x_2 - \alpha)'$ with $-4 \leq \alpha \leq 4$. For this edit operation, the system of transformed edits (23) is:

$$x_1 + \alpha \geq 0, \tag{30}$$

$$x_2 - \alpha \geq 0, \tag{31}$$

$$x_1 + x_2 \leq 5. \tag{32}$$

I also add the following restrictions of the form (3) on α:

$$\alpha \geq -4, \tag{33}$$

$$-\alpha \geq -4. \tag{34}$$

This yields five linear constraints (30)–(34) on x_1, x_2, and α, from which α may be removed by FM elimination. This yields the following implied edits:

$$x_1 \geq -4, \tag{35}$$

$$x_2 \geq -4, \tag{36}$$

$$x_1 + x_2 \geq 0, \tag{37}$$

$$x_1 + x_2 \leq 5. \tag{38}$$

According to the theory, any record $(x_1, x_2)'$ that satisfies (35)–(38) can be made consistent with the original edits (24)–(26) by transferring a certain amount $-4 \leq \alpha \leq 4$ between x_1 and x_2. As an illustration, the same example record as before $(x_1, x_2)' = (-2, 3)'$ satisfies (35)–(38). It is indeed possible to make this record consistent with (24)–(26) by applying g; in fact, any choice $2 \leq \alpha \leq 3$ will do. □

It should be noted that, for the special case that g is an FH operation, the above result is consistent with the existing theory reviewed in Section 4. In fact, for the FH operation that imputes x_j, the transformed system of edits (23) is obtained by replacing every occurrence of x_j in the original edits by an unrestricted parameter α. Eliminating α from (23) is equivalent in this case to eliminating x_j directly from the original edits. In this sense, the above result generalises the fundamental property of FM elimination to all edit operations of the form (2).

5.2 Several edit operations

Extending the result of Section 5.1 to a sequence of multiple edit operations is rather straightforward. Let $\boldsymbol{y}_t$ be the record that results from applying, in sequence, the edit operations $g_1, \ldots, g_t$ to the original record $\boldsymbol{x}$, for some $t \geq 2$. (In the terminology of Section 3, $P = [g_1, \ldots, g_t]$ is a path connecting $\boldsymbol{x}$ to $\boldsymbol{y}_t$.) Write $g_n(\boldsymbol{x}) = \boldsymbol{T}_n\boldsymbol{x} + \boldsymbol{h}_n$, for all $n \in \{1, \ldots, t\}$.

In general, the order in which a set of edit operations is applied may affect the outcome. It is assumed here that the edit operations are applied in the order in which they are numbered:

$$\boldsymbol{y}_t = g_t \circ g_{t-1} \circ \cdots \circ g_1(\boldsymbol{x}). \tag{39}$$

From (39) it follows immediately that

$$\begin{aligned}\boldsymbol{y}_2 &= \boldsymbol{T}_2\left(\boldsymbol{T}_1\boldsymbol{x}+\boldsymbol{h}_1\right)+\boldsymbol{h}_2\\ &= \boldsymbol{T}_2\boldsymbol{T}_1\boldsymbol{x}+\boldsymbol{T}_2\boldsymbol{h}_1+\boldsymbol{h}_2.\end{aligned} \tag{40}$$

Claim. *For any $t \geq 2$, $\boldsymbol{y}_t$ can be written as follows:*

$$\boldsymbol{y}_t = \boldsymbol{T}_t \cdots \boldsymbol{T}_1\boldsymbol{x} + \sum_{n=2}^{t} \boldsymbol{T}_t \cdots \boldsymbol{T}_n\boldsymbol{h}_{n-1} + \boldsymbol{h}_t. \tag{41}$$

Proof. The proof proceeds by induction on t. For $t = 2$, expression (41) is identical to (40), so the statement is trivial. For $t > 2$, assuming that the statement holds for $t-1$, I obtain:

$$\begin{aligned}\boldsymbol{y}_t &= \boldsymbol{T}_t\boldsymbol{y}_{t-1}+\boldsymbol{h}_t\\ &= \boldsymbol{T}_t\left(\boldsymbol{T}_{t-1}\cdots\boldsymbol{T}_1\boldsymbol{x}+\sum_{n=2}^{t-1}\boldsymbol{T}_{t-1}\cdots\boldsymbol{T}_n\boldsymbol{h}_{n-1}+\boldsymbol{h}_{t-1}\right)+\boldsymbol{h}_t\\ &= \boldsymbol{T}_t\cdots\boldsymbol{T}_1\boldsymbol{x}+\sum_{n=2}^{t-1}\boldsymbol{T}_t\cdots\boldsymbol{T}_n\boldsymbol{h}_{n-1}+\boldsymbol{T}_t\boldsymbol{h}_{t-1}+\boldsymbol{h}_t.\end{aligned}$$

Expression (41) now follows. □

Suppose first that the edit operations $g_1, \ldots, g_t$ do not contain any free parameters. Then, by applying (41), it is seen that $\boldsymbol{y}_t$ satisfies the edits (1) if, and only if, the original record $\boldsymbol{x}$ satisfies

$$\left(\boldsymbol{A}\boldsymbol{T}_t\cdots\boldsymbol{T}_1\right)\boldsymbol{x}+\left(\sum_{n=2}^{t}\boldsymbol{A}\boldsymbol{T}_t\cdots\boldsymbol{T}_n\boldsymbol{h}_{n-1}+\boldsymbol{A}\boldsymbol{h}_t+\boldsymbol{b}\right)\odot\mathbf{0}. \tag{42}$$

Moreover, if some of the vectors $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_t$ contain free parameters, these can be eliminated from (42), together with possible restrictions of the form (3), by means of FM elimination. Note that, just like (23), (42) is a system of linear constraints on an extended record consisting of $\boldsymbol{x}$ and any parameters that occur in $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_t$. Thus, the following extended version of the result of Section 5.1 holds: a record $\boldsymbol{x}$ can be made consistent with the original edits by applying the sequence of edit operations in (39) if, and only if, $\boldsymbol{x}$ satisfies the system of implied edits obtained by eliminating all free parameters from (42) and (3).

6 An error localisation algorithm

6.1 Preliminaries

In this section, I will outline a possible algorithm for solving the generalised error localisation problem of Section 3 when $\mathcal{G}$ contains other allowed edit operations in addition to the FH operations. In principle, this problem could be solved by an exhaustive search: for each *ordered* subset of $\mathcal{G}$, consider the corresponding path of edit operations starting in $\boldsymbol{x}$ and check whether $\boldsymbol{x}$ satisfies the associated implied edits obtained from (42) and (if relevant) (3). According to the theory of Section 5, the paths for which $\boldsymbol{x}$ satisfies these implied edits correspond exactly to the feasible solutions of the error localisation problem (5). In particular, if $P^* = [g_1^*, \ldots, g_t^*]$ has the shortest length among these paths, it follows that $\boldsymbol{x}^* = g_t^* \circ g_{t-1}^* \circ \cdots \circ g_1^*(\boldsymbol{x})$ is an optimal solution to the error localisation problem.

Following the common use of a restriction on the number of imputed variables in Fellegi-Holt-based editing (cf. Section 4), the above search could be limited to paths of at most R edit operations, for some pre-chosen (not too large) number R. Unfortunately, even with this restriction, the above "brute-force approach" is not efficient enough for realistic applications. Let N denote the number of distinct edit operations in $\mathcal{G}$. The number of paths of at most R steps that can be constructed from $\mathcal{G}$ equals:

$$\sum_{t=1}^{R} \frac{N!}{(N-t)!} = \sum_{t=1}^{R} t! \binom{N}{t},$$

since the ordering of edit operations within a path matters in general. In a realistic situation with, say, $p = 100$, $N = 200$, and $R = 10$, this would yield a search space of approximately 8.2×10^{22} possible paths.

To obtain a more efficient search algorithm, it is important to reduce the search space. One way to do this is by identifying as soon as possible paths that are uninteresting, either because they could never lead to a feasible solution or because their length would exceed that of the best solution found so far. In addition, a lot of extra work is introduced by the need to check every possible ordering of the same subset of edit operations, in particular when these subsets become larger. In fact, the order of the edit operations in a path matters only in *some* cases. Thus, another way to improve the efficiency of the algorithm may be by identifying paths where the ordering of edit operations does not matter. In the present subsection, both directions of improvement will be explored. The resulting improved algorithm will be described in Section 6.2.

First of all, it may be observed that, by extending the set of allowed edit operations beyond FH operations, one does not obtain any new feasible solutions to the error localisation problem. In fact, it was already noted in Section 3 that any two points in $\mathbb{R}^p$ can be connected using only FH operations. Therefore, any feasible solution corresponding to a path of general edit operations $P = [g_1, \ldots, g_t]$ can also be obtained by the path P' that consists of all FH operations associated with variables that are changed by one or more of the edit operations $g_1, \ldots, g_t$. On the other hand, the choice of an *optimal* solution among the feasible ones does depend in general on the set of allowed edit operations.

Based on this observation, one could use the following two-step approach to solve the error localisation problem:

1. Find all feasible solutions to the error localisation problem with only FH operations allowed.
2. For each feasible solution found in step 1, find the shortest path of general allowed edit operations that leads to (a special case of) this solution.

The overall shortest path found in step 2 would then correspond to the optimal solution. The addition "a special case" in step 2 is necessary because the optimal solution using general edit operations may be more restrictive than any feasible solution using only FH operations. For example, it might happen that "impute x_1 and x_2" is a feasible solution using only FH operations, while "interchange the values of x_1 and x_2" is the optimal solution using general edit operations. The latter solution may be seen as a special case of the first one. This feature will also be seen in the example discussed in Section 7.

For the first step in the above approach, an efficient algorithm for Fellegi-Holt-based error localisation may be used. As mentioned in Section 4, many such algorithms have been

developed in the past. Existing implementations of these algorithms may need to be modified slightly to find all feasible solutions instead of only the optimal ones. In the remainder of this section, I will focus on finding an algorithm for the second step.

Strictly speaking, the above observation about the set of feasible solutions only holds when the number of imputed variables is not restricted. Otherwise, it may happen that some solutions that are feasible when using general edit operations are not feasible when using only FH operations. As was seen above, restrictions on the number of imputed variables and/or edit operations are necessary (and, in fact, reasonable) in practice. The above two-step approach may still be used in this situation, provided that the following assumption is satisfied:

Assumption 1. *A solution to the error localisation problem is not acceptable if it involves either more than R edit operations or changes to the values of more than M variables. In practice, it will usually hold that $R \ll p$ and $M \ll p$.*

Note that, in practice, this assumption precludes the use of, for instance, an edit operation that changes the values of all variables in a record at the same time.

To simplify the problem in the second step, I also introduce the following assumption:

Assumption 2. *An optimal solution $\boldsymbol{x}^*$ to the error localisation problem for $\boldsymbol{x}$ can be obtained by a path $P = [g_1, \ldots, g_t]$ such that $x_j^* \neq x_j$ for any variable that is changed by one of the edit operations $g_1, \ldots, g_t$.*

This assumption precludes the occurrence of "diversions" where, for instance, a change made by g_1 to a certain variable is annulled by one of the other edit operations on the optimal path. This assumption seems reasonable, provided that $\mathcal{G}$ contains all edit operations that are relevant to a given application.[3)]

Finally, to find out whether the order of the edit operations on a path matters, it is convenient to introduce the concept of equivalent paths. For any subset $G \subseteq \mathcal{G}$, let $\mathcal{P}(\boldsymbol{x}; G)$ denote the set of paths that consist of all edit operations in G in some order. Note that if G contains t elements, $\mathcal{P}(\boldsymbol{x}; G)$ contains $t!$ paths. Two paths P_1 and P_2 in $\mathcal{P}(\boldsymbol{x}; G)$ are called *equivalent* if any record that can be reached from $\boldsymbol{x}$ by P_1 can also be reached by P_2, and vice versa. It is not difficult to see that this defines a proper equivalence relation on $\mathcal{P}(\boldsymbol{x}; G)$ (i.e., it defines a relation between paths that is reflexive, symmetric, and transitive). Let $\tilde{\mathcal{P}}(\boldsymbol{x}; G)$ denote a set that contains one representative of each of the equivalence classes in $\mathcal{P}(\boldsymbol{x}; G)$ under this relation.

Clearly, if one of the paths in an equivalence class of $\mathcal{P}(\boldsymbol{x}; G)$ leads to a feasible solution, then so do all the other paths in that class. Moreover, each path in $\mathcal{P}(\boldsymbol{x}; G)$ has the same length. To solve the error localisation problem, it is therefore sufficient to search for optimal paths in $\tilde{\mathcal{P}}(\boldsymbol{x}; G)$ instead of $\mathcal{P}(\boldsymbol{x}; G)$.

A set of representatives $\tilde{\mathcal{P}}(\boldsymbol{x}; G)$ may be constructed from $\mathcal{P}(\boldsymbol{x}; G)$ by the following algorithm.

[3)] At first glance, assumption 2 may seem to be quite harmless. Suppose, however, that $\mathcal{G}$ contains the following allowed edit operations: an operation that moves the values of x_1, x_2 and x_3 in a cycle, and an operation that interchanges the values of x_1 and x_2. Applying both edit operations to a record (a, b, c) yields first (c, a, b) and then (a, c, b). The latter record could also be obtained directly from (a, b, c) by interchanging only the values of x_2 and x_3. However, if that particular edit operation is not allowed, then the shortest path from (a, b, c) to (a, c, b) might consist of the two edit operations mentioned above. This path contains a "diversion" in the value of x_1, so it would not be found if assumption 2 were made.

1. Start with $\tilde{\mathcal{P}}(\boldsymbol{x}; G) = \emptyset$.
2. For each $P \in \mathcal{P}(\boldsymbol{x}; G)$, do the following:
   a. For each $P' \in \tilde{\mathcal{P}}(\boldsymbol{x}; G)$, test whether P and P' are equivalent. As soon as an equivalence is found, stop.
   b. If P is not equivalent to any $P' \in \tilde{\mathcal{P}}(\boldsymbol{x}; G)$, then add P to $\tilde{\mathcal{P}}(\boldsymbol{x}; G)$.

The only non-trivial part of this algorithm lies in testing whether two paths are equivalent. Appendix II discusses how this may be done in a straightforward manner.

6.2 The proposed algorithm

Figure 1 describes an algorithm for the second step of the above-mentioned two-step approach to solve the error localisation problem. This algorithm should be run separately for each feasible solution with FH operations found in the first step. The basic set-up of this algorithm was inspired by the so-called *apriori algorithm* of Agrawal and Srikant (1994) for data mining. Let J denote the index set of variables imputed by the given feasible solution with FH operations that serves as input for the algorithm. Upon completion, the algorithm returns a set $\mathcal{L}$ containing at least one path of allowed edit operations that leads to a consistent record for $\boldsymbol{x}$, as well as the length W of the shortest path in $\mathcal{L}$. The shortest path in $\mathcal{L}$ represents the best feasible solution that only affects the variables x_j ($j \in J$). As mentioned above, this best solution may be more restrictive than the initial solution that uses only FH operations.

In step 0 of the algorithm, the search space is limited by excluding all edit operations that affect variables other than x_j ($j \in J$). Under assumption 2 made above, it is guaranteed that the optimal solution to the error localisation problem can be found without looking at these edit operations. In addition, a path that contains all FH operations for the variables x_j ($j \in J$) is added to $\mathcal{L}$, since it is already known that this path leads to a feasible solution. It should be noted that the ordering of FH operations does not matter: any two paths that consist of the same FH operations are equivalent. For a direct proof, see the end of Appendix II.

In step 1 of the algorithm, the search space is limited further by using the following fact: if G has a proper subset $H \subset G$ for which $\mathcal{P}(\boldsymbol{x}; H)$ contains a path that leads to a consistent record, then $\mathcal{P}(\boldsymbol{x}; G)$ can contain only suboptimal solutions. Thus, any set G that has such a subset may be ignored by the algorithm. Similarly, G may also be ignored whenever the total weight of the edit operations in G exceeds the path length of the best feasible solution found so far (i.e., W).

To illustrate the algorithm of Figure 1, I will apply it to a small example in the next section. It remains to be seen whether this algorithm is computationally feasible in practice (e.g., in the above-mentioned realistic situation with $p = 100$, $N = 200$, and $R = 10$). During the t^{th} iteration, the number of subsets G encountered in step 1 of the algorithm equals $\binom{N_0}{t}$, with N_0 the number of edit operations in $\mathcal{G}_0$. For each of these subsets, the conditions in step 1 have to be checked. If a subset passes these checks, in step 2 an associated set of representative paths $\tilde{\mathcal{P}}(\boldsymbol{x}; G)$ has to be constructed and the paths in this set have to be evaluated using the theory of Section 5; in the worst case that none of the paths in $\mathcal{P}(\boldsymbol{x}; G)$ are equivalent, this step involves $t!$ different paths. The idea behind the apriori algorithm is that, as t becomes larger, the majority of subsets will not pass the checks in the first step, so that the total amount of computational work remains limited. In the context of data mining, this desirable behaviour has indeed been observed in practice.

Figure 1 An algorithm for finding the shortest path of allowed edit operations for a given feasible solution with FH operations.

Step 0 Let $\boldsymbol{x}$ be a given record with a feasible solution that imputes the variables x_j ($j \in J$) using FH operations. Let W be the value of expression (6) for this solution.[a)] Define $\mathcal{G}_0$ as the subset of edit operations in $\mathcal{G}$ that do not affect any variable x_j with $j \notin J$. Define $\mathcal{L}$ as a set that (initially) contains only a path consisting of all FH operations for $j \in J$. Define $\mathcal{B}_0$ as a collection that contains only the empty set. Finally, define $t := 1$.

a) If the algorithm is used as part of the two-step approach outlined in Section 6.1, its efficiency may be improved by choosing W equal to the path length of the overall best feasible solution found so far.

Step 1 Determine all subsets G of t elements of $\mathcal{G}_0$ that satisfy the following conditions:

1. Every subset of $t-1$ elements in G is part of $\mathcal{B}_{t-1}$.
2. $\sum_{g \in G} w_g \leq W$.

Step 2 For each G found in step 1, determine the set $\tilde{\mathcal{P}}(\boldsymbol{x}; G)$ and, for each P in this set, evaluate whether this path can lead to a consistent record when applied to $\boldsymbol{x}$. If so, then add P to $\mathcal{L}$. If *none* of the paths $P \in \tilde{\mathcal{P}}(\boldsymbol{x}; G)$ lead to a consistent record, then add G to $\mathcal{B}_t$.

Step 3 Compute $W := \min_{P \in \mathcal{L}} \ell(P)$. If $t < R$ and $\mathcal{B}_t \neq \emptyset$, define $t := t + 1$ and return to step 1.

Finally, it should be noted that, in theory, the algorithm of Figure 1 may also be used to solve the error localisation problem without resorting to the two-step approach of Section 6.1. In that case, the algorithm starts without a known feasible path of FH operations and one should set $\mathcal{G}_0 = \mathcal{G}$ in step 0. For the simulation study that will be discussed in Section 9 below – which involved only a small number of variables and allowed edit operations –, the algorithm was used in this stand-alone form.

7 Example

To illustrate the algorithm of Section 6, I revisit the three-variable example from Section 4. This time, suppose that, in addition to the three FH operations (abbreviated as FH1, FH2, and FH3), the following edit operations of the general form (2) are also allowed:

- an edit operation CS1 that changes the sign of x_1;
- an edit operation IC12 that interchanges the values of x_1 and x_2;
- an edit operation IC23 that interchanges the values of x_2 and x_3;
- an edit operation TF13 that transfers an amount of at most 15 units between x_1 and x_3 (in either direction).

Representations of these edit operations in matrix-vector notation can be derived from the examples given in Section 2. The weights of the allowed edit operations are specified as follows:

edit operation	FH1	FH2	FH3	CS1	IC12	IC23	TF13
weight	1	2	3	0.5	1	2	1

Although assumption 1 is not really needed in this small example, I restrict the maximum number of edit operations in a single path to $R = 2$ and retain the restriction that at most $M = 2$ variables may be imputed. Previously, it was stated that the error localisation problem with only FH operations has just one feasible solution in this example: "impute x_1 and x_3". With the above choice of weights, this solution has a path length of 4. I will now apply the algorithm in Figure 1 to find the optimal version of this solution when the above general edit operations are also allowed.

Since x_2 is not imputed under the present solution, any edit operation that affects this variable can be excluded from the search by assumption 2. Hence, in step 0 of the algorithm, the set $\mathcal{G}_0$ consists of the following edit operations: FH1, FH3, CS1, and TF13. In addition, I define $\mathcal{L} := \{[\mathrm{FH1}, \mathrm{FH3}]\}$, $W := 4$, $\mathcal{B}_0 := \{\emptyset\}$ and $t := 1$.

In the first iteration, the algorithm considers subsets of one edit operation from $\mathcal{G}_0$. There are four such subsets and they all satisfy the two conditions of step 1. (Note that the first condition is irrelevant when $t = 1$.) For subsets of cardinality one, $\mathcal{P}(\boldsymbol{x}; \{g\})$ contains a unique path; hence, there is no need to check for equivalent paths. For each of the paths $[\mathrm{FH1}]$, $[\mathrm{FH3}]$, $[\mathrm{CS1}]$, and $[\mathrm{TF13}]$, the theory of Section 5 can be used to see if it leads to a consistent record when applied to the original record $\boldsymbol{x} = (10, 1, -3)'$. Since the value of x_2 cannot be changed here, I use the original edits with $x_2 = 1$ fixed as a starting point, i.e., the edits (15)–(20) from Section 4.

It follows from the discussion of this example in Section 4 that the paths $[\mathrm{FH1}]$ and $[\mathrm{FH3}]$ do not lead to a consistent record here. For the path $[\mathrm{CS1}]$, working out expression (23) yields the following system of constraints:

$$
\begin{aligned}
-x_1 + x_3 &= 19,\\
-x_1 &\geq 4,\\
x_1 &\geq -7,\\
x_1 + x_3 &\geq 5,\\
-x_1 - x_3 &\geq -10,\\
x_3 &\geq 0.
\end{aligned}
$$

Since these constraints are not all satisfied by $x_1 = 10$ and $x_3 = -3$, I conclude that just changing the sign of x_1 does not lead to a consistent record in this example.

For the path $[\mathrm{TF13}]$, expression (23) yields a system that includes a free parameter:

$$
\begin{aligned}
x_1 + x_3 &= 19,\\
x_1 + \alpha &\geq 4,\\
-x_1 - \alpha &\geq -7,\\
-x_1 + x_3 - 2\alpha &\geq 5,\\
x_1 - x_3 + 2\alpha &\geq -10,\\
x_3 - \alpha &\geq 0,\\
\alpha &\geq -15,\\
-\alpha &\geq -15.
\end{aligned}
$$

Note that the last two inequalities are constraints of the form (3) that follow from the definition of TF13. By eliminating α from this system, an implied system of edits is found for x_1 and x_3. It turns out that the original values $x_1 = 10$ and $x_3 = -3$ do not satisfy this implied system, so the single edit operation TF13 does not lead to a consistent record either. This conclusion also follows directly from the first constraint above, since $x_1 + x_3 \neq 19$ independently of α.

In summary, applying single edit operations from $\mathcal{G}_0$ to $\boldsymbol{x}$ does not yield a feasible solution. At the end of the first iteration, it holds that $\mathcal{B}_1 = \{\{\mathrm{FH1}\}, \{\mathrm{FH3}\}, \{\mathrm{CS1}\}, \{\mathrm{TF13}\}\}$, $\mathcal{L} = \{[\mathrm{FH1}, \mathrm{FH3}]\}$, and $W = 4$.

In the next iteration, $t = 2$. There are $\binom{4}{2} = 6$ distinct subsets of two edit operations in $\mathcal{G}_0$:

$$\{\mathrm{FH1}, \mathrm{FH3}\},\ \{\mathrm{FH1}, \mathrm{CS1}\},\ \{\mathrm{FH1}, \mathrm{TF13}\},\ \{\mathrm{FH3}, \mathrm{CS1}\},\ \{\mathrm{FH3}, \mathrm{TF13}\},\ \{\mathrm{CS1}, \mathrm{TF13}\}.$$

Since no feasible solutions were found in the first iteration, all of these subsets satisfy the first condition of step 1. They also satisfy the second condition. For each subset, $\mathcal{P}(\boldsymbol{x}; G)$ consists of two ordered paths. However, by applying the results of Appendix II, it can be shown that for the first five subsets listed above, the two ordered paths are equivalent. The only pair of non-equivalent paths is found to be $[\mathrm{CS1}, \mathrm{TF13}]$ and $[\mathrm{TF13}, \mathrm{CS1}]$. Thus, in total, seven paths (out of a potential twelve) need to be evaluated in this iteration. For the sake of brevity, I discuss only one of these evaluations in detail.

Consider the path $[\mathrm{FH1}, \mathrm{TF13}]$. To see whether this path leads to a feasible solution, I apply the transformation (42) (with $t = 2$) to the edits (15)–(20), with $\boldsymbol{T}_1$ and $\boldsymbol{h}_1$ coming from the definition of FH1, and $\boldsymbol{T}_2$ and $\boldsymbol{h}_2$ coming from the definition of TF13. This yields the following system of constraints, with the parameter α_1 introduced by FH1 and the parameter α_2 introduced by TF13:

$$x_3 + \alpha_1 = 19, \tag{43}$$
$$\alpha_1 + \alpha_2 \geq 4, \tag{44}$$
$$-\alpha_1 - \alpha_2 \geq -7, \tag{45}$$
$$x_3 - \alpha_1 - 2\alpha_2 \geq 5, \tag{46}$$
$$-x_3 + \alpha_1 + 2\alpha_2 \geq -10, \tag{47}$$
$$x_3 - \alpha_2 \geq 0, \tag{48}$$
$$\alpha_2 \geq -15, \tag{49}$$
$$-\alpha_2 \geq -15. \tag{50}$$

Again, the last two restrictions are added from the definition of TF13. The two parameters have to be eliminated from (43)–(50). Elimination of α_1 yields the following non-redundant edits:

$$\begin{aligned} x_3 - \alpha_2 &\geq 12, \\ -2x_3 + 2\alpha_2 &\geq -29, \\ \alpha_2 &\geq -15, \\ -\alpha_2 &\geq -15. \end{aligned}$$

Elimination of α_2 from this system yields, upon simplification:

$$\begin{aligned} x_3 &\geq -3, \\ -2x_3 &\geq -59. \end{aligned}$$

It is seen that the original value $x_3 = -3$ satisfies this system of implied edits. Therefore, I conclude that a consistent record can be obtained by applying the edit operations FH1 and TF13

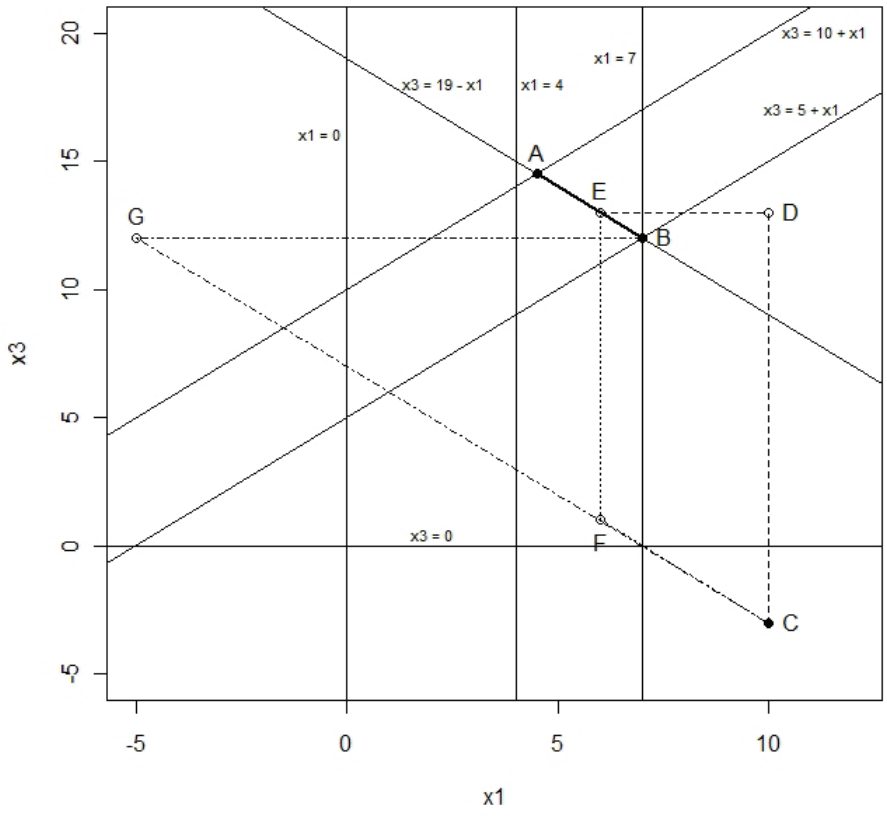

Figure 2 Illustration of three feasible solutions in the (x_1, x_3) plane.

to $\boldsymbol{x}$. The path $[\mathrm{FH1}, \mathrm{TF13}]$ is added to $\mathcal{L}$. The associated path length is 2, which is an improvement compared to the best solution found so far.

The other six paths mentioned above may be handled similarly. One of these paths actually corresponds to the Fellegi-Holt-based solution of Section 4. The remaining paths yield just one additional feasible solution, given by the path $[\mathrm{FH3}, \mathrm{TF13}]$. This solution has a path length of 4. At the end of the second iteration, it holds that: $\mathcal{B}_2 = \{\{\mathrm{FH1}, \mathrm{CS1}\}, \{\mathrm{FH3}, \mathrm{CS1}\}, \{\mathrm{CS1}, \mathrm{TF13}\}\}$, $\mathcal{L} = \{[\mathrm{FH1}, \mathrm{FH3}], [\mathrm{FH1}, \mathrm{TF13}], [\mathrm{FH3}, \mathrm{TF13}]\}$, and $W = 2$.

As $R = 2$, the algorithm is stopped after the second iteration. The optimal solution returned by the algorithm is: "impute x_1 and transfer an amount between x_1 and x_3". The corresponding distance $d(\boldsymbol{x}, D_0)$ equals 2. Note that the order in which the two edit operations are applied does not matter in this solution.

It was seen in Section 4 that the initial solution "impute x_1 and x_3" based on the Fellegi-Holt paradigm produces an amended record of the form $\boldsymbol{x}^* = (x_1^*, x_2, x_3^*)' = (7 - \beta, 1, 12 + \beta)'$ with $0 \leq \beta \leq \frac{5}{2}$. It is interesting to see whether the "improved" solution "impute x_1 and transfer an amount between x_1 and x_3" places additional restrictions on the values in the amended record.

For the solution "impute x_1 and transfer an amount between x_1 and x_3", the amended record has the form $(x_1^*, x_2, x_3^*)' = (\alpha_1 + \alpha_2, 1, -3 - \alpha_2)'$, where the parameters (α_1, α_2) satisfy the system of restrictions found by substituting $x_3 = -3$ in (43)–(50). From (43), it follows immediately that $\alpha_1 = 22$. The remaining restrictions for α_2 are satisfied only when $\alpha_2 = -15$. Thus, for this solution, the edits determine unique feasible values for the parameters. The

unique amended record is found to be $(x_1^*, x_2, x_3^*)' = (7, 1, 12)'$. This is a special case of the solution found previously, with $\beta = 0$. Hence, the optimal solution "impute x_1 and transfer an amount between x_1 and x_3" is found to be more restrictive than "impute x_1 and x_3". In a similar way, it can be shown that the other feasible (but suboptimal) solution found above – "impute x_3 and transfer an amount between x_1 and x_3" – is *not* more restrictive than "impute x_1 and x_3"; i.e., every record of the above form with $0 \leq \beta \leq \frac{5}{2}$ can be obtained using these edit operations.

Figure 2 summarises the results for this example in graphical form. Since the value $x_2 = 1$ is fixed here, records can be represented as points in the (x_1, x_3) plane. The boundary of the region defined by each edit from (15)–(20) is plotted as a solid line in Figure 2. The feasible region defined jointly by these edits is shown as the bold line segment AB; note that this segment consists of all points of the form $(x_1, x_3) = (7 - \beta, 12 + \beta)$ with $0 \leq \beta \leq \frac{5}{2}$. Geometrically, AB is found by intersecting the feasible region D_0 defined by the original edits (7)–(14) with the plane $x_2 = 1$. The original record $\boldsymbol{x}$ is plotted as point C.

The Fellegi-Holt-based feasible solution "impute x_1 and x_3" can be used to reach any point on AB from C. One potential path shown in Figure 2 consists of the line segment CD (i.e., an imputation for x_3) followed by DE (i.e., an imputation for x_1). The suboptimal solution "impute x_3 and transfer an amount between x_1 and x_3" also reaches any point on AB; a potential path is shown as CF (i.e., a transferred amount from x_1 to x_3) followed by FE (an imputation for x_3). The *optimal* solution "impute x_1 and transfer an amount between x_1 and x_3" reaches only the point B. The corresponding path is displayed as CG (a transferred amount from x_1 to x_3; note that the maximal allowed amount of 15 is needed here) followed by GB (an imputation for x_1).

In terms of distances, it holds that $d(C, B) = 2$ and $d(C, E) = 4$ for all points $E \neq B$ on AB. Apparently, it is considered better to adjust C towards B than towards any other point on AB.

8 A statistical interpretation of the error localisation problem

In motivating their paradigm for automatic error localisation, Fellegi and Holt (1976) did not provide any formal statistical argument. Their reasoning was more intuitive:

> "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." (Fellegi and Holt, 1976, p. 18)

In fact, error localisation under the Fellegi-Holt paradigm is often regarded as a "mechanical" approach without a clear statistical interpretation. Alternative error localisation procedures of a more statistical nature have been proposed by, e.g., Little and Smith (1987) and Ghosh-Dastidar and Schafer (2006). These procedures use outlier detection techniques and are based on an

explicit model for the true data. Unfortunately, they cannot handle edit rules such as (1) in a straightforward manner, which makes them unsuitable for most applications in official statistics.

The aim of the present section is to show that, under certain conditions, the optimal solution to the generalised error localisation problem of Section 3 may be interpreted as an approximate maximum likelihood estimator. The derivation of this result is based on Kruskal (1983, pp. 38-39), who gave a similar argument to justify the use of the so-called Levenshtein distance in string comparisons.

Let $\boldsymbol{x}$ be an observed record of data that may contain errors and let $\boldsymbol{y}$ denote the corresponding unobserved, error-free record. The goal of error localisation is to reconstruct, as well as possible, the unknown $\boldsymbol{y}$ from the observed $\boldsymbol{x}$. Assuming that the edits (1) are hard edits, i.e., edits that are failed only by erroneous values, it holds that $\boldsymbol{y} \in D_0$. Suppose that the stochastic "error generating process" $\mathcal{E}$ introduced in Section 2 has the following properties:

- There exists a one-to-one correspondence between the set of errors that can occur under $\mathcal{E}$ and the set of allowed edit operations $\mathcal{G}$. This means that every $g \in \mathcal{G}$ can be seen as a "corrector" for (exactly) one particular error under $\mathcal{E}$ and, on the other hand, every error under $\mathcal{E}$ has (exactly) one "corrector" in $\mathcal{G}$.
- Each error in $\mathcal{E}$ occurs independently of the other errors.
- The error that corresponds to edit operation g occurs with known probability p_g. It is assumed that $p_g \ll 1$; in other words, errors are assumed to be rare. It is also assumed that p_g does not depend on the true record $\boldsymbol{y}$.

Under the above assumptions, $\boldsymbol{x}$ is a random variable, obtained by drawing errors from $\mathcal{E}$ and applying them to $\boldsymbol{y}$. Consider the likelihood function of $\boldsymbol{y}$, given an observed record $\boldsymbol{x}$. Since the edits are assumed to be hard, it holds that $L(\boldsymbol{y}|\boldsymbol{x}) = 0$ for all $\boldsymbol{y} \notin D_0$. For $\boldsymbol{y} \in D_0$, $L(\boldsymbol{y}|\boldsymbol{x})$ is equivalent to the probability density of $\boldsymbol{x}$ given $\boldsymbol{y}$. This density consists of a sum over all paths in $\mathcal{P}(\boldsymbol{x}, \boldsymbol{y})$, with each path $P = [g_1, \ldots, g_t]$ contributing the probability of observing exactly the combination of errors corresponding to the edit operations $g_1, \ldots, g_t$. Since errors are assumed to occur independently of each other, one obtains:

$$\begin{aligned} L(\boldsymbol{y}|\boldsymbol{x}) &= \sum_{P=[g_1,\ldots,g_t]\in\mathcal{P}(\boldsymbol{x},\boldsymbol{y})} \left[p_{g_1} \cdots p_{g_t} \prod_{g\in\mathcal{G}\setminus\{g_1,\ldots,g_t\}} (1-p_g) \right] \\ &\approx \sum_{P=[g_1,\ldots,g_t]\in\mathcal{P}(\boldsymbol{x},\boldsymbol{y})} p_{g_1} \cdots p_{g_t}. \end{aligned} \tag{51}$$

In the second line, it was used that $p_g \ll 1$, so the product of terms $(1 - p_g)$ is approximately equal to 1.

Next, suppose that the weights of the edit operations $g \in \mathcal{G}$ are chosen as follows:

$$w_g = -\log p_g. \tag{52}$$

Expression (51) may then be rewritten as follows:

$$L(\boldsymbol{y}|\boldsymbol{x}) \approx \sum_{P=[g_1,\ldots,g_t]\in\mathcal{P}(\boldsymbol{x},\boldsymbol{y})} \exp\left\{-\sum_{n=1}^{t} w_{g_n}\right\} = \sum_{P\in\mathcal{P}(\boldsymbol{x},\boldsymbol{y})} \exp\{-\ell(P)\}. \tag{53}$$

Let P^* denote the shortest path from $\boldsymbol{x}$ to $\boldsymbol{y}$; that is to say, $d(\boldsymbol{x}, \boldsymbol{y}) = \ell(P^*)$ and $\ell(P) \geq \ell(P^*)$ for all $P \in \mathcal{P}(\boldsymbol{x}, \boldsymbol{y})$. The largest contribution to the sum in (53) is made by P^*. Now suppose that this term dominates the other terms in the sum, so that it holds approximately that

$$L(\boldsymbol{y}|\boldsymbol{x}) \approx \exp\{-\ell(P^*)\} = \exp\{-d(\boldsymbol{x}, \boldsymbol{y})\}. \tag{54}$$

This is equivalent to:

$$d(\boldsymbol{x}, \boldsymbol{y}) \approx -\log L(\boldsymbol{y}|\boldsymbol{x}).$$

Thus, under the assumptions made here, it is seen that *minimising* $d(\boldsymbol{x}, \boldsymbol{y})$ over all $\boldsymbol{y} \in D_0$ is approximately equivalent to *maximising* the (log)likelihood function of $\boldsymbol{y}$ given $\boldsymbol{x}$. In this sense, the optimal solution to error localisation problem (5) can be justified as an approximate maximum likelihood estimator for the unknown, error-free record corresponding to $\boldsymbol{x}$.

It should be acknowledged that the above argument is rather heuristic and based on assumptions that are unlikely to be fully met in practice. In particular, the approximation of (53) by (54) might introduce quite a large error if there exist many paths from $\boldsymbol{x}$ to $\boldsymbol{y}$ that are (almost) as short as P^*. The assumption that errors occur independently with probabilities that do not depend on the true record is also unlikely to hold exactly in practice. Unfortunately, these approximations are needed to obtain an optimisation problem that is computationally feasible.

Nevertheless, the above result may be of some interest, as it provides a statistical interpretation of a procedure that is commonly regarded as being "mechanical". Moreover, the assumptions made here[4)] may provide an idea of the circumstances under which the error localisation problem of Section 3 is likely to yield good results. In particular, (52) suggests a way to assign weights to edit operations, although it may be difficult to obtain information about the probabilities p_g in practice.

The above derivation may also be used to justify the Fellegi-Holt paradigm as an approximate maximum likelihood procedure, provided that it is reasonable to assume that the only errors that occur in a particular application are those corresponding to FH operations. These are errors that occur independently of each other and affect one variable at a time.

9 Simulation study

To test the potential usefulness of the new error localisation approach, I conducted a small simulation study, using the R environment for statistical computing (R Development Core Team, 2014). For this, I wrote a rough prototype implementation of the algorithm of Section 6.2, making use of the existing functionality for automatic editing available in the `editrules` package (De Jonge and Van der Loo, 2011; Van der Loo and De Jonge, 2012). In addition, I used the `lpSolve` package to solve the linear programming problems (71) and (72) in Appendix II. The prototype was not optimised for computational efficiency, but it turned out to work sufficiently fast for the relatively small error localisation problems encountered in this simulation study.

The simulation study involved records of five numerical variables that should satisfy the

4) It should be noted that these assumptions all refer to the error mechanism; I did not make any assumptions about the distribution of $\boldsymbol{y}$.

following linear edit rules:

$$
\begin{aligned}
x_1 + x_2 &= x_3, \\
x_3 - x_4 &= x_5, \\
x_1 &\geq 0, \\
x_2 &\geq 0, \\
x_3 &\geq 0, \\
x_4 &\geq 0, \\
x_1 &\geq x_2, \\
x_5 &\geq -0.1x_3, \\
x_5 &\leq 0.5x_3.
\end{aligned}
$$

These edits might typically be encountered in the SBS (as part of a much larger set of edit rules), where the variables could have the following interpretation:

- x_1 – turnover from main business activity;
- x_2 – turnover from other activities;
- x_3 – total turnover;
- x_4 – total costs;
- x_5 – profit.

I created an error-free data set of 2000 records by randomly drawing from a multivariate normal distribution (using the `mvtnorm` package) with the following parameters:

$$
\boldsymbol{\mu} = \begin{pmatrix} 500 \\ 250 \\ 750 \\ 600 \\ 150 \end{pmatrix} \quad \text{and} \quad \boldsymbol{\Sigma} = \begin{pmatrix} 10000 & -1250 & 8750 & 7500 & 1250 \\ -1250 & 5000 & 3750 & 4000 & -250 \\ 8750 & 3750 & 12500 & 11500 & 1000 \\ 7500 & 4000 & 11500 & 11750 & -250 \\ 1250 & -250 & 1000 & -250 & 1250 \end{pmatrix}.
$$

Note that $\boldsymbol{\Sigma}$ is a singular covariance matrix that incorporates the two equality edits $x_1 + x_2 = x_3$ and $x_3 - x_4 = x_5$. Only records that satisfied all of the above edits were added to the data set. Technically, the resulting data followed a so-called truncated multivariate singular normal distribution; see De Waal et al. (2011, pp. 318ff) or Tempelman (2007). Finally, the values in the data set were rounded to integers, taking the edit rules into account.

Table 1 lists the allowed edit operations that were considered in this study. As indicated in the table, each edit operation has an associated type of error. A synthetic data set to be edited was created by randomly adding errors of these types to the above-mentioned error-free data set, along the lines of the "error generating process" described in Section 8. The probability of each type of error is listed in the fourth column of Table 1. The associated "ideal" weight according to (52) is shown in the last column.

For the errors that correspond to FH operations, I changed the observed values to 0; in the exceptional case that the originally observed value equaled 0, the erroneous value became 1000. For TF21, the erroneously transferred amount was drawn from the uniform distribution on the interval $(x_1 - x_2, x_1]$, to ensure that the error resulted in an edit failure. For the remaining edit operations, the errors followed directly from the definition, as these operations do not contain free parameters.

To limit the amount of computational work, I only considered records that required three edit operations or less (i.e., that contained at most three errors). Records without errors were also

Table 1 Allowed edit operations for the simulation study.

name	description	associated error	p_g	w_g
FH1	impute x_1 (FH operation)	erroneous value of x_1	0.10	2.30
FH2	impute x_2 (FH operation)	erroneous value of x_2	0.08	2.53
FH3	impute x_3 (FH operation)	erroneous value of x_3	0.06	2.81
FH4	impute x_4 (FH operation)	erroneous value of x_4	0.04	3.22
FH5	impute x_5 (FH operation)	erroneous value of x_5	0.02	3.91
IC34	interchange x_3 and x_4	true values of x_3 and x_4 interchanged by mistake	0.07	2.66
TF21	transfer an amount from x_2 to x_1 (not vice versa)	part of the true value of x_1 reported as part of x_2	0.09	2.41
CS4	change the sign of x_4	sign error in x_4	0.11	2.21
CS5	change the sign of x_5	sign error in x_5	0.13	2.04

removed from the data set. This left 1025 records to be edited, with each record containing one, two, or three of the errors listed in Table 1.

Several different error localisation approaches were applied to this data set. First of all, I tested error localisation according to the Fellegi-Holt paradigm (i.e., using only the edit operations FH1–FH5) and according to the new paradigm (i.e., using all edit operations in Table 1). Both approaches were tested once using the "ideal" weights listed in Table 1 and once with all weights equal to 1. The latter case simulated a situation where the relevant edit operations would be known, but not their respective frequencies. In practice, it is unlikely that one would know exactly all relevant edit operations/types of errors in a particular application. To test the robustness of the new error localisation approach to a lack of information about relevant edit operations, I also applied this approach with one of the non-FH operations in Table 1 missing from the set of allowed edit operations.

The quality of error localisation was evaluated in two ways. Firstly, I evaluated how well the optimal paths of edit operations suggested by the error localisation algorithm matched the true distribution of errors for the synthetic data. This was done by calculating the following contingency table of edit operations per record:

	edit operation suggested	edit operation not suggested
associated error occurred	TP	FN
associated error did not occur	FP	TN

The information in this table was summarised by the following quality indicators:

$$\alpha = \frac{FN}{TP + FN}; \qquad \beta = \frac{FP}{FP + TN}; \qquad \delta = \frac{FN + FP}{TP + FN + FP + TN}.$$

Here, α measures the proportion of false negatives (edit operations that should have been suggested but were not), β measures the proportion of false positives (edit operations that were suggested but should not have been), and δ measures the joint proportion of wrong decisions. A good error localisation algorithm should have low scores on all three measures (De Waal et al.,

2011, pp. 410-411). As an alternative measure, I also computed the fraction of records in the data set for which the error localisation algorithm found exactly the right solution, i.e., a path of edit operations that corresponded exactly to the errors that occurred in a particular record. For this measure, denoted by ρ, higher scores indicate a better quality of error localisation.

It should be noted that the above quality indicators put the original Fellegi-Holt approach at a disadvantage, since this approach does not use all the edit operations listed in Table 1. Therefore, I also calculated a second set of quality indicators α, β, δ, and ρ that look at individual erroneous values rather than edit operations. In this case, α measures the proportion of values in the data set that were affected by errors but left unchanged by the optimal solution of the error localisation problem, and similarly for the other measures.

Table 2 displays the results of the simulation study for the first set of quality indicators (in terms of edit operations); Table 3 displays the second set of quality indicators (in terms of erroneous values). In both tables, a considerable improvement in the quality of the error localisation results is seen for the approach that used all edit operations, compared to the approach that used only FH operations. In addition, leaving one relevant edit operation out of the set of allowed edit operations had a negative effect on the quality of error localisation. In some cases this effect was quite large – particularly in Table 2 –, but the results of the new error localisation approach still remained substantially better than those of the Fellegi-Holt approach. Finally, contrary to expectation, not using different confidence weights actually improved the quality of the error localisation results somewhat for this data set under the Fellegi-Holt approach (both tables) and to some extent also under the new approach (Table 3 but not Table 2).

Table 2 Quality of error localisation in terms of edit operations used.

	quality indicators			
approach	α	β	δ	ρ
Fellegi-Holt	74%	12%	23%	20%
Fellegi-Holt (no weights)	70%	12%	21%	25%
all edit operations	14%	3%	5%	76%
all edit operations				
except IC34	29%	5%	8%	65%
except TF21	34%	5%	10%	63%
except CS4	28%	6%	9%	61%
except CS5	35%	7%	10%	53%
all edit operations (no weights)	27%	5%	8%	63%

Table 3 Quality of error localisation in terms of identified erroneous values.

	quality indicators			
approach	α	β	δ	ρ
Fellegi-Holt	19%	10%	13%	68%
Fellegi-Holt (no weights)	13%	8%	9%	75%
all edit operations	10%	5%	7%	83%
all edit operations				
except IC34	15%	8%	10%	71%
except TF21	10%	5%	7%	82%
except CS4	10%	5%	7%	83%
except CS5	11%	6%	7%	82%
all edit operations (no weights)	5%	4%	4%	88%

10Conclusion

In this paper, a new formulation was proposed of the error localisation problem in automatic editing. It was suggested to find the (weighted) minimal number of edit operations needed to make an observed record consistent with the edits. The new error localisation problem can be seen as a generalisation of the problem proposed by Fellegi and Holt (1976), because the operation that imputes a new value for one variable at a time is an important special case of an edit operation. The discussion in this paper was restricted to numerical data and linear edits. The original Fellegi-Holt paradigm has been applied also to categorical and mixed data (De Waal et al., 2011); in fact, the original article by Fellegi and Holt (1976) was devoted mainly to error localisation in categorical data. In principle, it should be possible to extend the approach of this paper to that context, using an appropriate class of edit operations, but this remains to be investigated.

The main focus in this paper has been on developing the mathematical theory behind the new error localisation problem. It turns out that the same elimination technique that is often used to solve the Fellegi-Holt-based error localisation problem can be applied also in the context of the new problem (Section 5). Nevertheless, the task of solving the new error localisation problem is challenging from a computational point of view, at least for the numbers of variables, edits, and edit operations that would be encountered in practical applications at Statistics Netherlands. A possible error localisation algorithm was outlined in Section 6. It is likely that the efficiency of this algorithm can be improved further.

There is an analogy between the error localisation problem proposed in this paper and the existing field of approximate string matching; see, e.g., Navarro (2001) for an overview. In approximate string matching, text strings are compared under the assumption that they may have been partially corrupted. Various distance functions have been proposed for approximate string matching. The Hamming distance, which counts the number of positions on which two strings differ, may be seen as an analogue of the Fellegi-Holt-based target function (6). The generalised error localisation problem defined in this paper has its counterpart in the use of the Levenshtein distance or "edit distance" for approximate string matching. It may be interesting to explore this analogy further. In particular, efficient algorithms have been developed for computing edit distances between strings; it might be possible to apply some of the underlying ideas also to the error localisation problem.

The new error localisation algorithm was applied successfully to a small synthetic data set (Section 9). Overall, the results of this simulation study suggest that the new error localisation approach has the potential to achieve a substantial improvement of the quality of automatic editing compared to the approach that is currently used in practice. However, this does require that sufficient information be available to identify all – or at least most – of the relevant edit operations in a particular application.

An obvious candidate for applying the new error localisation method in practice would be the SBS. As mentioned in the introduction, the automatic editing process that is currently used in the Dutch SBS is known to produce data that deviate in some respects from data that are edited manually. The method described in this paper has the potential to reduce these systematic differences between automatic and manual editing, by improving the flexibility of automatic editing in terms of the types of amendments that can be made to the data. A reduction of the

differences between automatic and manual editing would mean in turn that the efficiency of the editing process could be improved by increasing the fraction of data that is edited automatically.

However, more research is needed before the method described in this paper can be applied in practice. To apply the method in a particular context, it is necessary first to specify the relevant edit operations. Ideally, each edit operation should correspond to a combination of amendments to the data that human editors consider to be a correction for one particular error. In addition, a suitable set of weights w_g has to be determined for these edit operations. This would require information about the relative frequencies of the most common types of amendments made during manual editing. Both aspects could be investigated based on historical data before and after manual editing (including paradata logged during regular production), editing instructions and other documentation used by the editors, and interviews with editors and/or supervisors of editing.

On a more fundamental level, a question of demarcation arises between deductive correction methods and automatic editing under the new error localisation problem. In principle, many known types of error could be resolved either deductively or by means of edit operations: for instance, sign errors and interchanged values of revenues and costs (Scholtus, 2011) and, in principle, even unit of measurement errors. Each approach has its own advantages and disadvantages. Deductive correction by means of automatic correction rules provides a user with direct control over the amendments made to the data. On the other hand, this approach may require a very large collection of rules in practice to handle all possible errors correctly. Moreover, the amended records are not guaranteed to be consistent with all edit rules. Error localisation by means of edit operations is probably easier to maintain once it has been set up, and it automatically produces data that are consistent with the edits. On the other hand, this approach is less transparent and could produce unexpected (and undesirable) results when it is not properly set up. It is likely that some compromise between the two approaches will produce the best results, with some errors handled deductively and others by edit operations. However, it is not obvious how to make this division in practice.

Ultimately, the aim of the new methodology proposed in this paper is to improve the usefulness of automatic editing in practice. In this respect, the results of the above simulation study are promising. However, a number of issues remain to be investigated and resolved before the new error localisation approach could be applied in practice.

Acknowledgements

I would like to thank Jeroen Pannekoek, Ton de Waal, and Mark van der Loo for their comments on earlier versions of this paper.

References

Agrawal, R. and R. Srikant (1994). Fast Algorithms for Mining Association Rules. Technical report, IBM Almaden Research Center, San Jose, California.

Bikker, R. (2003a). Automatisch gaafmaken PS 2000 Bouwnijverheid: vier structurele problemen met oplossingen. Internal report (BPA-no. 2263-03-TMO), Statistics Netherlands, Voorburg; in Dutch.

Bikker, R. (2003b). Evaluatie automatisch versus handmatig gaafmaken van Productiestatistieken 2000 Handel & Transport – aanvullende verklaringen. Internal report (BPA-no. 1900-03-TMO), Statistics Netherlands, Voorburg; in Dutch.

Bruni, R. (2004). Discrete Models for Data Imputation. *Discrete Applied Mathematics 144*, 59–69.

Daalmans, J., B. Duyx, and S. Scholtus (2011). Gaafmaakregels afleiden voor de PS met datamining-technieken. Internal report (BPA-no. DMV-2011-09-07-JDAS-BDYX-SSHS), Statistics Netherlands, The Hague; in Dutch.

De Jonge, E. and M. Van der Loo (2011). Manipulation of Linear Edits and Error Localization with the Editrules Package. Discussion Paper 201120, Statistics Netherlands, The Hague.

De Jonge, E. and M. Van der Loo (2014). Error Localization as a Mixed Integer Problem with the Editrules Package. Discussion Paper (to appear), Statistics Netherlands, The Hague.

De Waal, T. (2003). Solving the Error Localization Problem by Means of Vertex Generation. *Survey Methodology 29*, 71–79.

De Waal, T. and W. Coutinho (2005). Automatic Editing for Business Surveys: An Assessment for Selected Algorithms. *International Statistical Review 73*, 73–102.

De Waal, T., J. Pannekoek, and S. Scholtus (2011). *Handbook of Statistical Data Editing and Imputation*. Hoboken, New Jersey: John Wiley & Sons.

De Waal, T. and R. Quere (2003). A Fast and Simple Algorithm for Automatic Editing of Mixed Data. *Journal of Official Statistics 19*, 383–402.

Fellegi, I. P. and D. Holt (1976). A Systematic Approach to Automatic Edit and Imputation. *Journal of the American Statistical Association 71*, 17–35.

Garfinkel, R. S., A. S. Kunnathur, and G. E. Liepins (1988). Error Localization for Erroneous Data: Continuous Data, Linear Constraints. *SIAM Journal on Scientific and Statistical Computing 9*, 922–931.

Ghosh-Dastidar, B. and J. L. Schafer (2006). Outlier Detection and Editing Procedures for Continuous Multivariate Data. *Journal of Official Statistics 22*, 487–506.

Granquist, L. and J. Kovar (1997). Editing of Survey Data: How Much is Enough? In L. E. Lyberg, P. Biemer, M. Collins, E. D. De Leeuw, C. Dippo, N. Schwartz, and D. Trewin (Eds.), *Survey Measurement and Process Quality*, pp. 415–435. John Wiley & Sons.

Kovar, J. and P. Whitridge (1990). Generalized Edit and Imputation System; Overview and Applications. *Revista Brasileira de Estadistica 51*, 85–100.

Kruskal, J. B. (1983). An Overview of Sequence Comparison. In D. Sankoff and J. B. Kruskal (Eds.), *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison*, pp. 1–44. Addison-Wesley.

Little, R. J. A. and P. J. Smith (1987). Editing and Imputation of Quantitative Survey Data. *Journal of the American Statistical Association 82*, 58–68.

Navarro, G. (2001). A Guided Tour to Approximate String Matching. *ACM Computing Surveys 33*, 31–88.

Pannekoek, J., S. Scholtus, and M. Van der Loo (2013). Automated and Manual Data Editing: A View on Process Design and Methodology. *Journal of Official Statistics 29*, 511–537.

R Development Core Team (2014). *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. URL: http://www.R-project.org/.

Ragsdale, C. T. and P. G. McKeown (1996). On Solving the Continuous Data Editing Problem. *Computers & Operations Research 23*, 263–273.

Riera-Ledesma, J. and J. J. Salazar-González (2003). New Algorithms for the Editing and Imputation Problem. Working Paper No. 5, UN/ECE Work Session on Statistical Data Editing, Madrid.

Schaffer, J. (1987). Procedure for Solving the Data-Editing Problem with Both Continuous and Discrete Data Types. *Naval Research Logistics 34*, 879–890.

Scholtus, S. (2011). Algorithms for Correcting Sign Errors and Rounding Errors in Business Survey Data. *Journal of Official Statistics 27*, 467–490.

Tempelman, D. C. G. (2007). Imputation of Restricted Data. PhD Thesis, University of Groningen.

Van der Loo, M. and E. De Jonge (2012). Automatic Data Editing with Open Source R. Working Paper No. 33, UN/ECE Work Session on Statistical Data Editing, Oslo.

Williams, H. P. (1986). Fourier's Method of Linear Programming and its Dual. *The American Mathematical Monthly 93*, 681–695.

Appendices

I Fourier-Motzkin elimination

As mentioned in Section 4, Fourier-Motzkin elimination (FM elimination) is a mathematical technique that transforms a system of linear constraints having p variables into a new system of linear constraints having at most $p-1$ variables (De Waal et al., 2011; Williams, 1986). Formally, FM elimination works as follows.

Consider a system of linear constraints (1) and let x_f be the variable to be eliminated. First, suppose that x_f is involved only in inequalities. In this case, one has to consider all pairs (r,s) of constraints in which the coefficients of x_f have opposite signs; that is, $a_{rf}a_{sf} < 0$. It may be assumed without loss of generality that $a_{rf} < 0$ and $a_{sf} > 0$. This means that the r^{th} constraint can be expressed as an upper bound on the value of x_f:

$$x_f \leq \frac{-1}{a_{rf}}\left(b_r + \sum_{j \neq f} a_{rj}x_j\right). \tag{55}$$

Similarly, the other constraint can be expressed as a lower bound on x_f:

$$x_f \geq \frac{-1}{a_{sf}}\left(b_s + \sum_{j \neq f} a_{sj}x_j\right). \tag{56}$$

Clearly, then, there exists a value for x_f that satisfies (55) and (56) if, and only if, the $p-1$ remaining variables satisfy:

$$\frac{-1}{a_{rf}}\left(b_r + \sum_{j \neq f} a_{rj}x_j\right) \geq \frac{-1}{a_{sf}}\left(b_s + \sum_{j \neq f} a_{sj}x_j\right).$$

This condition may be rewritten as a new constraint of the form (1):

$$\sum_{j=1}^{p} a_j^{\dagger}x_j + b^{\dagger} \geq 0, \tag{57}$$

with $a_j^{\dagger} = a_{sf}a_{rj} - a_{rf}a_{sj}$ and $b^{\dagger} = a_{sf}b_r - a_{rf}b_s$. Note that $a_f^{\dagger} = 0$, so x_f is not involved in (57). An inequality of the form (57) can be derived from each of the above-mentioned pairs (r,s). The implied system of constraints obtained by FM elimination now consists of these derived constraints, together with all original constraints that do not involve x_f.

Finally, suppose that there are linear equalities that involve x_f. It is possible to handle this case in the same way as before, by noting that each linear equality can be replaced by two equivalent linear inequalities. However, De Waal and Quere (2003) suggested an alternative elimination method that is more efficient in this case. Suppose that the r^{th} constraint in (1) is an equality that involves x_f. This constraint can be rewritten as

$$x_f = \frac{-1}{a_{rf}}\left(b_r + \sum_{j \neq f} a_{rj}x_j\right). \tag{58}$$

By substituting the expression on the right-hand-side of (58) for x_f in all other constraints, one obtains an implied system of constraints that does not involve x_f and that can be rewritten in the form (1).

For a proof that, both for equalities and inequalities, FM elimination has the fundamental property mentioned in Section 4, see, e.g., De Waal et al. (2011, pp. 69-70).

II Testing the equivalence of two paths

In this appendix to Section 6, a procedure is described for testing whether two paths in $\mathcal{P}(\boldsymbol{x}; G)$ belong to the same equivalence class. Let $G = \{g_1, \ldots, g_t\}$ $(t \geq 2)$, with $g_n(\boldsymbol{x}) = \boldsymbol{T}_n\boldsymbol{x} + \boldsymbol{h}_n$ for $n \in \{1, \ldots, t\}$. It is convenient here to distinguish between constants and parameters in $\boldsymbol{h}_n$. Let

$$\boldsymbol{h}_n = \boldsymbol{c}_n + \boldsymbol{S}_n\boldsymbol{\alpha}_n, \tag{59}$$

with $\boldsymbol{c}_n$ a p vector of known constants, $\boldsymbol{\alpha}_n$ an m_n vector of parameters, and $\boldsymbol{S}_n$ a $p \times m_n$ matrix of known coefficients. In the special case that g_n does not involve any free parameters, one has $m_n = 0$ and $\boldsymbol{h}_n = \boldsymbol{c}_n$. If relevant, a set of linear restrictions of the form (3) may be posited for the parameters in $\boldsymbol{\alpha}_n$:

$$\boldsymbol{R}_n\boldsymbol{\alpha}_n + \boldsymbol{d}_n \odot \boldsymbol{0}. \tag{60}$$

For simplicity, I first consider the situation that $t = 2$. As defined in Section 6.1, the two paths $P_{12} = [g_1, g_2]$ and $P_{21} = [g_2, g_1]$ are equivalent in $\mathcal{P}(\boldsymbol{x}; \{g_1, g_2\})$ when any record that can be reached from $\boldsymbol{x}$ by P_{12} can also be reached by P_{21}, and vice versa. That is to say, P_{12} and P_{21} are equivalent if there exists a one-to-one correspondence between records of the form

$$\boldsymbol{T}_2\boldsymbol{T}_1\boldsymbol{x} + \boldsymbol{T}_2\boldsymbol{h}_1 + \boldsymbol{h}_2 = \boldsymbol{T}_2\boldsymbol{T}_1\boldsymbol{x} + \boldsymbol{T}_2\boldsymbol{c}_1 + \boldsymbol{c}_2 + \boldsymbol{T}_2\boldsymbol{S}_1\boldsymbol{\alpha}_1 + \boldsymbol{S}_2\boldsymbol{\alpha}_2 \tag{61}$$

and records of the form

$$\boldsymbol{T}_1\boldsymbol{T}_2\boldsymbol{x} + \boldsymbol{T}_1\boldsymbol{h}_2 + \boldsymbol{h}_1 = \boldsymbol{T}_1\boldsymbol{T}_2\boldsymbol{x} + \boldsymbol{T}_1\boldsymbol{c}_2 + \boldsymbol{c}_1 + \boldsymbol{T}_1\boldsymbol{S}_2\boldsymbol{\alpha}_2 + \boldsymbol{S}_1\boldsymbol{\alpha}_1; \tag{62}$$

cf. expressions (40) and (59). Note that in (61) and (62), the record $\boldsymbol{x}$ is fixed.

I now consider three different cases.

Case 1: g_1 and g_2 contain no free parameters.
In this case, it holds that $\boldsymbol{h}_1 = \boldsymbol{c}_1$ and $\boldsymbol{h}_2 = \boldsymbol{c}_2$, and the last two terms in expressions (61) and (62) vanish. This means that these expressions do not contain any variables. Hence, the two paths P_{12} and P_{21} are equivalent if, and only if, (61) and (62) are identical. Let

$$\boldsymbol{\Delta} = (\boldsymbol{T}_2\boldsymbol{T}_1 - \boldsymbol{T}_1\boldsymbol{T}_2)\,\boldsymbol{x} + (\boldsymbol{T}_2 - \boldsymbol{I})\,\boldsymbol{c}_1 - (\boldsymbol{T}_1 - \boldsymbol{I})\,\boldsymbol{c}_2. \tag{63}$$

Then P_{12} and P_{21} are equivalent if, and only if, it holds that $\boldsymbol{\Delta} = \boldsymbol{0}$.

Case 2: g_1 and g_2 contain only unrestricted parameters.
In this case, expressions (61) and (62) describe sets of different records that can be reached from $\boldsymbol{x}$ by varying the choice of $\boldsymbol{\alpha}_1$ and $\boldsymbol{\alpha}_2$. (If only one of the edit operations involves free parameters, one of these vectors vanishes; for notational simplicity, I do not treat this as a separate case.) No restrictions of the form (60) have to be taken into account.

Consider the record that is reached by the path P_{12} by choosing $\boldsymbol{\alpha}_1 = \boldsymbol{\alpha}_1^{(1)}$ and $\boldsymbol{\alpha}_2 = \boldsymbol{\alpha}_2^{(1)}$ in (61). If the path P_{21} is equivalent to P_{12}, it should be possible to find values for $\boldsymbol{\alpha}_1$ and $\boldsymbol{\alpha}_2$ such that (62) yields the same record. That is to say, it should hold that the system

$$\boldsymbol{T}_1\boldsymbol{S}_2\boldsymbol{\alpha}_2^{(2)} + \boldsymbol{S}_1\boldsymbol{\alpha}_1^{(2)} = \boldsymbol{\Delta} + \boldsymbol{T}_2\boldsymbol{S}_1\boldsymbol{\alpha}_1^{(1)} + \boldsymbol{S}_2\boldsymbol{\alpha}_2^{(1)} \tag{64}$$

has a solution for $\boldsymbol{\alpha}_1^{(2)}$ and $\boldsymbol{\alpha}_2^{(2)}$, given $\boldsymbol{\alpha}_1^{(1)}$ and $\boldsymbol{\alpha}_2^{(1)}$. Here, $\boldsymbol{\Delta}$ is given by (63). Let $\boldsymbol{M}_1 = (\boldsymbol{T}_2\boldsymbol{S}_1, \boldsymbol{S}_2)$ and $\boldsymbol{M}_2 = (\boldsymbol{T}_1\boldsymbol{S}_2, \boldsymbol{S}_1)$. By a standard result from linear algebra, (64) has a solution if, and only if, the vector on the right-hand-side is contained in the column space of the matrix $\boldsymbol{M}_2$, say $\mathcal{C}(\boldsymbol{M}_2)$. Since this has to hold for any choice of $\boldsymbol{\alpha}_1^{(1)}$ and $\boldsymbol{\alpha}_2^{(1)}$, one obtains the

condition that $\boldsymbol{\Delta} + \boldsymbol{v} \in \mathcal{C}(\boldsymbol{M}_2)$ for all $\boldsymbol{v} \in \mathcal{C}(\boldsymbol{M}_1)$. It is not difficult to show that this condition is equivalent to the following: $\boldsymbol{\Delta} \in \mathcal{C}(\boldsymbol{M}_2)$ and $\mathcal{C}(\boldsymbol{M}_1) \subseteq \mathcal{C}(\boldsymbol{M}_2)$.

Under the above condition, any record that is reached from $\boldsymbol{x}$ by P_{12} can also be reached by P_{21}. For the paths to be equivalent, it should also hold that any record that is reached by P_{21} can be reached by P_{12}. Thus, the previous argument may be repeated in the other direction. This yields the alternative condition: $\boldsymbol{\Delta} \in \mathcal{C}(\boldsymbol{M}_1)$ and $\mathcal{C}(\boldsymbol{M}_2) \subseteq \mathcal{C}(\boldsymbol{M}_1)$. Combining both conditions, I conclude that the two paths are equivalent if, and only if, it holds that $\mathcal{C}(\boldsymbol{M}_1) = \mathcal{C}(\boldsymbol{M}_2)$ and $\boldsymbol{\Delta} \in \mathcal{C}(\boldsymbol{M}_1)$ [$= \mathcal{C}(\boldsymbol{M}_2)$]. In practice, this condition may be verified by checking whether the four matrices $\boldsymbol{M}_1$, $\boldsymbol{M}_2$, $(\boldsymbol{M}_1, \boldsymbol{M}_2)$, and $(\boldsymbol{M}_1, \boldsymbol{\Delta})$ all have the same rank.

Case 3: g_1 and g_2 contain restricted parameters.
In this case, I consider the following system of (in)equalities:

$$\boldsymbol{T}_2\boldsymbol{S}_1\boldsymbol{\alpha}_1^{(1)} + \boldsymbol{S}_2\boldsymbol{\alpha}_2^{(1)} - \boldsymbol{T}_1\boldsymbol{S}_2\boldsymbol{\alpha}_2^{(2)} - \boldsymbol{S}_1\boldsymbol{\alpha}_1^{(2)} + \boldsymbol{\Delta} = \boldsymbol{0}, \tag{65}$$

$$\boldsymbol{R}_1\boldsymbol{\alpha}_1^{(1)} + \boldsymbol{d}_1 \odot \boldsymbol{0}, \tag{66}$$

$$\boldsymbol{R}_2\boldsymbol{\alpha}_2^{(1)} + \boldsymbol{d}_2 \odot \boldsymbol{0}, \tag{67}$$

$$\boldsymbol{R}_1\boldsymbol{\alpha}_1^{(2)} + \boldsymbol{d}_1 \odot \boldsymbol{0}, \tag{68}$$

$$\boldsymbol{R}_2\boldsymbol{\alpha}_2^{(2)} + \boldsymbol{d}_2 \odot \boldsymbol{0}. \tag{69}$$

By a similar argument as in case 2, the two paths are equivalent if, and only if, the following two conditions hold:

- for any fixed $\boldsymbol{\alpha}_1^{(1)}$ and $\boldsymbol{\alpha}_2^{(1)}$ that satisfy (66) and (67), it is possible to find values $\boldsymbol{\alpha}_1^{(2)}$ and $\boldsymbol{\alpha}_2^{(2)}$ that satisfy (65)–(69);
- for any fixed $\boldsymbol{\alpha}_1^{(2)}$ and $\boldsymbol{\alpha}_2^{(2)}$ that satisfy (68) and (69), it is possible to find values $\boldsymbol{\alpha}_1^{(1)}$ and $\boldsymbol{\alpha}_2^{(1)}$ that satisfy (65)–(69).

Both conditions may be verified with the aid of FM elimination. Namely, the first condition holds if, and only if, FM elimination of $\boldsymbol{\alpha}_1^{(2)}$ and $\boldsymbol{\alpha}_2^{(2)}$ from (65)–(69) yields a system of (in)equalities that is equivalent to (66) and (67). Analogously, the second condition holds if, and only if, FM elimination of $\boldsymbol{\alpha}_1^{(1)}$ and $\boldsymbol{\alpha}_2^{(1)}$ from (65)–(69) yields an equivalent system to (68) and (69). Thus, the problem of testing whether two paths are equivalent is reduced in this case to the problem of verifying whether two pairs of systems of linear (in)equalities are equivalent.

In more general notation, the latter problem consists of verifying whether two sets of the form $\{\boldsymbol{v} \mid \boldsymbol{A}\boldsymbol{v} + \boldsymbol{b} \odot_1 \boldsymbol{0}\}$ and $\{\boldsymbol{v} \mid \boldsymbol{C}\boldsymbol{v} + \boldsymbol{d} \odot_2 \boldsymbol{0}\}$ contain the same vectors. This problem may be solved in (at least) two ways. One possibility is to examine the following optimisation problems:

$$\left\{ \begin{array}{l} \max \sum_{j=1}^{p} z_{1j}(\boldsymbol{v}) \\ \text{under } \boldsymbol{C}\boldsymbol{v} + \boldsymbol{d} \odot_2 \boldsymbol{0} \end{array} \right\} \quad \text{and} \quad \left\{ \begin{array}{l} \max \sum_{j=1}^{p} z_{2j}(\boldsymbol{v}) \\ \text{under } \boldsymbol{A}\boldsymbol{v} + \boldsymbol{b} \odot_1 \boldsymbol{0} \end{array} \right\}, \tag{70}$$

where

$$z_{1j}(\boldsymbol{v}) = \begin{cases} \max\{0, -(\boldsymbol{A}\boldsymbol{v} + \boldsymbol{b})_j\} & \text{if } \odot_{1j} = \text{“}\geq\text{”}, \\ |\boldsymbol{A}\boldsymbol{v} + \boldsymbol{b}|_j & \text{if } \odot_{1j} = \text{“}=\text{”}. \end{cases}$$

and

$$z_{2j}(\boldsymbol{v}) = \begin{cases} \max\{0, -(\boldsymbol{C}\boldsymbol{v} + \boldsymbol{d})_j\} & \text{if } \odot_{2j} = \text{“}\geq\text{”}, \\ |\boldsymbol{C}\boldsymbol{v} + \boldsymbol{d}|_j & \text{if } \odot_{2j} = \text{“}=\text{”}. \end{cases}$$

If both maxima in (70) are equal to 0, then it follows that any vector that satisfies the first system of constraints must also satisfy the second system of constraints, and vice versa. In that case, both systems are equivalent. If at least one of the maxima in (70) is positive, then it follows

that the two systems are not equivalent. Both problems in (70) can be rewritten as mixed integer linear programming (MIP) problems and may be solved by standard software.

An alternative approach[5] is to consider each constraint separately. If the j^{th} constraint in the first system is an inequality (i.e., $\odot_{1j} =$ "$\geq$"), examine the following linear programming (LP) problem:

$$\left\{ \begin{array}{l} \max\ -(\boldsymbol{A}\boldsymbol{v} + \boldsymbol{b})_j \\ \text{under } \boldsymbol{C}\boldsymbol{v} + \boldsymbol{d} \odot_2 \boldsymbol{0} \end{array} \right\}. \tag{71}$$

If the maximum in (71) exceeds 0, it follows that there exists a vector that satisfies the second system of constraints but violates the j^{th} constraint in the first system; hence, the two systems are not equivalent. If the j^{th} constraint in the first system is an equality (i.e., $\odot_{1j} =$ "$=$"), one should examine (71) and the following LP problem:

$$\left\{ \begin{array}{l} \max\ (\boldsymbol{A}\boldsymbol{v} + \boldsymbol{b})_j \\ \text{under } \boldsymbol{C}\boldsymbol{v} + \boldsymbol{d} \odot_2 \boldsymbol{0} \end{array} \right\}. \tag{72}$$

If one of the maxima in (71) and (72) exceeds 0, it follows again that the two systems are not equivalent. Moreover, if this procedure is repeated for all constraints in the first system and none of the maxima are found to be positive, it follows that any vector that satisfies the second system also satisifies the first system. Finally, the same procedure may be applied in the other direction, i.e., by considering each constraint in the second system separately, given that the first system of constraints holds. If no positive maxima are found in either direction, it follows that the two systems are equivalent. Note that this alternative approach may be more convenient in practice, since LP problems can be solved with much less computational effort than MIP problems.

This concludes the discussion for the situation that $t = 2$. The situation that $t > 2$ is very similar. To test whether two paths in $\mathcal{P}(\boldsymbol{x}; \{g_1, \ldots, g_t\})$ are equivalent, expressions (61) and (62) should be replaced by analogous expressions that follow from (41). Again, the above three cases may be distinguished, depending on the occurrence of (restricted) parameters in the edit operations $g_1, \ldots, g_t$. Each case may be handled in the same way as before, except that the expressions for $\boldsymbol{\Delta}$, $\boldsymbol{M}_1$, $\boldsymbol{M}_2$, etc. become more complex. However, no new theoretical difficulties are encountered by going from $t = 2$ to $t \geq 2$.

In practice, it is likely that most edit operations will affect only a small number of variables at a time and involve only a small number of parameters. Hence, the mathematical problems described in this appendix should be computationally easy to solve, at least in comparison to the error localisation problem itself. Finally, it may be interesting to note that the record $\boldsymbol{x}$ occurs in the above conditions for equivalence only through its contribution to $\boldsymbol{\Delta}$ in expression (63). If the matrices $\boldsymbol{T}_1$ and $\boldsymbol{T}_2$ are commutative, this contribution vanishes. Hence, in this case the two paths P_{12} and P_{21} are equivalent or not, independently of the record to which the edit operations are applied.

As a simple application, I use the above result to show that the ordering of FH operations on a path does not matter. For notational convenience, I restrict attention to the case $t = 2$ but, again, a similar proof applies to longer paths. Let g_1 and g_2 be the FH operations that impute new values for the variables x_i and x_j, respectively. The second case above applies here, since each FH operation involves one unrestricted parameter. For the first edit operation, it holds that $\boldsymbol{T}_1 = \boldsymbol{I} - \boldsymbol{e}_i \boldsymbol{e}_i'$, $\boldsymbol{c}_1 = \boldsymbol{0}$, and $\boldsymbol{S}_1 = \boldsymbol{e}_i$, with $\boldsymbol{e}_i$ the i^{th} standard basis vector in $\mathbb{R}^p$. Similarly,

[5] Thanks to Mark van der Loo for suggesting this.

$\boldsymbol{T}_2 = \boldsymbol{I} - \boldsymbol{e}_j\boldsymbol{e}_j'$, $\boldsymbol{c}_2 = \boldsymbol{0}$, and $\boldsymbol{S}_2 = \boldsymbol{e}_j$. Using the fact that $\boldsymbol{e}_i'\boldsymbol{e}_j = 0$, it is easily seen that $\boldsymbol{T}_1\boldsymbol{T}_2 = \boldsymbol{T}_2\boldsymbol{T}_1$. Thus, the choice of input record $\boldsymbol{x}$ is not important here. It also follows that $\boldsymbol{\Delta} = \boldsymbol{0}$. Again using the fact that the standard basis vectors are orthogonal, it is found that

$$\boldsymbol{M}_1 = \left((\boldsymbol{I} - \boldsymbol{e}_j\boldsymbol{e}_j')\boldsymbol{e}_i, \boldsymbol{e}_j\right) = \left(\boldsymbol{e}_i, \boldsymbol{e}_j\right),$$

$$\boldsymbol{M}_2 = \left((\boldsymbol{I} - \boldsymbol{e}_i\boldsymbol{e}_i')\boldsymbol{e}_j, \boldsymbol{e}_i\right) = \left(\boldsymbol{e}_j, \boldsymbol{e}_i\right).$$

Clearly, it holds that $\mathcal{C}(\boldsymbol{M}_1) = \mathcal{C}(\boldsymbol{M}_2)$; in addition, $\boldsymbol{\Delta}$ is trivially contained in this subspace. Thus, the above condition for equivalence is satisfied and it follows that the order of g_1 and g_2 on a path does not matter.

Explanation of symbols

.	Data not available
*	Provisional figure
**	Revised provisional figure (but not definite)
x	Publication prohibited (confidential figure)
–	Nil
–	(Between two figures) inclusive
0 (0.0)	Less than half of unit concerned
empty cell	Not applicable
2013–2014	2013 to 2014 inclusive
2013/2014	Average for 2013 to 2014 inclusive
2013/'14	Crop year, financial year, school year, etc., beginning in 2013 and ending in 2014
2011/'12–2013/'14	Crop year, financial year, etc., 2011/'12 to 2013/'14 inclusive

Due to rounding, some totals may not correspond to the sum of the separate figures.

Publisher
Statistics Netherlands
Henri Faasdreef 312, 2492 JP The Hague
www.cbs.nl

Prepress: Statistics Netherlands, Grafimedia
Design: Edenspiekermann

Information
Telephone +31 88 570 70 70, fax +31 70 337 59 94
Via contact form: www.cbs.nl/information

Where to order
verkoop@cbs.nl
Fax +31 45 570 62 68

60083201414 X-10