Automatic editing with hard and soft edits

Some first experiences

Sander Scholtus and Sevinç Göksen

Discussion paper (201225)

Statistics Netherlands

The Hague/Heerlen, 2012

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
2011–2012	2011 to 2012 inclusive
2011/2012	average for 2011 up to and including 2012
2011/'12	crop year, financial year, school year etc. beginning in 2011 and ending in 2012
2009/'10– 2011/'12	crop year, financial year, etc. 2009/'10 to 2011/'12 inclusive

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

Publisher
Statistics Netherlands
Henri Faasdreef 312
2492 JP The Hague

Prepress
Statistics Netherlands
Grafimedia

Cover
Teldesign, Rotterdam

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

Where to order
E-mail: verkoop@cbs.nl
Telefax +31 45 570 62 68

Internet
www.cbs.nl

ISSN: 1572-0314

60083201225 X-10

Automatic editing with hard and soft edits – Some first experiences

Sander Scholtus and Sevinç Göksen

Summary: Soft edit rules, i.e. constraints which identify (combinations of) values that are suspicious but not necessarily incorrect, are an important element of many traditional, manual editing processes. It is desirable to use the information contained in these edit rules also in automatic editing. However, current algorithms for automatic editing are not well suited to use soft edits because they treat all edit rules as hard constraints: each edit failure is attributed to an error. Recently at Statistics Netherlands, a new automatic editing method has been developed that can distinguish between hard and soft edits. A prototype implementation of the new algorithm has been written in the R programming language. This paper reports some results of an application of this prototype to data from the Dutch Structural Business Statistics. The paper also introduces and tests several size measures of soft edit failures that can be used with the new automatic editing method.

Keywords: Automatic error localisation; Fellegi-Holt paradigm; Soft constraints; Numerical data; Simulation study

1. Introduction

Many national statistical institutes (NSIs) nowadays use automatic editing as a partial alternative to manual editing, to increase the efficiency of their editing processes. Most automatic editing methods treat a record of data in two steps: first, an attempt is made to identify the variables with erroneous or missing values (the *error localisation problem*) and subsequently new values are imputed for these variables to obtain a valid record (the *consistent imputation problem*). We shall focus on the error localisation problem here; for the consistent imputation problem, see De Waal *et al.* (2011) and the references therein.

For the data sets that occur in official statistics, one can typically formulate a large number of *edits*. Edits describe constraints that should be satisfied by the data. Both manual and automatic editing are guided by the information contained in the edits. In particular, attention is focused on records that do not satisfy – *i.e.* fail – certain edits. In practice, some edits are only failed by erroneous combinations of values; these are known as *hard edits* or fatal edits. An example of a hard edit is:

Profit = Turnover – Costs.

Other edits are failed by combinations of values that are unusual – and therefore suspicious – but that also have a nonzero probability of being correct; these are known as *soft edits* or query edits. An example of a soft edit could be:

$$Profit \,/\, Turnover \leq 60\%.$$

This edit would identify as suspicious all records for which the value of *Profit* exceeds 60% of the value of *Turnover*.

Current algorithms for automatic editing treat all edits as hard constraints: each edit failure is attributed to an error. On the other hand, soft edits are an important element of manual editing, which means that it is desirable to use the information contained in these edits also during automatic editing. Actually, there are currently only two ways to handle soft edits during automatic editing: either ignoring them or treating them as if they were hard edits. This is in fact an important practical difference between manual and automatic editing, which is likely to produce systematic differences between data that is edited manually and data that is edited automatically.[1]

Recently, a new error localisation method has been developed at Statistics Netherlands which can take soft edits into account in a more meaningful way, with the aim to improve the quality of automatically edited data. In this paper, we describe some of the first empirical results obtained with this new method. The remainder of this paper is organised as follows. Section 2 describes the error localisation problem as formulated by Fellegi and Holt (1976), as well as a generalisation that can distinguish between hard and soft edits. The latter formulation does not yet specify precisely how soft edit failures should be handled in the error localisation problem. Section 3 suggests several ways to do this. In Section 4, the suggested approaches are tested in a simulation study with data from the Dutch structural business statistics. Concluding remarks follow in Section 5.

2. Error Localisation Problems

2.1 An Error Localisation Problem without Soft Edits

We will assume throughout this paper that the data consists of numerical variables $(x_1, \ldots x_j, \ldots, x_p)$ and that the edits can be written as linear equalities or inequalities. That is, if we denote the set of edits by Ψ, each edit $\psi_k \in \Psi$ has one of the following two forms:

$$a_{k1}x_1 + \cdots + a_{kp}x_p + b_k = 0 \qquad (1)$$

[1] In analyses of the data editing process, it is customary to take the outcome of manual editing as the 'gold standard' with which the results of automatic editing should be compared. A critical evaluation of this assumption is beyond the scope of this paper, but see *e.g.* EDIMBUS (2007).

or

$$a_{k1}x_1 + \cdots + a_{kp}x_p + b_k \geq 0\,, \tag{2}$$

where $a_{k1}, \ldots, a_{kp}, b_k$ are numerical constants. A variable x_j is said to be *involved* in an edit ψ_k if and only if $a_{kj} \neq 0$. Clearly, the failure or non-failure of an edit is completely determined by the values of the variables involved in that edit.

For a given record $(x_1^0, \ldots, x_p^0)$, which might contain both errors and missing values, it is easily assessed which edits are failed and which edits are satisfied. (We make the working assumption that if an edit involves a variable with a missing value, the edit is failed.) If none of the edits are failed, the record is *consistent* with the edits and is supposed to require no editing. If the record is not consistent with the edits, then we would like to identify the erroneous values that cause the edit failures, *i.e.* to solve the error localisation problem. Actually, this problem is impossible to solve with certainty in any realistic application. In practice, we can only solve a less ambitious problem: to identify a subset of the variables which

(*i*) can be imputed so that the record becomes consistent with the edits, and

(*ii*) is 'optimal' according to some chosen criterion.

A solution to the latter problem (which is also commonly referred to as 'the error localisation problem') may or may not identify the actual erroneous values in a record. This depends in particular on the optimisation criterion and the set of edits used.

Fellegi and Holt (1976) proposed to solve the above-mentioned problem by minimising the number of variables to impute. This optimisation criterion has become widely-used for automatic editing at NSIs. Often a generalised version of the Fellegi-Holt paradigm is used, for which each variable is given a so-called *confidence weight*. These weights provide a means to take into account that some variables naturally contain more errors than others, for instance because they correspond to questions that many respondents find difficult to answer. Higher confidence weights are attached to variables that supposedly contain fewer errors, and vice versa. The objective now becomes to find a set of variables to impute that minimises the following distance measure:

$$D_{FH} = \sum_{j=1}^{p} w_j y_j\,, \tag{3}$$

where w_j denotes the confidence weight of variable x_j, and y_j is a binary variable with $y_j = 1$ if x_j is imputed and $y_j = 0$ otherwise. The original Fellegi-Holt paradigm is recovered as a special case by choosing all w_j equal, *e.g.* $w_j = 1$.

Mathematically, the problem of minimising (3) under the condition that the imputed record satisfies a given set of edits of the forms (1) and (2) can be written as a mixed integer linear programming problem (see *e.g.* Riera-Ledesma and Salazar-González,

2003). As such, it can be solved using commercially available solvers. In addition, some NSIs have developed and implemented specialised algorithms for solving the Fellegi-Holt based error localisation problem; see De Waal *et al.* (2011) for an overview. One example is a branch-and-bound algorithm due to De Waal and Quere (2003), which has been implemented at Statistics Netherlands in the automatic editing tool SLICE and, more recently, in the R package `editrules` (De Jonge and Van der Loo, 2011; Van der Loo and De Jonge, 2012).

2.2 An Error Localisation Problem with Soft Edits

In the above error localisation problem, it is tacitly assumed that all edits are hard edits. Hence, a subset of variables is only considered a feasible solution if it can be imputed to make the record consistent with all edits [*cf.* condition (*i*) above]. If soft edits have been specified for the data at hand, then these have to be either discarded or interpreted as hard edits during automatic editing. As mentioned in the introduction, this rather crude way of handling soft edits may be a source of systematic differences between automatic editing and manual editing. Di Zio *et al.* (2005) noted that the outcome of automatic editing may be significantly different depending on whether certain soft edits are included or excluded.

Scholtus (2011, 2012) proposed an alternative formulation of the error localisation problem which provides room for a more meaningful use of soft edits. For this formulation, it is assumed that the edit set Ψ is partitioned into two disjoint subsets: $\Psi = \Psi^H \cup \Psi^S$. The set Ψ^H contains the hard edits; the set Ψ^S contains the soft edits. The error localisation problem is now stated as the problem of identifying a subset of the variables which

(*i*) can be imputed so that the record becomes consistent with the edits in Ψ^H, and

(*ii*) minimises the following distance measure:

$$D = \lambda D_{FH} + (1-\lambda) D_{soft} . \tag{4}$$

In this expression, D_{FH} is given by (3), D_{soft} represents the costs associated with failed edits in Ψ^S, and $\lambda \in [0,1]$ is a parameter that balances the contributions of both terms to D.

In order to apply expression (4) in practice, a cost function D_{soft} has to be specified. The next section introduces several possible choices for D_{soft}. These choices can be divided into two broad classes: cost functions that only depend on the status of each soft edit (*i.e.* failed or satisfied), and cost functions that also depend on the amount by which each soft edit is failed. At first glance, cost functions from the latter class are the most attractive. Human editors interpret soft edit failures as measures of suspicion, and the degree of suspicion becomes higher as the size of the edit failure increases. For example, suppose that the soft edit $x_1 \leq 9$ is failed by two records with $x_1^0 = 10$ and $x_1^0 = 100$, respectively. Both records fail the edit and are therefore suspicious, but an editor would usually consider the second record more likely to be

in error than the first. By taking the sizes of soft edit failures into account in D_{soft} in (4), we can attempt to mimic this behaviour during automatic editing.

An algorithm for solving the error localisation problem of this subsection is presented in Scholtus (2011). This is in fact an extension of the above-mentioned branch-and-bound algorithm of De Waal and Quere (2003). Scholtus (2011) also noted that the second class of cost functions mentioned above, where D_{soft} depends explicitly on the sizes of soft edit failures, requires a more complex version of the algorithm than the first class. Thus, in spite of the intuitive appeal of using the sizes of edit failures in D_{soft}, it might be impractical to do so in realistic applications. Hence, it is also interesting to look at cost functions that do not depend on the sizes of edit failures, because they can be handled more efficiently.

3. Possible Measures of Soft Edit Failure

3.1 Preliminary Notation

Suppose that Ψ^S consists of the following soft edits: $\psi_1^S, \ldots, \psi_k^S, \ldots, \psi_{K_S}^S$. Let the binary variable z_k be defined so that $z_k = 1$ if edit ψ_k^S is failed and $z_k = 0$ otherwise. Since all edits are assumed to be linear equalities or inequalities, there exists a natural measure of the amount of edit failure. A given record $(x_1^0, \ldots, x_p^0)$ may be said to fail an edit of the form (1) by the amount

$$e_k = \left| a_{k1} x_1^0 + \cdots + a_{kp} x_p^0 + b_k \right| \tag{5}$$

and an edit of the form (2) by the amount

$$e_k = \max \left\{ 0, -(a_{k1} x_1^0 + \cdots + a_{kp} x_p^0 + b_k) \right\}. \tag{6}$$

In both cases, the value of e_k equals the absolute amount by which the left-hand-side of expression (1) or (2) would have to be shifted in order for the edit to become satisfied. In particular, $e_k = 0$ if the edit is satisfied and $e_k > 0$ otherwise. For the example from Subsection 2.2, the first record would have $e_k = 1$ and the second record would have $e_k = 91$ according to (6).

In Subsection 3.2, we will consider some simple variants of D_{soft} that can be computed from the values of $z_1, \ldots, z_{K_S}$ alone. Subsection 3.3 introduces a trick whereby the amount of edit failure can be taken into account to some extent while still using only the values of $z_1, \ldots, z_{K_S}$. Finally, Subsection 3.4 considers variants of D_{soft} that depend directly on $e_1, \ldots, e_{K_S}$. Göksen (2012) described some other variants which are omitted here to save space.

3.2 Cost Functions that Depend on $z_1, \ldots, z_{K_S}$

3.2.1 Sum of Failure Weights

Given the definition of D_{FH} as a sum of confidence weights in (3), it seems natural to use an analogous definition for D_{soft} in (4). That is, we may associate a weight s_k (which we will call a *failure weight*) to each soft edit ψ_k^S, and define D_{soft} as the sum of weights of failed soft edits:

$$D_{soft} = \sum_{k=1}^{K_S} s_k z_k \ . \tag{7}$$

The failure weights in (7) have a similar interpretation to the confidence weights in (3). Just as it may be thought that the variables are not a priori equally reliable, so it may be that some soft edits are considered 'harder' than others. In other words, the probability that a particular soft edit failure is caused by an erroneous combination of values may be different for different soft edits. Note that for hard edits this probability equals 1 by definition.

To illustrate the use of expression (4) if D_{soft} is defined by (7), we consider a small example. Suppose that the error localisation algorithm has found the following feasible solutions:

(*a*) It is possible to satisfy all edits (hard and soft) by imputing x_1 and x_2.

(*b*) It is possible to satisfy all edits except for ψ_1^S by imputing x_3.

(*c*) It is also possible to satisfy all edits except for ψ_4^S and ψ_5^S by imputing x_3.

Obviously, a different value would be imputed for x_3 in solutions (*b*) and (*c*). Expression (4) would yield the following values for these feasible solutions:

(*a*) $D = \lambda(w_1 + w_2)$.

(*b*) $D = \lambda w_3 + (1 - \lambda)s_1$.

(*c*) $D = \lambda w_3 + (1 - \lambda)(s_4 + s_5)$.

Hence, the choice of confidence weights w_j, failure weights s_k, and balancing parameter λ determines which of the three feasible solutions is the optimal one.

As a general point of interest, it should be noted that – since expression (4) is minimised – an edit with a high failure weight is less likely to be failed by the optimal solution to the error localisation problem than an edit with a low failure weight. Thus, soft edits that are relatively 'hard' in the above sense should be given a relatively high weight. We will now consider several ways to choose the failure weights in (7). We will distinguish different variants by adding superscripts to s_k.

If we have no prior information about the error detecting probability of the soft edits, then we may choose $s_k^A = 1$ for all $k = 1, \ldots, K_S$ by default. In the remainder of this section, we will assume that prior information is available in the form of a reference data set $\Re$ that has already been manually edited. Given this information, one obvious choice for s_k is to take the proportion of records in the edited data set that satisfy soft edit ψ_k^S:

$$s_k^B = \frac{1}{|\Re|} \sum_{i \in \Re} (1 - z_{ki}^{clean}) , \tag{8}$$

where $|\Re|$ denotes the number of records in $\Re$ and z_{ki}^{clean} denotes the value of z_k for the i^{th} record after manual editing. A high value of $0 \leq s_k^B \leq 1$ indicates a soft edit that is often satisfied in the edited reference data. Alternatively, s_k^B can be interpreted as an empirically estimated probability that a record satisfies edit ψ_k^S after manual editing. We may write this succinctly as $s_k^B = \hat{P}(z_k^{clean} = 0)$. This probability is estimated from the reference data and subsequently used during automatic editing to predict whether a record should be made to satisfy ψ_k^S or not.

Interestingly, the above method of prediction can be improved, at least from a theoretical point of view. Analogous to (8), the following empirically estimated probabilities may be computed from the reference data:

$$\hat{P}(z_k^{raw} = 1) = \frac{1}{|\Re|} \sum_{i \in \Re} z_{ki}^{raw} ,$$

$$\hat{P}(z_k^{raw} = 1 \wedge z_k^{clean} = 0) = \frac{1}{|\Re|} \sum_{i \in \Re} z_{ki}^{raw} (1 - z_{ki}^{clean}) .$$

The first expression estimates the probability that ψ_k^S is failed prior to editing; note that z_{ki}^{raw} is the counterpart of z_{ki}^{clean} in the unedited version of the reference data set. The second expression represents the estimated probability that a record fails edit ψ_k^S prior to editing *and* satisfies the edit after editing. According to the definition of conditional probability,

$$\hat{P}(z_k^{clean} = 0 \mid z_k^{raw} = 1) = \frac{\hat{P}(z_k^{raw} = 1 \wedge z_k^{clean} = 0)}{\hat{P}(z_k^{raw} = 1)} .$$

When solving the error localisation problem for a given record, the value of z_k^{raw} is known. Moreover, only the failure weights of the edits with $z_k^{raw} = 1$ are relevant for evaluating expression (7). Thus instead of using the above unconditional probability $\hat{P}(z_k^{clean} = 0)$ to predict whether ψ_k^S should be satisfied or not, we may also use the conditional probability $\hat{P}(z_k^{clean} = 0 \mid z_k^{raw} = 1)$, which might give a more accurate prediction. With this in mind, we define the following failure weight:

$$s_k^C = \hat{P}(z_k^{clean} = 0 \mid z_k^{raw} = 1) = \frac{\sum_{i \in \Re} z_{ki}^{raw}(1 - z_{ki}^{clean})}{\sum_{i \in \Re} z_{ki}^{raw}}.$$

As we have seen, both s_k^B and s_k^C can be interpreted as estimated probabilities. Inserting these probabilities directly into the linear expression (7) might not give the best results. A simple alternative approach is to derive categorical versions of these weights, which we denote by $s_k^{B(cat)}$ and $s_k^{C(cat)}$, respectively. Formally, we map the interval $[0,1]$ onto a finite (in fact rather small) set of values $\{s^{(1)}, \ldots, s^{(m)}\}$. For instance, taking $m = 3$ we could choose two cut-off values $a_{low} \leq a_{high}$ and define

$$s_k^{B(cat)} = \begin{cases} 0.5 & \text{if } s_k^B < a_{low} \\ 1 & \text{if } a_{low} \leq s_k^B \leq a_{high} \\ 1.5 & \text{if } s_k^B > a_{high} \end{cases} \tag{9}$$

with an analogous definition for $s_k^{C(cat)}$. This might give better results than using s_k^B and s_k^C directly if the handling of soft edit failures during manual editing depends on the above estimated probabilities in a strongly non-linear manner.

3.2.2 A Nearest-Neighbour Method

A common feature of the above methods for defining s_k is that every record that fails a particular soft edit receives the same failure weight. It seems interesting to also look at methods that allow the failure weights to be different across records. One relatively simple method works as follows. First, we identify, for each unedited record, its nearest neighbour in the edited reference data set. Next, we assign a failure weight s_k to edit ψ_k^S separately for each unedited record, based on the status of that edit for the nearest-neighbour record: if the nearest neighbour satisfies ψ_k^S then we take $s_k = t_{high}$ and otherwise we take $s_k = t_{low}$, for some chosen values $t_{low} << t_{high}$. That is to say, a soft edit receives a high (low) failure weight – and is thus more (less) likely to be satisfied after editing – for records with nearest neighbours that satisfy (fail) that edit. In this way, the method tries to infer the correct interpretation of a soft edit failure for an unedited record (*i.e.* whether the involved values are incorrect or merely unusual) from its edited nearest neighbour. Recall that a soft edit failure in an edited reference record corresponds with a combination of values that is unusual but correct.

Apart from t_{low}, t_{high}, this method is parametrised further by the choice of a distance function that is to be minimised by the nearest neighbour. In the simulation study of Section 4, we will define the distance between an unedited record $(x_{l1}^{raw}, \ldots, x_{lp}^{raw})$ and an edited reference record $(x_{i1}^{clean}, \ldots, x_{ip}^{clean})$ as follows:

$$d_{NN}(l,i) = \sum_{j=1}^{p} | \tilde{x}_{lj}^{raw} - \tilde{x}_{ij}^{clean} | ,$$

where the values $\tilde{x}_{lj}^{raw}, \tilde{x}_{ij}^{clean}$ are obtained by standardising $x_{lj}^{raw}, x_{ij}^{clean}$ according to the observed means and standard deviations in the reference data:

$$\tilde{x}_{lj}^{raw} = \frac{x_{lj}^{raw} - \mu_j^{clean}}{\sigma_j^{clean}} \quad \text{and} \quad \tilde{x}_{ij}^{clean} = \frac{x_{ij}^{clean} - \mu_j^{clean}}{\sigma_j^{clean}},$$

with

$$\mu_j^{clean} = \frac{1}{|\Re|} \sum_{i \in \Re} x_{ij}^{clean} \quad \text{and} \quad (\sigma_j^{clean})^2 = \frac{1}{|\Re| - 1} \sum_{i \in \Re} (x_{ij}^{clean} - \mu_j^{clean})^2 .$$

The nearest neighbour of the l^{th} unedited record is thus obtained as the edited reference record with index

$$i_l = \arg \min_{i \in \Re} d_{NN}(l,i) .$$

Using $\tilde{x}_{lj}^{raw}, \tilde{x}_{ij}^{clean}$ instead of $x_{lj}^{raw}, x_{ij}^{clean}$ ensures that all variables have similar-sized contributions to d_{NN}. Of course, more elaborate distance functions are also possible.

3.2.3 Maximum of Failure Weights

To conclude Subsection 3.2, we note that alternative functions of fixed failure weights could be used for D_{soft} instead of expression (7). One interesting choice that will be considered in the simulation study below is the maximum function:

$$D_{soft} = \max_{k=1,\ldots,K_S} s_k z_k . \tag{10}$$

In this case, the soft edits' contribution to (4) is completely determined by the failed soft edit with the highest failure weight. In the example given at the beginning of this subsection, the value of D would remain the same for solutions (*a*) and (*b*). Solution (*c*) on the other hand would now have either $D = \lambda w_3 + (1-\lambda) s_4$ or $D = \lambda w_3 + (1-\lambda) s_5$, depending on whether $s_4 > s_5$ or $s_4 < s_5$.

3.3 Quantile Edits

A drawback of the variants considered so far is that they do not take the sizes of soft edit failures into account. In Subsection 3.4, we will discuss variants of D_{soft} that take the sizes of soft edit failures into account directly. As mentioned above, this approach requires a more complex error localisation algorithm. The current subsection describes an indirect way to take the edit failure sizes into account, without increasing the complexity of the algorithm.

The basic idea is as follows. Consider a soft edit ψ_k^S of the form (2)[2] and suppose that this edit is replaced by the following system of soft edits:

$$\begin{cases} \psi_{k(1)}^S: & a_{k1}x_1 + \cdots + a_{kp}x_p + b_{k(1)} \geq 0 \\ \psi_{k(2)}^S: & a_{k1}x_1 + \cdots + a_{kp}x_p + b_{k(2)} \geq 0 \\ \quad\vdots & \qquad\qquad\qquad\qquad\quad \vdots \\ \psi_{k(R)}^S: & a_{k1}x_1 + \cdots + a_{kp}x_p + b_{k(R)} \geq 0 \end{cases} \tag{11}$$

where the constant terms are chosen so that

$$b_k = b_{k(1)} < b_{k(2)} < \cdots < b_{k(R)}. \tag{12}$$

Suppose in addition that the edits in this system are given failure weights $s_{k(1)}, s_{k(2)}, \ldots, s_{k(R)}$, and that expression (7) is used for D_{soft}. It is not difficult to show that, for $r = 2, \ldots, R$, any record which fails edit $\psi_{k(r)}^S$ automatically also fails the edits $\psi_{k(1)}^S, \ldots, \psi_{k(r-1)}^S$. Hence, a record which fails edit $\psi_{k(r)}^S$ actually receives a contribution of $s_{k(1)} + \cdots + s_{k(r)}$ to cost function (7). In this way, larger edit failures with respect to the original edit ψ_k^S implicitly receive larger contributions to D_{soft}, for the simple reason that they fail more edits in system (11).

Assuming again that prior information is available in the form of a reference data set that has been edited manually, an interesting application of the above idea could be as follows. Compute the value of $Q = a_{k1}x_1 + \cdots + a_{kp}x_p$ for each record in the edited data set. Next, consider the univariate distribution of Q and let $Q^{(\alpha)}$ denote the $\alpha \times 100\,\%$ quantile of this distribution (for $\alpha \in [0,1]$). Let α_1 be the largest percentage point for which $Q^{(\alpha)} < -b_k$. Clearly, soft edit (2) is failed by $\alpha_1 \times 100\,\%$ of the edited records. Now choose specific values $0 \leq \alpha_R < \cdots < \alpha_2 < \alpha_1$ and define $b_{k(r)} = -Q^{(\alpha_r)}$ for $r = 1,2,\ldots,R$. These values satisfy property (12) and therefore may be inserted as constant terms into system (11). Göksen (2012) suggested the term *quantile edits* to refer to a system of edits obtained in this manner. An advantage of using quantile edits is that these are soft edits with prescribed failure rates: by construction, the quantile edit $\psi_{k(r)}^S$ is failed by $\alpha_r \times 100\,\%$ of the edited records.

In practice many soft edits are *ratio edits*, *i.e.* bivariate edits of the form $x_{j_1} / x_{j_2} \geq c$, where the variables x_{j_1} and x_{j_2} are constrained to be non-negative by the hard edits. A ratio edit can be linearised as $x_{j_1} - cx_{j_2} \geq 0$, which is an edit of the form (2). For a soft edit ψ_k^S of this kind, a variation on the above idea can be applied by varying the value of coefficient c:

[2] Note that we can restrict attention to inequality edits without essential loss of generality, since an equality edit can always be replaced by two equivalent inequality edits.

$$\begin{cases} \psi^{S}_{k(1)}: & x_{j_1} - c_{(1)} x_{j_2} \geq 0 \\ \psi^{S}_{k(2)}: & x_{j_1} - c_{(2)} x_{j_2} \geq 0 \\ \quad \vdots & \quad \vdots \\ \psi^{S}_{k(R)}: & x_{j_1} - c_{(R)} x_{j_2} \geq 0 \end{cases}$$

where the coefficients have to be chosen so that $c_{(R)} < \cdots < c_{(2)} < c_{(1)}$. Again it can be shown that $\psi^{S}_{k(r)}$ is failed only if $\psi^{S}_{k(1)}, \ldots, \psi^{S}_{k(r-1)}$ are also failed (for $r = 2, \ldots, R$). As before, we can use this fact to ensure that larger edit failures with respect to the ratio edit receive larger contributions to D_{soft}. In particular, quantile edits can be generated from the univariate distribution of $Q' = x_{j_1} / x_{j_2}$ in a reference data set, by taking $c_{(r)} = Q'^{(\alpha_r)}$ for some choice of values $0 \leq \alpha_R < \cdots < \alpha_2 < \alpha_1 < 1$. Again, it automatically holds that the quantile edit $\psi^{S}_{k(r)}$ is failed by $\alpha_r \times 100\,\%$ of the edited records. An analogous construction exists for ratio edits of the form $x_{j_1} / x_{j_2} \leq c$; see Göksen (2012).

As an example, suppose that one intends to use a ratio edit of the form $x_1 / x_2 \geq c$. Upon examining the univariate distribution of $Q' = x_1 / x_2$ in the edited reference data, it is found that the 10%, 5% and 1% quantiles of Q' are 0.75, 0.60, and 0.10, respectively. Consider the following system of soft edits:

$$\begin{cases} x_1 - 0.75\, x_2 \geq 0 \\ x_1 - 0.60\, x_2 \geq 0 \\ x_1 - 0.10\, x_2 \geq 0 \end{cases}$$

Given our reference data, we expect that these edits are failed by 10%, 5%, and 1% of records without errors, respectively. Suppose in addition that each of the above edits is given a failure weight of 1. An unedited record with, say, $x_1^0 / x_2^0 = 0.65$ only fails the 10% quantile edit and thus receives a total failure weight of 1 during automatic editing. Similarly, an unedited record with $x_1^0 / x_2^0 = 0.40$ fails the 5% and 10% quantile edits and receives a total failure weight of 2, while an unedited record with $x_1^0 / x_2^0 = 0.05$ receives a total failure weight of 3. Thus the total failure weight of a record increases as its ratio x_1 / x_2 becomes less likely to be correct. On the other hand, the methods discussed in Subsection 3.2 would assign the same failure weight to each of the above records (with the possible exception of the nearest-neighbour method).

A potential drawback of the quantile edit approach is that it increases the number of edits. In general, error localisation algorithms tend to show a dramatic increase in the amount of required computing time and memory as the number of edits becomes larger (for an explanation of this effect, see *e.g.* De Jonge and Van der Loo, 2011). Therefore, it seems advisable to limit the number of additional edits introduced by this method. In the simulation study to be discussed below we used $R = 3$.

3.4 Cost Functions that Depend on $e_1, \ldots, e_{K_S}$

3.4.1 Introduction

We will now consider three variants of D_{soft} that depend on $e_1, \ldots, e_{K_S}$, the measures of individual edit failure size introduced in Subsection 3.1. First of all, it should be noted that e_k as defined by (5) or (6) is not invariant to arbitrary changes in the formulation of the soft edits. In particular, we can multiply the left-hand-side of (1) or (2) by an arbitrary constant $\theta > 0$ to obtain an equivalent edit. However, all values of e_k would be inflated by a factor θ under this operation. By the same token, the typical magnitude of e_k may not be directly comparable for different soft edits. Hence, it seems appropriate to standardise the values of $e_1, \ldots, e_{K_S}$ in some way before using them in an expression for D_{soft}.

We assume as before that there is a reference data set $\Re$ available. From this data set we may estimate the mean and variance of e_k in manually edited data:

$$\hat{\bar{e}}_k^{clean} = \frac{1}{|\Re_0|} \sum_{i \in \Re_0} e_{ki}^{clean} , \qquad \hat{\sigma}_k^2 = \frac{1}{|\Re_0| - 1} \sum_{i \in \Re_0} (e_{ki}^{clean} - \hat{\bar{e}}_k^{clean})^2 ,$$

where e_{ki}^{clean} denotes the value of e_k for the edited version of the i^{th} record in the reference data; $\Re_0$ denotes the subset of edited records in $\Re$ that fail at least one (soft) edit. That is, we estimate the above quantities conditional on the event that a record has at least one non-zero e_k. The rationale behind this is that the expression used for D_{soft} in (4) is only relevant for records that satisfy this condition. Therefore, we are not interested in the distribution of e_k outside $\Re_0$.

3.4.2 Sum of Standardised Edit Failures

One possible method to standardise e_k for use in D_{soft} is as follows. Dividing each e_k by its estimated standard deviation to obtain $e_k^s = e_k / \hat{\sigma}_k$, we may define in analogy with (7):

$$D_{soft} = \sum_{k=1}^{K_S} e_k^s . \tag{13}$$

Note that by construction only the failed soft edits have a non-zero contribution to (13), just as in (7).

3.4.3 Mahalanobis Distance of Edit Failures

A second possible method computes the well-known *Mahalanobis distance* between the vector $\mathbf{e} = (e_1, \ldots, e_{K_S})'$ of edit failure sizes and a vector of zeros, corresponding with no soft edit failures:

$$D_{soft} = D_M(\mathbf{e}, \mathbf{0}) = \sqrt{\mathbf{e}' \hat{\boldsymbol{\Sigma}}^{-1} \mathbf{e}}\,, \tag{14}$$

where $\hat{\boldsymbol{\Sigma}}$ denotes the variance-covariance matrix of e_k in $\Re_0$. This distance measure also takes potential correlations between edit failures into account in the standardisation. Unlike the variants of D_{soft} considered so far, expression (14) cannot be written as a linear combination of $z_1, \ldots, z_{K_S}$ or $e_1, \ldots, e_{K_S}$. Taking the Mahalanobis distance of edit failures was suggested by Hedlin (2003) in a different context.

3.4.4 Logistic Regression

A third possible method proceeds by fitting a logistic regression model to a combined data set containing both the edited and unedited reference data:

$$\log \frac{P(q=1)}{1-P(q=1)} = \beta_0 + \beta_1 e_1^s + \cdots + \beta_{K_S} e_{K_S}^s + \varepsilon\,. \tag{15}$$

The binary target variable q in (15) is defined so that $q = 1$ if a record contains at least one error and $q = 0$ otherwise. Note that the standardised soft edit failures $e_1^s, \ldots, e_{K_S}^s$ are used as explanatory variables. Having obtained estimates $b_0, b_1, \ldots, b_{K_S}$ for the parameters in (15), we define

$$D_{soft} = \hat{P}(q=1) = \frac{1}{1 + \exp[-(b_0 + b_1 e_1^s + \cdots + b_{K_S} e_{K_S}^s)]}\,. \tag{16}$$

When applied during error localisation, the value of D_{soft} according to expression (16) can be interpreted as an estimated probability that the current soft edit failures are caused by an error rather than an unusual but correct combination of values. Göksen (2012) also considered logistic regression models with other explanatory variables.

4. Simulation Study

In this section, we will present some results of a simulation study in which the above methods for defining D_{soft} were tested. For the purpose of this simulation study, a prototype implementation of the error localisation algorithm from Scholtus (2011) was written using the R programming language. This prototype makes extensive use of the error localisation functionality of the `editrules` package[3] (De Jonge and Van der Loo, 2011; Van der Loo and De Jonge, 2012).

[3] The prototype was based on version 0.8 of the `editrules` package, being the most recent version that existed at the time. Since then, the package has received many updates. At the time of writing (September 2012), `editrules` v2.5 is available on the CRAN website.

We used two data sets in the simulation study. All data was obtained from manually edited records on medium-sized businesses (*i.e.* businesses with 10–99 employed persons) from the Dutch wholesale structural business statistics of 2007. The manually edited data was assumed to be error-free. In both cases, we selected half of the original edited data as reference data. The other half was used to provide test data for automatic editing. For data set 1, we introduced synthetic errors in the edited data by randomly modifying about 20% of the values; see Scholtus (2011) for details on the error mechanism used. For data set 2, we used the unedited data – containing actual errors made by actual respondents – as test data. The number of records to edit equalled 728 for data set 1 and 580 for data set 2. Missing values only occurred in data set 2.

Tables A.1 through A.4 in the Appendix display the variables and edits for both data sets. As can be seen, most soft edits for data set 1 are linearised ratio edits. For instance, the first soft edit in Table A.2 expresses that net turnover from wholesale should represent at least 50% of net total turnover. For data set 2, many edits were originally specified in IF-THEN form. Since the prototype could only handle linear numerical edits (but see Section 5), we rewrote the IF-THEN edits so that the IF conditions only contained auxiliary variables which, for the purpose of the simulation study, could be considered error-free. In this way, we obtained a different set of linear numerical edits for each record, depending on the values of the auxiliary variables.

For both data sets, we initially chose all confidence weights equal to 1 and experimented only with the specification of D_{soft}. Unless otherwise indicated, the results reported here were obtained with $\lambda = 1/2$ in (4), *i.e.* with D taken as the unweighted mean of D_{FH} and D_{soft} (or, equivalently, their unweighted sum).

Since the distribution of errors was assumed known for the data sets in this simulation study, we could directly evaluate the success of each error localisation effort. Consider the following 2×2 contingency table for a given editing approach:

		detected:	
		error	no error
true:	error	*TP*	*FN*
	no error	*FP*	*TN*

where the classification used refers to individual values in the data set. From this table, we computed the following quality indicators:

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

These three indicators evaluate how well an editing approach manages to identify individual values as correct or erroneous: α measures the proportion of true errors that were missed (false negatives); β measures the proportion of correct values mistaken for errors (false positives); and γ measures the overall proportion of wrong decisions. Clearly, a successful editing approach should achieve low values of α, β, and γ. For an alternative evaluation measure, we also calculated the proportion of records for which exactly the right solution was found – that is, the solution that identifies as erroneous all erroneous values and only these. This fourth indicator is denoted by δ. To be successful, an editing approach should achieve a high value of δ.[4]

Table 1 displays the evaluation results for a number of error localisation approaches for both data sets. The first two lines present the results of, respectively, not using soft edits and using all soft edits as if they were hard edits. As noted above, these are the two ways that are available to handle soft edits in the error localisation problem based on (3). For data set 1, it is clear that using the soft edits as hard edits is not a good idea. In fact, simply ignoring the soft edits gave better results on all evaluation measures except α. For data set 2, using all soft edits as if they were hard edits was not always possible, because for some records an infeasible problem was encountered if all hard and soft edits had to be satisfied simultaneously. Hence, we did not evaluate this approach for data set 2.

The next ten lines contain results for the methods introduced in Subsection 3.2, with D_{soft} defined by expression (7). For both data sets, it is seen that a significant improvement compared to not using soft edits was already obtained by taking the soft edits into account with all failure weights equal to $s_k^A = 1$. Differentiating the failure weights of soft edits according to the choice s_k^B led to a further improvement for data set 1, but not for data set 2. Contrary to our expectations, the alternative failure weight s_k^C did not improve on s_k^B; in fact, this choice made the results significantly worse for data set 1.

We also experimented with several categorised versions $s_k^{B(cat)}$ and $s_k^{C(cat)}$. The results in Table 1 were obtained with expression (9) and its analogue for $s_k^{C(cat)}$, where the optimal choices of a_{low} and a_{high} depended on the data set (see Göksen, 2012). For data set 1, the categorised versions did not perform better than the original weights, but for data set 2 we observed a clear improvement.

[4] In the presence of missing data, a choice has to be made whether or not to include the missing values – which are always correctly identified as erroneous – in the count of true positives (*TP*). Somewhat arbitrarily, we chose not to include them. More precisely, we calculated the above contingency table for the *non-missing* values only. In addition, higher values of δ may be expected in the presence of missing data for the reason given in the first sentence of this footnote. Again somewhat arbitrarily, we did not adjust the calculation of δ to compensate for this effect.

Table 1. Evaluation results for both data sets

editing approach	data set 1				data set 2			
	α	β	γ	δ	α	β	γ	δ
no soft edits used	0.364	0.047	0.115	40.2%	0.551	0.003	0.131	58.4%
all edits used as hard edits	0.232	0.131	0.153	36.8%	n/a	n/a	n/a	n/a
<u>failure weights: sum (Subsections 3.2.1–3.2.2)</u>								
weights A	0.227	0.060	0.096	47.3%	0.435	0.014	0.121	63.4%
weights B	0.253	0.037	0.083	52.1%	0.508	0.004	0.125	60.9%
weights C	0.332	0.038	0.101	43.3%	0.524	0.003	0.126	60.7%
weights B(cat)	0.224	0.053	0.089	50.0%	0.432	0.009	0.116	64.5%
weights C(cat)	0.333	0.038	0.101	43.1%	0.444	0.007	0.117	64.5%
weights nearest-neighbour (t_{high} = 1.25, t_{low} = 0.1)	0.264	0.048	0.094	48.9%	0.443	0.007	0.117	64.5%
weights B, λ = 0.75	0.318	0.034	0.095	43.4%	0.551	0.003	0.131	58.6%
weights B(cat), λ = 0.75	0.262	0.036	0.084	51.6%	0.536	0.002	0.128	60.2%
weights B, λ = 0.3	0.234	0.094	0.124	41.9%	0.303	0.057	0.129	55.3%
weights B(cat), λ = 0.3	0.231	0.082	0.114	44.5%	0.298	0.043	0.117	60.2%
<u>failure weights: max (Subsection 3.2.3)</u>								
weights B	0.336	0.039	0.103	43.1%	0.538	0.002	0.128	59.7%
weights B(cat)	0.254	0.046	0.091	50.8%	0.487	0.005	0.122	63.3%
<u>quantile edits (10%, 5%, 1%) (Subsection 3.3)</u>								
weights 1/3, 1/3, 1/3	0.247	0.032	0.078	54.4%	0.481	0.005	0.121	63.4%
weights 0.9, 0.05, 0.05	0.214	0.037	0.075	56.5%	0.466	0.005	0.119	63.8%
<u>amounts of edit failure</u>								
sum of standardised edit failures (Subsection 3.4.2)	0.273	0.050	0.098	49.2%	-	-	-	-
Mahalanobis distance (Subsection 3.4.3)	0.328	0.049	0.109	46.8%	-	-	-	-
logistic regression (Subsection 3.4.4)	0.307	0.034	0.092	45.9%	-	-	-	-

For the nearest-neighbour method we experimented with several choices of (t_{high}, t_{low}). Table 1 shows the results for $(t_{high}, t_{low}) = (1.25, 0.1)$; see Göksen (2012) for several other combinations of values. This method also appeared to give better results for data set 2 than for data set 1.

Experiments with different values of λ in (4) did not yield significant improvements over the previous results, as indicated by the selected results with $\lambda = 0.75$ and $\lambda = 0.3$ shown in Table 1. The same can be said of the results obtained with expression (10) for D_{soft} instead of expression (7). More and similar results with λ and expression (10) can be found in Göksen (2012). A clear effect can be seen in Table 1 of the choice of λ on the values of α and β for the second data set and, to a lesser extent, for the first data set.

The next two lines relate to quantile edits. For both data sets, we replaced each soft edit with $R = 3$ quantile edits, namely those corresponding to the 10%, 5%, and 1% quantile of the underlying distribution (*i.e.* $\alpha_1 = 0.1$, $\alpha_2 = 0.05$, and $\alpha_3 = 0.01$). The first line shows the results obtained by giving each quantile edit a failure weight of 1/3; the second line shows the results when the 10%, 5%, and 1% quantile edits were given failure weights of 0.9, 0.05, and 0.05, respectively. These choices produced the best results seen so far for data set 1. Moreover, the results for data set 2 also ranked among the best seen so far. Both of the above sets of weights are such that the maximal implicit failure weight for a record that fails the original soft edit equals 1. Göksen (2012) also reported results for other choices of weights, but these did not improve on the results shown here.

The final three lines present the results of approaches for which D_{soft} depends explicitly on the sizes of soft edit failures. Due to time constraints, we could only evaluate these approaches for the first data set. Somewhat contrary to expectation, the results did not improve on the best results seen so far. In particular, the results of the Mahalanobis distance and the logistic regression approach were rather poor in comparison with some of the other approaches that use soft edits.

Finally, as noted above, the results presented so far were obtained with all confidence weights equal to 1. If the confidence weights are not all equal to 1, this should be taken into account in the definition of D_{soft}, since otherwise the two terms in (4) are weighted unintentionally. An easy solution, which works for the above approaches that do not depend explicitly on the sizes of soft edit failures, is to multiply each failure weight with an appropriate factor. This factor should inflate D_{soft} to the same magnitude as D_{FH} in (4). Experimental results with the above data sets (not reported here, but see Göksen, 2012) suggested that a good approach is to multiply the failure weight of soft edit ψ_k^S with the harmonic mean of the confidence weights of all variables involved in ψ_k^S. That is, if we denote the set of variables involved in ψ_k^S by $\Im_k$, the failure weight s_k should be multiplied by

$$\frac{|\Im_k|}{\sum_{j\in\Im_k} w_j^{-1}} = \frac{\sum_{j\in\Im_k} 1}{\sum_{j\in\Im_k} w_j^{-1}} = \frac{\sum_{j\in\Im_k} w_j^{-1} w_j}{\sum_{j\in\Im_k} w_j^{-1}} = \sum_{j\in\Im_k} p_j w_j \,, \tag{17}$$

with $p_j = w_j^{-1} \Big/ \sum_{j'\in\Im_k} w_{j'}^{-1}$. The rightmost expression in (17) shows that this factor may be interpreted as follows: it is the expected value of the contribution to D_{FH} when exactly one of the variables involved in ψ_k^S is imputed, under the assumption that, for each variable, the probability to be selected for imputation is proportional to the reciprocal of its confidence weight.

5. Conclusion

In this paper, we presented several ways to use soft edits together with hard edits in automatic error localisation. We also reported some empirical results for these methods in a simulation study. These results showed that, at least with respect to the four quality indicators defined in Section 4, all methods considered here have the potential to improve the quality of automatic error localisation in comparison with the method that uses only the hard edits, which is currently the most common approach in automatic editing.[5] Some methods were more successful in this respect than others, however. In addition, the results depended on the data set at hand and on the type of quality indicator that is considered the most important (*e.g.* false positives versus false negatives).

Overall, the method of quantile edits appears to be the best choice of the methods considered here. But there may be room for further improvement. In particular, it seems likely that the approach to let D_{soft} depend explicitly on the sizes of soft edit failures could be used to more effect than we have done here.

The description in this paper was restricted to numerical data and edits, but this restriction is not necessary. A similar error localisation problem with hard and soft edits can be formulated and solved for categorical and mixed data; see Scholtus (2011). Future work will include a simulation study that involves mixed data and edits.

In summary, we can say that the methodology presented here has the potential to increase the quality of automatic error localisation. More research is needed, however, to find a generic way to choose the most appropriate method in concrete applications.

[5] In fact, if δ is used to measure the quality of error localisation, then using any of the methods with soft edits considered in Table 1 would yield better results than using only hard edits, with one exception: taking expression (7) for D_{soft} with failure weights according to expression (8) and with $\lambda = 0.3$ would slightly deteriorate the results for data set 2.

References

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

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

Di Zio, M., U. Guarnera, and O. Luzi (2005), Improving the Effectiveness of a Probabilistic Editing Strategy for Business Data. Report, ISTAT, Rome.

EDIMBUS (2007), *Recommended Practices for Editing and Imputation in Cross-Sectional Business Surveys*. Manual, prepared by ISTAT, Statistics Netherlands, and SFSO.

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

Göksen, S. (2012), Automatic Error Localisation with Hard and Soft Constraints at Statistics Netherlands. Master Thesis (in Dutch), Statistics Netherlands, The Hague.

Hedlin, D. (2003), Score Functions to Reduce Business Survey Editing at the U.K. Office for National Statistics. *Journal of Official Statistics* **19**, pp. 177-199.

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.

Scholtus, S. (2011), Automatic Editing with Soft Edits. Discussion Paper 201130, Statistics Netherlands, The Hague.

Scholtus, S. (2012), Automatic Editing with Hard and Soft Edits. *Survey Methodology*, accepted for publication.

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.

Appendix: Metadata for the data sets used in the simulation study

Table A.1. Variables in data set 1

	variable description
x_1	net turnover from wholesale
x_2	net turnover from other activities
x_3	net total turnover
x_4	net turnover from wholesale[a)]
x_5	internal revenues
x_6	external revenues, type 1
x_7	external revenues, type 2
x_8	total other revenues
x_9	total operating revenues
x_{10}	total operating costs
x_{11}	pre-tax results
x_{12}	number of employees on payroll

[a)] Alternative calculation as total of a product group breakdown.

Table A.2. Edits for data set 1

hard edits		soft edits	
$x_1 + x_2 = x_3$	[1]	$x_1 - 0.5x_3 \geq 0$	[1]
$x_1 = x_4$	[2]	$x_3 - 0.9x_9 \geq 0$	[2]
$x_5 + x_6 + x_7 = x_8$	[3]	$x_5 + x_6 \geq x_7$	[3]
$x_3 + x_8 = x_9$	[4]	$x_9 - 50x_{12} \geq 0$	[4]
$x_9 - x_{10} = x_{11}$	[5]	$5000x_{12} - x_9 \geq 0$	[5]
$x_j \geq 0$ (for all $j \neq 11$)	[6–16]	$0.4x_9 - x_{11} \geq 0$	[6]
		$x_{11} + 0.1x_9 \geq 0$	[7]
		$x_{12} \geq 1$	[8]
		$x_{12} \geq 5$	[9]
		$x_{12} \leq 100$	[10]

Table A.3. Variables in data set 2

	variable description[a)]
h_1	social security payments
h_2	gross wages of employees on payroll
h_3	costs of temporary employees
h_4	costs of hired employees
h_5	net total turnover
h_6	fee for lending out employees
h_7	other revenues
h_8	total personnel costs
sc	size class
x_1	total number of employed persons
x_2	total employed persons (in FTE)
x_3	employees on payroll (in FTE)
x_4	number of employees on payroll
x_5	number of other employed persons
x_6	number of hired employees
x_7	number of employees lent out
x_8	number of temporary employees
x_9	subtotal 1
x_{10}	subtotal 2

[a)] The variables h_1–h_8 and *sc* were used as auxiliary variables during editing (*cf.* Section 4).

Table A.4. Edits for data set 2

hard edits		soft edits	
$x_4 - x_7 + x_8 + x_6 + x_5 = x_1$	[1]	$x_1 - x_4 \geq 0$	[1]
$x_4 - x_7 = x_9$	[2]	$x_2 - x_3 \geq 0$	[2]
$x_8 + x_6 + x_5 = x_{10}$	[3]	$x_2 \geq 0{,}001$	[3]
$x_1 - x_2 \geq 0$	[4]	IF ($h_3 \neq 0$) THEN ($x_8 \geq 1$)	[4]
$x_4 - x_3 \geq 0$	[5]	IF ($h_6 = 0$ & $h_7 = 0$) THEN ($x_7 = 0$)	[5]
IF ($h_1 = 0$) THEN ($x_3 \leq 2$)	[6]	IF ($h_5 = 0$ || $h_8 = 0$) THEN ($x_1 = 0$)	[6]
IF ($h_2 = 0$) THEN ($x_4 = 0$)	[7]	IF ($h_5 = 0$ || $h_8 = 0$) THEN ($x_2 = 0$)	[7]
$x_j \geq 0$ (for all j)	[8–17]	IF ($h_2 \neq 0$) THEN ($x_4 \geq 1$)	[8]
		IF ($h_2 \neq 0$) THEN ($x_3 \geq 0{,}001$)	[9]
		IF ($h_4 \neq 0$) THEN ($x_6 \geq 1$)	[10]
		IF ($h_6 \neq 0$) THEN ($x_7 \geq 1$)	[11]
		IF ($h_3 = 0$) THEN ($x_8 = 0$)	[12]
		IF ($h_4 = 0$) THEN ($x_6 = 0$)	[13]
		IF ($sc = 4$) THEN ($x_1 \geq 10$)	[14]
		IF ($sc = 4$) THEN ($x_1 \leq 19$)	[15]
		IF ($sc = 5$) THEN ($x_1 \geq 20$)	[16]
		IF ($sc = 5$) THEN ($x_1 \leq 49$)	[17]
		IF ($sc = 6$) THEN ($x_1 \geq 50$)	[18]
		IF ($sc = 6$) THEN ($x_1 \leq 99$)	[19]