UNITED NATIONS ECONOMIC COMMISSION FOR EUROPE
CONFERENCE OF EUROPEAN STATISTICIANS
Expert Meeting on Statistical Data Editing
3-6 October 2022, (virtual)

The SCIA system implementing the Fellegi and Holt methodology compared to the recent R packages

M. Teresa Buglielli, Romina Filippini, and Simona Rosati (Istat – Italian National Institute of Statistics, Italy)[1]
bugliell@istat.it, filippini@istat.it, sirosati@istat.it

I. Introduction

1. The SCIA system (Sistema per il Controllo e l'Imputazione Automatici) is an open-source software developed at Istat for statistical data editing and imputation. It is completely based on the Fellegi-Holt methodology, and it is widely used in survey data with qualitative variables. The high quality of this system is directly related to the good properties of the methodology it is based on. However, only qualitative variables can be handled correctly by the system and the programming language is too obsolete to proceed with a new release. A new software system implementing the Fellegi-Holt methodology would be necessary.

2. Before planning a new project, we investigated among the most recent R packages for editing and imputation in order to find an alternative option. To this aim, we chose to adopt several R packages, such as *validate*, *validatetools* and *errorlocate* to confront data with respect to a set of rules and to localize the errors. Once the errors were identified (including missing values), the imputation was carried out with the R package VIM. All these R packages were first used for the process of data editing and imputation of the Agricultural Census 2020.

3. We present the SCIA system in Section II. Then, in Section III, we provide a description of the editing and imputation process with R. Section IV, after introducing the data input, highlights the main results. A more in-depth study is devoted to how the two software solved the error localization problem. We conclude with a discussion of our findings and how we plan to proceed for future work in Section V.

II. The SCIA system

4. SCIA is an open source software, developed at Istat in the 90s, for automatic editing and imputation of random errors in qualitative variables (Istat, 2004).
It applies the Fellegi-Holt methodology based on the minimum change principle: the correctness of the final data set is ensured by the algorithm that searches for the minimum number of variables to be modified to restore consistency, and by imputing erroneous values preserving the marginal and joint distributions of the variables of interest (Fellegi and Holt, 1976).

5. The software is very user-friendly and flexible, adaptable to different surveys and questionnaires. It operates only on qualitative variables or quantitative variables which can assume a finite number of real values within a given interval (e.g. age). In some cases, it may be convenient to recode a quantitative variables into classes. Moreover, the system is not directly applicable in presence of systematic errors or inconsistencies between different units.

[1] The views expressed in this paper are those of the authors and do not necessarily reflect the views or policies of the Italian National Institute of Statistics.

6. The editing and imputation (E&I) process with SCIA goes through several phases. Specifically, the following SCIA functions are applied sequentially:

a. Definition of the variables involved in the process.
b. Definition of the explicit edit rules.
c. Check of the correctness of the explicit edit rules.
d. Identification of implied edit rules.
e. Check of the data set with respect to the complete set of edits (explicit plus implied edits).
f. Localization of erroneous or missing data.
g. Imputation of erroneous and missing data.

7. First of all, the set of admissible values is defined for each variable involved in the E&I process. Since the data set required in input is in fixed format, the path (name, position, length) of the variables must also be defined. If necessary, the software itself provides a function to convert the data set format from delimited to fixed and vice versa.

8. Then the *explicit edits*, specified by subject matter experts, must be entered in SCIA. This set of edit rules (commonly known as edits) indicate conditions that should be satisfied by combinations of variables in a record. If a record does not satisfy the condition specified by an edit, the edit is said to be failed by that record. Data items that fail any of the edits must be inspected to find errors. Edits must be written in normal form (Fellegi and Holt, 1976), in a text format file, according to the terminology conventionally adopted by the software.

9. The subsequent step is to check the set of explicit edits, i.e. checking for syntax errors and the presence of inconsistencies or redundancies among themselves. This is automatically done by the software. If errors are found, then the edits must be manually modified by the user. The definition step can be considered concluded only when this checking step has been completed successfully.

10. To guarantee the minimum change according to the Fellegi-Holt principle the derivation of the implied edits is necessary. The initial rules defined by the user and the implicit rules derived automatically from the software constitute the *complete set* of edits.

11. Once the correctness of the edits has been verified, the subsequent step is to check the correctness of the data with respect to the predefined edits. This step consists of partitioning the data into *exact records* (records that do not fail any of the edits) and erroneous records (records that fail at least one edit). These erroneous records need to be modified.

12. The last step of data correction is in turn divided into two sub-steps: the localization of errors and the imputation of erroneous or missing values. First, for each erroneous record the set of variables to be modified is identified, according to the minimum change principle, then their values are changed in such a fashion that the resulting record would satisfy all the edits. The correction step can be controlled by the user by setting some parameters by choosing the fixed variables, i.e. those variables whose values must be left unchanged, or by defining matching variables.

13. Data imputation is performed by sequentially applying three types of approaches: conjoint imputation, sequential imputation, imputation based on marginal distributions. The first two approaches are hot-deck donor imputations: in the first case all the erroneous variables are imputed by selecting a single donor record; in the second case one variable at a time is imputed using different donors. If no suitable donor can be identified with any of these two approaches, the software automatically imputes the erroneous variables by randomly extracting a suitable value from the simple marginal distribution computed on the exact records.

14. At the end of the E&I process, SCIA ensures that the minimum number of values for each record has been modified, that the original data distributions remain unchanged, and that the records satisfy all the defined edits.

III. R for editing and imputation

15. The R packages used for the E&I process are as follows:

- **validate**, whose the basic workflow is to create a rule set, confront a data set with the rules in the rule set, and then analyze or use the results further (Van der Loo and De Jonge, 2021; Van der Loo *et al*., 2022). The rules must be written following the R syntax bearing in mind that any expression that gives a logical value as a result can be used as a rule. Main functions are:

    - *validator*, reads the rules and stores them in an R object.
    - *confront*, evaluates data against rules.
    - *summary*, gets a report, for each rule, with number of records who pass checks, fail them or have missing.
    - *violating*, extract from original data set the records with errors.
    - *satisfying*, extract from original data set the records without errors.

- **validatetools**, a set of functions for finding redundancies or contradictions between the rules formulated with *validate* (de Jonge and van der Loo, 2020).

- **errorlocate**, that allows to localize errors given a set of rules. It works in tandem with *validate* package: while validate can identify if a record is valid or not, it does not identify which of the variables are responsible for the invalidation. This package provides a small framework for record based error detection and implements the Fellegi-Holt algorithm. This algorithm assumes there is no other information available then the values of a record and a set of validation rules. The algorithm minimizes the (weighted) number of values that need to be adjusted to remove the invalidation (De Jonge and Van der Loo, 2022). Main functions are:

    - *locate_errors*, which locates erroneous field in records.
    - *replace_errors*, which replaces erroneous values with missing values or a suggestion that is provided by the error detection algorithm. We chose to replace the erroneous fields with missing values and then apply a proper imputation method, as also suggested by the authors.

16. For the imputation step the VIM package was used (Templ *et al*. 2022; Kowarik and Templ, 2016).

IV. Results

A. Input data

17. R and SCIA were both applied to the section D of the questionnaire of the Agricultural Census 2020 that collected information on the farm manager, if not the same as the holder, and other gainful activities (OGA) directly related to the farm. Example of treated variables are: sex, age, citizen, education, agricultural training, the time spent on farm work, provision of health, social or educational services, tourism and accommodation, handicraft, production of renewable energy, wood processing and so on. These are mainly qualitative variables.

18. The number of explicit edits specified by the experts was 119 and 83 in total, for R and SCIA, respectively, including field validation rules. Failures expressed by the edits were conceptually identical, while the different number of edits was due to the specific syntax required by the two software to write the edits. Due to the different syntax, in some cases, the same consistency rule required one edit in SCIA and two edits in R. Specifically, rules of the type *'If A>0 then B between 1 and 100'* must be divided into two edits in R, i.e. *'If A>0 then B>=1'* and *'If A> 0 then B<=100'*. This explains the different number of edits between R and SCIA.

19. As far as SCIA is concerned, the complete set of edits, containing also the essential implied edits logically derived from the explicit edits, amounted to 104 edit rules in all.

20. More than 1 million of records were submitted to the E&I process.

B. Main results

21. Performance of the two software was compared in terms of edit failures and localized errors. As expected, data check carried out through R and SCIA returns similar results, meaning that the two set of rules are essentially the same. The number of failures per edit are reported in Tables 1 and 2.

22. The highest number of failed edits concerned the range of admissible values of the variables involved (Table 1). These errors were mainly due to item non-responses.

Table 1. Field validation rules failures

Edit rule	Number of failures
0 < CAPO_MEDIAORE <=14	82,471
1 <= CAPO_NUMGIO <=365	79,389
1 <= TITSTU_CAPO <=9	8,667
1<= TEMPO_ATT_EXTRA <= 3	1,370
1<= INPS_CAPO <= 4	1.300
1930 <= ANNO_CAPO_AZIENDA <= 2020	191
1 <= CORSI_CAPO <= 2	12
0 <= TEMPO_ATT_CONN <=100	7
1904 <= CAPO_ANNO<=2004	2
1 <= CAPO_CITTAD <= 3	1

23. As regards consistency rules, the edits with at least one failure were only eight (Table 2). The edit with the highest number of failures concerned the "Time dedicated to related activities" (TEMPO_ATT_CONN) that takes values greater than zero when no related activities (ATT_CON) are declared.

Table 2: Consistency rules failures

Edit rule*	Number of failures
1. if (ATT_CON == 2) TEMPO_ATT_CONN <= 0	70,191
2. if (TEMPO_ATT_EXTRA < 3) SETT_AGR + SETT_EXTRA_AGR >=1 and SETT_AGR + SETT_EXTRA_AGR <4	979
3. if (ATT_CONR + ATT_CONQ < 4 and (ATT_CONR >0 or ATT_CONQ>0)) ORE_TERZATT >= 1	224
4. if (ATT_CON ==1) PERC_ATT_CON_REN >= 1 and PERC_ATT_CON_REN <= 100	74
5. if (ATT_CONQ == 1) ORE_TERZATT >= 0 and ORE_TERZATT <= 99999	62
6. if (ATT_CONR == 1) ORE_TERZATT >= 0 and ORE_TERZATT <= 99999	17
7. if (CAPO_AZ == 2) CAPO_REL >=1 and CAPO_REL <=5	3
8. if (ATT_CON ==1) ATT_CON_REN in (A,B,C,D,E,F,G,H,I,L,M,N,O,P,Q,R,S,T,U,V,Z)	1

*For the sake of simplicity, edit rules are expressed according to the R syntax.

24. It should be noted that in some cases, specific additional variables were needed to correctly specify the consistency edit rules in SCIA. Indeed, SCIA treats exclusively qualitative variables and only logical edits can be specified, while mathematical operators are not allowed. For example, the specification of edits 2 and 3 (in Table 2) needed the construction of additional variables. In particular:

(a) Variable SETT was computed as the sum of variables SETT_AGR and SETT_EXTRA_AGR. Consequently, edit 2 in SCIA became: if TEMPO_ATT_EXTRA<3 then SETT in (1-3).

(b) Variable ATT_CON_TERZ was computed as the sum of variables ATT_CONR and ATT_CONQ. Consequently, edit 3 in SCIA became: If ATT_CON_TERZ in (1-3) then ORE_TERZATT>=1.

25. The errors localization process showed different behaviour between R and SCIA (Table 3). Two tests were carried out with R: in the first, fixity weights were specified for those more reliable variables that could not be changed, while in the second, weights were not specified. Weights allow for "guiding" the error localization process, so that less reliable variables with less weight are selected first (De Jonge and Van der Loo, 2022).

26. In some cases, the choice of variables to be modified through the localization process was not the same between R and SCIA, although the total number of values localized as erroneous was very similar for both the software.

27. When weights were specified in R (R1 in Table 3) slight differences were found between R and SCIA in terms of the number of localizations for some variables. Obviously, those variables that are fixated such as ATT_CON, could not be localized because could not be changed.

28. The test performed without weights (R2 in Table 3) resulted in an unexpected outcome: the variable ATT_CON was localized as an error by R, meaning that (looking at the edits) if it is modified by imputation, one of the explicit edits will fail. This was probably due to a lack in the error localization algorithm, which did not consider the entire set of explicit edits, which are instead considered by SCIA together with implied edits. This is the main reason why SCIA found a consistent solution in any case.

29. On the contrary, when weights were specified in R (R1 in Table 3), the *errorlocate* algorithm found a consistent solution with respect to the set of the specified edits.

Table 3: Errors localization: number of erroneous values per variable (excluding field errors)

Variable	SCIA	R1*	R2	R1 - SCIA	R2-SCIA
TEMPO_ATT_CONN	70,198	70,198	70,198	0	0
SETT_AGR plus SETT_EXTRA_AGR	908	938	935	30	27
ORE_TERZATT	146	188	178	42	32
ATT_CONQ plus ATT_CONR	80	38	48	-42	-32
PERC_ATT_CON_REN	74	74	37	0	-37
TEMPO_ATT_EXTR	71	52	52	-19	-19
CAPO_AZ	3	3	3	0	0
ATT_CON	0	0	37	0	37
Total	71,480	71,491	71,488	11	8

*The variable ATT_CON was fixated (it cannot be changed).

30. Once the error localization has been carried out, then the imputation is done. During the imputation phase SCIA took into account the complete set of edits and imputed only with consistent values, whereas with R, two or more steps were required to obtain a global solution consistent with respect to the entire set of edits when weights were not specified. A further imputation step may be required in SCIA when the additional variables (SETT and ATT_CON_TERZ) are modified, in order to restore the original variables.

V. Conclusions and further developments

31. This experience revealed different performance between R and SCIA. Although the SCIA system requires more effort to set up the input metadata, it provides an effective and efficient one-step solution. On the contrary, R approach may take more than one step to arrive at an acceptable solution. This is mainly due to not considering the complete set of edits to localize the errors, given that the implied edits are not derived.

32. The complete set of edits is one of the basic principle of the Fellegi and Holt methodology. As stated by the authors, in attempting to correct a record which fails one or more edits, we will select fields which, between

them, explicitly figure in all the failed edits. We say that such fields "cover off" the failed edits (Fellegi and Holt, 1976).

33. On the other hand, the derivation of the complete set of edits may be too cumbersome and not so obvious, especially when dealing with a large number of variables and edits. In this case, an alternative procedure may consist, for example, by cutting the set of explicit edits into smaller, independent subsets of edits.

34. As far as R is concerned, although it may require more than one processing step and the locate procedures may take long time, it offers more flexible tools than SCIA for managing large and complex data sets, and it allows mixed types of variables to be treated jointly. Moreover, the R packages are extensible, hence other functionalities or features can be developed. For these reasons, we can conclude that more investment should be made in using R in E&I processes.

References

E. de Jonge and M.P.J. van der Loo. *validatetools: Checking and Simplifying Validation Rule Sets.* R package version 0.5.0, URL https://CRAN.R-project.org/package=validatetools, 2020.

E. de Jonge and M.P.J. van der Loo. *errorlocate: Locate Errors with Validation Rules*. R package version 1.1, URL https://CRAN.R-project.org/package=errorlocate, 2022.

I.P. Fellegi and D. Holt. A Systematic Approach to Automatic Edit and Imputation. *Journal of the American Statistical Association*, 71(353): 17–35, 1976.

Istat. CONCORDJava, Controllo e correzione dati, versione 2.2, URL https://www.istat.it/it/files//2014/03/Manuale-utente-CONCORDJava.pdf, 2004.

A. Kowarik and M. Templ. Data Validation Infrastructure for R. *Journal of Statistical Software*, 74(7), 1–16, 2016.

M. Templ, A. Kowarik, A. Alfons, G. de Cillia, B. Prantner, and W. Rannetbauer. *VIM: Visualization and Imputation of Missing Values*. R package version 6.2.2, URL https://CRAN.R-project.org/package=VIM, 2022.

M.P.J. van der Loo and E. de Jonge. Data Validation Infrastructure for R. *Journal of Statistical Software*, 97(10), 1–31, 2021.

M.P.J. van der Loo, E. de Jonge and P. Hsieh. *validate: Data Validation Infrastructure.* R package version 1.1.1, URL https://CRAN.R-project.org/package=validate, 2022.