Methodology for data validation 1.0

Revised edition June 2016

Essnet Validat Foundation

Marco Di Zio, Nadežda Fursova, Tjalling Gelsema, Sarah Gießing, Ugo Guarnera, Jūratė Petrauskienė, Lucas Quensel-von Kalben, Mauro Scanu, K.O. ten Bosch, Mark van der Loo, Katrin Walsdorfer

Table of contents

Foreword

Data validation is a task that is usually performed in all National Statistical Institutes, in all the statistical domains. It is indeed not a new practice, and although it has been performed for many years, nevertheless, procedures and approaches have never been systematized, most of them are ad-hoc and also its conceptualisation within a statistical production process is not always clear. This is a cause of inefficiency both in terms of methodologies and of organization of the production system. There is an urge of producing a generic framework for data validation in order to have a reference context, and to provide tools for setting an efficient and effective data validation procedure.

The first part of the document is devoted to establish a *generic reference framework* for data validation. Firstly, the main elements needed to understand clearly *what* is data validation, *why* data validation is performed and *how* to process data validation are discussed. To this aim a definition for data validation is provided, the main purpose of data validation is discussed taking into account the European quality framework, and finally, for the 'how' perspective, the key elements necessary for performing data validation, that are validation rules, are illustrated.

Afterwards, data validation is analysed within a statistical production system by using the main current references in this context, i.e., GSBPM for the business process and GSIM for defining the informative objects of data validation described as a process with an input and an output. Connections with statistical data editing are clarified by considering the GSDEMs (Generic Statistical Data Editing Models). Finally, the data validation process life cycle is described to allow a clear management of such an important task.

The second part of the document is concerned with the measurement of important characteristics of a data validation procedure (*metrics for data validation*). The introduction of characteristics of a data validation procedure and the proposal of indicators providing quantitative information about them is useful to help the design, maintenance and monitoring of a data validation procedure.

In this second part, the reader can find a discussion of concepts concerning the properties of validation rules such as complexity, redundancy, completeness, and suggestions about how to analyse them. Moreover, once a validation process has been designed, there is the need to analyse its performance with respect to data. In the document, suggestions on indicators based only on observed data are proposed. These indicators are particularly useful for tuning the parameters of the data validation rules. Finally, in order to measure the quality of a data validation procedure, indicators based both on observed and reference data (e.g., simulated or cleaned data) are illustrated.

The document is intended for a broad category of readers: survey managers, methodologists, statistical production designers, and more in general for all the people involved in a data validation process. In fact, the first important objective of the document is to provide a common language about data validation that can be used in the design phase, and in the production phase as well.

The common language we are referring to is concerned with concepts. The introduction of a common technical language for expressing validation rules is certainly an essential part, however it is beyond of the scope of this paper.

Finally, it is worthwhile to remark that more research is still needed, especially in the field concerned with the development of metrics for measuring the quality of a data validation procedure.

This document was developed between January and December 2015 as an output of the Essnet project Validat foundation primarily financed by Eurostat, which involved CBS, Destatis, Istat and Statistic Lithuania.

A generic framework for data validation

1 What is data validation

(Marco Di Zio, Nadežda Fursova, Tjalling Gelsema, Sarah Gießing, Ugo Guarnera, Jūratė Petrauskienė, Lucas Quensel-von Kalben, Mauro Scanu, K.O. ten Bosch, Mark van der Loo, Katrin Walsdorfer)

The first important concept to clarify is concerned with the definition of data validation.

A definition for *data validation* is given in the Unece glossary on statistical data editing (UNECE 2013):

An activity aimed at verifying whether the value of a data item comes from the given (finite or infinite) set of acceptable values.

In this definition, the validation activity is referred to a single *data item* without any explicit mention to the verification of consistency among *different data items*. If the definition is interpreted as stating that validation is the verification that values of single variables belong to set of *prefixed sets of values* (domains) it is too strict since important activities generally considered part of data validation are left out. On the other hand, if the acceptance/rejection of a data item were intended as the final action deriving from some complex procedure of *error localization*, the previous definition of validation would be too inclusive since it would involve also phases of the editing process not strictly related to the validation process.

A proposal for an operational definition of data validation was recently provided by Simon (2013a).

“Data validation could be operationally defined as a process which ensures the correspondence of the final (published) data with a number of quality characteristics.”

It is related to the objective of validation, that is to ensure a certain level of quality of the final data. We did not adopt this definition since many processes different than data validation may be characterised by using this definition.

The definition we provide is reported in the following box, it is similar to that given by Unece, but with a modification that allows to disregard the correction activities, and to focus only on verification:

Data Validation is an activity verifying whether or not a combination of values is a member of a set of acceptable combinations.

The set of 'acceptable values' may be a set of possible values for a single field. But under this definition it may also be a set of valid value combinations for a record, column, or larger collection of data. We emphasize that the set of acceptable values does not need to be defined extensively. This broad definition of data is introduced to make data validation refer both to micro and macro (aggregated) data.

Data validation assesses the plausibility of data: a positive outcome will not guarantee that the data is correct, but a negative outcome will guarantee that the data is incorrect.

Data validation is a decisional procedure ending with an acceptance or refusal of data as acceptable. The decisional procedure is generally based on rules expressing the acceptable combinations of values. Rules are applied to data. If data satisfy the rules, which means that the combination expressed by the rules is not violated, data are considered valid for the final use they are intended to. There is of course the possibility of using the complementary approach in which rules are expressed in "negative form": in this case data are validated by verifying that predefined non-acceptable combinations of values do not occur.
Sometimes the rules used in a validation procedure are split in hard/fatal edits and soft/query edits and the not acceptable values are classified either as 'erroneous' or 'suspicious' depending on whether they fail hard edits or soft edits. Hard edits are generally rules that *must* necessarily be satisfied for logical or mathematical reasons (e.g., children cannot be older than their parents). An example of query edits taken from the Unece glossary on statistical data editing is "a value that, compared to historical data, seems suspiciously high" while for fatal edits is "a geographic code for a Country province that does not exist in a table of acceptable geographic codes". This distinction is an important information for the related 'editing' phase. In addition to this information, a data validation procedure may assign a degree of failure (severity) that is important for the data editing phase and for the tuning of data validation. Taking the example previously mentioned for soft edits, the severity can be evaluated by measuring the distance of the actual values with respect to the historical one. A more detailed discussion on metrics for measuring severity is given in the 'Metrics' Section.

In case of failure of a rule, data are exported from the data validation procedure, or marked respectively, and are handled by the editing staff in order to correct values to make the rules satisfied, or data are considered acceptable and the rules of the data validation are updated. The data validation process is an iterative procedure based on the tuning of rules that will converge to a set of rules that are considered the minimal set of relations that must be necessarily satisfied.

The relation with statistical data editing must be clarified. We take as reference for statistical data editing the Generic Statistical Data Editing Models (GSDEMs v. 0.1) and the Generic Statistical Business Process Model (GSBPM). In the GSBPM, the process 'Validate and Review' is distinguished from the process 'Edit and Impute'. In the 'Validate and review phase' there is data validation as it is previously described, while the 'edit and impute phase' includes the action of 'changing data'. This is the idea underlying our definition.

In the GSDEMs, statistical data editing is described as composed of three different function types: Review, Selection and Amendment.
The review functions are defined as:
Functions that examine the data to identify potential problems. This may be by evaluating formally specified quality measures or edit rules or by assessing the plausibility of the data in a less formal sense, for instance by using graphical displays.

Among the GSDEMs different function categories there is 'Review of data validity' that is

Functions that check the validity of single variable values against a specified range or a set of values and also the validity of specified combinations of values. Each check leads to a binary value

(TRUE, FALSE).

This is in fact what we have just defined for data validation. In the GSDEMs, other 'review' functions are introduced: 'review of data plausibility' and 'review of units'.

In review of data plausibility and review of units the output is a degree of 'plausibility', they are not seen as a final step, but as an intermediate step necessary for further work on data. In other words, the GSDEM review category includes also functions that typically are used to produce elements (such as scores or thresholds) that are needed in the validation procedure.
The connection of data validation with statistical data editing depends on the reference framework that is taken into account. According to GSBPM they are related but distinct, according to the GSDEMs data validation is a part of statistical data editing.

We notice that, the step of validation in statistical data editing generally assumes that IT structural requirements are satisfied and it does not deal with the related errors (formal checks), while we include this type of check in the data validation procedure.

2 Why data validation - Relationship between validation and quality

(Marco Di Zio, Ugo Guarnera)

The purpose of data validation is to ensure a certain level of quality of the final data.
Nevertheless, quality has several dimensions in official statistics: relevance, accuracy, timeliness and punctuality, accessibility and clarity, comparability, coherence, completeness. Hence, it is important to establish on which components data validation is concerned with.

Data validation focuses on the quality dimensions related to the 'structure of the data', that are accuracy, comparability, coherence. Data validation does not directly focus on quality aspects from the ESS QAF (see ESS QAF) that concern processes (e.g., timeliness). It is worthwhile to see in detail to which extent data validation is related to the different quality dimensions.

Accuracy.
The general definition of accuracy refers to the measurement of the difference between the 'target parameter' and the 'estimated parameter' (in principle these parameters are aggregates). This difference is potentially due to many error components that can be divided in 'sampling and non-sampling errors'.
The validation procedure we are going to discuss does not take into account the sampling errors that are in fact evaluated generally through variance estimation methods. By definition sampling errors are not generated by errors in data, and are out of the scope of data validation procedure.

Non-sampling errors are composed of different components (coverage errors, measurement errors, processing errors, non-response) but generally validation procedures are aimed at verifying the presence of those errors in that data that are defined in literature as 'measurement errors' (only a part of non-sampling errors).

On the contrary to what happens with sampling errors, for non-sampling errors and of course as a specific case for measurement errors, it is usually difficult and expensive to provide a direct measure. Evaluation of non-sampling error is often based on 'repeated measures' on the same individuals. Hence, it is worthwhile to exploit indirect measures to identify errors in data by using consistency rules concerning micro and macro aspects. This explains the usefulness of a data validation procedure.

It is worthwhile to remark that, while strictly speaking measurement errors refer only to observed data, in our more general context measurement errors refer also to errors produced in the production procedure, for instance coding, imputation. Actually, from the point of view of the validator, also data produced during the statistical production process are indeed observations.

Coherence and comparability

The general definition of coherence and comparability claims that statistics should be consistent internally, over time and comparable between regions and countries.

Coherence and comparability aspects are definitely important for the data validation process. Validation rules and the process of confronting the data set with validation rules, the process of detecting errors and flagging them should be coherent and consistent internally and between countries, based on common standards with respect to the scope and national deviations.

Clarity and accessibility

'Accessibility and clarity' is a quality dimension checked in a data validation process, it is related to the IT formal checks that are performed in order to be able to read without any misunderstanding the data.

Timeliness

Timeliness is not a quality dimension checked by a validation procedure. Nevertheless, it is important to remark that it has a strong connection with a data validation procedure. Timeliness can be seen as a constraint when designing a data validation procedure. For instance, in case of complex checks and time demanding editing procedures, a less restrictive data validation process allowing a higher amount of errors in data may be designed to meet the expected timing of the release of final data.

A final remark about the concept that data validation "aims at verifying whether data have a certain level of quality": It is indeed true that data validation cannot 'ensure' a level of quality. What it can more realistically provide is that at least a certain level of data consistency considered as the minimum requirement for having acceptable data, is reached. This results not in perfect data, but in 'plausible' data.

3 How to perform data validation: validation levels and validation rules

(Marco Di Zio, Mark Van der Loo)

Because of the variety of validation steps and procedures, and because of the way validation procedures pervade statistical production processes, it is desirable to be able to judge to what extent a data set has been validated (validation level) by validation procedures applied to it. Moreover, as statistical processes age and mature, the number of validation procedures and rules tend to grow organically, generating a need for maintenance. Finally, one would like to be able to compare statistical processes and statistical software in view of their abilities to validate data.

Clearly, the above tasks would be easier, if there was some sort of system that classifies validation levels, validation rules and procedures into disjoint subtypes.
To develop any classification system, one needs to consider what principle or principles separate the different classes. For validation rules and procedures, the following come to mind or have been encountered in literature:

- automated versus manual;
- objective versus subjective/expert opinion;
- structural validation versus content validation;
- set being validated: in-field, in-record, cross-record, cross-data set, etc.;
- place in the statistical process: input, throughput, output;
- type of validation rule: equality, inequality, logical rule,…

and sure there are many more options. Depending on the task at hand, different classifications may be useful, as long as they are both exhaustive and mutually disjoint.

In the recent literature, classifications have been developed from both a business perspective (Simon 2013a, 2013b) and from a more abstract perspective (Van der Loo, 2015). Moreover, Daas *et al.* (2011) produced an extensive checklist describing quality aspects to be validated, encompassing both structural (e.g. file format compliance) and subject-matter based content validation in the context of administrative data sources.

The works of Simón and Daas *et al.* take a business perspective, encompassing both structural and subject-matter based content validation. The work of Van der Loo is focused on validation of data for which the technical structure has already been fixed. The advantage of the latter work is that it abstracts away from particular processes, files, or quality requirements and only looks at the structure of the domain of validation functions. This makes it more suitable for comparing, for example validation tools and languages or comparing validation across wildly different production processes.

4 Validation levels from a business perspective

(Sarah Giessing, Katrin Walsdorfer)

As it was observed in the introduction, data validation is not a new process, and some common elements can be derived from the existing practice of data validation. To this aim, a survey on the data validation procedures currently used in the National Statistical Institutes has been carried out in the Essnet Validat Foundation to find out how the typical validation process is practically implemented in the Member States of the ESS, see (Giessing and Walsdorfer, 2015) for information on design and results of this survey.

Looking at the practical implementation of the validation process means to take a business perspective. In the business perspective the attention is focused on the validation activities. The amount of information needed and the phases of the validation process are important for determining the validation levels. This approach is particularly useful for classifying and designing validation activities within an organization.

It is generally assumed that there are basically two general validation levels:

A. Technical integrity of the file, i.e., consistency with the expected IT structural requirements
B. Logical and statistical consistency of the data

The second category is generally split into different sub-categories (levels) involving more and more information. The two general categories can then be expanded forming the following validation levels.

Valid. Lev 0 : consistency with the expected IT structural requirements

Valid. Lev. 1: consistency within the data set

Valid. Lev. 2: consistency with other data sets within the same domain and within the same data source

Valid lev. 3: consistency within the same domain between different data sources

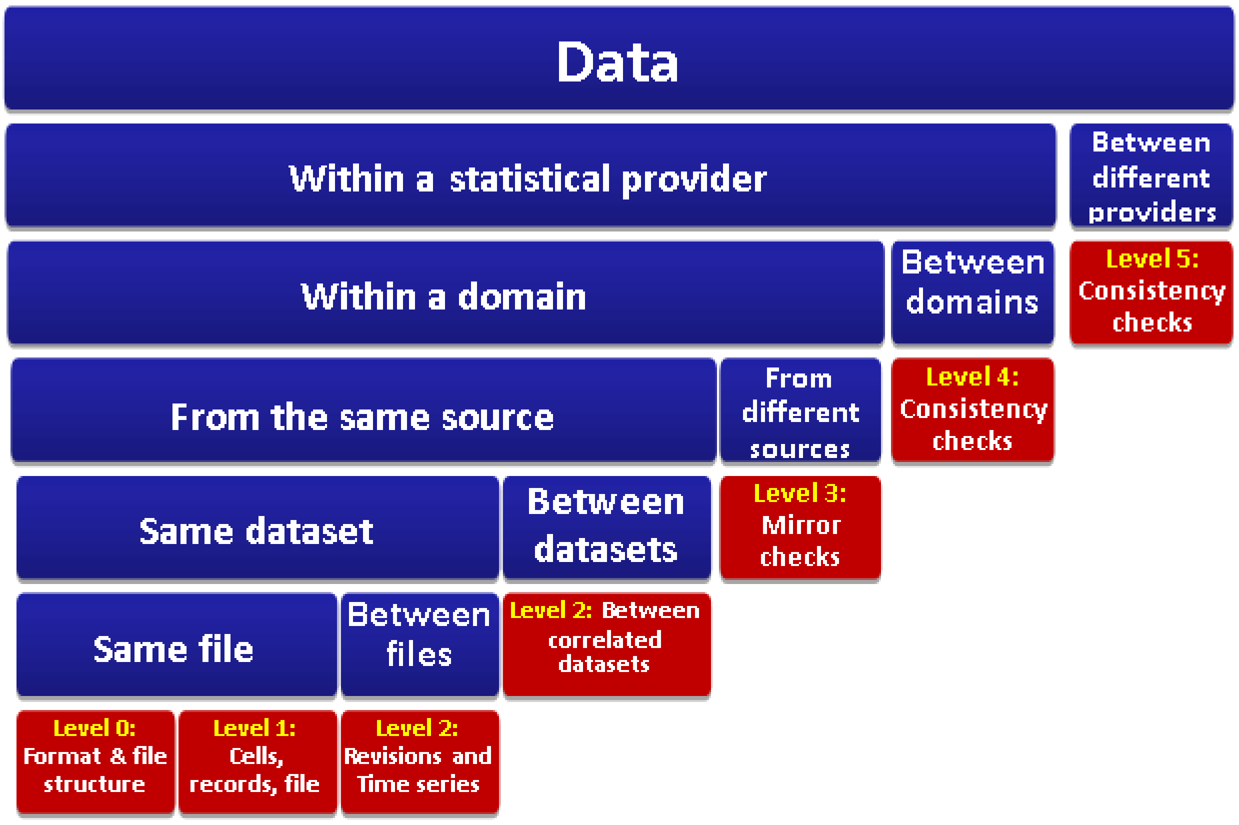
Valid lev. 4: consistency between separate domains in the same data provider

Valid. Lev 5: consistency with data of other data providers.

We notice that validation level 0 corresponds to the general validation level A earlier mentioned.

In Figure 1, validation levels are represented according to their degree of complexity. The figure and the examples illustrating the levels have been taken from Simon (2013a).

Figure 1. Graphical representation of validation levels



Data
Within a statistical provider
Between different providers
Within a domain
Between domains
Level 5: Consistency checks
From the same source
From different sources
Level 4: Consistency checks
Same dataset
Between datasets
Level 3: Mirror checks
Same file
Between files
Level 2: Between correlated datasets
Level 0: Format & file structure
Level 1: Cells, records, file
Level 2: Revisions and Time series

The description of the validation levels is reported in the following, for a detailed discussion see Simon (2013a).

Validation level 0.

At this level, it is checked the consistency of the data with their expected IT requirements, for instance

- if the file has been sent/prepared by the authorised authority (data sender);
- if the column separator, the end of record symbol are correctly used;
- if the file has the expected number of columns (agreed format of the file);
- if the column have the expected format of the data (i.e., alphanumeric, numeric, etc.)

For these quality checks only the structure of the file or the format of the variables are necessary as input.

Validation level 1

It is checked the consistency within the elements of the data set. For these quality checks, it is needed only the (statistical) information included in the file itself.

For instance:

- check whether the number included in column 4 is not negative (as expected);

- check whether the year in the second column is 2011, as in the file name;

- check whether the content of the third column is one of the codes of the dictionary "Sex";

- check whether the content of the first column is consistent with the data sender (let's assume that there is a dictionary including the list of the data senders associated to the specific data set): data for Luxembourg should not be sent by another country.

- based on information available before data collection (for example from previous survey or other sources) one could establish a "plausibility range" for a certain variable (for instance number of components of a household).

- check consistency at (micro-level) of two (or more) variables: a certain combination of codes is illogical, a variable has to be reported only for a certain combination of codes.

- check consistency at macro-level of two (or more) variables: Total inhabitants = male inhabitants + female inhabitants, or Female inhabitants = (total inhabitants / 2) +/- 10%

Validation level 2

Validation levels 2 is concerned with the check of consistency based on the comparison of the content of the file with the content of "other files" referring to the same statistical system (or domain) and the same data source.

For instance:

Case a) the "other files" can be other versions of exactly the same file.

In this case the quality checks are meant to detect "revisions" compared to previously sent data. Detection and analysis of revisions can be useful for example to verify if revisions are consistent with outliers detected in previous quality checks (corrections) or to have an estimate of the impact of the revisions in the "to be published" results, for the benefit of the users.

Case b) the "other files" can be versions of the same data set referring to other time periods.

These checks are usually referred to as "time series checks" and are meant to verify the plausibility of the time series.

Case c) the "other files" can refer to other data sets from the same data provider (e.g., Countries in the ESS), referring to the same or other correlated time periods. Sometimes a group of data sets (same country, same reference period) is sent at the same time.

Example: three files could be sent at the same time, from the same country and referring to the same time period: one file includes data for "females", one for "male" and one for "total". Consistency between the results of the three files can be checked.

Another example: results from annual data sets can be compared with the results of the corresponding quarterly data sets.

Validation level 3

Validation levels 3 is concerned with the check of consistency based on the comparison of the content of the file with the content of "other files" referring to the same statistical system (or domain) but with a different data source.

For instance:

Case d) the "other files" can refer to the same data set, but from another data provider (e.g., Countries of the ESS).

Mirror checks are included in this class. Mirror checks verify the consistency between declarations from different sources referring to the same phenomenon, e.g., export declared by country A to country B should be the same as import declared by country B from country A.

Validation level 4

Validation level 4 could be defined as plausibility or consistency checks between separate domains available in the same Institution. The availability implies a certain level of "control" over the methodologies by the concerned Institution.

These checks could be based on the plausibility of results describing the "same" phenomenon from different statistical domains. Examples: unemployment from registers and from Labour Force Survey, or inhabitation of a dwelling (from survey of owners of houses and dwellings vs. from population register)

Checks could also be made between results from correlated micro-data and macro-data sources.

Other plausibility checks could be based on known correlations between different phenomena: for example external trade and international transport activity in ports.

Validation level 5

Validation level 5 could be defined as plausibility or consistency checks between the data available in the data provider (Institution) and the data / information available outside the data provider (Institution). This implies no "control" over the methodology on the basis of which the external data are collected, and sometimes a limited knowledge of it.

Statistical indicators collected by Eurostat might also be compiled for their own needs by national institutions such as National Statistical Institutes or Ministries; by private entities (ports, airports, companies, etc.) and also by international organisations (World Bank, United Nations, International Monetary Fund, etc.).

For example, EU road freight statistics are prepared by Member States according to the EU Commission legal acts and in addition countries can carry out specific surveys for national purposes. A benchmarking between indicators common to these different surveys allows assessing the coherence of these data and could help improving the methodologies for data collection.

To summarize, the classification of validation levels presented above implicitly assumes a growing degree of complexity from one level to another. However, this must not necessarily be reflected by a growing technical complexity of the validation checks themselves. From the technical point of view, the distinction made with respect to data sets is an artifice, since data sets and files could be merged into single databases in advance of implementing the checks.

A likely rise in complexity might be regarding organizational and management matters. On the higher levels of this classification more parties and stake-holders will be involved, potentially with different needs and requirements regarding data validity. This certainly tends to make it more difficult to harmonize technical and methodological concepts. However, this may depend very much on the concrete situation and circumstances.

4.1 Validation rules

The validation levels, as anticipated in the examples of validation levels, are verified by means of rules. Rules are applied to data, a failure of the rule implies that the corresponding validation level is not attained by the data at hand.
As explained in the beginning of section 4., a first broad classification of validation rules distinguishes rules to ensure technical integrity of the data file (type A.) and rules for logical/statistical consistency validation (type B). The distinction is useful since the rules used in the two contexts can be very different. Examples for the different rule types have been reported by the respondents of the ESSNET survey. Some of them will be presented further below:

A. Rules to ensure technical integrity of a data file
   - formal validity of entries (valid data type, field length, characters, numerical range)
   - presence of an entry
   - no duplicate units
   - all the values in a field of one data set are contained in a field of another data set (for instance contained in a codelist)
   - each record has a valid number of related records (in a hierarchical file structure)

B. Rules for logical validation and consistency could be classified using the two typology dimensions presented in table 1, e.g. identity vs. range checks (1) and simple vs. complex checks (2).

Table 1: Categories of a 2-way typology for validation rules for logical validation and consistency

Typology dimension	Types of checks	
1	Identity checks	Range checks • bounds fixed • bounds depending on entries in other fields
2	Simple checks, based directly on the entry of a target field	More “complex” checks, combining more than one field by functions (like sums, differences, ratios)

Also, rules are often implemented as conditional checks, i.e. they are only checked, if a certain condition holds. This can be regarded as another property of a rule and might be considered as additional “dimension” of the rule typologies (for both rule sets, A. and B.).

Typical conditions of a conditional check mentioned by the ESSNET survey respondents are

-
- if “age under 15” (then marital status must be not married), or
- if “legal form: Self-Employed” (then number of self-employed" must exceed 0), or
- if “status in employment = compulsory military service” (then sex must be male), or
- if “no. of employees not zero” (then wages and salaries must be greater than zero), or
- if “enterprise reports production of goods” (then it should also report costs for raw material), etc.

Of course there might be several conditions combined by logical AND or OR statements.
Table 2 below presents at least one example[1] for each rule type in set A.
For the rule types of set B, table 3 provides examples[2].

[1] Examples selected from the examples provided by the respondents to the survey in the ESSNET project.

Table 2: Examples of rules to ensure technical integrity of a data file

Formal validity of… - data type - field length - characters - numerical range	 *Telephone number*: Numerical data. *Date:* If Date is given as text it should be 8 characters long *Date:* If Date is given as text it should contain only numbers. *Month*: Month of arrival in the country must be in {1,...,12}
Presence of an entry	*Persons in households:* It is checked whether all have responded *Code for Sex:* no missing data.
No duplicate units	*Holding ID*: Each holding has a unique ID number, duplicate ID numbers are not allowed within the data set
All the values in a field of one data set are contained in a field of another data set (for instance contained in a codelist) *"code list check"*	*Occupation*: Field "Occupation" must contain only entries from a list of valid ISCO-08(COM) codes at 3 digits level *Country of origin:* Field "country of origin" must contain only entries from a list of valid ISO country codes
Each record has a valid number of related records (in a hierarchical file structure) "*Cardinality check*"	*Number of members of a family:* the aggregated number of persons in each family must be equal to the number of individual rows in the data set corresponding to the members of that family

Table 3: Examples of rules for logical validation and consistency

	"Simple"	**"Complex" (checks involving functions on field entries)**
Identity check	*In a sheep survey :* "Milk production" must be equal to "milk disposal"	*Employment:* "Number of persons engaged" must be equal to the **sum** of "employees" and "self-employed persons"
Range checks - **bounds fixed**	 *Working hours (monthly)*: "Hours worked" must be between 0 and 168	 *Average poultry weight*: "weight of poultry" **divided by** "number of poultry" must be between 0.03 and 40
- **bounds depending on entries in other fields**	*Cereal production:* "Organic cereals" must be less or equal to "cereals"	*External services:* "Expenses on external services" must be greater or equal to "payment for agency workers" **plus** "telecommunications" **plus** "business trips of company personnel"

Notably, not all cross-combinations in the 2-way representation of rule types used to define the fields in table 3 are "necessary" from a language perspective. For example, any range check of the type "complex" can be expressed as range check with fixed bounds. For illustration, consider the instance provided in table 3 for checking expenses on external services. This rule would be equal to the following rule with a fixed bound of zero:

> "Expenses on external services" **minus** "payment for agency workers" **minus** "telecommunications" **minus** "business trips of company personnel" must be greater or equal to zero.

Moreover, any check of the type "complex" might be implemented as well as check of the type "simple": According to our definition, a "complex" check is a check combining more than one field by functions (like sums, differences, ratios). Of course one might implement into the procedures of a statistic a step which derives new variables implementing such "functions". If the validation step is carried out after such a preparatory step, all "complex" checks will be "simple". This has also been reported as actual practice by one of the survey respondents: "Variables that are a combination of elementary ones are always implemented by combining elementary variables in the output process".

Also, from a pure technical perspective, a conditional check may have the same logical form as an unconditional one. For example an unconditional check may have the logical form: if C1 and C2. Any range check could be expressed this way, for example C1: age difference between spouses ≥0, C2: age difference between spouses ≤20. On the other hand also the conditional check "If a person is aged under 16 the legal marital status must be never married" can be expressed this way, if we define: C1: age < 16 and C2: legal marital status not married.

An extended list of validation rules is provided in Appendix A. It has been obtained combining the lists of tables 2 and 3, taking into account a few more examples provided by the survey respondents, and combining this with the list of rules Simon (2013b).

In the extended list, we classify the rules according to the rule typology of table 1, with some occasional comments, if the check might be typically implemented not simply as intra-file check (i.e. on level 1 of the validation levels discussed in 2.3.1), but might perhaps fall into the categories defined as levels 2 to 5 in (Simon, 2013a), c.f. 2.3.1.

However, unlike for the examples directly taken from (Simon, 2013b), constructed to explain the different levels, for the examples provided by the survey respondents, this is always just a guess. The survey questionnaire did not "cover" the "levels" dimension. Consequently, respondents did not bother to explain, if the data used for a particular check are stored in the same, or in different files, whether they come from the same or from different sources or even from different data collecting organizations. Nor did they explain explicitly, if a check is carried out on the microdata- or on the macro-data-level. Rather on the contrary, one respondent reported for a certain type of check (i.e. complex range check with bounds depending on other fields) that "This is performed on micro, macro and output editing. In micro-editing relevant ratios are tested in order to qualify the quality of a firms answer to the yearly survey. In macro and output editing phases, these ratios are used in order to identify the firms/sectors that have a big influence on the results."

So far we have discussed the validation levels and rules from a business perspective, which means to describe the validation as it is usually discussed in the practice of surveys. This perspective is particularly relevant with all the practical aspects of a survey, for instance for doing a check-list in the design phase. On the other hand, it is limited in terms of abstraction and this may be inconvenient for generalizing the concepts and results.

In the following section a generic framework for validation levels and rules is presented.

5 Validation levels based on decomposition of metadata

(Mark Van der Loo)

For this typology, we use the following technical definition of a data validation function (Van der Loo, 2015). Denote with S the class of all data sets. That is, if $s \in S$, then s may be a single data field, a record, a column of data, or any other collection of data points. A *data validation function* v is defined as a Boolean function on S, that is:

$$v : S \to \{0,1\}.$$

Here, 0 is to be interpreted as FALSE, or *invalid*, and 1 is to be interpreted as TRUE, or *not invalidated by* v.

Such a functional definition may feel unnatural for readers who are used to validation rules, for example in the form of (non)linear (in)equality constraints. However, a comparison operator such as the "$>$" in the rule $x > y$ may be interpreted as a function $>(x,y)$, yielding 0 if $x \leq y$ and 1 otherwise. Likewise, comparison operators $\leq, <, =, >, \geq$ and the membership operator $\in$ can be interpreted as functions.

The action of a validation function thus consists of testing whether a certain relation on a (set of) data points holds. As such it's result does not usually permit an immediate interpretation of failures in terms of what data point, or set of points cause the violation(s). The latter task is called error localization which is an entirely different task.

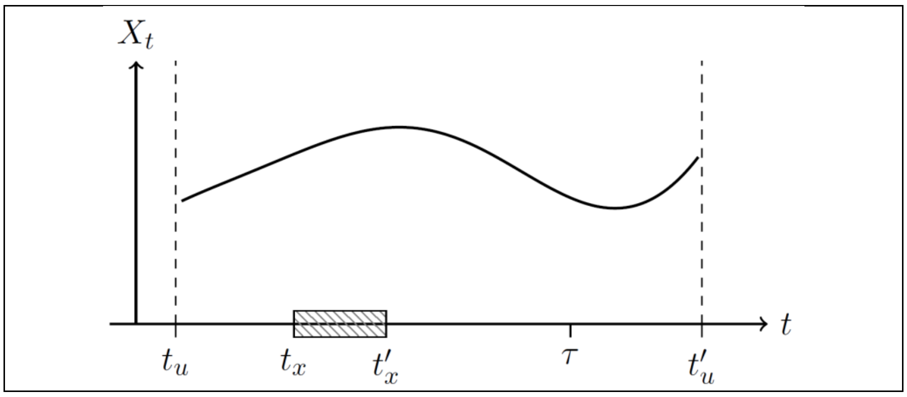
We may distinguish different types of validation functions by considering what type of elements from the class of all data sets (S) it validates. In other words, we need to decompose the metadata that defines the data points making up a data set $s \in S$. Van der Loo (2015) proposes a basic but extensible metadata decomposition that is based on an analysis of the measurement process.

In this model, which is depicted in **Figure 2**, the horizontal axis is the time line with time points t. At time t_u an element of a statistical universe U is born. From that time on, it contains certain properties, say X, that may depend on time. At time τ, u is selected from the then current population and a measurement of X on u takes place.

This model yields four essential, and not completely independent metadata aspects that are necessary to identify a data point, namely:

1. The universe U from which a statistical object originates. This roughly determines the type of object that is investigated: household, person, company, e-mail, phone call, and tweet are all examples of valid statistical objects from certain universes. The choice of U determines the set of properties X for statistical objects.
2. The time τ of selecting an element u from the then current population $p(\tau)$. The moment of selection determines about what population one is producing a statistical statement later on.
3. The selected element $u \in p(\tau)$. This determines the value of variables X over time that may be observed.
4. The variable selected for measurement.

Figure 2: A simple model of a measurement process (Van der Loo, 2015)



It is important to point out a subtlety regarding time in this model. Observe that the value that is obtained by measurement may or may not pertain to the measurement time τ. For example, if a questionnaire is the instrument of measurement, one may ask about a subject's past, current or future (expected) place of residence. In general then, the measurement pertains to a period $[t_x, t'_x)$ or a moment in time if one lets $t'_x \to t_x$ that need not coincide with τ. In the context of this simple model, the time to which a value pertains is a part of the definition of variable X and therefore non-essential. The time of measurement τ is considered essential since it both fixes the population and the value that is to be recorded. Indeed, Zhang and Pritchard (2013) for example point out that in the case of administrative data, a recorded value may be updated over time.

Following the model described above, we define a *data point* as recorded value endowed with the four indices (U, τ, u, X) for Universe, time of measurement, selected element, and observed variable. A *data set* $s \in S$ is now formally defined as a set of data points.

5.1 A formal typology of data validation functions

We may classify data sets in S according to which indices are constant for all data points $x \in s$. And classify validation functions accordingly. For example, the rule

$$x_{U,\tau,u,X} > 0,$$

states that individual values have to be larger than zero. The corresponding validation function can be executed on the simplest of data sets: a single observation. To execute the validation

$$x_{U,\tau,u,X} + x_{U,\tau,u,Y} = x_{U,\tau,u,Z},$$

we need to collect values for variables X, Y, and Z for the same element $u \in U$, measured at the same time τ (in short: it is an in-record validation rule). Hence, only the indices (U, τ, u) are constant over the set that is validated by this rule.

Generalizing from these examples, we see that validation functions may be classified according to which of the metadata indices need to be varied to be able to execute a validation function. Since we have four indices, this in principle yields $2^4 = 16$ possible rule types.

There are however some restrictions since the indices cannot be varied completely independent from each other. The first restriction is that a statistical element u cannot be a member of two universes, except in the trivial case where one universe is a subset of the other (for example: take the universe of all households, and the universe of all households with more than 3 members). The second restriction stems from the fact that U determines what variables can be measured. The exact same variable cannot be a property of two types of objects (*e.g.* even though one may speak of an *income* for either persons or households, one would consider them separate objects and not combine them to say, compute a total).

Taking these restrictions into account yields 10 disjunctive classes of validation functions. Using the index order $U\tau uX$, each indicate class is indicated with a quadruplet of $\{s, m\}$, where s stands for single and m for multiple. An overview of the classes, with examples on numerical data is given in the Table 4.

Table 4: Overview of the classes and examples of numerical data

Class $(\boldsymbol{U\tau uX})$	**Description of input**	**Example function**	**Description of example**
$ssss$	Single data point	$x > 0$	Univariate comparison with constant
$sssm$	Multivariate (in-record)	$x + y = z$	Linear restriction
$ssms$	Multi-element (single variable)	$\sum_{u\in s} x_u > 0$	Condition on aggregate of single variable
$ssmm$	Multi-element multivariate	$\frac{\sum_{u\in s} x_u}{\sum_{u\in s} y_u} < \epsilon$	Condition on ratio of aggregates of two variables
$smss$	Multi-measurement	$x_\tau - x_\nu < \epsilon$	Condition on difference between current and previous observation.
$smsm$	Multi-measurement multivariate	$\frac{x_\tau + y_\tau}{x_\nu + y_\nu} < \epsilon$	Condition on ratio of sums of two currently and preciously observed observations.
$smms$	Multi-measurement multi-element	$\frac{\sum_{u\in s} x_{u\tau}}{\sum_{u\in s} x_{u\nu}} < \epsilon$	Condition on ratio of current and previously observed aggregate.
$smmm$	Multi-measurement multi-element, multivariate	$\frac{\sum_{u\in s} x_{u\tau}}{\sum_{u\in s} x_{u\nu}} - \frac{\sum_{u\in s} y_{u\tau}}{\sum_{u\in s} y_{u\nu}} < \epsilon$	Condition on difference between ratios of previous and currently observed aggregates.
$msmm$	Multi-universe multi-element multivariate	$\frac{\sum_{u\in s} x_u}{\sum_{u'\in s'} y_{u'}} < \epsilon$	Condition on ratio of aggregates over different variables of different object types.
$mmmm$	Multi-universe multi-measurement multi-element multi-time	$\frac{\sum_{u\in s} x_u}{\sum_{u'\in s'} y_{u'}} - \frac{\sum_{u\in s} x_{u\tau}}{\sum_{u'\in s'} y_{u'\nu}} < \epsilon$	Condition on difference between ratios of aggregates of different object types measured at different times.

5.2 Validation levels

The typology described in the previous subsection lends itself naturally for defining levels of validation in the following way. Given a data set $s \in S$ that is validated by some validation function v, count the number of indices $U\tau uX$ that vary over s. The result can be summarized in the following table.

Table 5: Validation levels

	Validation level				
	0	1	2	3	4
$U\tau uX$	$ssss$	$sssm$	$ssmm$	$smmm$	$mmmm$
		$ssmd$	$smsm$	$msmm$	
		$smss$	$smms$		

Observe that the order of validation levels correspond with the common practice where one tests the validity of a data set starting with simple single-field checks (range checks) and then moves to more complex tests involving more versatile data.

6 Relation between validation levels from a business and a formal perspective

(Mark Van der Loo)

In the previous sections, validation levels have been discussed from both a business and a more formal view. A natural question to ask is how these two perspectives interrelate and what the merits and demerits of these contrasting views are. In the following, we discuss both viewpoints from a theoretical perspective, and we will correlate the levels amongst each other, illustrated by examples obtained in the survey undertaken in this ESSnet.

From a theoretical point of view, the difference lies in the chosen principals that are used to separate validation rules. The validation levels that are derived from a business perspective (see Figure 1) are motivated by the viewpoint of a statistician: data are obtained in batches of *file*s (a *data set*) from a *source*, pertaining to a *domain*, and data may or may not come from within the same institute (*statistical provider*). Now, validation rules are categorized by judging whether they pertain to data within a single or multiple files, within or across sources, and so on. The main merit of this division is that it appeals closely to the daily routine of data management and handling. It is likely that the business-based subdivision can therefore be easily understood by many practitioners in the field. The main demerit is that it is to some extent subjective. For example, consider the division of rules between in-file and cross-file is spurious: one statistician may receive two files, and apply a cross-file validation, merge the files and pass it to a second statistician. The second statistician can perform the exactly the same validation, but now it is classified as an in-file validation. This subjectivity is especially problematic when one wishes to compare validation levels across production processes.

The validation levels that are derived from a formal viewpoint are based on a decomposition of metadata that minimally consists of a *domain,* a *measurement time*, the observed *statistical object* and the measured *variable*. Rules are categorized according to whether they pertain to one or more domains, objects, and so on, taking into account that the four aspects cannot be chosen completely independently. The merits and demerits mirror the demerits and merits derived from the business point of views and can in that sense be seen as complementary: being based on formal consideration, this typology's demerit is that it may take more effort to analyse practical situations. The main merit is its objectivity: it allows for comparison of different production processes.

Table 6 shows a correlation chart between the typologies driven by business and formal considerations: an 'x' marks where levels in the business-driven typology have matches in the formal typology. A capital 'X means 'full overlap' while a small 'x' marks partial overlap.

Since the formal typology is defined on a logical level, file format and file structure is not a part of the typology. Hence, the first row in the table is empty. The business-typology level 1 includes checks that can be performed in-file, where based on the examples given in section 4.1, it is assumed that a single file contains only data from a single domain. This means that there is no overlap with formal typology level 4, in which the domain must vary in order to perform a check.

There is partial overlap with format-typology level 3, since it contains one category where domains a varied and one where this is not the case. The same holds for business-level 2, where it is stated explicitly that the checks have to pertain to within a single domain. The most important difference between business-level 2 and 3 is the physical origin of the data (*source*). Since there is no equivalent of this in the formal typology, the correspondence is again the same as for level 1 and level 2.

Business-level 4 is explicitly reserved for checks that use data across statistical domains. There is therefore full overlap with formal level 4 and partial overlap with formal level 3. Business level 5 finally, mentions explicitly the use of data 'information outside the institution'. Since the formal typology makes no difference between such physical aspects, there is at least partial overlap with all formal levels.

Table 6: Cross-correlation of the business-driven and formal typology. Small 'x' means partial overlap, large 'X' indicates full overlap

		Formal typology				
		L0. single data point	L1. vary one key	L2. vary two keys	Level 3. vary three keys	Level 4. vary four keys
Business-based typology	L0. Format & File structure					
	L1. Cells, records, file	X	X	X	x	
	L2. Consistency across files and data sets		X	X	x	
	L3. Consistency across sources		X	X	x	
	L4. Consistency across domains				x	X
	L5. Consistency with external data	x	x	x	x	x

6.1 Applications and examples

To further clarify the above typology, we classify a number of rules that were submitted by NSI's during the stocktaking work of the ESSnet on validation. In the below examples, we copy the description of rules exactly as they were submitted by respondents and analyse their coding in terms of the typology. Since the descriptions do not always provide all the necessary information for a complete classification, we make extra assumptions explicit.

Example 1

Field for country of birth should contain only entries from code list of countries

This rule is used to check whether a single variable occurs in a (constant) set of valid values. Hence it is coded *ssss* (level 0) in the formal typology. In the business typology it would fall in Level 1.

Example 2

if a price of reference period is different to price of last period, the Code of price change must be completed

We assume that the *price of reference period* is recorded during a different measurement then *price of last period*. If this assumption holds, the rule encompasses two measurement times and two variables: *price* and *Code of price change*. Hence, in the formal typology the rule type is classified as $smsm$ (level 2). In the business-typology it is a level 2 check, since it can be thought of as involving a sort of 'time series check'.

Example 3

If in a household record the number of persons living is that household is 2, there must be 2 records in the file of the person record

This rule concerns objects from two different universes: *households* and *persons*, so we have two options in the formal typology: $mmmm$ or $msmm.$ Assuming that the data is collected in a single household survey where various persons were interviewed, the correct type is $msmm$ (level 3) in the formal typology. In the business-typology this would probably be perceived as an in-domain check (a check on households, or demography). Assuming all data is stored in a single file, it would be a business-typology level 1 check.

Example 4

unit price = total price / quantity

Assuming the three variables *unit price*, *total price*, and *quantity* are collected in a single measurement, this rule is coded as $sssm$ (formal typology level 1). Similarly, this is an in-file check for the business typology and therefore level 1.

Example 5

We do check for duplication of respondents by checking the 'person_id'.

This concerns a single variable, *person_id*, and multiple units that presumably are collected at a single measurement time. Hence the rule is of type $ssms$ (level 1) in the formal typology. Checking for duplicates is interpreted as a structural requirement from the business perspective (considering that checking whether all columns are present is also considered structural), so in the business-typology it would be level 0.

Example 6

Number of animal at the end of reference period == number of animal at the beginning of following reference period.

This rule concerns a single variable, measured at two instances of time. The classification is therefore $ssms$ (level 1) in the formal typology. In the business typology it is a level 2 check as well, for similar reasons as in Example 2.

Example 7

If a person states that his/her mother lives in the household and states her interview-id the person with this id needs to be female.

This rule concerns a single object type (*person*), two objects, and two variables: whether a person's mother lives in the household and the recorded gender of the person's mother. Hence the classification is $ssmm$ (level 2) in the formal typology. In the business-typology it is an in-file check, and therefore level 1.

7 Data validation as a process

7.1 Data validation in a statistical production process (GSBPM)

(Marco Di Zio, Ugo Guarnera)

The business processes for the production of official statistics are described in the GSBPM (UNECE 2013).
The schema illustrated in the GSBPM is useful to see that data validation is performed in different phases of a production process. The phases where validation is performed are the following:

GSBPM: phase 2.5
The first phase when data validation is introduced is the 'design' phase and more specifically in sub-phase 2.5, that is '*design processing and analysis*'. The description in GSBPM is:
"*This sub-process designs the statistical processing methodology to be applied during the "Process" and "Analyse" phases. This can include specification of routines for coding, editing, imputing, estimating, integrating, validating and finalizing data sets*".

This is of course related to the design of a validation procedure, or more properly, of a set of validation procedures composing a validation plan.

GSBPM: phase 4.3
The first sub-phase of GSBPM where validation checks are performed is the 4.3. As described in the GSBPM document, checks are concerned with formal aspects of data and not on the content:

“Some basic validation of the structure and integrity of the information received may take place within this sub-process, e.g. checking that files are in the right format and contain the expected fields. All validation of the content takes place in the Process phase”

The other two sub-phases where validation procedures are applied are ‘Process’ and ‘Analyse’.

GSBPM: phase 5.3

In the process phase, the sub-phase 5.3 is specifically referred to validation, it is in fact named ‘review & validate’.

The description given in the document GSBPM (2013) is:

“*This sub-process examines data to try to identify potential problems, errors and discrepancies such as outliers, item non-response and miscoding. It can also be referred to as input data validation. It may be run iteratively, validating data against predefined edit rules, usually in a set order. It may flag data for automatic or manual inspection or editing. Reviewing and validating can apply to data from any type of source, before and after integration. Whilst validation is treated as part of the “Process” phase, in practice, some elements of validation may occur alongside collection activities, particularly for modes such as web collection. Whilst this sub-process is concerned with detection of actual or potential errors, any correction activities that actually change the data are done in sub-process 5.4*”

Some remarks can be done on the previous description.

i) The term input validation proposes an order in the production process. The term and the idea can be used in the handbook.
ii) Input data concern with any types of source
iii) Validation may occur alongside collection activities
iv) The distinction between validation and editing is provided, and it is in the action of ‘correction’ that is made in the editing phase, while validation only says if there is (potentially) an error or not. The relationship between validation and data editing will be discussed later on.

GSBPM: phase 6.2

The last sub-phase is the 6.2 (‘validate outputs’).

“*This sub-process is where statisticians validate the quality of the outputs produced, in accordance with a general quality framework and with expectations. This sub-process also includes activities involved with the gathering of intelligence, with the cumulative effect of building up a body of knowledge about a specific statistical domain. This knowledge is then applied to the current collection, in the current environment, to identify any divergence from expectations and to allow informed analyses. Validation activities can include:*

- *checking that the population coverage and response rates are as required;*
- ✓ *comparing the statistics with previous cycles (if applicable);*

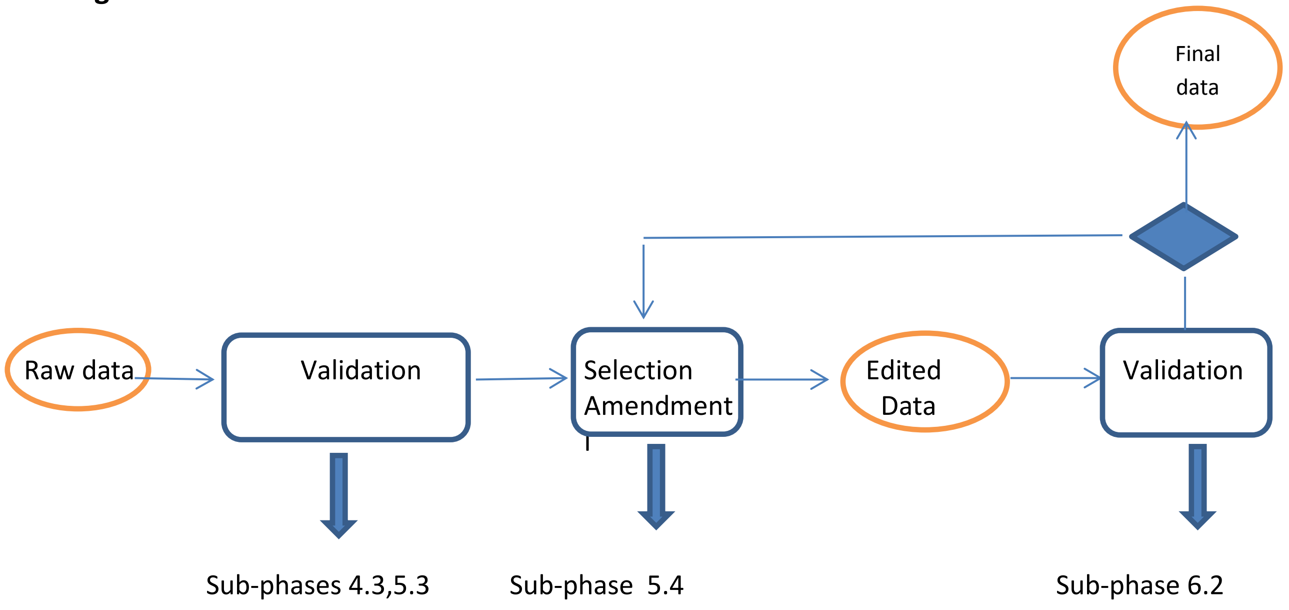
- *checking that the associated metadata and paradata (process metadata) are present and in line with expectations*
- ✓ *confronting the statistics against other relevant data (both internal and external);*
- ✓ *investigating inconsistencies in the statistics;*
- ✓ *performing macro editing;*
- ✓ *validating the statistics against expectations and domain intelligence"*

The checks that are not usually considered as a part of a 'data validation' procedure (i.e., the first and the third item where emphasis is not on data) are marked with "•".

Remark. The attention of this validation step is on the output of the 'process' step. It means that data are already processed, e.g., statistical data editing and imputations are done.

In figure 1,a flow-chart is depicted describing the different validation phases in connection with statistical data editing as described in the GSBPM.

Figure3: Flow-chart describing the different validation phases in connection with statistical data editing



In principle, there should be a decisional step addressing the end of the process also after input validation, but it is rare that input data are free of errors especially by considering also the non-response in the non-sampling errors.

This process flow is related to a data set, however it can be easily adapted to a more complex situation where more than a single provider is responsible of the release of data. An important case is the European statistical system, where each single NSI applies the process described in figure 3 and send the final data to Eurostat. Eurostat has the possibility of making further comparisons, having data from different Countries. Hence, the final collector may repeat the data validation process, with the usual exception of performing the phase of data editing (sub-phase 5.4). A similar example can be that of aggregates provided by National Accounts, in general NA collects data from different sources and for this they have the possibility of making further consistency checks that are not possible within each single part of the production chain.

From the previous considerations, the validation process is considered as a set of validation procedures.

7.2 The informative objects of data validation (GSIM)

(Marco Di Zio, Ugo Guarnera, Mauro Scanu)

According to GSIM, each data is a result of a *Process step* through the application of a *Process method* on the necessary *Inputs*.
A data validation procedure can be interpreted according to the GSIM standard (Unece GSIM 2013) that provides a set of standardized, consistently described information objects, which are the inputs and outputs in the design and production of statistics.
To this aim the validation process can be represented by an *Input,* a *process,* and an *output.*

The relevant informative objects are those characterising the *input*, the *process* and the *output*, and more in general all the objects used to describe the validation procedure in the statistical production procedure.

At first, we introduce the concept of Statistical program cycle that is for instance the survey at a certain time within a *Statistical Program.* A statistical program cycle is typically performed by means of several *Business Processes*. A Business process corresponds to the processes and sub-processes found in the Generic Statistical Business Process Model (GSBPM).
Process steps address the question: how to process the business process.
Each *Process Step* in a statistical *Business Process* has been included to serve some purposes. The purpose is captured as the *Business Function* (what) associated with the Process Step (how to do it).

According to these definitions, data validation can be interpreted as a *business function* corresponding to different business processes, which means that data validation can be performed at different stages of the production chain, in fact data validation refers to different phases of GSBPM. These phases, composed of process steps, are distinguished by their process inputs, i.e., any instance of the information objects supplied to a *Process Step Instance* at the time its execution is initiated.

Data validation is in general described by an application of rules to a set of data to see whether data are consistent with the rules. Data fulfilling rules are considered validated. The output of the process is a data set with indicators addressing which data are considered acceptable and hence validated, indicators about metrics computing..., and indicators measuring the severity of the failure (if any).

The data sets are to be considered in a broad way, they can be composed of microdata or aggregates, they can have a longitudinal part or not.

GSIM defines data set as:
"*A Data Set has Data Points. A Data Point is placeholder (for example, an empty cell in a table) in a Data Set for a Datum. The Datum is the value that populates that placeholder (for example, an item of factual information obtained by measurement or created by a production process). A Data Structure describes the structure of a Data Set by means of Data Structure Components (Identifier Components, Measure Components and Attribute Components). These are all Represented Variables with specific roles.*

Data Sets come in different forms, for example as Administrative Registers, Time Series, Panel Data, or Survey Data, just to name a few. The type of a Data Set determines the set of specific attributes to be defined, the type of Data Structure required (Unit Data Structure or Dimensional Data Structure), and the methods applicable to the data."

This definition is broad enough to include the elements data validation is supposed to analyse.

The input of a validation procedure must include the variables to be analysed, however it is worthwhile to notice that GSIM defines separately informative objects for the meaning and the concrete data-representation, i.e., it distinguishes between conceptual and representation levels in the model, to differentiate between the objects used to conceptually describe information, and those that are representational.

The validation procedure requires the description of the variables at representation level. Furthermore, it is necessary to associate variables to the data set(s). The GSIM informative object is the *instance variable.* From GSIM we have that "a*n Instance Variable is a Represented Variable that has been associated with a Data Set. This can correspond to a column of data in a database. For example, the "age of all the US presidents either now (if they are alive) or the age at their deaths" is a column of data described by an Instance Variable, which is a combination of the Represented Variable describing "Person's Age" and the Value Domain of "decimal natural numbers (in years)"*.

Finally, the *parameter* object is an essential input for the process since it is concerned with the parameters required by the rules used in the process.

In GSIM, a *Process Method* specifies the method to be used, and is associated with a set of Rules to be applied. For example, any use of the Process Method 'nearest neighbour imputation' will be associated with a (parameterized) Rule for determining the 'nearest neighbour'. In that example the Rule will be mathematical (for example, based on a formula). Rules can also be logical (for example, if Condition 1 is 'false' and Condition 2 is 'false' then set the 'requires imputation' flag to 'true', else set the 'requires imputation flag' to 'false'). In case of validation, a common example of process method is a balance edit, Closing inventory = Opening Inventory + Purchases - Sales.

Process Outputs can be composed of reports of various types (processing metrics, reports about data validation and quality, etc.), edited *Data Sets*, new *Data Sets*, new or revised instances of metadata, etc.
More precisely, in data validation process outputs are metrics measuring the severity of possible failures and a set of logical values addressing whether the unit/variables are acceptable or not.

The data set is the same as the one in the input, the same holds for data structure and variables

- We are now able to represent a validation procedure as a generic process defined in terms of an input, a process and an output, characterised respectively by GSIM informative objects, see Table 7

Table 7. GSIM informative objects characterising a data validation procedure

INPUT	**PROCESS**	**OUTPUT**
– Data set, – Data structure – Instance variable – Parameter (of the rules)	– Process method (defined by the rule) – Rule	– Process output – Data set – Data structure Variable

This generic procedure may be applied in different business processes for the production of official statistics (different phases of GSBPM). Each of these different applications will be characterised by specific input parameters, process objects and also outputs.

While the previous informative objects are sufficient to describe a validation procedure, in order to place precisely a validation procedure in a statistical production process, further information is needed, in fact there is the need to associate the procedure to a *cycle* of *statistical program (e.g., a survey at a certain time)*, a *business process (a phase of GSBPM) and process step (steps performed in the business process).* These informative objects represent the coordinates to place exactly the validation procedure a production process.

8 The data validation process life cycle

(Nadežda Fursova, Jūratė Petrauskienė)

In order to improve the performance of a statistical production process by managing and optimising the data validation process, it is useful to describe the data validation process life cycle. First, the process should be seen as a dynamic and complex process. Adapting validation rules may influence not only in the scope of one data set or one statistical domain, but also to all statistical domains. For instance, the optimization of efficacy and efficiency of the validation rules should take into account their assessment in the previous occasion, relations of indicators, etc. Second, the process should be seen as an integral part of the whole statistical information production process.

The data validation life cycle involves the activities directly linked to each statistical domain for the definition and execution of data validation. This cycle starts by **designing** the data validation process for the statistical domain or inter statistical domain, with an overall study of the data sets, variables and its relations to find a list of suitable and effective validation rules. In the **implementation** phase, these validation rules are described in common syntax, formalised, tested and refined, discussed and evaluated by stakeholders. During the **execution** phase, data are checked against the rules; validation results are measured and quantified. These outputs will be **reviewed** to improve the list of validation rules.

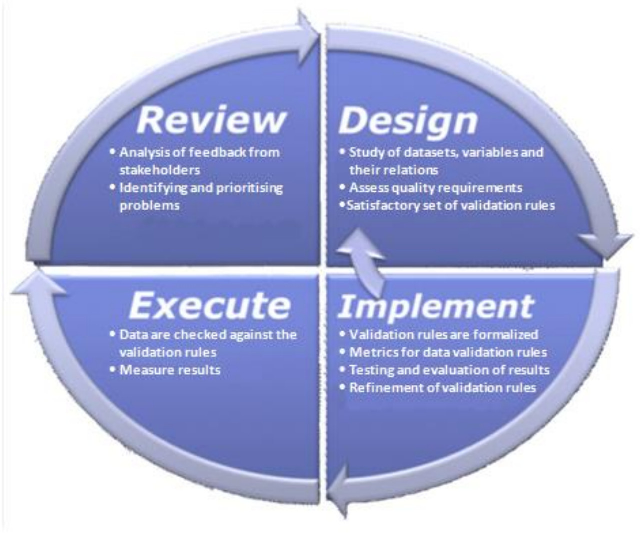
The data validation life cycle process includes the review of obtained statistical data through data editing, in fact the output of this task is used to improve the data validation procedure in an iterative way.

Data validation process is an integral part of the whole statistical information production process. Validation tasks and controls are performed by several stakeholders with a wide range of responsibilities. The data validation process life cycle should provide clear and coherent allocation of actions and responsibilities to ensure the highest performance, while reducing the possibility of mistakes. Though, allocation of responsibilities for each phase of data validation life cycle is hardly possible due to the complexity of data validation procedure and because this is strongly related to the particular organization structure.

Designing validation rules and rule sets for a data set implies the distribution of validation tasks in the statistical production chain to be proposed to the decision making structures. This distribution of responsibilities should be designed following the principle of *"the sooner the better"* as it is commonly agreed that the cost in terms of resources, time and quality of fixing data errors is lower as closer it is to the data source.

The data validation process life cycle is represented in figure 4.

Figure 4: Data validation process life cycle


Review
• Analysis of feedback from stakeholders
• Identifying and prioritising problems
Design
• Study of datasets, variables and their relations
• Assess quality requirements
•Satisfactory set of validation rules
Execute
• Data are checked against the validation rules
• Measure results
Implement
• Validation rules are formalized
• Metrics for data validation rules
• Testing and evaluation of results
• Refinement of validation rules

8.1 Design phase

The design of a data validation process is a part of the design of the whole survey process. The data validation process has to be designed and executed in a way that allows for control of the process. The design of the validation process for a data set in or between the statistical domains requires setting up the validation rules to be applied to the data set.

These set of validation rules should be complete, coherent, and efficient and should not contain any inconsistencies. Designing a set of validation rules is a dynamic process. Validation rules should be designed in collaboration with subject matter specialists and should be based on analysis of previous surveys. Consistency and non-redundancy of rules should be verified. Validation rules should be designed cautiously in order to avoid over-editing. Effective validation rules can be obtained by differently combining approaches and “best practices”.

In this phase the validation process should be planned and documented for further progress monitoring. The overall management of the process and the interfaces with the other sub-processes should be considered. For each phase the resources and time needed to implement, test, execute, review and document should be planned.

This is the phase where survey designers, questionnaire designers, validation and editing specialists and subject matter experts have to co-operate.

Activity descriptions

- ✓ **Assess quality requirements for data sets**

- ✓ **Overall study of data sets, variables and their relations**

- ✓ **Determine satisfactory set of validation rules for the data.** In order to make data production process more efficient, reducing time and human resources, but considering quality requirements.

- ✓ **Assess responsibilities and roles.** Document who is doing what; who is responsible for different actions; who is accepting and adopting the validation rules, etc.

- ✓ **Integrate the data validation process in the overall statistical production process.** Design the connections with other phases of the statistical production processes.

- ✓ **Improvement of the validation according to the results of the review phase**

A document with the form of guidelines with some theoretical background, examples and best practices could support the task of the domain manager when designing the entire validation process.

8.2 Implementation phase

Once the data validation process has been designed, it has to be implemented with a parameterisation, thoroughly tested, tuned and become productive.

The validation process should be tested before it is applied. Validation rules and editing techniques and methods should be tested separately and together. It is important to realize that once the validation process is implemented in the actual survey process, only slight changes should be made to monitoring and tuning in order to avoid structural changes.

Common definitions and descriptions applied to data validation are required for a common understanding of the whole validation process.

A proper documentation of the validation process is an integral part of the metadata to be published. The aim of documentation is to inform users, survey managers, respondents, validation and editing specialists about the data quality, the performance of the process, its design and adopted strategy. The documents can be of three types: methodological, reporting and archiving.

The validation rules should be written in an unambiguous syntax that could allow communicating the rules amongst the different actors in the production chain and could also be interpreted by IT systems.

People working on validation and related aspects should have a sound knowledge of the methods that can be adopted, aware about the links between the validation and the other parts of the statistical production process. At this phase cooperation from methodologist and IT specialist should be very concise.

Activity descriptions

- ✓ **Validation rules are formalized and described in a common syntax.**

- ✓ **Determine metrics for data validation rules, assessment of validation process and validation rules.** Validation rules should be assessed for quality (clear, unambiguous and consistent, saving time resources).

- ✓ **Testing. Apply validation rules to test data (real data, artificial data) and producing indicators.**

- ✓ **Test results (indicators, validation rules, metrics, quality aspects, etc.) are evaluated by stakeholders (Eurostat, Member states, Domain managers, etc.).** Reporting documents on test results and evaluation should be prepared and saved for review phase.

- ✓ **Refinement of validation rules according to the test results and consultations with stakeholders**

- ✓ **Documenting.** Data validation rules should be well documented – documents depend on the purpose and the final user: producers, users of the results, survey managers or methodologists.

8.3 Execution phase

The execution phase consists of identifying values that are not acceptable with respect to rules expressing logical, mathematical or statistical relationships. This process usually consists of a set of integrated validation methods dealing with different type of errors. This allows assessing the quality of the data and helps to identify error sources for future improvements of statistical production process.

The result of execution phase is a flag pointing out acceptable and not acceptable data, and generally a score measuring the degree of severity of failure.

A standard communication of error/warning messages may increase the global efficiency of statistical production and impacts directly the time required for understanding and locating the source of the error. As well, this standardisation may lead to an automatic treatment of validation messages by IT tools.

It would be desirable to reach certain level of harmonisation in the presentation of validation results with agreed validation metrics. More about metrics on validation could be found in the second part of this handbook: Metrics for a data validation procedure. The quality measures could be used as standard reporting devices which are routinely calculated.

The part of this phase is gathering the statistics on validation outcomes to assess the quality of data sets and quality of validation rules.

Data, programs and the corresponding metadata have to be documented and archived if the process should be repeated or if new methods will be tested for a data sets. It is desirable to have common approach for validation procedure to keep validation rules in one place maintained and supported continuously, friendly users' application and specification written in understandable language for different users of the application.

Activity descriptions

- ✓ **Data are checked against the validation rules.** Validate data against predefined validation rules.

- ✓ **Summarising results.** It depends on the user of the results (staff, management or methodologist).

8.4 Review phase

This phase is aimed at continuous improvement of validation process efficacy and data quality. During the review phase needs for new design elements are established. This phase includes identification of problems using feedback from the users and other stakeholders and analysing outcomes from the execution phase. The identified problems are prioritised and dealt with in the design phase.

Examples of revisions are:

Improvement of validation rules due to:

- Replacing those that detect few errors by others more powerful
- Replacing those that 'mislead': detect errors that are not real errors
- Increase efficiency of validation rules
- Improvements in validation rules: detecting more possible errors
- Changes in the data file or regulations

Changes in the validation process originated by:

- Changes in validation tools
- Changes in file formats
- Improving efficiency

Changes in the validation workflow due to:

- Better assignment of responsibilities in validation tasks
- Efficiency gains in the chain

Activity descriptions

- ✓ **Analysis of feedback from stakeholders.** Feedback gathered in previous phases.
- ✓ **Analysing of outcomes from the execution phase.** Identified potential problems, errors, discrepancies, detected systematic problems are analysed in order to decide whether validation rules should be reviewed.
- ✓ **Identifying and prioritising problems.**

Metrics for data validation

9 Metrics for data validation

(Lucas Quensel-von Kalben)

The objective of the good design of a set of validation rules is to achieve a satisfactory quality that would permit the statisticians to have a reasonable confidence that the outcome of the validation process is free of important errors and that the cost of that process is not disproportionate satisfying some requirements of efficiency (fast, low detection of false errors) and completeness (most true errors are detected).

Designing and maintaining a set of validation rules is a dynamic learning validation process as it can be improved through the experiences drawn from the checking of successive data vintages. In fact, evaluation of the existing validation rules should be periodically performed on the basis of the validation results (e.g., flag and hit rates and any external feedback on presence of errors that have survived the validation process).

The following actions are performed:

- Less effective/efficient rules are replaced
- More effective/efficient rules are incorporated to detect systematic problems
- New rules may be added to detect errors that escaped from previous checks.

It is important to have indicators providing quantitative information to help the design and maintenance of a data validation procedure, and for monitoring the data validation procedure as well.

Indicators should measure the efficacy and efficiency of a data validation procedure. The indicators may refer either to a single rule or to the whole data validation procedure, that is the set of validation rules.

Indicators may be distinguished in

- Indicators taking into account only validation rules (properties)
- Indicators taking into account only observed data
- Indicators taking into account both observed and reference data (e.g., imputed data, simulated data).

The first two are generally used to fine tune the data validation procedure, for instance in the design phase and by using pilot survey.

The indicators of the third type are used to obtain a more precise measure of the effectiveness of a data validation procedure, but are dependent on the method chosen to obtain amended plausible data, or synthetic data. The method is composed of an error localization procedure and an imputation procedure.

Evaluation of validation rules can be done by looking at their efficacy, i.e., the capacity of reaching the target objective. However, when evaluating a validation rule, it should be considered also its capacity to find important errors. These two aspects, already defined with the term ‘severity’, are to be considered jointly when evaluating the efficacy of a validation rule. As an example, let’s look at a balance edit $x+y=z$ and at a ratio edit $a<x/y<b$ defined to check outliers. The first one addresses an error with a probability 1, the second, by the definition of outliers, identifies potential errors (probability less than 1) but given that the values are in the tails, it can be useful to find important errors. A balanced analysis of these two aspects is necessary to evaluate the efficacy of validation rules.

10 Properties of validation rules

(Mark Van der loo)

Since data validation is an important and intrinsic part of statistical production and data exchange, it is worthwhile to regard the building blocks of data validation, the data validation rules, as object of study. In particular, it is interesting for reasons related to quality of both the statistical process and the data to have insight into the (over)completeness, effectiveness, internal consistency and cohesion and complexity of a set of rules. The ability to describe such properties, regardless of to what data set the rules are applied can help practitioners to weed out redundancies and understand the implications of combinations of rules.

There seems to be little literature in official statistics that aims directly at these goals, although a number of the basic techniques that are necessary for such investigations have been discussed in the context of other problems where rule manipulation is a basic requirement. In particular, several techniques have been developed in the context of error localization (see De Waal et al., 2011 and references therein). Since the literature on these techniques is limited to what are commonly called ‘in-record’ (cross-variable) validation rules, the current section is limited to these rules as well[2] except when otherwise indicated. For example, a rule such as

Profit + Cost = Total Revenue,

is included in the discussion, but a rule stating

The average price of this month must not differ from last month’s price by more than 50%,

is not covered in this discussion. That is to say, the latter example is covered to the extent in which they can be represented as in-record rules. One may always create a data set where the average price of the current and previous month are stored in a single record. However, all the information used to compute the averages is then lost and not part of computing the outcome of the

[2] In terms the formal typology of Section **5**, Part I of this handbook we restrict ourselves to rules of the type *sssm*.

validation rule. For example, in case the rule is violated there is no way based on that rule and average data alone to localize the record or records that cause the violation (if any).

Since the explicit literature on rule maintenance and properties is limited, so is the discussion in the current section of this handbook. However, we have attempted to gather a number of approaches and techniques that appear useful when describing rule sets – but necessarily on a shallow level. In the following Sections we cover techniques to determine the completeness, redundancy, consistency, and complexity of a set of rules.

10.1 Completeness

With *completeness*, we mean the **extent** to which prior knowledge about a data set has been expressed in terms of a set of validation rules. Since the term 'prior knowledge' is hard to quantify, it will in general be difficult to find tools or methods to systematically asses completeness. Worse than that, with the current state of practice it is hard to encode all domain knowledge in (hard) validation rules. Knowledge that is easily encoded includes restrictions that follow from physical or logical facts such as: `age cannot be negative', or: `the profit must be smaller than or equal to revenue'. Knowledge that is hard to encode in rules include facts or events whose influence on the values recorded in a data set can only be stated in imprecisely. For example, although it is clear that a takeover, fusion, or acquisition influences the cost structure of a company, it is difficult to quantify this in anyway precisely. Another example, consider political choices and economic circumstances that influence the purchasing power of certain households. An expert judging household records or aggregates may take knowledge of such situations into account, but quantifying is exceedingly difficult. With these limitations in mind, in this Section we focus primarily on knowledge that can be stated with precision in terms of in-record validation rules.

Two obvious completeness-related problems may occur. The first is *incompleteness*, by which we mean that there are restrictions, implied by physics or logic that have not been stated explicitly or implicitly by a set of rules. The second is *over-completeness*, which means that the set of rules is too restrictive and value combinations that are actually valid are unjustly excluded.

10.1.1 Peer review

The basis of finding out whether a set of rules in- or overcompletes with respect to the knowledge of a domain is by knowledgeable peers. Given a rule set that is produced by an expert or team of experts, one can follow two approaches for judging completeness.

In the first approach a second team of peers (or peer) sets up a set of rules independently of the original team. The differences between the rule sets: rules that occur in one, but not in the other can then be discussed in terms of necessity and validity. It is important that the focus of such a discussion is on real world assumptions that underlie the rule sets rather than the way they have been stated. In fact, we will show in Section [ref to sub section on redundancy], that it is possible to determine whether two sets of rules are equivalent (in a sense to be stated more precisely), so such discussions can be avoided.

In the second approach, a rule set and its documentation is reviewed by a (team of) peer(s). For each rule the underlying assumption is judged against expert knowledge, and it is checked whether these assumptions are made sufficiently clear by either the form of the rule or clarity of documentation. One might score each rule, indicating to what extent the underlying assumption is hard, amenable to change over time, or is `soft' for example because it depends on a threshold of which the value is to some extent subjective. Such indicators can be used to judge as to how often a (subset of) the rule set must/should be revised.

10.1.2 Formal methods

On the more formal side, one may investigate to what extent the variables in a data set are covered by the rules in a set. Recall that we may associate with an in-record validation rule a *validation function* that takes a record of data and returns a value in $\{0,1\}$, where 0 indicates failure (the rule is violated) and 1 indicates success (the rule is satisfied). For example, given a record with the variables *profit (p), staff cost (s), total cost (c), and total revenue (t).* We can define the following rules

$$
\begin{aligned}
&p + c = t \\
&s \leq c \\
&s \geq 0 \qquad\qquad\qquad\qquad (1)\\
&t \geq 0 \\
&c \geq 0
\end{aligned}
$$

For the rule $p + c = t$, we can define the validation function v as

$$v\colon \mathbb{R}^4 \to \{0,1\}$$

$$v(p,s,c,t) = \textbf{if } p + c = t \textbf{ then } 1 \textbf{ else } 0.$$

Now, we say that a variable *occurs* in a validation function (or rule) when the value of the validation function can change when the variable is changed. In this example, the variable s does not occur in v and variables p, c, and, t do occur in v.

As a first metric for coverage, one check for each variable whether it occurs in at least one of the explicitly defined rules. It is reasonable to assume that for each measured or observed variable occurs in at least one check. In the example, all four variables occur in at least one rule.

It is tempting to extend this check and to tabulate the number of occurrences for each variable. However, one must be careful when interpreting such tables since these numbers are not in general uniquely defined. For instance from the rule set above the numbers of occurrences is $(p = 1, s = 2, c = 3, t = 2)$. However, we can solve c from the demand that $p + c = t$ and substitute it in the demands $s \leq c$ and $c \geq 0$. This yields the equivalent[3] rule set

$$p + c = t$$

$$s \leq t - p$$

$$s \geq 0 \qquad\qquad (1b)$$

$$t \geq 0$$

$$t - p \geq 0.$$

Counting the occurrences now yields $(p = 3, s = 2, c = 1, t = 4)$. In fact, the number of rules can be variable as well. Consider the following subset of rule set (1)

$$s \leq c$$

$$s \geq 0$$

$$c \geq 0$$

The rule $c \geq 0$ is redundant, since there is no way that rule $s \leq c$ and $s \geq 0$ can be satisfied for negative c. In other words, rule set (1) is also equivalent to the set

$$p + c = t$$

$$s \leq c$$

$$s \geq 0 \qquad\qquad (1c)$$

$$t \geq 0,$$

Yielding yet another variable (and rule!) count, namely $(p = 1, s = 2, c = 1, t = 2)$.

The above indicates that counting variable occurrences (or rules) has little meaning, unless one somehow declares a standard (possibly minimal or irredundant) way for formulating rules. This conclusion generalizes to any data validation rule set since it really only depends on the question whether logical deductions can be made from a set of rules. As far as the author is aware, there is currently no method or algorithm described in literature that allows one to derive a unique representation from any set of in-record validation rules.

[3] Equivalence meaning that the same set of records satisfy the set of rules.

10.2 Redundancy

A set of validation rules divides the space of all possible records into a valid, or acceptance region and an inacceptable region. Suppose we have a set of validation rules V. We say that a rule is *redundant* in V if removing it from the set of rules does not alter the acceptance region.

There are two reasons to remove redundancies from a set of rules. First, redundancy removal yields a more compact and elegant rule set that expresses the necessary restrictions in some sense minimally. Secondly, the time and/or memory consumption of some algorithms that make use of rule sets can depend strongly on the number of rules provided. One example is the branch-and-bound error localization algorithm of De Waal and Quere (2003), whose time and memory consumption grow exponentially with the number of rules if redundancies are not taken care of at runtime.

The downside of removing redundant rules is that they may be less understandable by experts. For example, as stated above, the rules

$$s \leq c$$

$$s \geq 0,$$

together imply that $c \geq 0$. This becomes however only apparent after some reasoning. In practice, rule sets with more than 100 (in)equalities are not uncommon so one can hardly expect a domain expert to produce or interpret an irredundant rule set. Moreover, when interpreting the output of a validation step (that is, the set of Boolean values that result from confronting data with a set of restrictions) one would like to connect that output directly with the defined rules, rather than with a reduced set.

This means there is a trade-off between the mathematical rigor of obtaining an absolutely irreducible set of rules and a user-friendly set that is more directly connected to the assumptions formulated by domain experts. A reasonable compromise is to let domain experts formulate the rules in a formal language that is close to their own, but to remove redundancies automatically as much as possible when this is beneficial for the algorithms making use of such rules.

10.2.1 Methods for redundancy removal

Several methods for removal of redundant constraints have been described in literature. A recent comparative overview in the context of linear programming is given by Paulraj and Sumathi (2010). As an illustration on how to approach redundancy removal, we follow Daalmans (2015) and describe shortly a method that has been described by Chmeiss et al. (2008) and also by Felfernig (2014).The idea of the method depends on the ability to find inconsistencies (see next Section).

Consider a set $V = \{v_1, v_2, \dots, v_m\}$ of validation rules. Suppose that one of the $v_i \in V$ is redundant, that is, it must always yield 1 when the other rules yield 1 (we also say that v_i is implied by the other rules). Then, if we replace v_i in V with the opposite rule, the resulting set of rules becomes *infeasible*. That is, the acceptance region is the empty set: no record can ever satisfy all rules in this adapted set. As an example, consider again the following set of rules, which we now give names, for convenience.

$$V = \{\, v_1 := \; s \leq c,\; v_2 := \; s \geq 0,\; v_3 := \; c \geq 0\}.$$

Now, v_2 and v_3 imply that any combination (s, c) must lie in the first quadrant of the plane, while $v_1$implies that only points on and above the line $c = s$ are valid. Let us now replace v_3 with its opposite and define

$$V^* = \{\, v_1 := \; s \leq c,\; v_2 := \; s \geq 0,\; \neg v_3 := \; c < 0\}.$$

Now, v_2 and $\neg v_3$ imply that every point must lie in the fourth quadrant of the $s - c$ plane, but not on the s axis. This conflicts with v_1 which implies that every point (s, c) must be on or above the line $c = s$.

This idea, that replacing an implied rule with its negation yields an infeasible rule set is fully general, and is thus not limited to numerical examples as given above. Given that we are able to establish the feasibility of a set of rules, the following procedure for redundancy detection readily presents itself (the backslash indicates set difference, and $\neg$ indicates negation).

Procedure: Identification and removal of redundant validation rules

```
Input : Set of rules V
Output:  A set of rules V, without redundant rules
1 For each rule $v \in V$ do
2   $V^* \leftarrow \{V \backslash v\} \cup \neg v$
3   if $V^*$ is not feasible then $V \leftarrow V \backslash v$
```

Daalmans (2015) discusses several variants of the above procedure that employ a mixed integer programming approach to determine feasibility and redundancies for rule sets that are common in official statistics.

As a corollary, we obtain a method to determine whether two rule sets are equivalent, in the sense that they have the same acceptance region. Consider two rule sets $V = \{v_1, v_2, \dots, v_m\}$ and $W = \{w_1, w_2, \dots, w_n\}$. A rule w_i is redundant in V if $\neg w_i \cup V$ is infeasible. More general, we say that V is redundant in W when every $v_i \in V$ is redundant in W. Now, we can define that V and W

are *equivalent* when both V is redundant in W and W is redundant in V. The following procedure tests such equivalence.

Procedure: Testing equivalence of two rule sets

```
Input : Sets of rules V and W
Output: A Boolean, EQ, indicating equivalence of V and W.
1 EQ ← TRUE
1 For each rule $v \in V$ do
2   $W^* \leftarrow W \cup \neg v$
3   if $W^*$ is not feasible then EQ ← FALSE; STOP.
4 For each rule $w \in W$ do
5   $V^* \leftarrow V \cup \neg w$
6   if $V^*$ is not feasible then EQ ← FALSE; STOP.
```

The procedure terminates when one non-redundant rule in one set is found in the other. Determining equivalence this way is computationally expensive, since the computational time necessary for determining feasibility of a rule set grows exponentially with the number of rules.

10.3 Feasibility

A rule set is called *feasible,* or informally also *consistent,* when the acceptance region defined by a rule set is nonempty. Infeasibility occurs for instance when a rule set contains a rule that is contradictory in itself, for example the rule that states $x > x$ is clearly contradictory. As a second example, consider the following rule set has rules that are perfectly feasible by themselves but their combination is contradictory:

$$\{x > 1, x < 0\}$$

Clearly, there is no number x that can satisfy both rules. In practice, rule sets can be much more complicated than this and contradictions rarely present themselves in such a clear form.

Now, consider a general set of validation rules $V = \{v_1, v_2, \dots, v_m\}$. We may think of these rules as functions that take a data record, say $\boldsymbol{r}$, and return a value $v_i(\boldsymbol{r})$ in $\{0,1\}$, where 0 means the rule is violated and 1 means the rule is satisfied. We emphasize that $\boldsymbol{r}$ is not necessarily numeric, but comes from a domain D containing all possible records that may have been measured. For example, consider a survey where we ask n persons about their age and whether they are employed. The domain for a single record might be described by $D = \mathbb{R} \times \{\text{yes}, \text{no}\}$. An example record is $\boldsymbol{r} = (38, \text{yes})$ and we have the rule set

$$V = \{v_1 \coloneqq age \geq 0, v_2 \coloneqq \text{if } (age < 15) \text{ then } job = \text{no}\}.$$

Now we can associate with each rule a subset of D for which the rule is valid:

$$v^{-1}(1) = \{\boldsymbol{r} \in D \colon v(\boldsymbol{r}) = 1\}.$$

In our example we have

$v_1^{-1}(1) = [0, \infty) \times \{\text{yes}, \text{no}\}$

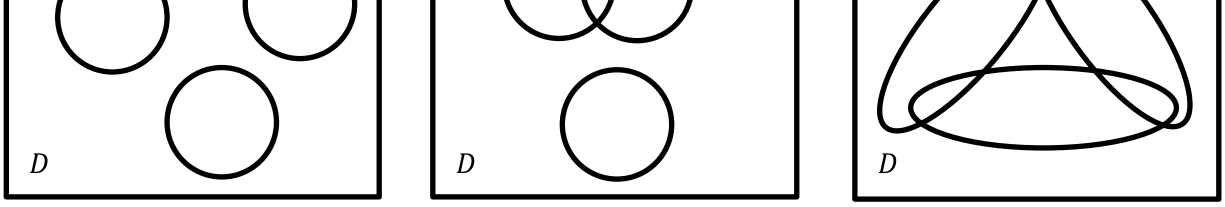
$v_2^{-1}(1) = (-\infty, 15) \times \{\text{no}\} \cup [15, \infty) \times \{\text{yes}, \text{no}\}$.

Since we want all rules to be satisfied simultaneously, the acceptance region for V is given by the nonempty intersection $v_1^{-1} \cap v_2^{-1} = [0,15) \times \{\text{no}\} \cup [15, \infty) \times \{\text{yes}, \text{no}\}$.

We can generalize this idea to give a more general definition of feasibility, and state that a set of rules $V = \{v_1, v_2, \ldots, v_n\}$ is infeasible when the intersection of all valid regions is empty, or in notation when

$$\bigcap_{i=1}^{m} v_i^{-1}(1) = \emptyset.$$

With this general definition, we can draw pictures to see in what ways a collection of rules might be contradictory. Below we draw three situations for a rule set consisting of three rules. The rectangle depicts the valid domain D and the circles or ellipses indicate the valid regions associated with each rule. In the first situation. All three cases show a situation where the rule set is infeasible. In the first case there is no overlap between any of the rules. In the second case, two rules overlap and have a combined valid region, but the third rule has no overlap with the others so the set as a whole is infeasible. In the third situation, each pair of rules can be satisfied, but there is no common overlap between the three rules, hence the rule set is infeasible.



These pictures illustrate why determining the feasibility of a set of rules is in general a hard problem: determining feasibility may involve computations on every subset of rules. Nevertheless, for certain classes of rule sets strategies for determining feasibility have been developed.

10.3.1 Methods for finding inconsistencies

There are two common strategies for determining the feasibility of a set of in-record feasibility rules. Both strategies have in common that the rules must be expressible in a single general form. To denote this general form, we first write a general record so that all the categorical variables c_i come first, and the numerical variables x_i come next.

$$\boldsymbol{r} = (c_1, c_2, \dots, c_m, x_1, x_2, \dots, x_n) = (\boldsymbol{c}, \boldsymbol{x}). \qquad (2)$$

Now, we assume that rules can be written in the form (De Waal, 2002)

$$\text{if } \boldsymbol{c} \in F \text{ then } \boldsymbol{x} \in \left\{\boldsymbol{x} \in \mathbb{R}^n : \sum_{i=1}^{n} a_i x_i \leq 0\right\}.$$

Here, F is a subset of combinations of categories and the a_i are real coefficients. It can be shown [see e.g. De Waal (2002), De Waal et al (2011)] that many commonly occurring in-record validation rules can be written in such a form. Amongst others, this form includes multivariate conditions on categorical variables, linear equality and/or inequality restrictions and conditional rules involving both categorical and numerical linear restrictions. A slight generalization was recently formulated by De Jonge and Van der Loo (2014) in the context of software for error localization problems.

The first strategy to determine feasibility is based on methods for simplifying rule sets by eliminating variables. For example, consider the rule set

$$\{x \geq 1, x < 0\}.$$

We write this set as a system of linear inequalities, and add the two demands together as follows

$$\begin{array}{r} -x < 1 \\ x < 0 \quad + \\ \hline 0 < 1. \end{array}$$

The resulting demand contains one variable less than the two original rules, and it is an obvious contradiction. Since the original system implies a contradiction, the system must be infeasible.

This example can be generalized in the following ways. First of all, variables can always be eliminated from systems of inequalities through a procedure called Fourier-Motzkin elimination (see e.g. Williams 1986). Also, it can be shown that if a set of linear inequalities is infeasible, repeated Fourier-Motzkin elimination will always lead to a simple contradictions such as $0 < 1$. For rules concerning categorical variables, a procedure called multivalent resolution (Hooker, 2000) can be used to eliminate a variable, which again can be repeated to generate obvious contradictions if the original set was infeasible to begin with. Finally, De Waal (2002) shows how these elimination methods may be combined to include rules of the general form of Equation (2).

The second strategy relies on mixed-integer programming (MIP), and existing mixed-integer problem solving software. The idea is to make the set of validation rules part of the restrictions in a specific optimization problem which is then fed to a MIP solver. If no solution can be found by the solver the set of rules is contradictory. See Daalmans (2015) and references therein for examples.

Advantages of the first strategy include reliability and numerical stability. Fourier-Motzkin elimination is available in free software, such as the R-package *editrules* (De Jonge and Van der Loo, 2011) or in a more basic interface in the lintools package (Van der Loo and De Jonge, 2015). However, variable elimination methods can be both computationally intensive and may consume a lot of memory. In principle, the number of rules can grow exponentially with each elimination step unless redundancies are taken care of (see e.g. Van der Loo and De Jonge, 2014 for a discussion of performance in the context of error localization).

Advantages of the second strategy include the ability to use existing (possibly free) solvers for MIP problems which are often well-developed and high performing. The drawback involves possible lack of numerical stability. De Jonge and Van der Loo (2014) discuss that solving MIP problems where coefficients differ more than 8 orders of magnitude are likely to yield spurious solutions. Such coefficients are not unlikely in economic data, where for example billions of currency occur in the same rule as staff numbers. One then needs to carefully choose the unit of measurement to avoid such problems, as scaling is hard to fully automate.

10.4 Complexity

Unlike completeness, redundancy, and feasibility, there is no single definition of the complexity of a set of validation rules, although most statisticians or analysts probably have an intuitive feeling as to what such a complexity might entail. In the following subsections we discuss several notions of complexity regarding validation rules or validation rule sets.

10.4.1 Information needed to evaluate a rule

The first notion of complexity is related to the variety of information that is necessary to evaluate a validation rule. In the most simple case, a rule can be evaluated by comparing a single data value with a fixed range of values. Stepping up a level in complexity, we find rules that compare a data value by one or more other values, for example in balance edits, or rules where an observation at time t is compared to an observation at an earlier time $t-1$. Going up in complexity we find rules where a value is compared with (functions of) ranges of other values. For example, a value can be compared with a location estimator of a different variable, computed from the current data set.

This notion of the 'variety of information' necessary to compute a validation rule is precisely the principle behind the formal typology that has been described in the first part of this handbook, in Section 5**.** There, a minimal set of four labels are identified that characterize a data point: the Universe (or domain, type of statistical objects), the time of measurement, the actual unit on which a measurement was performed and the variable that was measured. The levels of complexity then correspond to the number of labels that must be varied in order to compute the validation function.

We refer to Section 5 for further explanation and elaborated examples.

10.4.2 Computational complexity

The second notion of complexity pertains to the amount of computational effort and/or the amount of memory that is necessary to compute a validation rule. One speaks of *time complexity* to indicate the amount of computational effort necessary and *space complexity* to indicate the amount of memory necessary.

Here, we will concern ourselves with time complexity. Computational complexity is a well-established subfield of computer science, and standard textbooks are available (See for instance the book by Arora and Barak (2007), for which a draft copy is freely available online).

Instead of determining the precise amount of computational effort necessary to complete a programme, the field of computational complexity studies how the computational effort to complete a programme *grows* when the input of a programme grows. The most commonly used concept is the so-called big-O notation. Given two functions f and g. We say that a function is $f=\mathcal{O}(g)$ if there is a constant c such that $f(n)\leq cg(n)$ for every sufficiently large n. In computational complexity, n represents the amount of input to a programme.

Below are some examples concerning common operations in statistics and data processing, that may also be part of a validation rule.

Operation	**Complexity**
Comparing a value to a constant	$\mathcal{O}(1)$
Comparing n values to a constant	$\mathcal{O}(n)$
Computing the mean over n values	$\mathcal{O}(n)$
Computing the variance over n values	$\mathcal{O}(n)$
Sorting a set of n records	$\mathcal{O}(n\,log\,n)$
Computing the median over n values	$\mathcal{O}(n\,log\,n)$
Joining n with m records	$\mathcal{O}(n+m)$
Computing m linear regression coefficients (once the design matrix is available, of which the construction takes $\mathcal{O}(n)$ steps, where n is the size of the training set).	$\mathcal{O}(m^3)$

To compare a value to a constant, only a single basic operation is necessary. Doing this a number of times makes the computational time necessary grow linearly. To compute the mean over n values we need to go over all values, add them, and divide the results (or add them each time after division by n). Hence the number of basic operations grows as n. Computing the variance in principle takes longer since we need to compute both the mean over the squared values and the square of the mean. However, the amount of time necessary still grows as n. Of course, the actual complexity also depends on the algorithm used to perform an operation. For example, there are many sorting algorithms, and only the fastest known have complexity of $\mathcal{O}(n\,log\,n)$. Computing the median is of the same order as a sort since it involves a sorting step. In the case of the join we assume an algorithm using hash tables is used since a naive approach would yield $\mathcal{O}(nm)$ complexity.

Overall, the computational complexity of operations typically involved in the evaluation of validation rules is on the low side of the computational spectrum, given that many other computational problems are of exponential or higher order. Most of the time in a data validation step is probably spent on data retrieval and collection of results. When data validation is becoming a performance-critical step, this is probably the best place to start looking.

10.4.3 Cohesion of a rule set

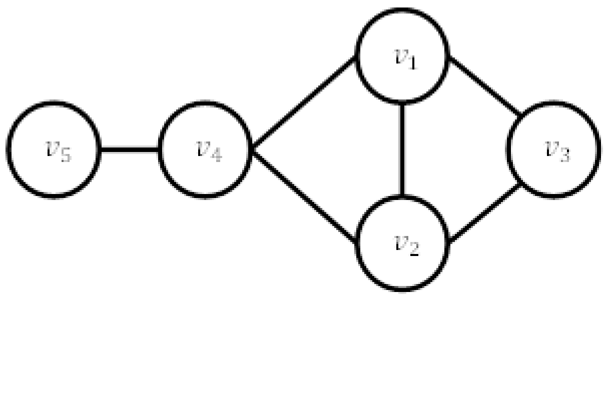
This notion of complexity of a rule set relates to the sense of connectedness of a set of rules, through shared variables. We already defined that a variable *occurs* in a validation rule when altering that variable can alter the outcome of the rule. We say that two rules *share* a variable when that variable occurs in both rules. Two rule sets V and W are *independent* when there is no $v \in V$ and $w \in W$ that share a variable.

This idea of independence is of importance, especially when validation rules are used as input for algorithms for redundancy or feasibility detection, data correction, or error localization. The time (and space) complexity of such algorithms can be exponential in the number of rules. If for each rule, the variables to which they pertain can be established, it is not difficult to split a rule set in independent groups. Software to do so is also available, for example in the R package *validate* for data validation (Van der Loo and de Jonge, 2015) and *editrules* (De Jonge and Van der Loo, 2011).

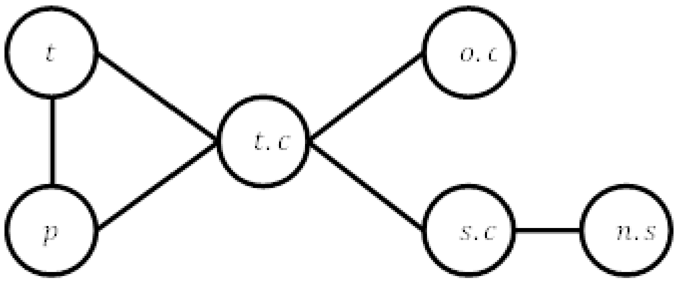
The dependency between rules or variables can be depicted in by two graphs which are in a sense dual to each other. As an example, consider the variables *total costs (t.c), total revenue (t.r), profit (p), staff costs (s.c), other costs (o.c), number of staff (n.s)*. We use the following rule set.

Name	**Rule**
v_1	$t.c + t = p$
v_2	$t.c \geq 0$
v_3	$t \geq 0$
v_4	$s.c + o.c = t.c$
v_5	if $n.s > 0$ then $s.c > 0$

The interdependency between rules and variables can be depicted in a graph (see also Hooker, 2000). Below in the first graph, two rules are connected when they share at least one variable. For example, v_1 and v_4 are connected since they share the variable $t.c$. This graph shows that for example, if we try to solve a violation of rule v_4, then we may affect the status of v_5, v_1 and v_2.



In the second graph, two variables are connected when they occur together in at least one rule. For example, the variable $t.c$ occurs together with $o.c$ in rule v_4. This graph indicates that if we change the value of $o.c$, we may have to change the value of $t.c$ to compensate for violating one or more of the rules connecting them. In turn this may imply that t, p, and $s.c$ must be altered, and so on. It is also clear that variable $t.c$ is in some sense `central' to the graph. If we were to fill in a fixed value for this variable in all rules, then the variables t and p would be disconnected from the rest of the variables, and they may be adapted without regarding $o.c$ (which is also disconnected) or the pair $s.c$ and $n.s$.



Summarizing, the graph representation of a set of rules can give some insight into the cohesive structure of a rule and variable set. Software that can automatically produce such graphs includes the R packages *editrules* mentioned before and *validate.viz* (Van der Loo and De Jonge, 2015). It is possible that descriptive from graph theory, such as connectivity, graph diameter, and measures of centrality provide some quantitative insight into the cohesive structure and relative importance of variables or rules. However, no investigations in that direction are currently known to us.

11 Metrics for a data validation procedure

(Nadežda Fursova, Jūratė Petrauskiene)

Once a validation process has been designed for a given dataset, it needs to be tested in order to assess its suitability for the observed data and to find the best set of techniques and/or parameters able to optimize its performance. The testing aims at evaluating the performance of the validation rules in terms of efficacy (ability to achieve the objectives) and efficiency. Based on the results from testing, some of the design decisions and/or parameters could be revisited to optimize quality. Two types of analysis can be performed: in a first evaluation only observed data are used, in a second type of evaluation both observed and reference data (true, or synthetic) are used.

11.1 Indicators on validation rule sets derived from observed data

Investigation based only on observed data may provide useful insight for tuning the validation rules. Indicators taking into account observed data and treated data, i.e. edited data, are not included in this section. In fact, our aim is to evaluate the rule introduced in a data validation procedure, and according to the definition, the editing step is not a part of the process. Indicators based on the comparison of observed and treated data are useful to assess the efficacy of an editing and imputation procedure. This topic will be addressed in section 11.2. See also Edimbus (2007) and references therein.

The indicators based only on the observed data exploit only information related to the application of rules to data.

They can be calculated overall or per variable/observation, but as stated at the beginning of this document, they refer only to in-record rules. The extension to inter-record rules is an interesting topic that deserves further studies.

In the following list, some examples for validating numerical and logical variables are reported:

1. Number of failed records
2. Minimum number of variables to be changed in order to make records pass the given set of rules (cardinality of solution)
3. Counts of records that passed, missed and failed for each rule
4. Distribution of records that passed missed and failed *k* rules
5. Counts of rules applications of status pass, miss, fail
6. Counts of records of status pass, miss or fail for which field *j* contributed to the overall record status
7. Ratio of missing records against failed record counts (NAs/number of_failed_records) – a measure of (non-)responsiveness against erroneous records.
8. Difference (or percent) of 1-6 indicators between current and previous period observed datasets.

In the following two examples in the given context are reported.

Example 1

Quality indicators (metrics) on data validation that could be computed separately, e.g. for each statistical survey derived from observed data based on questionnaires:

- The number and share (%) of statistical questionnaires validated due to respondent or data entry mistakes compared to the total number of questionnaires (*a questionnaire is considered as erroneous if at least one fatal edit rule has been unsatisfied*).
- The number and share (%) of statistical questionnaires validated due to respondent mistakes compared to the total number of questionnaires.
- The number and share (%) of statistical questionnaires validated by the specialists compared to the total number of questionnaires.
- The number and share (%) of values validated by the specialists divisions compared to the total number of entered values.

A discussion on the use in practice of those indicators should be included. For instance, a high number of failures of a validation rule may suggest either the presence of a systematic error or the inappropriateness of the validation rule.

Other remarkable examples of indicators developed for measuring the impact of validation rules on observed data can be found in BANFF, a processing system for editing and imputation developed at Statistics Canada

Example 2

<u>Description of the Method of the Banff System (see Banff 2008):</u>

For a validation rule set of *m* positivity validation rules and *n* user-specified validation rules with no redundant validation rules, a total of *m + n + 1* status codes is assigned to each record. Redundant validation rules should be removed from the validation rules group by this stage, but if they are still included, the number of status codes to be generated will be reduced because redundant validation rules do not appear in the tables. The following procedure is performed independently for each data record.

- One status code is assigned to the data record for each validation rule, including the positivity rules. There are *m + n* of these codes in total. The status is:
    - PASS, if the record passes the validation rule,
    - MISS, if the record has one or more missing fields involved in the validation rule, or
    - FAIL, if the record fails the validation rule because of one or more non-missing values.
- An overall record status is derived from the *m + n* status codes, which have been assigned to the record based on the results of the application of the individual validation rules to that record. The status is:
    - PASS, if each validation rule status is PASS, i.e., if the record has passed all the validation rules in the group,

- MISS, if one or more validation rule status is MISS and no validation rule status is FAIL, i.e., each validation rule which the record did not pass involved missing fields, or
- FAIL, if one or more validation rule status is FAIL, i.e., if the record failed validation rule because of non-missing values and possibly had failures due to missing values.

<u>Example of Creation of Validation Rule Status Codes</u>

Consider the following group of validation rules and respondent records.
Positivity validation rules: x1>=0 (1); x2>=0 (2); x3>=0 (3);
User validation rules: x1+1>=x2 (4); x1<=5 (5); x2>=x3 (6); x1+x2+x3<=9 (7);
There are three positivity validation rules and four user-specified validation rules, so there are 3 + 4 + 1 = 8 status codes to be assigned to each record. These codes are marked as Validation Rule Status (1) to (7) and Overall Status in the following table.

	Respondent Records			**Validation Rule Status**							**Overall Status**
	x1	**x2**	**x3**	**(1)**	**(2)**	**(3)**	**(4)**	**(5)**	**(6)**	**(7)**	
record 1	**4**	**3**	**2**	**P**	**P**	**P**	**P**	**P**	**P**	**P**	**P**
record 2	**4**	**3**	**missing**	**P**	**P**	**M**	**P**	**P**	**M**	**M**	**M**
record 3	**6**	**3**	**2**	**P**	**P**	**P**	**P**	**F**	**P**	**F**	**F**
record 4	**6**	**3**	**missing**	**P**	**P**	**M**	**P**	**F**	**M**	**M**	**F**

Record 1 passes the validation rules and therefore has a pass status for all seven validation rules and for overall status. Record 2 passes all the validation rules except those involving the variable x3, which is missing. Therefore validation rules (1), (2), (4) and (5) have PASS status and validation rules (3), (6) and (7) have MISS status. The overall status for record 2 is also MISS. Record 3 has no missing values, but fails validation rules (5) and (7). All validation rules have a PASS status except validation rules (5) and (7) which have FAIL status, so the overall status is FAIL. Record 4 fails validation rule (5) due to "incorrect" non-missing values and fails other validation rules due to missing values. Validation rules (1), (2) and (4) have PASS status, Validation rule (5) has FAIL status and validation rules (3), (6) and (7) have MISS status. The overall status of record 4 is FAIL because FAIL takes precedence over the MISS status code when deriving the overall record status code.

<u>Example of the Tables</u>

A short description of the five Validation rule Summary Statistics Tables is given here, along with the tables which would be produced for the validation rules and data records used in the above example of the creation of status codes. The user should keep in mind that Tables 1-1 and 1-2 are based on the validation rule status codes while Table 1-3 is based on the overall record status codes. Table 2-1 is based on validation rule status codes which have been assigned to each field in the record according to rules described below. Table 2-2 is based on the record status codes, which have been assigned to each field of the record according to rules described below.

TABLE 1-1. COUNTS OF RECORDS THAT PASSED, MISSED AND FAILED FOR EACH VALIDATION RULE

VALIDATION RULE	RECORDS PASSED	RECORDS MISSED	RECORDS FAILED
(1)	4	0	0
(2)	4	0	0
(3)	2	2	0
(4)	4	0	0
(5)	2	0	2
(6)	2	2	0
(7)	1	2	1

Table 1-1 gives the number of records, which passed, missed or failed each validation rule. The table is based on counts of the PASS, MISS and FAIL validation rule status codes, which occur for each individual validation rule. For any given validation rule, each record is counted as passed, missed or failed, so the total in each row should be the same as the total number of records.
For example, for validation rule (1), the positivity validation rules on the variable x1, all four records passed so all four are counted in the "RECORDS PASSED" column and no records are counted in the "RECORDS MISSED" or "RECORDS FAILED" columns. On the other hand, only record 1 passed validation rule (7), while records 2 and 4 had validation rule status MISS and record 3 failed. The last line of Table 1-1 counts one under "RECORDS PASSED", two under "RECORDS MISSED" and one under "RECORDS FAILED".

From Table 1-1, the user may determine if records tend to fail or miss some validation rules more often than others. This could indicate that the validation rules are too restrictive or that the data are of poor quality (validation rule focus).

TABLE 1-2. DISTRIBUTION OF RECORDS THAT PASSED, MISSED AND FAILED *K* VALIDATION RULES

NUMBER OF VALIDATION RULES (K)	RECORDS PASSED	RECORDS MISSED	RECORDS FAILED
0	0	2	2
1	0	0	1
2	0	0	1
3	1	2	0
4	1	0	0
5	1	0	0
6	0	0	0
7	1	0	0
TOTAL RECORDS	4	4	4

Table 1-2 gives the distribution of records, which pass, miss or fail a given number of validation rules. The table is based on the validation rules status codes and gives the number of records for

which each status code occurs zero times, once, twice, etc. Each record is counted once in each column (in the row corresponding to the number of validation rules it failed, missed or passed) so that the total of each column is equal to the number of records.
In this example, the first row of Table 1-2 indicates that there were no records, which passed zero validation rules, while two records missed zero validation rules and two records failed zero validation rules. Moving down the table to the row with "3" in the left-hand column, one can see that one record passed three validation rules, two records missed three validation rules and no records failed three validation rules. The line beginning with "7" says that one record passed all seven validation rules, no records missed seven validation rules and no records failed seven validation rules.

From Table 1-2 the user may determine, if the data being analysed consist of a moderate number of records which pass all validation rules and a moderate number of records which fail a few validation rules, or if the data consist of a large group of records which pass all the validation rules and a much smaller group of records which fail most or all of the validation rules (data and error distribution focus).

TABLE 1-3. OVERALL COUNTS OF RECORDS THAT PASSED, MISSED AND FAILED

RECORDS PASSED	RECORDS MISSED	RECORDS FAILED	TOTAL
1	1	2	4

Table 1-3 gives the number of records with PASS, MISS or FAIL overall record status. Each record is counted once so the single row of this table should add to the total number of records.
In the example being used here, one record had a PASS record status code, one had a MISS and two had record status codes of FAIL. Note that records which have a FAIL record status may have one or more MISS validation rule status codes as well as at least one FAIL validation rule status code.
The sum of the RECORDS MISSED and RECORDS FAILED cells gives a preview of the number of failures to expect when the Error Localization procedure is run.

TABLE 2-1. COUNTS OF VALIDATION RULE APPLICATIONS OF STATUS PASS, MISS OR FAIL THAT INVOLVE EACH FIELD

FIELD	VALIDATION RULE APPLIC PASSED	VALIDATION RULE APPLIC MISSED	VALIDATION RULE APPLIC FAILED	VALIDATION RULE APPLIC NOT INVOLVED	NUMBER OF VALIDATION RULES INVOLVED
x1	11	2	3	12	4
x2	11	4	1	12	4
x3	5	6	1	16	3

Table 2-1 gives the number of times each variable was involved in a validation rule, which passed, missed or
failed. For this tabulation the PASS, MISS and FAIL validation rule status codes, which were generated for a particular record and validation rule are assigned to each field which is involved in the validation rule. A NOT INVOLVED code is assigned if the variable did not appear in the validation rule. Then the occurrences of these codes are summed for each variable. For a given row of the table (i.e., a given variable), each record is counted with each validation rule, so the total of all cells except the last is the number of records times the number of validation rules. The last column gives a count of the number of validation rule in which the particular variable is involved.
In the example being used here, x1 was involved in a PASS validation rule status code 11 times - record 1 in validation rules (1), (4), (5) and (7), record 2 in validation rules (1), (4) and (5), record 3 in validation rules (1) and (4) and record 4 in validation rules (1) and (4). The variable x1 was involved in a MISS validation rule status code twice – validation rule (7) with records 2 and 4 - and was involved in a FAIL validation rule status code three times - record 3 with validation rules (5) and (7) and record 4 with validation rule (5). There are three validation rules in which x1 was not involved, giving a count of 3 validation rules x4 records = 12 validation rule applications which did not involve x1. The last column gives the number of validation rules which did involve x1. There are four records in this example, so there should be 4 x 4 = 16 validation rule applications which involved x1, and this should always be equal to the sum of the first three columns.

From Table 2-1, the user may judge if some variables tend to be included in validation rules, which fail or miss more often than others.

TABLE 2-2. COUNTS OF RECORDS OF STATUS PASS, MISS, OR FAIL FOR WHICH FIELD *j* CONTRIBUTED TO THE OVERALL RECORD STATUS

FIELD	RECORDS PASSED	RECORDS MISSED	RECORDS FAILED	RECORDS NOT APPL
x1	1	1	2	0
x2	1	1	1	1
x3	1	1	1	1

Table 2-2 gives the number of times each variable contributed to the overall record status. For this tabulation, the PASS, MISS and FAIL overall record status codes are assigned to each field according to the following rules.

- If the overall record status is PASS then the record passed all validation rules, all fields must be good and all fields are assigned a PASS status for the purposes of this table.
- If the overall record status is MISS, then MISS and PASS are the only possible values for validation rule status. The fields involved in the validation rule set

with status MISS are assigned a MISS and the fields not involved in any validation rule set with status MISS are assigned a NOT APPLICABLE status.

- If the overall record status is FAIL, then at least one validation rule status must be FAIL and one or more validation rule status may be MISS. The variables involved in validation rule set with FAIL validation rule status are assigned a FAIL and the variables not involved in any validation rule with a FAIL validation rule status are assigned a NOT APPLICABLE status.

Each record is counted once for each field, so the row total is equal to the number of records. For example, since record 1 passed all validation rules, it is counted under "RECORDS PASSED" for all variables. Record 2 has a MISS record status and all its fields are involved in at least one validation rule which had a MISS status, so record 2 is counted in the "RECORDS MISSED" column for all variables, even though only x3 is actually missing on the respondent data record. Record 3 has a FAIL record status and each of its fields is involved in at least one validation rule which has a FAIL validation rule status. Therefore, record 3 is counted in the "RECORDS FAILED" column for all variables. Record 4 also has a FAIL record status. The variable x1 is involved in validation rule (5) which has a FAIL validation rule status, so it is counted in the "RECORDS FAILED" column for x1. The remaining two fields are not involved in any validation rules which have a FAIL validation rule status, so record 4 is counted in the "RECORDS NOT APPLICABLE" column for the rows referring to x2 and x3.

From Table 2-2, the user may determine if some variables tend to contribute to the MISS or FAIL record status more often than others. It should be noted that this table counts all fields involved in a FAIL or MISS validation rule as contributing to the overall record FAIL or MISS status. The Error Localization procedure will likely identify a subset of these fields as requiring imputation and allow the other fields involved in the validation rule to remain as they are.

The first major use is to assess the suitability of individual validation rules or of a group of validation rules
by observing the failure rates which occur when the validation rules are applied to a set of data records. A high failure rate for one particular validation rule might indicate that the user should modify that validation rule by changing the constants or by altering the form of the validation rule. If this is done, it would then be necessary to change the validation rule, and resubmit the new validation rules group to validation rule analysis. It is also possible that anomalies observed in the Validation rule Summary Statistics Tables do not represent problems with the group of validation rules itself. For example, if one variable is involved in a large percentage of the validation rule failures, it might indicate that a review should be done of the questionnaire definitions or of the collection procedures used in the field. Of course, it might not be possible to do this during the later stages of a survey.

An interesting example of indicators built on observed data is reported in the following

Example 3.

If we perform a number, say *N*, validations on a data set, we obtain *N* values. Here, we assume that each value is either 0 (invalid), 1 (valid) or NA (missing). An outcome is missing when one of the variables involved in the validation is missing. In general, if we have *n* in-record rules (*ssss* or *sssm* in our formal classification) and *m* records, we have *N = nm*.
Now, suppose we have a data set A, perform some process step on it, giving us data set B'. The following table gives a hierarchical partition of validation results going from A to B (van de Broek et al., 2014).

Total number of rules checked					
Total verifiable				Total not verifiable	
Failed		satisfied			
Still failed	Extra failed	Still satisfied	Extra satisfied	Stil not verifiable	Extra verifiable

The information reported in the table are essentially that described in Banff, however this way of reporting has the property that there is partition of the initial 'total number of rules checked', this makes the interpretation of the results easier.

The three examples above represent different ways of calculating the impact of a set of validation rules on a set of observed data. There is a considerable amount of overlap and the indicators do differ mainly in the form of representation. No standard form has been developed yet and this is a clear indicator that this field of research not yet very well understood. Some of the concepts developed earlier ("properties of validation rules") like redundancy and complexity need to be better understood and integrated in the approaches used here to improve and harmonize the methodology of validation metrics in future. The Indicators introduced in this section may be used in the testing phase to optimize the parameters of the rules, and in the execution phase to monitor the process and results. At the moment they are of more explorative nature and should be analysed in more detail.

The indicators used in this step can be interpreted as quality measures and used as standard reporting devices, which are routinely calculated.

Reporting on the validation process should inform the user about the main aspects of the validation process and data quality. This kind of documentation is the minimal documentation and should be added to the general description of the survey or to the main statistical publication.

11.2 Indicators on validation rule sets derived from observed and reference data

(Marco Di Zio, Ugo Guarnera)

While the assessment of the *impact* of a data validation procedure requires only the availability of the observed data, measures of *efficiency* and *efficacy* should in principle involve comparison between "true" data and observed data.
Since true data are usually not available (otherwise no validation procedure would be necessary), some set of reference data have to be used in place of the true data. A possible approach is to revise accurately a small subset of the observed data and to use the resulting cleaned dataset as reference dataset.
Two other approaches are illustrated in Subsections 11.2.2 and 11.2.3. In Subsection 11.2.1 some classical indicators are introduced for the ideal case when true data are available.

11.2.1 True values as reference data

A data validation procedure can be viewed, at least at record level, as a classification process, according to which records are classified as erroneous or not. Thus, we could adopt some usual measures for classification procedures like for instance confusion matrix or ROC curve. However, while the validation procedure often refers to complex data objects (typically records), errors are generally referred to single items. Thus, evaluating efficacy of a validation procedure should also involve the analysis of changes of single values resulting from the procedure. In order words, the metrics should be extended to the classification of single variables as erroneous or not. Unfortunately, this makes the evaluation task somewhat ambiguous. In fact, apart the special case of rules involving only one variable (*domain rule*s), generally application of a validation rule (edit) does not imply any localisation step, that is, any decision about which variable(s) is responsible for the edit failure. This is typically the objective of a separate localisation procedure, often based on some mathematical algorithm (e.g. Fellegi-Holt methodology). Thus, assessing the capability of correctly classifying single values as erroneous or not would actually result in evaluating the whole data editing procedure including both data validation and error localisation. Furthermore, the quality of the final data to be used for statistical purposes also depends on the method(s) used to replace values flagged as erroneous with plausible ones (imputation). Thus, efficacy indicators related to the accuracy of the final data would also include evaluation of the imputation methodology.
From the last observations, it follows that it is difficult to assess the efficacy and efficiency of a data validation procedure independently from the evaluation of other phases of the editing and imputation process. Nevertheless, it is always possible to evaluate a validation procedure in terms of its capability of correctly identifying records containing *at least* one error ("erroneous records"). Of course, this approach does not distinguish records containing a different number of errors.
Using symbols C and E to denote absence or presence of error respectively, we can define in the usual manner the "confusion table" as the 2X2 cross table of the true vs predicted values for the error indicator:

Table 8. Confusion Matrix for classification of erroneous record: E = positive (erroneous), C = negative (correct)

		Predicted	
		C	E
Actual	C	TN: True negative	FP: False positive (Type II error)
	E	FN: False negative (Type I error)	TP: True positive

Common summarising statistics based on the counts in the cells of Table 8 are:

Precision or *Positive Predictive Value* (PPV) = TP / (TP+FP)
Sensitivity or *True Positive Rate* (TPR)= TP / (TP+FN)
Accuracy (ACC) = (TP+TN) / (TP+TN+FN+FP)
Specificity or True Negative Rate (TNR) = TN / (TN+FP)

Cohen's Kappa statistics K is also frequently used to assess the reliability of a classification procedure.
It is defined as:

$$K = (p_o - p_e)/(1-p_e),$$

where p_o = ACC = (TP+TN)/U is the observed rate of agreement between actual and predicted values, and $p_e = [TP+TN+FN+FP]^{-2} [(TN+FN) \times (TN+FP) + (TP+FN) \times (TP+FP)]$ is the rate of agreement than one would obtain if predictions and actual values were independent one of each other, estimated on the observed data. Thus, the statistic measures the "excess" of agreement between actual and predicted value with respect to the situation where agreement is achieved by chance.

These statistics can be used as indicators to assess some characteristics of the validation procedure. For instance, TPR can be viewed as an effectiveness indicator (it attains its maximum value 1 if all errors are detected regardless of the number of records erroneously classified), while PPV can be interpreted as an efficiency indicator (it is low if the number of false positive is high).

Remark 1
Note that the above setup is independent of the specific method used to classify the records as erroneous or not. Thus, it can refer to one single validation rule as well as to some set of different rules. In particular, the confusion matrix can be used to assess the effectiveness/efficiency of the whole data validation strategy.

Remark 2

When the confusion matrix refers to a validation rule which depends on some discriminant threshold (e.g., as in case of a ratio edit), one can obtain the Receiver Characteristic Curve (ROC) by plotting the TPR against the *False Positive Rate* (FPR=1-TNR) for different values of the threshold. The ROC curve can be analysed in the design phase in order to obtain "optimal" validation rules.

Remark 3

The above analyses is limited to rules that when failed, lead to the record causing the failure. This excludes rules that have a cross-record validation character, for instance violating the rule

$$0.9 < \text{mean}(y_t)/\text{mean}(y_{(t-1)}) < 1.1$$

does not single out any record as erroneous and therefore cannot be treated with the measures mentioned.

Number of errors at item level and severity

The indicators so far introduced are appropriate for evaluating classification procedures with binary outcomes, where the objects to be classified are records of a dataset, and the predicted dichotomous variable is "presence of at least an error" in each record. As already mentioned, this metrics does not take into account differences in the number of erroneous values within the records. Moreover, in case of quantitative variables, it could be desirable to have metrics capable of distinguishing large and small discrepancies between true and observed values (i.e., measures of "severity" of the errors). As noticed above the number of incorrect values in a single record, and in case of numerical variables the error magnitude, can only be estimated if some localization procedure is used. Localizing erroneous variables however, is not part of the validation phase, whose only outcome is the split of the data in validated and not validated records. A possible approach to overcome these difficulties is to assume (ideally) that "perfect" procedures for error localization and imputation are available, so that once a record is flagged as "not valid", the true value can be restored with certainty. In this manner, we could evaluate the validation procedure by comparing the observed data with the true data. Classical indicators for the evaluation of editing procedures can be used. Below, some examples are provided.

Indicators for both categorical and numerical variables

Single variable (Y):
Indicators based on the confusion matrix (PPV, ACC, TNR,..), where the categories for the binary variables are "actual presence of error in variable Y" (rows) and "flag associated with the output of the validation procedure" (columns). This confusion matrix corresponds to considering the variable Y incorrect in each not validated record. For instance, if Y is the variable AGE, the counts

in the corresponding confusion matrix refer to number of times that records are validated or not and variable AGE is correctly reported or not.

Overall:
The same indicators as in the previous case but now applied to *all the data items* simultaneously. The confusion matrix corresponds to considering all variables of not validated records as erroneous.

Severity indicators for numerical variables

In case of numerical variables it would be desirable to have some indicator measuring the *importance* of the errors that cause records to be rejected by the validation procedure. When true data are available, natural way to define "importance" is in terms of absolute difference between observed and true value. Again, in order to have meaningful metrics, we should consider *all* the variables in not validated records as erroneous and derive indicators by comparing the dataset "corrected" dataset with the "true" dataset. Here the corrected dataset is to be intended as the one obtained by replacing each non validated record with the corresponding record in the true dataset. Thus, if for the i*th* unit, Y_i^*, Y_i^0,and Y_i^c denote true, observed and corrected value for the variable *Y* respectively, Y_i^c will be Y_i^0 or Y_i^* depending on whether the *i*th record has been validated or not respectively. In this setting, classical indicators for the evaluation of editing and imputation procedures can be used. For instance, for a dataset of size *n,* possible severity indicators for the variable Y are :

$I_Y = \frac{\sum_{i=1}^{n} | Y_i^c - Y_i^* |}{n}$, or $I_Y^r = \frac{\sum_{i=1}^{n} | Y_i^c - Y_i^* |}{n | Y_i^* |}$ measuring the mean error and the relative mean error respectively remaining in data after the validation procedure (zero value for these indicators correspond to optimal procedures). An overall indicator can be obtained through some averaging procedure of the indicators that refer to single variables. In order to assess also the efficiency of the procedure, indicators taking into account the "cost" associated to rejection of records would be appropriate. For instance, in case of equal cost for all the record, an efficiency indicator for the variable Y could be:

$I_Y^e = \frac{\sum_{i=1}^{n} | Y_i^c - Y_i^0 |}{m \sum_{i=1}^{n} | Y_i^o - Y_i^* |}$, where *m* is the number of not validated records. This indicator represents the *(relative) "amount" of error removed per rejected record*.

The previous metrics evaluate the micro-accuracy of the validation procedure. In fact they are defined in terms of amount of error removed at micro level. In an estimation-oriented perspective, it can be also useful to introduce indicators that are specific of the particular estimate(s) of interest. In this case, what is to be measured is the impact of the validation procedure on the accuracy of the target estimates instead of the amount of errors removed from data. For instance, if the total $T_Y = \sum_{i=1}^{n} Y_i$ is to be estimated, a corresponding indicator could be:

$$I_{T_Y} = \frac{\sum_{i=1}^{n} Y_i^c - \sum_{i=1}^{n} Y_i^*}{|\sum_{i=1}^{n} Y_i^*|}$$

measuring the (relative) distance between the estimate based on true data and the corresponding estimate based on the data after validation. The same approach can also be adopted for distributional characteristics of the population instead of finite population quantities.

11.2.2 Plausible data as reference data

When true data are not available, even for a small subsample of the data, a common approach to analyse a validation procedure is to use the indicators discussed in the previous section by applying them to “plausible” data in place of true data. Starting from a set of observed data, a set of plausible data is generally obtained by flagging all the not acceptable values (localization) and replacing them with plausible values (imputation). Thus, the metrics based on comparing the observed data with the plausible data depend on the method used for the error localization and imputation that is the method used to obtain the plausible data. However, once a set of plausible data is available, one can forget the specific method used to obtain it and treat them as if they were error-free. For a given fixed reference dataset it is possible to compare different sets of validation rules by applying the metrics introduced in subsection 11.2.1.

As for the methodologies to be adopted to obtain plausible data, some descriptions can be found in (Edimbus, 2007). Here we only mention the Fellegi-Holt approach for error localization (and imputation) of categorical variables (Fellegi and Holt, 1976), and its extension for error localization of numerical variables. Extensive discussion on the imputation methods can be found in the literature on incomplete data (see, for example Little-Rubin 2002).

11.2.3 Simulation approach

Simulation is another strategy for the assessment of a validation procedure based on comparing observed data with reference data. In practice, one tries to artificially reproduce a situation, which is likely to be similar in terms of true data distribution and error mechanism. Specifically, there are two elements for the simulation approach:

1) a set of true reference data and
2) some procedure for perturbing data by introducing errors.

For the first element, it is important that the reference data are “similar” in terms of distributional characteristics to the real data that have to be validated. To this aim, an explicit model could be fitted on a set of real data and new synthetic data could be obtained from random drawing from the (robustly) estimated model. Alternatively, a set of accurately edited data can be used as reference true data. For the error simulation, a possible approach is the probabilistic selection of the subset of data to be perturbed and the introduction of “typical” errors in all the selected records. Frequent errors in statistical data are, for instance, swap of consecutive digits, or unity measure errors.

The indicators described in Section 11.2.1 can be computed comparing reference and perturbed data. In order to assess the efficacy and efficiency of a validation procedure, it is convenient to replicate the simulation several times (Monte Carlo iterations) and obtain the evaluation by averaging over the resulting values of the indicators. The general scheme for deriving metrics in a simulation framework can be summarized as in the following Monte Carlo (MC) scheme based on a ***n.sim*** iterations:

```
Repeat n.sim times:
 {
  1) Draw a random sample D* of data to be considered as error-free
  2) Simulate a contaminated dataset D by introducing errors in D*
  3) Apply the validation procedure to D
  4) Simulate a “perfect” localization and imputation procedure, i.e., replacing non-validated
     values in D with the corresponding true values from D*. Let $D^c$ the corresponding cleaned
     dataset
  5) Compute indicators based on datasets $D^c$, D, D* (see Section 11.2.1)
 }
Compute global metrics by taking averages of the elementary indicators over the nsim MC
iterations.
```

The simulation approach allows to analyze some statistical properties of the validation procedure. The underlining random elements are the probability distribution generating the true data and the error mechanism. In the latter case, randomness refers both to the selection of the particular set of values that are contaminated and to the realization of the specific erroneous values.

In these framework, some metrics can be related to statistical properties of the validation procedure such as bias and variance. For instance, if $I_{T_Y}^k$ denotes the value of the elementary indicator I_{T_Y} at the k*th* simulation, one can relate the overall indicators

$$\frac{\sum_{k=1}^{n.sim} I_{T_Y}^k}{n.sim}, \quad \sqrt{\frac{(\sum_{k=1}^{n.sim} I_{T_Y}^k)^2}{n.sim}}$$

to the bias and the mean square error, respectively, of the estimator due to the validation procedure.

12 Assessment of validation rules

(Lucas Quensel-von Kalben)

In the second part of this document, beginning with section 9., several approaches to measure efficiency and effectiveness of validation rules and rule sets have been presented and critically discussed. In section 10. key concepts (properties) of validation rule sets have been identified that are usually intuitively interpreted among statisticians but not really made explicitly open for discussion. Completeness, redundancy, feasibility/coherence and complexity are explored in detail and properly defined. The concepts are at times rather difficult to grip precisely and have sometimes to be defined in a manner that is not completely in accordance with our primal perception. While completeness and complexity seem to be easy to define, they remain more vague concepts. Redundancy and feasibility are instead of a more practical value.

The methods invoked to analyze these main properties of validation rule(s) (sets) can be split into formal and non-formal (human intuition) methods. Questions of completeness (and the related concepts of in-completeness and over-completeness) and complexity are at the moment better to be evaluated (rather than tested) by expert knowledge and can be supported by network graphs of the usage of variables and rules. More formal methods can be used to address questions of redundancy and feasibility. Both properties are prerequisites of more elaborate methods (mentioned later). Interestingly a complete elimination of all redundancies of a rule set might not be the best solution because of the reduced readability of the optimized rule sets for human actors.

Section 11. provides three examples of how to present the relationship between validation rule sets and observed data. Especially interesting are the tabular summaries provided by the BANFF system and the resulting hints for the statistician about the quality of the rules and/or the data (and/or ambiguous questionnaires). The list of key indicators (example 1) and the table of example 3 are variants of the BANFF tables and more compact in character. Probably starting with compact measures first and later going to BANFF-like tables is the most promising strategy. Some kind of statistical “error report” are probably already part of most offices common procedure. However, a standard form and systematic evaluation of these reports is currently not established.

The comparison between validation rules, observed data and "real" data is particularly helpful from an efficiency point of view. The major obstacle is – of course – the availability of "real" data. A small sample of thoroughly cleaned observed data or a set of imputed data could be a used instead. The comparison to cleaned data of older surveys is a standard procedure in most offices already. Confusion table and ROC-curves, severity indicators and simulation approaches explore the interrelationship between the three elements to different data types and scope. The first two methods address the optimization problem of validation rules being either too strict or too loose (the pitfall of increasing type 1 or type 2 errors). They can be used directly to identify more optimal validation strategies. The other methods discussed add some detail to the more general methods like the severity of errors or the question of multiple errors in one record.

The methods presented in this section are similar to statistical methods used in predictive analytics and machine learning. This hints to kinds of solutions that might be used in future to optimize a set of validation rules.

The methodological tool-kit could be applied in different phases of the validation life cycle and support an overarching validation strategy. When a new survey is designed the methods discussed in the second part of the document to explore the properties of a new set of validation rules and optimize the sets can be used (network graphs of validation rules and peer reviews). Later, during data collection and the processual and analytical phases, the methods described for comparing validation rules and observed data can be used. After a particular survey, in the evaluation phase, the comparisons between observed and real data could be applied. Clearly, these methods can be used in other phases as well. A pretest with a sub-sample can already provide some valuable insight to the efficiency of a validation rule set.

Several limitations have been mentioned explicitly in the sections before. The authors had to restrict their scope to the relatively simple case of in-record validation and left out for the present work more complex validation rules (the higher levels of our formal typology of part one of the handbook). This restriction might be neglected in practical terms, assuming the majority of rules being used is of the more simple type. The efficiency of a validation strategy should be output-oriented. Therefore, a comparison of the impact of different validation rule sets of the final output (mainly macro data) would be useful. And a final restriction concerns the interplay of data quality on the one side (and in the focus of this document) and effort on the other side. With effort, the total of all human input (across all validation related processes of the GSBPM) is meant. Typically, this is measured in time required to identify and localize errors and edit/impute data accordingly. Paradata and process metadata could be used for this purpose. We have not covered this issue yet and leave it for future exploration.

The interrelationship of the methods discussed in this document with methods used in neighboring processes (data editing and imputation) has also been mentioned several times. This is not purely a question of joint effects of these methods regarding data quality but also a question of efficiency. To optimize some processes might influence the efficiency of these other processes.

The very practical question of effectiveness of a given rule set regarding its contribution to data quality at reasonable costs is currently not easy to answer. Most methods are more explorative and presentational in character (heuristic in a way) and need knowledgeable persons for interpretation. Probably this expert knowledge not just for applying the methods but even more for interpreting will be necessary in future as well. Besides further research on the methodological edge, educating statisticians on national and international level in the usage of metrics will be essential to improve the efficiency and effectiveness of validation procedures in the field.

Several times in this document, existing IT-solutions (tools and services) were mentioned that try to tackle some of the issues raised from a methodological perspective. A common IT-infrastructure in the ESS should try to assist the assessment of validation rules by the statistical community by providing strong (and adaptable) solutions. This could be achieved by using existing solutions (BANFF, Validate and others) integrated as CSPA-compliant services within the infrastructure or by building new services that incorporate the advancements in methodology in the coming years.

Appendix A: List of validation rules

Table A.2.3.2: Examples of checking rules for logical validation and consistency

Check	Comment and remarks regarding the level and type of the check (c.f. 2.3.1)
Simple Identity Check	
In a sheep survey : "Milk production" must be equal to "milk disposal"	*Mirror check.* This might be an example for levels 4 or 5, if "production" and "disposal" data come from different data sources or even from different data providers
Year Year = Year (in the filename)	
Population: The total for each population (persons, households, families, dwellings) is checked to be consistent throughout the process.	This might be an example for level 2 or higher, if data on persons, households, families and dwellings are stored in different files and come perhaps form different data sources
Enterprises: An enterprise included in the admin data must be part of the predetermined population (from the Business Register)	This might be an example for level 2 or higher, if "admin data" and "business register" are organized as different data sets, or even managed by different units in the institute.
Number of livestock and animal production survey: Number of animal at the end of reference period == number of animal at the beginning of following reference period.	*Mirror check.* Level 2, if we assume that data from different periods are stored in different files
Simple range check - bounds fixed	
Working hours (monthly): "Hours worked" must be between 0 and 168	
Number of inhabitants for country LU: Total number of inhabitants should be in the range 100,000 – 1,000,000.	

Check	Comment and remarks regarding the level and type of the check (c.f. 2.3.1)
Simple Range check - bounds depending on entries in other fields	
Cereal production: “Organic cereals” must be less or equal to “cereals”	
Price changes: Price changes between -30% and +50%	*Time series check.* Level 2, if we assume that data from different periods are stored in different files
Inhabitants (Micro level): Total inhabitants > = Male inhabitants (on the micro record level)	Level 2, if we assume that data for "total inhabitants" are stored in another file as "male inhabitants"
Inhabitants (Macro level, i.e. after aggregation across all records of the micro data file): Total inhabitants > = Male inhabitants (on the macro record level)	Level 2, if we assume that data for "total inhabitants" are stored in another file as "male inhabitants". “Complex” check, if organized as check on the micro data set.
“Complex” identity check (involving functions on field entries)	
Inhabitants: Total = Male + Female	Level 2, if we assume that data for "total inhabitants" are stored in another file as "male inhabitants"
Employment: “Number of persons engaged” must be equal to the **sum** of “employees” and “self-employed persons”	
Households: Number of members in the household file = persons who are members (not guests) in the persons file	*Cardinality Check* Level 2, if we assume that persons and households data are separate data sets.
Profit/Loss: Profit/Loss from Profit and Loss account should be equal to Profit/Loss from the Balance sheet	Level 2, if we assume "Profit and Loss account" being not the same data set as "Balance sheet"
Social protection expenditure: Total social protection expenditure= Social protection benefits + Administration costs + Other expenditure	

Check	Comment and remarks regarding the level and type of the check (c.f. 2.3.1)
“Complex” range check (involving functions on field entries) - bounds fixed	
Average poultry weight: “weight of poultry” **divided by** “number of poultry” must be between 0.03 and 40	
0.82<V11110(t)/V11110(t-1)<1.22	*Time series check* Level 2, if we assume that data from different periods are stored in different files
Milk: Quantity of produced drinking milk}<={unskimmed milk as raw material}+{skim milk as raw material}	This might be an example for levels 4 or 5, if "production" and "raw material" data come from different data sources or even from different data providers
Milk: (milk quantity/no of dairy cows) is between 3000 and 10000	This might be an example for levels 4 or 5, if "milk quantity" and "no of cows" data come from different data sources or even from different data providers
Data revision: Value_t2 – Value_t1)/Value_t1 <= 0.005, where Value_t2 is revised version of value_t1	*Revision check*
Weight of goods loaded in a origin-region minus the sum of weight of goods in all records having the same region as destination should be with in a small range (i.e. rounding error)	*Mirror check*

Check	Comment and remarks regarding the level and type of the check (c.f. 2.3.1)
“Complex” range check (involving functions on field entries - bounds depending on entries in other fields -	
External services: “Expenses on external services” must be greater or equal to “payment for agency workers” **plus** “telecommunications” **plus** “business trips of company personnel”	
Inhabitants: Female inhabitants = (total inhabitants / 2) +/- 10%[4]	Level 2, if we assume that data for "total inhabitants" are stored in another file as "female inhabitants"
Milk: Fat quantity in processed milk > fat quantity of all milk products	Level 5, if we assume that processed milk data come from a different data provider as milk product data

[4] This check is the same as: 0.4*total inhabitants <= Female inhabitants<=0.6*total inhabitants

13 References

Arora, S. and Barak, B. (2007). *Computational Complexity: a modern approach.* Cambridge University Press. http://theory.cs.princeton.edu/complexity/book.pdf

Banff (2008). Functional description of the Banff system for edit and imputation. Statistics Canada

Chmeiss, A., Krawczyk V., Sais L. (2008). *Redundancy in CSPs*. In: *Proceedings of the 18th European Conference on Artificial Intelligence (ECAI 2008),* IOS Press, Amsterdam

Daalmans, J. (2015) Simplifying constraints. *Discussion paper 2015, Statistics Netherlands In press.*

Daas P., Tennekes M., Zhang L.C. , Hendriks C., Foldal Haugen K., Cerroni F., Di Bella G, Laitila T., Wallgren A. and Wallgren B. (2011) Report on methods preferred for the quality indicators of administrative data sources. *Deliverable 4.2 of the BLUE-ETS project*.

De Jonge, E., Van der Loo M. (2011). R package editrules. http://github.com/data-cleaning/editrules

De Jonge,E., Van der Loo M. (2014). *Error localisation as a mixed integer problem in the editrules package.* Discussion paper 201407, Statistics Netherlands Den Haag/Heerlen.

De Waal T., Quere R. (2002). A fast and simple algorithm for automatic editing of mixed data. *Journal of Official Statistics* 4, 383-402.

De Waal T., Pannekoek J., Scholtus S. (2011). *Handbook of statistical data editing.* John Wiley & Sons, Inc.

De Waal T. (2003) *Processing of unsafe and erroneous data*. PhD thesis, Erasmus University Rotterdam.

EDIMBUS (2007). *Recommended Practices for Editing and Imputation in Cross-sectional Business Surveys*. http://epp.eurostat.ec.europa.eu/portal/page/portal/quality/documents/RPM_EDIMBUS.pdf.

ESS QAF. Quality Assurance Framework of the European Statistical System, Version 1.2 http://ec.europa.eu/eurostat/web/quality

Felfernig A., Hotz L., Bagley C., Tiihonen J., (2014). *Knowledged-based configuration: from research to business cases*. Morgan Kaufmann Publishers Inc., San Francisco.

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

Giessing S., Walsdorfer K. (2015). The ValiDat Foundation Project: Survey on the Different Approaches to Validation Applied Within the ESS, *UNECE Work Session on Statistical Data Editing, Budapest, 2015*

GSBPM version 5.0, 2013. http://www1.unece.org/stat/platform/display/GSBPM/GSBPM+v5.0.

GSIM Version 1.1, December 2013 http://www1.unece.org/stat/platform/display/metis/Generic+Statistical+Information+Model

Hooker, J. (2000) *Logic based methods for optimization: combining optimization and constraint satisfaction.* John Wiley & Sons.

Pannekoek J., Scholtus S., van der Loo M. (2013). Automated and Manual Data Editing: A View on Process Design and Methodology. *Journal of official statistics*, Vol 29, No 4, pp. 511-537

Paulraj S., Sumathi P. (2010*). A comparative study of redundant constraints identification methods in linear programming problems. Mathematical problems in engineering Article* ID 723402.

Simon A., (2013a) Definition of validation levels and other related concepts v01307*. Working document*. Available at https://webgate.ec.europa.eu/fpfis/mwikis/essvalidserv/images/3/30/Eurostat_-_definition_validation_levels_and_other_related_concepts_v01307.doc

Simon A., (2013b*)* Exhaustive and detailed typology of validation rules v01306. *Working document* available at https://webgate.ec.europa.eu/fpfis/mwikis/essvalidserv/images/3/30/Eurostat_-_definition_validation_levels_and_other_related_concepts_v01307.doc

UNECE 2013 Glossary of terms on statistical data editing http://www1.unece.org/stat/platform/display/kbase/Glossary

Van den Broek, B., van der Loo, M., Pannekoek, J., (2014) Kwaliteitsmaten voor het datacorrectieproces, Statistics Netherlands,

Van der Loo M. (2015). A formal typology of data validation functions, *UNECE Work Session on Statistical Data Editing, Budapest, 2015*

Van der Loo M., De Jonge E. (2015). R package lintools. http://github.com/data-cleaning/lintools

Van der Loo M., De Jonge E. (2015). R package validate. http://github.com/data-cleaing/validate

Van der Loo M., De Jonge E. (2015). R package validate.viz. http://github.com/data-cleaing/validate.viz

van der Loo M., Pannekoek J. (2014). Towards generic analyses of data validation functions, *UNECE Work Session on Statistical Data Editing, Paris, 2014*

Williams, H.P. (1986*). Fourier's method of linear programming and its dual. The American Mathematical Monthly* 93, 681-695.

Zhang L.C. and Pritchard A. (2013). The integration of survey and administrative data. *In 3rd European Establishment Statistics Network, Nuremburg, Germany*.