CWI Tracts

Centrum voor Wiskunde en Informatica
Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

The CWI is a research institute of the Stichting Mathematisch Centrum, which was founded on February 11, 1946, as a nonprofit institution aiming at the promotion of mathematics, computer science, and their applications. It is sponsored by the Dutch Government through the Netherlands Organization for the Advancement of Pure Research (Z.W.O.).

CWI Tract 54

Computational aspects of survey data processing

L.C.R.J. Willenborg

Centrum voor Wiskunde en Informatica
Centre for Mathematics and Computer Science

1980 Mathematics Subject Classification: 68M50, 68P40, 69H21, 69G12, 69F20, 62D05, 62P99.
ISBN 90 6196 356 7
NUGI-code: 811

PREFACE

This monograph was originally written as a Ph.D. thesis, which was submitted to Tilburg University. As such it is the result of a large-scale research project carried out at the Netherlands Central Bureau of Statistics (CBS) with the aim of understanding and improving (in particular: computerizing as much as possible) the collection of survey data and the production of statistical information from these data. In view of the enormous amount of survey data processed at CBS each year this research is important from an economic perspective. The possibilities provided by current computer technology make automatic survey data processing an interesting and challenging research topic as well.

The purpose of this monograph is to create a theoretical framework for the description of survey data production processes, and to give an onset to the development of the relevant theory. The emphasis is primarily on computational aspects, although some statistical aspects are considered as well. An important topic in the area of survey data processing is not treated here, namely automated coding. The reasons for this purposive omission are two-fold. In the first place it does not fit into the framework created in this book, and secondly, for adequate treatment it requires a monograph in its own right.

The research project mentioned above was carried out by a team which, apart from myself, included: Wouter Keller, Jelke Bethlehem, Dee Denteneer, Anco Hundepool and Albert Verbeek. To all these people I am grateful for their insight, dedication and stimulation. The four first-mentioned people are responsible for the conception and/or implementation of the computer program BLAISE. This program is a tool for the design of questionnaires in a PASCAL-like language. In turn, such questionnaires are used by BLAISE to automatically generate several programs which can be used in survey data production processes (cf. chapter 0).

Furthermore I want to thank Dirk Sikkel, a former colleague of mine at CBS, for his constructive criticism on several technical reports which I wrote on the subject of the present monograph. I also enjoyed the many helpful and inspiring discussions with him on the most diverse aspects of survey data collection and processing.

My thesis advisors Arie Kapteyn, Jan Karel Lenstra and Theo Nijman had a major influence on the final form and contents of my Ph.D. thesis. I am grateful to them for their advice and guidance, and I appreciate the pleasant cooperation with them.

Leon Willenborg

0. INTRODUCTION

The purpose of this book is to describe certain aspects in the production of 'clean', i.e. checked and corrected, survey data. The red thread running through this process, as well as through this book, is the logical structure of questionnaires. This structure in fact defines which data are acceptable and which are not. This is of importance when the collected survey data are being checked and corrected at a statistical office. Furthermore it is of importance for the collection of the data themselves through a questionnaire. The logical structure prescribes precisely which questions should be posed to which persons in which order and (optionally) which answers are simultaneously acceptable and which are not.

The logical structure of a questionnaire can be very complicated. A questionnaire designer should have some tools at his disposal which can help him in defining and checking the logical structure of a questionnaire. In the present book we shall formalize the logical structure and suggest tests for certain aspects. These should be useful when trying to build a computer aided questionnaire design system.

Designing and testing a questionnaire is the only activity considered in this book which takes place prior to the field work of the survey. The remaining topics all deal with handling of the survey data, i.e. after they have been gathered. More particularly they deal with what we shall term the survey data production process. This process aims at coding, checking (or editing) and correcting of survey data. The only aspect we do not consider here is coding.

In the discussion of the survey production processes we assume that they are largely computerized. That is, the editing and correction of data is done by computer and not, or only in exceptional(ly difficult) cases by manual intervention. Such a state of affairs is highly desirable when large amounts of survey data have to be processed in limited time.

In the discussion of automatic data correction (imputation) we

especially keep in mind that it should be applied in a survey data production process. In particular this means that the statistical procedures which might be applied should not be too complex. Furthermore they should be computationally tractable.

The purpose of the present chapter is twofold. Each purpose is explained in its own section. Section 0.1 presents in more detail the sort of problems this book is about and provides a perspective from which the present work should be seen. In section 0.2 an outline of the present work is given.

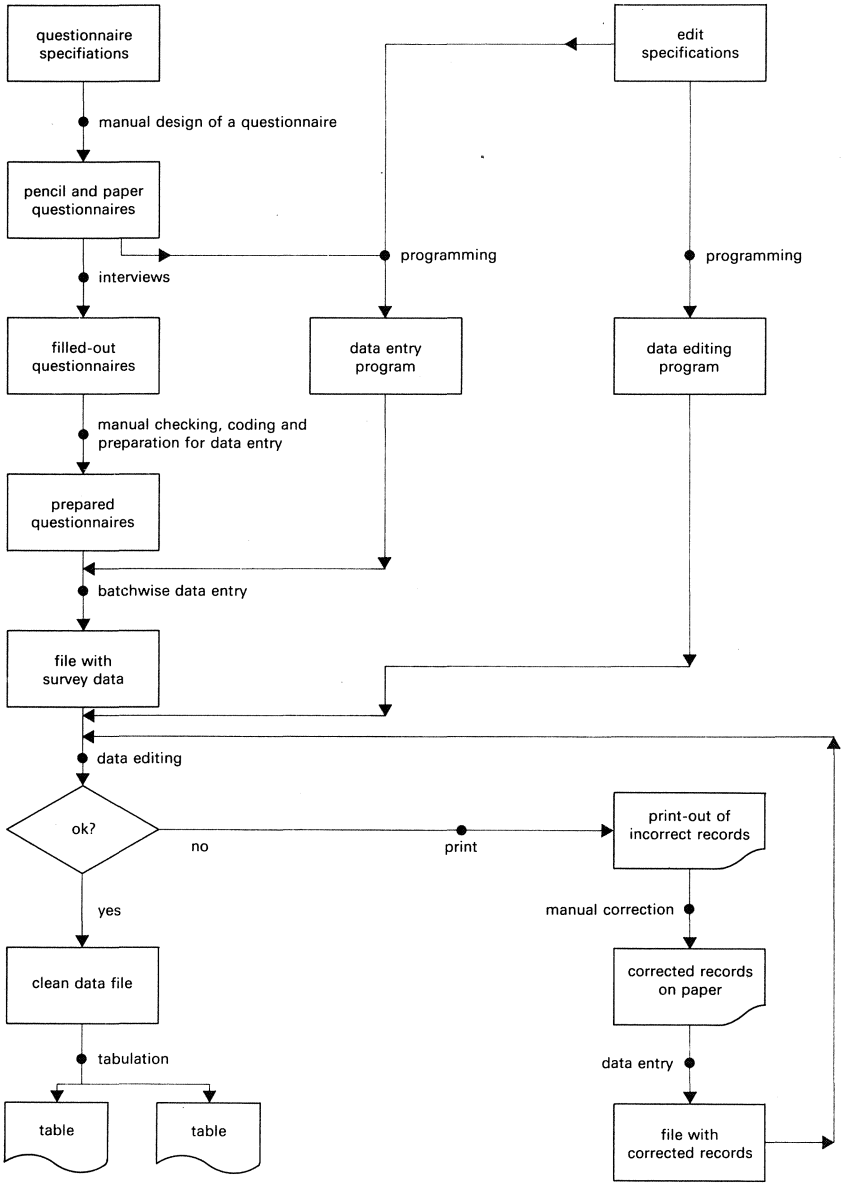
0.1 Historical perspectives

As a government agency responsible for the collection and dissemination of statistical data with respect to the Dutch society, the Netherlands Central Bureau of Statistics (CBS) carries out a great number of surveys every year. Giant amounts of data have to be collected and processed yearly. It is clear that these activities should flow as smoothly as possible, if one wants to minimize the time between collection and dissemination of high quality data.

In order to be able to reach this goal it is necessary that first of all an insight into the important activities of these processes as well as their organization is obtained. After this insight has been gained one can then try to automate as many of these activities as possible and redesign the whole production process on the basis of these changes.

With this idea in mind a study group was initiated at CBS in the fall of 1984. The production processes of four surveys carried out at CBS were studied first. The impression obtained was that such a process was typically being carried out as shown schematically in exhibit 0.1.1. No doubt this applies to other statistical agencies as well.

Exhibit 0.1.1 Schematic representation of a typical traditional survey data production process



The following features of the typical production process as shown in exhibit 0.1.1 are noteworthy:

1. The questionnaire is hand-made and to be used for paper and pencil interviewing.

2. The filled-out questionnaires are manually checked and corrected if necessary prior to data entry.

3. The coding of the data is also a manual activity.

4. Data entry is carried out batchwise by data typists, i.e. non-subject matter experts.

5. The data are automatically checked batchwise after data entry.

6. The correction of errors in these data is performed manually by subject-matter experts, who indicate the corrections on computer listings. These corrections are entered into a computer system. And the checking and correction process is started again. On the average, it takes about 2 to 3 such cycles until no more errors are found (or the process is aborted by eliminating certain data). However the cycling process may very well take several more cycles until it is finished.

7. Questionnaires on the one hand and programs for data entry and data checking on the other hand, are designed separately, and in general by different people.

8. The meta-information, comprising the definition of the variables in the questionnaire and the description of the meaning of their answer categories, is kept separate from the questionnaire and is not explicitly added to the finished data set.

9. Tabulation programs and questionnaires are designed separately.

10. Interesting information pertaining to the production process itself, such as the number of records checked, the number of errors observed in the records, etc. , is not systematically retained; and if it is retained it is not always available in an accessible form.

On the basis of these findings and with an eye toward new technological developments which allow the possibility of alternate forms of interviewing, viz. using computers and telephones, the following improvements were suggested:

1. The information on the variables etc. in a questionnaire should be brought together into a sort of 'knowledge base'. This knowledge base should be used as a source from which various programs, that are required in the production process, are automatically generated. For instance, it should be possible to generate the following programs, setups and other output: various types of questionnaires, viz. for paper and pencil interviewing (*PAPI*), computer assisted telephone interviewing (*CATI*) and computer assisted personal interviewing (*CAPI*), data entry programs (in case of PAPI), decoders (in case of CATI and CAPI), data editing programs (in case of CATI and CAPI), imputation programs (in case of CATI and CAPI), tabulation setups for various statistical packages (such as SPSS).

2. The design process of questionnaires should be carried out with the aid of computers. This requires a suitable language for the specification of a questionnaire. The design environment should provide assistance in checking a newly designed questionnaire; it should carry out as many checks as are possible, in particular in relation to the logical structure of a questionnaire. It may also prove to be very helpful if e.g. the routing structure of a questionnaire could be drawn automatically by the design system. Visualization of this structure is a great aid in perceiving it globally.

3. In the PAPI case interactive data entry, combined with automatic coding (this is optional; see also point 4.) and checking and correction of data should be employed instead of their batchwise processing. (This is also called computer assisted data input, abbreviated to *CADI*.) Using

CADI in a PAPI survey avoids lengthy and time consuming cycles in the processing of the data. It entails a major change in the survey data production process.

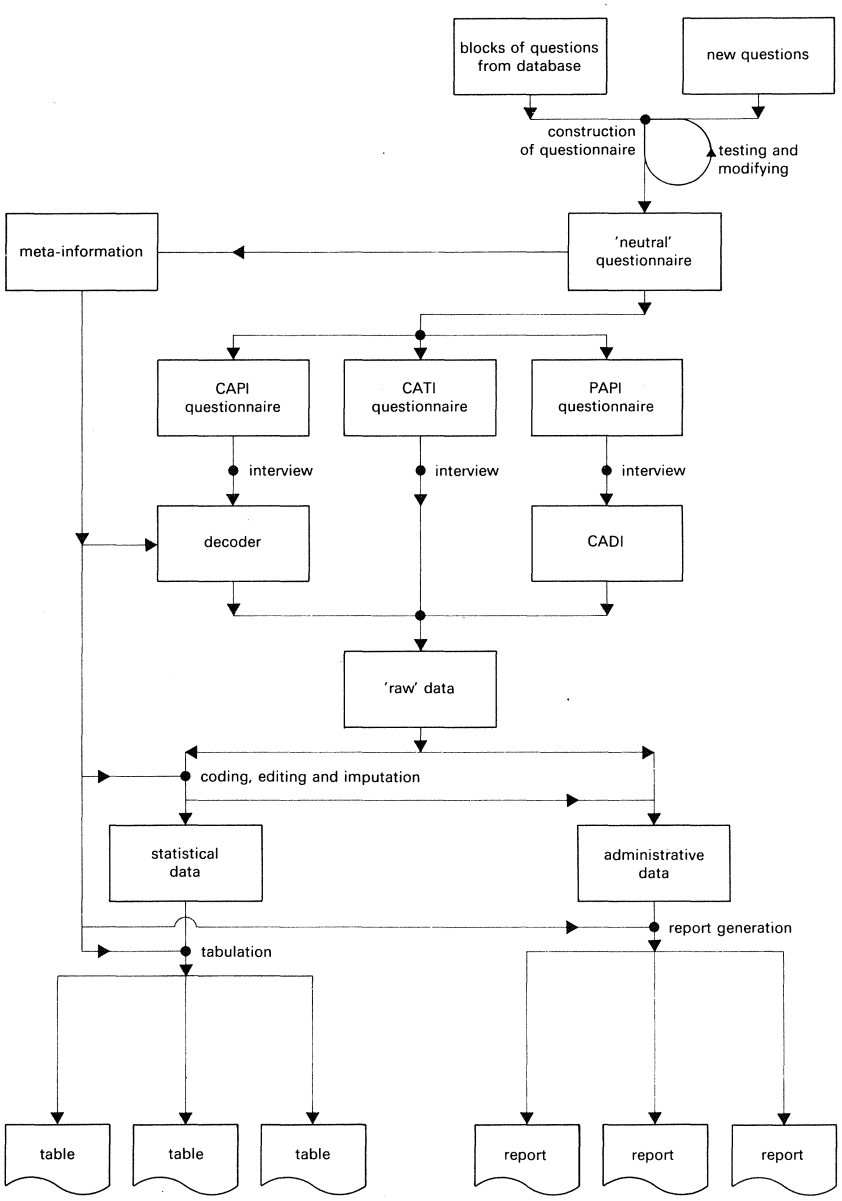
4. It should be investigated to what extent automatic data coding can be applied. Even if this process cannot be carried out completely automatically in all applications, i.e. without the intervention of human experts, it still might be possible to automate it to such a degree that a significant improvement of the speed of the coding process can be obtained.

In exhibit 0.1.2 it is shown how a production process would look like in view of the suggestions for improvements stated above.

When these insights had been gained, the development of the various software tools required was initiated at CBS. An important step that was taken then was the definition of a language, christened BLAISE, for the specification of questionnaires. It is essentially based on the well-known programming language PASCAL. Soon after the inception of the idea to use a PASCAL-like language to specify questionnaires, the implementation of a package for computer aided design of questionnaires was started. This package is also called BLAISE.

The development of an automatic coding system was started at CBS within the framework of a labour force survey 'new style' (LFS). This coding system can be used for the coding of professions and firms. This in turn was followed by the development of another coding system at CBS, viz. one that is used in the family expenditure survey. It is used to code the products people buy for consumption.

Exhibit 0.1.2 A schematic view of a survey data production process 'new style'



As another part of the activities of the LFS project an automatic imputation system was also developed, to eliminate the missing values in incomplete records by substituting suitable values. This system is based on a form of stochastic imputation. LFS is suited for automatic imputation because it is a CAPI survey. Along with the development of such an imputation system a tool was constructed to test it, by generating a data set consisting of test records which are incomplete.

In the present book we deal with several topics in the field of survey data processing, notably with respect to questionnaire design and testing, data editing and imputation. Our main interest is in computational aspects, although organizational and statistical aspects are considered as well. The aim is in the foremost place to provide a conceptual framework to study the processing of survey data. In fact the so-called logical structure of questionnaires is the unifying theme of the various topics of this book. Within the conceptual framework developed, we derive a few technical results. In the next section we briefly consider the organization of the book and the contents of the respective chapters.

0.2 Outline of the book

In the present section we first give a short description of a survey data production process. We shall only consider those elements in the process that are of importance for our study and leave out the more practical details. The contents of each chapter is briefly described subsequently.

We assume that a survey data production is started with the design of the questionnaire to be used. Designing and testing a questionnaire is a rather difficult job, in which computers can be of great assistance. After the questionnaire has finally been designed, it is possible to start the field work of the survey. The survey data can be collected in various forms, e.g. as written information or as machine readable code. But finally they are available in a form which is suitable for further

processing. We assume that the information provided by a single respondent is then present in a single record. We shall now consider the processing of such a record.

We assume that this record is automatically checked with respect to its consistency, and corrected if necessary. The questionnaire which is used in a survey is assumed to define what is to be considered consistent information. The consistency checks that are carried out are the following:

1. Range checks, which simply check whether the answers to individual questions are within the respective ranges.

2. Routing check, to verify whether the record is in accordance with the routing structure of the corresponding questionnaire. This structure defines which questions will be asked to which respondents and in which order. The routing structure can be considered to define the syntax of the questionnaire.

3. Edit checks. These checks are only carried out if the questionnaire contains certain constraints on the answers, called edits. Edits involve at least two variables in the questionnaire. Edits are constraints on the answers which are in general related to the semantics of the data. They are normally formulated by experts on (parts of) the subject of the survey. Edits are not strictly necessary to define a questionnaire, contrary to routing.

It is assumed that the checks are carried out in the same order as they are presented above. If no error is found then there is no problem. If a record contains an error then it depends on the type of error which action is undertaken. If a range check is violated, the corresponding variable is assigned a particular missing value, viz. 'missing due to out of range'. If the routing is found to be incorrect, a procedure is started to identify the variables which are possibly incorrect. These variables will also receive certain missing values, some of which can be identified by (unions of) transition sets. In fact after the routing has been checked and

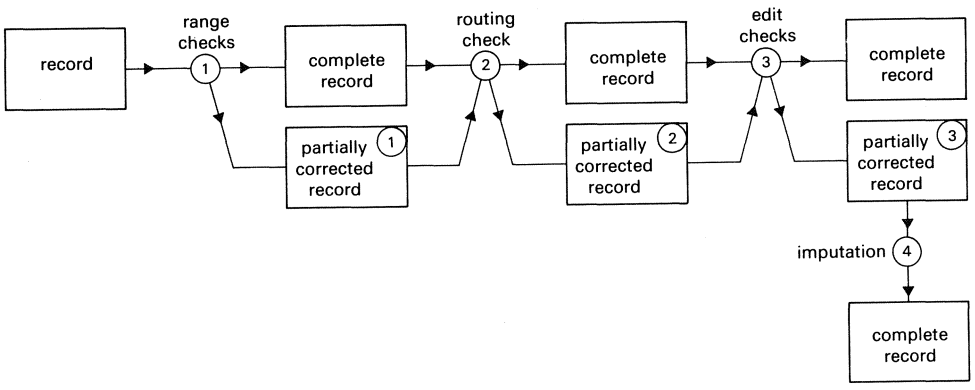
corrected, each missing value appearing in the record can be identified by an appropriate transition set. An error localization procedure is also applied when one or more edit errors have occurred. The variables which are supposed to be in error will also receive certain missing values, which indicate the reason for missingness. Such missing values can also be identified by appropriate transition sets. The process of eliminating errors in a record, by replacing non-missing values by missing ones is called partially correcting this record.

After these checks have been carried out on a record which initially contained incorrect values, it is assumed that a so-called partially corrected record has been produced which can be 'repaired' so as to yield a complete and correct record (with respect to the logical structure). That is, it should be possible to substitute values for the missing values created in the process just described. If we do not substitute these values then we have a data file containing incomplete records. If this is not satisfactory, then it is possible to complete the data file by applying a so-called imputation procedure, in which suitable values are substituted for the missing ones. 'Suitable' means in particular that the values should satisfy the constraints imposed by the logical structure. Furthermore they should be acceptable from a statistical point of view. In exhibit 0.2.1 an overview is given of the edit and imputation process as it is assumed in this book.

The reason to impute values is both pragmatically and statistically motivated. A complete file is much easier to handle than an incomplete one. Furthermore, the decision of a statistical bureau not to complete a file puts the responsibility (and the burden) to deal with the missing data in the hands of secondary data analysts. Of course, there is a danger associated in carrying out imputations, such as to overestimate the precision of the data. In order to avoid this, one could e.g. carry out multiple imputations (see e.g. Rubin, 1987). This amounts to the association of several imputed values with each missing one, instead of a single one. These multiple imputed values reflect better the true nature of a missing value, namely that it is a random variable. Although the idea of multiple imputation is theoretically attractive, it is less so from a

practical point of view.

Exhibit 0.2.1 Schematic overview of a data editing and imputation process



After this general introduction, we are in a position to discuss the organization of the remainder of the book more closely.

In chapter 1 the logical structure of a questionnaire is formally introduced. We start our discussion by considering questionnaires and their building blocks, the questions, more closely. Then we define the routing structure of a special type of questionnaire, which we shall call Markovian. Markovian questionnaires will be assumed throughout the book. The next structure to be introduced is the edit structure. The routing structure together with the edit structure comprises the logical structure of the questionnaire.

In chapter 2 some tests are discussed to check the correctness of the logical structure of a Markovian questionnaire. In particular we investigate the worst-case complexity of some decision problems related to the logical structure of a questionnaire.

In chapter 3 data editing is discussed. The approach given here is an extension of those that have hitherto appeared in the literature, in the

sense that the routing structure of a questionnaire is explicitly taken into account. The problem of localizing an error in a record is split into three steps, as was discussed above. Localizing and correcting routing errors can be solved in polynomial time (and space). Localizing and correcting edit errors, is NP-hard. So it is unlikely that there is an efficient algorithm for localizing edit errors. Nevertheless we discuss two procedures which have been proposed in the literature to solve this problem in case all domains of the variables in a questionnaire are finite. One method is due to Fellegi and Holt and the other one to Garfinkel.

As an aid in the identification of variables in error in a record which violates at least one edit, error models can be used. In chapter 3 we briefly discuss this topic, concentrating in particular on an error model proposed by Naus, Johnson and Montalvo. It is also shown in chapter 3 that this model yields another edit error localization procedure. Finally, a probabilistic method is discussed in chapter 3, which can be viewed as an approximation method for edit error localization.

Chapters 4 and 5 consider two aspects of incomplete data files in more detail. In these chapters the logical structure of a questionnaire does not explicitly play a role. In fact these chapters illustrate some of the problems created by an incomplete file. Furthermore chapter 5 forms a prelude to chapter 6, in particular to section 6.2.

In chapter 4 we concentrate on two issues pertaining to databases (or simply files) containing missing data. In the first place we attempt to give a complete enumeration of all types of missing values that can be found in such databases, in the light of the theory developed in this book. In the second place we discuss four approaches to deal with missing data in databases, in view of manipulations performed with such databases (joining, selecting, projecting, etc.) and when querying such a database.

In chapter 5 we consider an estimation prcblem in the presence of missing data in the data set, viz. the estimation of a probability density of a discrete variable y. Two approaches are discussed to attack this problem, which are called the quasi-randomization and the label independent

approach respectively. In order to apply either of these approaches it is necessary that the response propensities for each category of y are known, or can be estimated.

In chapter 6 we pick up the main thread of our story, and consider some statistical and computational aspects with respect to imputation. In particular we focus on several issues which are considered to be of importance for an imputation system which is to be used in a survey data production process.

In chapter 7 we formulate some conclusions that can be drawn from the results in the previous chapters, and discuss some directions for future research.

The book is completed by three appendices and a list of references. Appendix A contains a few fundamental graph theoretical algorithms, which are used in procedures and algorithms in chapters 1, 2, 3 and 6. In appendix B two numerical characterizations of the routing structure of a Markovian questionnaire, viz. the balance and complexity, are discussed. In appendix C, finally, a discussion of some ideas from discriminant analysis is given, to provide the necessary background for a part of chapter 6.

1. THE LOGICAL STRUCTURE OF A QUESTIONNAIRE

1.1 Questionnaires

From an abstract point of view a questionnaire is a collection of questions with a structure imposed on it. This structure, which we shall call the logical structure, is defined to insure that one obtains relevant and consistent information from respondents, given a certain set of questions. (In fact, the logical structure itself defines what is considered relevant and consistent by the designers of the questionnaire.)

A questionnaire as it is used in a survey is a lot more than a set of questions with a structure imposed on it. To design such a questionnaire means that there are more aspects to consider than those related to its logical structure. For instance the wording of its questions, the anwers to its questions and its physical layout (either on paper or on a screen) are all important aspects to be considered. Such matters contribute to the success of a questionnaire as well. But they seem to belong more to the realm of 'art' than to that of 'science'. In the present book we shall deal only with the logical structure of a questionnaire. In particular we shall study the problems which this structure entails when designing a questionnaire and when data, which constitute the answers provided by respondents, are being checked and corrected.

The logical structure of a questionnaire consists, in its most general form, of two components, viz.

1. The routing structure.
2. The edit structure.

The routing structure defines which questions will be asked in which order to which respondents. The next question to be posed in an interview is entirely determined by the answers which have, up to that moment, been provided by the respondent. The routing structure is an essential part of a questionnaire and cannot be left out. This is different for the edit structure, which is optionally included in a questionnaire. The edits

constitute additional checks on the answers provided by respondents. They can be introduced to increase the consistency of the answers provided by respondents, beyond the constraints imposed by the routing structure. In this book we shall make an explicit distinction between two types of questionnaires, viz. questionnaires without edits and questionnaires with edits. The first type shall be referred to as *q-type*, and the second as *q&e-type* questionnaires.

As a rule the questionnaires considered in this book are assumed to be *Markovian*. This means that they have the property that each transition to the next question depends solely on the answer to the question from which the transition is made. (Note the similarity with Markov chains, from which the name is borrowed.) This property is common to many traditional questionnaires, but not to all. In *non-Markovian* questionnaires, on the contrary, a transition to the next question is, for at least one question, dependent on at least two answers to previous questions. In order to avoid technical complications we shall not consider non-Markovian questionnaires here. So we adopt the following convention in this book:

Convention 1.1.1 The only questionnaires considered in this book are Markovian questionnaires, unless explicitly stated otherwise. ///

In the following sections of this chapter we shall define several concepts mentioned above, as well as several unmentioned ones, more precisely. Then we will be prepared to study various aspects of and related to the logical structure of questionnaires.

1.2 Questions

The building blocks for questionnaires are questions. Let the set of questions in a given questionnaire be denoted by $\{v_1,\ldots,v_n\}$. For each question v_i there is a set R_i of possible answers. Such a set will be called a *domain*. Depending on the nature of its domain, we assume that the corresponding question v_i can be classified as one of the following types:

a. v_i is *closed*. In this case R_i consists of a finite and modest number of possibilities (answer categories), which can be explicitly enumerated. The answers to such questions shall be called *precoded*.

b. v_i is *open*. In this case R_i consists of a (very) large number of elements, which are not explicitly enumerated. The answers to such questions shall be called *uncoded*.

c. v_i is *partly open*. In this case some of the elements in R_i are explicitly enumerated (the precoded answers) and the rest is not (the uncoded answers).

In the discussion below we shall identify 'questions' also with 'variables', although formally these concepts differ in meaning. Partly open and open questions are the most difficult ones to handle, basically as a result of the following limitation. The non-precoded answers to such questions have to be coded by experts or by automatic coding systems. This process, however, cannot be carried out during an interview, at least not at the present state of affairs and neither, probably, for some time to come. Therefore we have to assume that the coding has to take place at a later phase, viz. when the survey data are being processed at the agency which conducts the survey. This limitation implies that it is impossible to base certain decisions with respect to the logical structure of the questionnaire on uncoded answers while an interview is being carried out. It is then for instance impossible to let the routing depend on their interpretation or to involve them in an edit, i.e. an on-line edit. Of course it is still possible then to use the information as to whether or not a question has been answered or as to whether a particular answer category was used.

In the sequel we adopt the following convention in order to facilitate the discussion:

Convention 1.2.1 The questionnaires considered consist of closed questions only, unless explicitly stated otherwise. ///

Adopting this convention avoids to repeatedly single out open or partly open questions from the closed ones because they may require a different treatment. This does not limit the scope of the treatment below, but only alleviates the exposition. In practice, however, careful attention should be devoted to the proper treatment of open and partly open questions.

1.3 Routing

We shall now define the routing structure of a questionnaire more formally. Let Q denote a questionnaire. We shall assume that the routing structure in Q is represented by a *routing graph* G which consists of a pair (V,E) with the following interpretations:

1. The finite set $V = \{v_1, \ldots, v_n\}$ of vertices corresponds to the set of questions in Q.

2. The finite set E of edges represents the possible transitions between questions in Q. Each edge e in E can be represented by an ordered pair of vertices (v_i, v_j), indicating that it is possible in Q to jump from question v_i to question v_j (but not necessarily reversely).

To indicate under which conditions there will be a possible transition from v_i to v_j the *transition set* $R_{ij} \subset R_i$ is required. This set consists of all those answers to v_i that imply that v_j is the next question to be posed.

The points 1 and 2 above express that G is a finite directed graph, for short a *digraph*. In addition, we require G to possess the following properties:

1. G is an acyclic digraph, which prevents the possibility to pose a question twice.

2. G has exactly one vertex with indegree zero, which will be called the *source*, and exactly one vertex, the *sink*, with outdegree zero. This means that every question, except the source, has a predecessor, and that

every question, except the sink, has a successor. The source corresponds to the first question in Q and the sink to the last one.

Together both properties imply that G is *connected*, i.e. every vertex in V can be reached from the source, and the sink can be reached from every vertex in V. If a questionnaire Q^* has a routing graph $G^*=(V^*,E^*)$ which is connected but has at least two vertices with indegree or outdegree zero, G^* can be easily extended to one of the above type, by adding dummy vertices and edges, as follows. Assume that G^* has e.g. two vertices v and w with outdegree zero and one source, then it can be extended to a routing graph G', simply by adding a dummy vertex u and connecting v and w to u, i.e. by adding the edges (v,u) and (w,u) to those already present in E^*. The vertex u will be the sink in G'.

A *path* π *from* v_{i_1} *to* v_{i_k} is a sequence of pairs of vertices (v_{i_1},v_{i_2}) $(v_{i_2},v_{i_3})\ldots(v_{i_{k-1}},v_{i_k})$ or, alternatively, a sequence of pairs $(v_{i_1},R_{i_1,i_2})(v_{i_2},R_{i_2,i_3})\ldots(v_{i_{k-1}},R_{i_{k-1},i_k})(v_{i_k},R_{i_k})$, where the v_{i_j} denote questions, the $R_{i_j,i_{j+1}}$ transition sets and where R_{i_k} is the domain of variable v_{i_k}. Instead of $(v_{i_1},v_{i_2})\ldots(v_{i_{k-1}},v_{i_k})$ we shall also write $(v_{i_1},\ldots,v_{i_k})$. It depends on the application at hand which of these definitions is most suitable. If v_{i_1} and v_{i_k} are the source and sink of the routing graph then π is simply called a *path*. Any vertex which appears in this sequence is said to *lie* on π. Alternatively expressed, π is said to *cut* such a vertex. The *length* of a path π is the number of pairs $(v_i,R_{i,j})$ of which it consists. For a routing graph G we define the *path set* Π_G, or just Π, which consists of all paths in G from source to sink.

Remark 1.3.1 If the transition sets R_{ij} corresponding to the transitions in a questionnaire are mutally disjoint then a path can be represented as a sequence of such transition sets. That is, the v_i can be discarded. ///

A routing graph G can be *contracted* if it contains vertices v and w and a path π from v to w with a length greater than 1, such that any vertex lying on π, except w, has exactly one ingoing and one outgoing edge in G. A path π from v to w in G which obeys this condition will be called a *linear part* of G. A linear part π can be *removed* from G by replacing it by

the edge (v,w).

A routing graph F, which has been obtained from a routing graph G by removing a number of linear parts from G, will be called a *contraction* of G. A routing graph which cannot be contracted is said to be *maximally contracted*.

A *cut-point* in a routing graph G is a vertex v such that every path $\pi \in \Pi$ cuts v. The source and the sink are trivial examples of cut-points. In appendix A an algorithm and a listing of a PASCAL program are given to determine all cut-points in a routing graph.

The vertices in a routing graph can be linearly ordered by applying a *topological sort* (see appendix A). The reason for this is the acyclicity of a routing graph. In the sequel we shall often implicitly assume that a routing graph has been topologically sorted in one way or another. We shall sometimes denote the resulting linear order in the vertices by $\leq$, without referring explicitly to the particular topological sorting that has been used, because it is assumed to be fixed.

In appendix B two characterizations of the structure of a routing graph are defined and some of their properties are given.

In exhibit 1.3.1 the routing graph of questionnaire B of the Labour Force Survey 1983 is presented as an example of a routing graph in a real-life Markovian questionnaire. Note that it has 13 cut-points, viz. 1, 62, 63, 65, 70, 72, 92, 103, 105, 106, 107, 108 and 109.

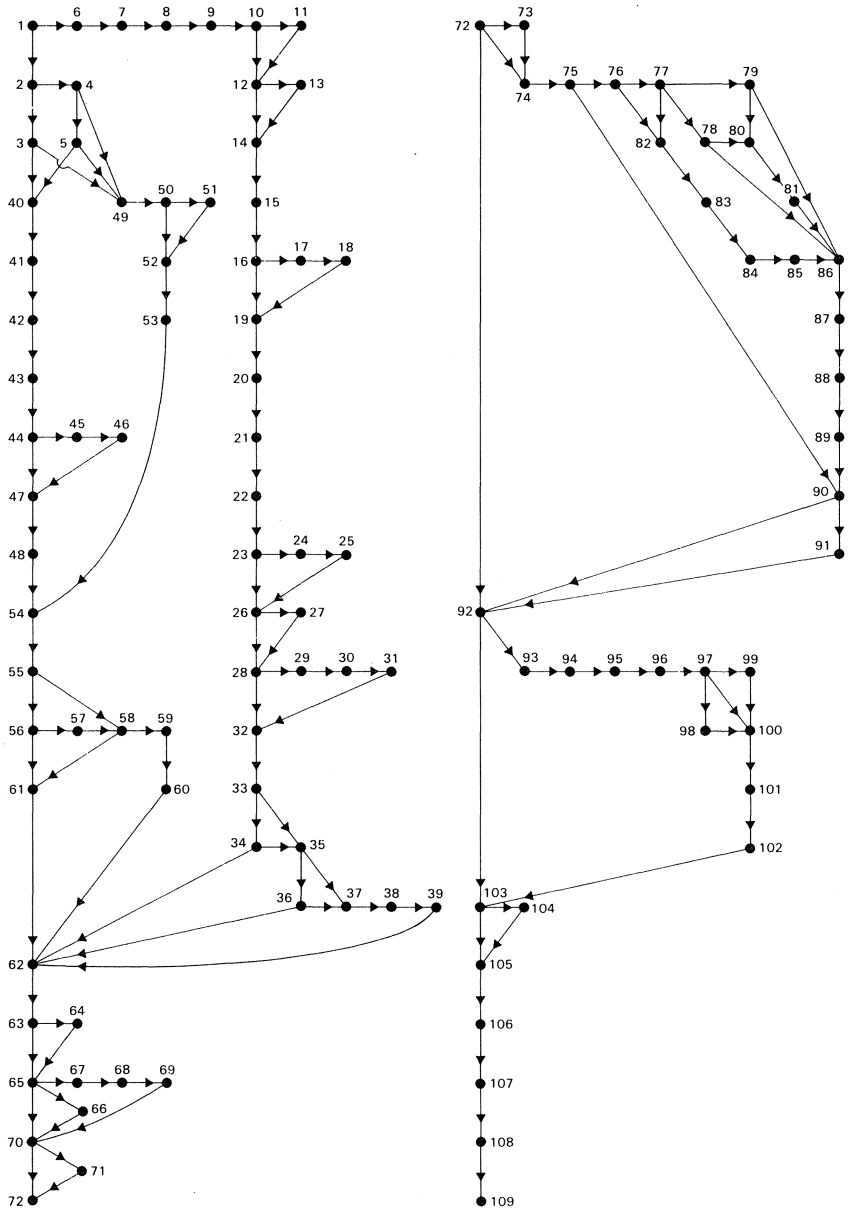
Now let G_1 and G_2 be two routing graphs. The *series composition* of G_1 and G_2, denoted by $G_1 * G_2$, is the routing graph obtained from G_1 and G_2 by identifying the sink of G_1 and the source of G_2. We shall also refer to series composition as *glueing*. Glueing is obviously associative, i.e. $(G_1 * G_2) * G_3 = G_1 * (G_2 * G_3)$ for all routing graphs G_1, G_2 and G_3. But glueing is not commutative, i.e. there are routing graphs G_1 and G_2 such that $G_1 * G_2 \neq G_2 * G_1$. Note that the vertex in $G_1 * G_2$ at which G_1 and G_2 are glued together, i.e. the sink of G_1 and the source of G_2, is a cut-point.

Instead of glueing different routing graphs together, we can also consider the opposite operation of *splitting* or *decomposing* a routing graph into *components* or *routing sub-graphs*. Such a decomposition of a routing graph can be important if the original one is too big to consider it as a whole. For it might be easier to deal with the routing sub-graphs than with the original routing graph (cf. section 1.6).

A routing graph G is said to be *decomposable* if there exist graphs G_1 and G_2, neither of them being a *point-graph* (which consists of a single point), such that $G=G_1*G_2$. G is called a *prime* if such a decomposition does not exist. A *canonical decomposition* of a routing graph G is a decomposition of G as $G_1*\ldots*G_m$ ($m\geq 2$) for certain prime routing graphs $G_1,\ldots,G_m$. It is not difficult to verify that for any routing graph G such a decomposition is unique if G is decomposable.

A canonical decomposition of a routing graph can be found by calculating its cut-points. Suppose the cut-points of a routing graph G are $c_1,\ldots,c_{m+1}$, such that $c_1<c_2<\ldots<c_{m+1}$. Then the i-th prime routing graph G_i in the canonical decomposition of G consists of the vertices v such that $c_i\leq v\leq c_{i+1}$, and those edges of G which connect vertices in G_i.

Exhibit 1.3.1 Example of a routing graph (from a questionnaire used in the Dutch Labour Force Survey 1983).



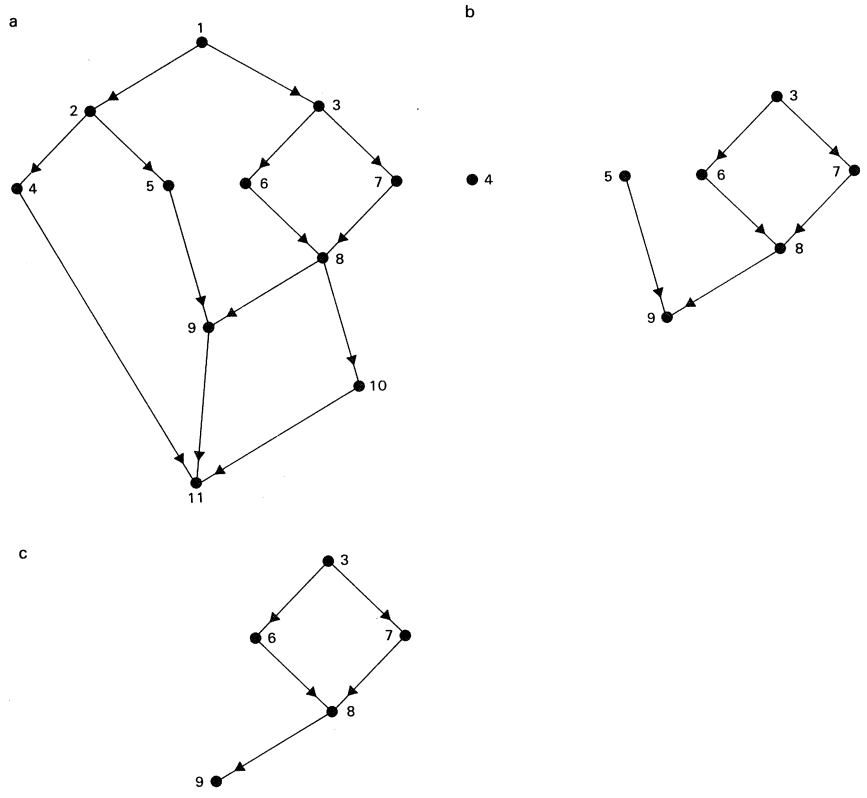
Let G be a routing graph and A its adjacency matrix. The *inverted routing graph* G^{-1} of G is obtained from G by inverting its edges, i.e. by replacing each edge (v,w) in G by (w,v), where v and w are vertices in G. It is straightforward to see that G^{-1} is also a routing graph. The adjacency matrix B of G^{-1} is associated to A by the identity B=PAP, where P is the square matrix (of the same order as A), which has 1's on the main anti-diagonal and 0's elsewhere, i.e.

$$P = \begin{pmatrix} 0 & \cdots & 1 \\ \vdots & \iddots & \vdots \\ 1 & \cdots & 0 \end{pmatrix} . \qquad (1.3.1)$$

Note that premultiplication of A by P amounts to inversion of the rows in A, and postmultiplication by P to inversion of the columns. Furthermore it holds that $P^2=I$, the identity matrix of the same order as P.

Let G be a routing graph, and let V denote its set of vertices and E its set of edges. Let A be its adjacency matrix, and let $\alpha,\beta\in V$ with $\alpha\leq\beta$. The *slice* ($[\alpha,\beta]$) in G is the sub-digraph of G with vertex set $\{v\in V:\alpha\leq v\leq\beta\}$ and edge set $\{(v,w):\alpha\leq v,w\leq\beta$ and $(v,w)\in E\}$. The adjacency matrix of ($[\alpha,\beta]$) is obtained from restricting A to the rows and columns in the vertex set of the slice. Note that a slice is not necessarily a routing graph, because there may be several sources or sinks. The *segment* $[\alpha,\beta]$ in G is the maximal sub-digraph in G with the single source α and the single sink β, i.e. the sub-digraph of G with vertex set $V'=\{v\in V:A^*_{\alpha v}=1$ and $A^*_{v\beta}=1\}$ and edge set $\{(v,w)\in E:v\in V'$ and $w\in V'\}$, where A^* is the *transitive closure* of A. That is, $A^*_{i,j}=1$ if and only if there is a path from i to j, and 0 otherwise. In appendix A an algorithm is presented to calculate the transitive closure of a digraph. A segment is a routing graph and its adjacency matrix is obtained from A by restricting it to the rows and columns corresponding to the vertices in V'. Trivially, a segment $[\alpha,\beta]$ is also a subdigraph of the corresponding slice ($[\alpha,\beta]$). In exhibit 1.3.2 an example of a slice and of a segment is shown.

Exhibit 1.3.2 Example of a slice (b) and a segment (c) in a routing graph (a)



Remark 1.3.2 If the transition sets are mutually disjoint, then a record can be unambiguously represented as a sequence of answers. The v_i are not required then to indicate to which question a certain answer corresponds.

///

We assume that the information provided by a respondent in an interview

which was based on a questionnaire Q, is ultimately represented as a record in a computer. (Of course, there may actually be several physical records containing the response of each respondent participating in the survey; yet we may assume that for each respondent the information is contained in a single, symbolical, record.) In that form the information is most conveniently subjected to further processing and analysis. In fact we shall talk about a *record* when the collection of answers provided by a respondent in an interview is meant. For convenience we shall assume that a record is represented by a sequence of pairs of questions and corresponding answers. An answer can be either of two types:

1. An element of a transition set (or domain).
2. A special type of missing value, which indicates that the question was originally answered by a respondent but this answer was incorrect.

If a variable has a value of the second type, it will receive a *regular value*, i.e. a value of the first type, after an imputation procedure has been applied (see chapter 6). In section 4.2 a list of various types of missing values can be found.

A *complete record* r is a sequence of pairs $(v_{i_1},a_{i_1})(v_{i_2},a_{i_2})\ldots(v_{i_{k-1}},a_{i_{k-1}})(v_{i_k},a_{i_k})$, where the v_{i_j} denote questions and the a_{i_j} are answers to v_{i_j}, i.e. elements in R_{i_j} (j=1,...,k-1), such that:

1. v_{i_1} is the source of G.
2. v_{i_k} is the sink of G.
3. $a_{i_j}\in R_{i_j,i_{j+1}}$ for j=1,...,k-1, if k>1.
4. $a_{i_k}\in R_{i_k}$

From this and the assumed acyclicity of G, it follows that a variable v_i appears at most once in r. Note also that if we replace each a_{i_j} by the transition set $R_{i_j,i_{j+1}}$ to which it belongs we obtain a path in G. In fact records which yield the same path in this way can be considered equivalent and are said to possess the same *routing structure*. For a routing graph G we define the *set* REC_G, or REC, *of records with a correct routing structure*, which consists of sequences $(v_{i_1},\alpha_{i_1})\ldots(v_{i_{k-1}},\alpha_{i_{k-1}})(v_{i_k},\alpha_{i_k})$,

where v_{i_1} is the source of G and v_{i_k} its sink, and where α_{i_j} is either the transition set $R_{i_j,i_{j+1}}$ or an element in this set. Hence $\Pi \subset REC$ for a routing graph G. A record which does not obey conditions 1 through 4 above will be called an *incomplete record*.

1.4 Edits

An *edit* e is a constraint on the joint values of two or more variables, which is not derived from the routing structure, and which defines which combinations of values are not acceptable. Formally an edit e, defined on a set of variables $v_{i_1},\ldots,v_{i_k}$, is a Boolean function

$$e : R_{i_1} \times \ldots \times R_{i_k} \rightarrow \{\text{TRUE, FALSE}\} , \qquad (1.4.1)$$

where the R_{i_j} denote the domains of the v_{i_j}. The set $E=e^{-1}(\text{TRUE})$ is called the *edit set* of e and consists of values in $R_{i_1}\times\ldots\times R_{i_k}$ which are considered unacceptable. The variables $v_{i_1},\ldots,v_{i_k}$ are said to be *involved* in e.

Edits constitute additional constraints on the answers provided by respondents, additional to the constraints imposed by the routing structure. They can be used in CAPI or CATI situations to check the consistency of combinations of answers provided by respondents. Inconsistencies found in certain answers of respondents can be rectified on the spot. But edits can also be used when checking the survey data at the data editing phase, e.g. in a CADI situation. If inconsistencies are found then they generally have to be corrected without consultation of the respondent who provided them, but by using statistical means.

The routing structure is essential for questionnaires, but an edit structure is not. Edits can optionally be added to a questionnaire. Of course, if edits have been defined in a questionnaire they count as much as the routing does.

Technically, routing and edits differ in several ways. On the one hand the routing structure in a Markovian questionnaire defines a partial ordering on its questions, i.e. a binary relation. Edits, on the contrary, may involve any number of variables greater than one. Even within a single questionnaire the edits need not all involve the same number of variables. Furthermore edits only restrict the joint values of variables with regular values. The routing structure however does not only impose such a restriction, it also indicates which variables should be skipped, i.e. may not appear in a complete record. The *exclusive or property*, for short the *xor-property*, of the routing structure in a questionnaire excludes the possibility that in a complete record the variables v_i, v_j, and v_k all three have regular values if (v_i, v_j) and (v_i, v_k) are edges in the corresponding routing graph.

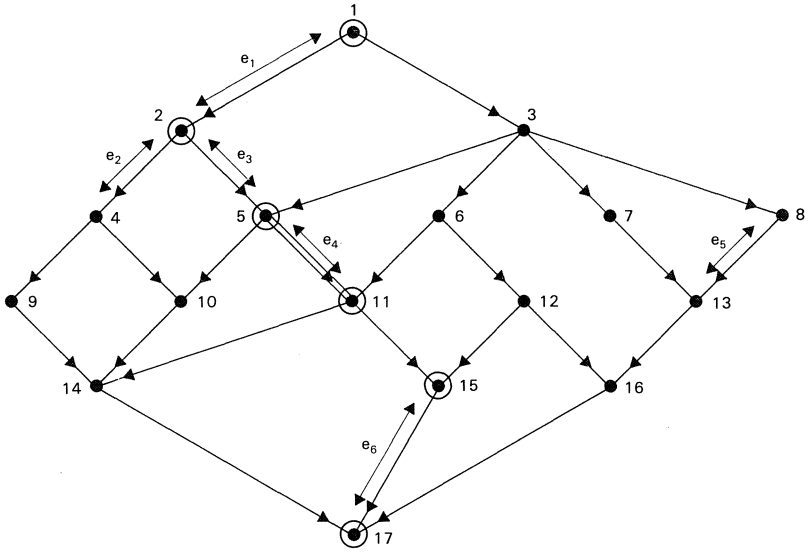
When applied to check the answers provided by a respondent, assumed to be available in a record, edits can be in several states, depending on the information present in the record. As was already said in the previous section the values of the variables in a record are essentially of two types: regular values and missing values which should be replaced by regular ones. An edit e is said to be *activated* by a record r if the variables involved in e are present in r. An edit e should be activated by a record r before r can be checked to satisfy or violate e. In order that this check be carried out, it is necessary that the variables involved in the edit have regular values. An activated edit which, for this reason, cannot be applied to check the consistency of certain values in a record, is said to be *idle*. If a record r (activates and) violates an edit e, it can make e idle, by replacing the regular value of at least one variable in r, which also appears in e, by a so-called regular missing value (see section 4.2; it is a missing value which should be replaced by a regular value).

If in the record r at least one variable which is involved in e is not represented in r, then e is called *invisible*. An edit e is said to be *activatable* if the variables involved in it lie on a path in the routing graph of the corresponding questionnaire. This means that there exists a correctly completed questionnaire (with respect to the routing structure)

which can activate this edit. Any correctly specified edit is activatable. In exhibit 1.4.1 each of these states of an edit is illustrated.

Let Q&E denote a questionnaire and let REC denote the set of records in this questionnaire with a correct routing structure. Let e be an edit. Instead of taking the domain of e as the cartesian product set in (1.4.1), we may assume that its domain is REC and its codomain the set {TRUE, FALSE, NEUTRAL}, i.e. e: REC→{TRUE, FALSE, NEUTRAL}. It is assumed that if r activates e then either e(r)=TRUE or e(r)=FALSE, otherwise e(r)=NEUTRAL. The subset e^{-1}(FALSE) of REC consists of the records which *satisfy* e and are considered to be acceptable with respect to e. The set e^{-1}(TRUE) consists of the records which *violate* e and are considered to be unacceptable with respect to e.

Exhibit 1.4.1 Examples of activated, idle and invisible edits

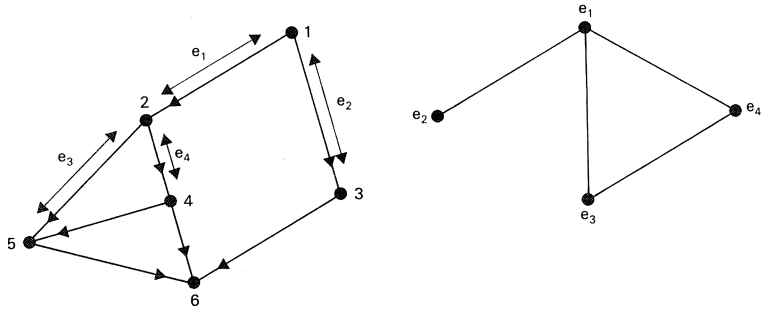


Remark: indicated is the path π=(1,2,5,11,15,17). It is assumed that there is an incomplete record r associated with π, for which the value of variable/question 5 is missing (hence the double edge in the figure). The edits activated by π are e_1, e_3, e_4, e_6; the edits which are invisible for r are: e_2, e_5; and the edits which are idle: e_3, e_4.

The edits in a q&e-type questionnaire can be ordered via the lexicographical ordering of the sets of variables involved in the edits, which are numbered themselves through a topological sorting procedure. We shall refer to such an ordering of the edits as a *natural ordering*. In general it is not unique. A natural ordering can be useful in a CAPI or CADI situation, when the edits are invoked on-line.

For a set of edits in a q&e-type questionnaire we can define an *edit graph* as follows. The edits themselves are the vertices of the edit graph. Let W_e and W_f denote the set of variables involved in e and f respectively. Now (e,f) is an edge in the edit graph if $W_e \cap W_f \neq \emptyset$. In exhibit 1.4.2 an example of an edit graph is shown. Note that an edit graph is defined independently of the routing structure in the corresponding questionnaire.

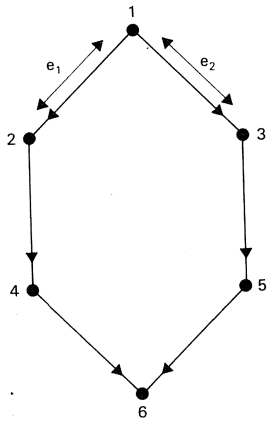
Exhibit 1.4.2 Example of a routing graph with edits and the corresponding edit graph



A picture of an edit graph visualizes how the edits are interrelated. The edit graph can also be used to calculate which variables are related to each other via the edit structure but independently of the routing structure: calculate the connected components of the edit graph by a depth-first search (see appendix A) and take the union of the sets of variables corresponding to the vertices in each component. The sets of variables calculated for each component are disjoint. As far as the edit structure is concerned variables in different component sets are independent of each other and should be treated as such. However, because the routing

structure is discarded in an edit graph, the edits which can be simultaneously activated cannot be determined from it. Exhibit 1.4.3 illustrates this point. The xor property of a routing graph is reponsible for this phenomenon.

Exhibit 1.4.3 Example of two edits which cannot be activated simultaneously



Two edits e_1 and e_2 can be *simultaneously activated*, or are *simultaneously activatable*, if there is a path π in the routing graph such that the variables involved in both e_1 and e_2 lie on π. It is easy to see that this property is symmetric but not necessarily transitive. The following theorem holds.

Theorem 1.4.1 Let $e_1,..,e_p$ be activatable edits such that for every pair e_i and e_j a path $\pi_{i,j} \in \Pi$ exists which simultaneously activates e_i and e_j. Then there is a path $\pi_{1,\ldots,p} \in \Pi$ which simultaneously activates $e_1,\ldots,e_p$.

Proof We prove the assertion by contradiction. So suppose that such a path $\pi_{1,\ldots,p}$ does not exist for $e_1,\ldots,e_p$. Consider $\cup_i W_i$, where W_i is the set of variables involved in e_i. Let the elements in this union be $\alpha_1<\alpha_2<\ldots<\alpha_t$. Because we have assumed that there is no path $\pi_{1,\ldots,p}$ which cuts these vertices it follows that there is at least one j such that (α_j,α_{j+1}) is not an edge in the transitive closure G^* of G, i.e. there is no path in G from α_j to α_{j+1} for some $j \in \{1,\ldots,t-1\}$. Then α_j and α_{j+1} cannot belong to the same set W_i, because edit e_i is activatable. Therefore

we have that $\alpha_j \in W_i$ and $\alpha_{j+1} \in W_k$ for some $i \neq k$. However there is a path $\pi_{i,k} \in \Pi$ simultaneously activating e_i and e_k, which, alternatively expressed, means that $W_i \cup W_k \subset \pi_{i,k}$. In particular this implies that there is a path in G from α_j to α_{j+1}. Contradiction. Hence such a path $\pi_{1,\ldots,k}$ does exist. ///

If e_1 and e_2 are edits which are always simultaneously activated we can replace them by a single edit $e_1 \vee e_2$, defined as

$$(e_1 \vee e_2)(r) = e_1(r) \vee e_2(r) \ , \tag{1.4.2}$$

for $r \in REC$ which activates both e_1 and e_2. The new edit $e_1 \vee e_2$ does not distort the constraint structure as defined by the original logical structure of the questionnaire in which e_1 and e_2 appear. The set of variables involved in $e_1 \vee e_2$ is the union of the sets of variables involved in e_1 and e_2. Applying this disjunction operation repeatedly to edits in a questionnaire eventually yields a set with a minimal number of edits.

Remark 1.4.1 Note that if we define the composite edit $e_1 \vee e_2$ for edits e_1 and e_2 which cannot always be activated simultaneously by

$$(e_1 \vee e_2)(r) = \begin{cases} e_1(r) \vee e_2(r) \text{ if } r \in REC \text{ activates } e_1 \text{ and } e_2 \\ \qquad\qquad\qquad\qquad \text{simultaneously} \\ \text{NEUTRAL otherwise} \ , \end{cases} \tag{1.4.3}$$

then $e_1 \vee e_2$ is not equivalent to the situation with two separate edits e_1 and e_2. The reason for this is that a record $r \in REC$ which activates e_1 but not e_2 would be checked against e_1 but not against $e_1 \vee e_2$, which is not activated. ///

Remark 1.4.2 On the one hand it is advantageous to form composite edits from edits which are always simultaneously activated, because this increases the transparancy of the logical structure in the questionnaire. On the other hand it is also helpful to split more complicated edits into smaller ones, especially when localizing errors in questionnaires. The reason for this is that an error localization procedure is based on the

sets of variables involved in edits. The smaller such sets the better it is to pin-point possibly faulty values in a record. For details refer to chapter 3. ///

The foregoing shows that it is of some interest to consider the problem which edits in a given q&e-type questionnaire are always activated simultaneously. In chapter 2 we shall study this problem.

It is implicitly assumed that an edit e contains a minimal set of variables, i.e. such that no variable involved in e can be removed from e without creating an edit e' which is different from e, where e and e' are interpreted as functions REC→{TRUE, FALSE, NEUTRAL}. But this minimality condition is not a formal requirement for an edit. It is only advantageous for edits to have this property when localizing errors in a record (cf. chapter 3).

Example 1.4.1 (Minimality of set of variables involved in edit)

Consider a q&e-type questionnaire containing, among others, the following six variables and corresponding domains: $v_1 \in [0,100]$, $v_2 \in \{0,1,2\}$, $v_3 \in \{1,2\}$, $v_4 \in [0,100]$, $v_5 \in \{0,1,2,3\}$ and $v_6 \in \{1,2,9\}$. Suppose that $(v_1,\ldots,v_6)$ forms a linear part of the routing graph of the questionnaire. Suppose furthermore that the following two edits are defined in this questionnaire:

e_1 : $0 \leq v_1 \leq 50 \wedge v_2 \in \{0,1\} \wedge v_3 \in \{1,2\}$;

e_2 : $0 \leq v_4 \leq 50 \wedge v_5 \in \{0,1\} \wedge v_6 \in \{1,2\}$.

Now edit e_1 does not obey this minimality condition, whereas e_2 does. Note that e_1 can be replaced by the equivalent edit

e_1' : $0 \leq v_1 \leq 50 \wedge v_2 \in \{0,1\}$,

because v_3 is not constrained by e_1. ///

If we want to understand the logical structure in a q&e-type questionnaire

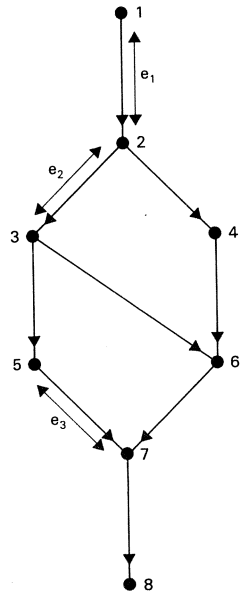
we cannot restrict our attention to individual edits or to edits which are always activated simultaneously. We should also consider edits which can be activated simultaneously by some records. The reason is that if a record violates several edits, one or more of the variables involved in these edits should be assigned a different value, if we want to reach a situation in which no edit is violated. If one considers the edits one at a time, and one adjusts the values of some variables in the record so as to satisfy the edit in consideration, one might be going on for quite a while searching for an acceptable record, which closely resembles the original one. It is possible that a change of the value of some variable can cause an edit, which was previously satisfied, to be violated. This cannot happen if one considers certain edits jointly. In the following example this is illustrated.

Example 1.4.2 (Simultaneous activation of edits)

Consider the routing graph G in exhibit 1.4.4. Let there be three edits defined in a questionnaire which has G as its routing graph. Let W_i denote the set of variables involved in edit e_i (i=1,2,3). We have $W_1=\{1,2\}$, $W_2=\{3,4\}$ and $W_3=\{5,7\}$.

Obviously the path (1,2,3,5,7,8) cuts each of the W_i. Edit e_1 defines a constraint on the joint values of variables v_1 and v_2. In fact it limits the values of these variables within the set $R_1 \times R_2$. Similarly e_2 defines a constraint on the joint values of v_2 and v_3 within $R_{12} \times R_3$, and e_3 defines a constraint on the joint values of $v_5 \times v_7$ within $R_5 \times R_7$. Now e_1 and e_2 involve a common variable, namely v_2, and neither e_1 nor e_2 involve a variable which is also involved in e_3. If we consider e_1 and e_2 jointly, we find that it restricts the values of v_1, v_2 and v_3 within the set $R_1 \times R_{23} \times R_3$.

Exhibit 1.4.4 Edits which can be activated simultaneously



Assume that a record r∈REC has activated both e_1 and e_2. There are four possibilities now:

1. Both e_1 and e_2 are satisfied.
2. e_1 is satisfied and e_2 is violated.
3. e_1 is violated and e_2 is satisfied.
4. Both e_1 and e_2 are violated.

The first case need not worry us. The second case seems to indicate that the values for v_1 and v_2 in r are correct but that the value of v_3 is not. Case 3 is similar, but the values of v_2 and v_3 seem to be correct, whereas the value of v_1 seems to be incorrect. In the fourth case the value of v_2 is suspicious. To illustrate how such values can formally be found consider case 3. Because e_1 is violated both v_1 and v_2 are suspicious, and because e_2 is not violated both v_2 and v_3 are not suspicious, i.e. we can write

suspicious: $(v_1 \vee v_2) \wedge \neg(v_2 \vee v_3)$,

which is equivalent to

suspicious: $v_1 \wedge \neg v_2 \wedge \neg v_3$.

It should be remarked that such a reasoning does not always lead to an unambiguous determination of suspicious variables (cf. section 3.4.4).

Of course there is no guarantee that the variables found in this way are the only ones which are suspicious. Nevertheless they are the variables that at least should be assigned a different value in order to create an acceptable record. Of course, it should certainly be possible to obtain a correct record in cases 2, 3 and 4, if we can change all three variables v_1, v_2 and v_3. But this practice may spoil more information in the original record than necessary. In assigning missing values to variables we want to be as parsimonious as possible (cf. chapter 3).

So it is of importance to consider the edits e_1 and e_2 jointly. However, it is of no special interest to consider any other combination of two or more edits simultaneously in this example. ///

Example 1.4.2 shows that in order to understand the edit structure in a questionnaire it is not sufficient to consider only the individual edits but also certain combinations of these edits. Such a combination S of edits is called an *edit cluster*, and has the following two properties:

1. It consists of edits which can be simultaneously activated by records in a certain path π in the routing graph.

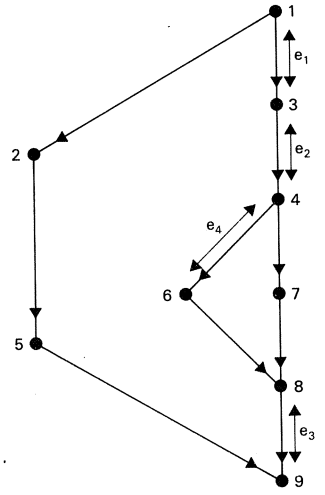
2. It is maximal, in the sense that there are no more edits outside S which can also be activated by records in π.

A *maximal edit cluster* is an edit cluster which is not properly contained in any other edit cluster. Example 1.4.3 illustrates the concepts of edit cluster and maximal edit cluster.

Example 1.4.3 (An edit cluster and a maximal edit cluster)

In exhibit 1.4.5 a routing graph is shown, and the sets of variables of three edits defined in a corresponding questionnaire have been indicated.

Exhibit 1.4.5 Routing graph with four edits



Note that the edits e_1, e_2 and e_3 in exhibit 1.4.5 are always simultaneously activated, either by records in the path π_1=(1,3,4,7,8,9) or by records in the path π_2=(1,3,4,6,8,9). Edit e_4 is only activated by records in π_2. Both $\{e_1, e_2, e_3\}$ and $\{e_1, e_2, e_3, e_4\}$ form edit clusters: the former set is maximal with respect to the records in π_1 and the latter set with respect to the records in π_2. $\{e_1, e_2, e_3, e_4\}$ is also maximal, i.e. not properly contained in any other edit cluster. ///

In order to be able to judge whether the routing and edit structure in a questionnaire are formally correct, we have to formulate a criterion. In fact we shall require that the routing and the edit structure in a questionnaire have the following property, which is of central importance to the theory in this book.

Property 1.4.1 Let Q&E be a q&e-type questionnaire, with a routing graph G. Any path in G contains at least one correct record. Formally: if $(v_{i_1},R_{i_1,i_2})(v_{i_2},R_{i_2,i_3})\ldots(v_{i_k},R_{i_k})$ is a path in G, then there are values $a_{i_j}\in R_{i_j,i_{j+1}}$ for $j=1,\ldots,k-1$ and $a_{i_k}\in R_{i_k}$ such that the record $(v_{i_1},a_{i_1})(v_{i_2},a_{i_2})\ldots(v_{i_k},a_{i_k})$ satisfies the edits in Q&E. ///

This property rules out the possibility that a path in a routing graph cannot actually occur because each record in such a path violates certain edits. We do not want that the edits interfere in this way with the routing structure. This convention provides a criterion for checking the formal correctness of the logical structure of a questionnaire. Furthermore it yields a more conspicuous data editing process, by allowing this to be neatly divided into two parts (if we forget the trivial range checking). The first part checks and fixes the routing structure in a record (if necessary). The second one deals with the edit structure in this record, without any need to adjust the routing structure of a record in this second phase.

Although in many instances we are not especially interested in the form of the edits, it is sometimes important to restrict the attention to special types of edits. The following two types seem to be most common in the context of survey data processing. The edits of the first type shall be called *CP-edits* (CP= cartesian product). A CP-edit can be defined for those variables for which a total ordering has been defined on their corresponding domains. The edit set of a CP-edit is the finite union of cartesian product sets. A CP-edit is said to be *normal*, if its edit set is a cartesian product set. The second type will be called polyhedral. A *polyhedral edit* is defined by a finite set of linear inequalities (in disjunction). The corresponding edit set is the union of a number of half-spaces, each defining a set of unacceptable records. Its complement (in the appropriate set) is a polyhedron, which explains the name of this type of edit. Polyhedral edits can only be defined for those variables such that on their corresponding domains the following structures have been defined:

1. A suitably rich algebraic structure, which includes operations of addition and multiplication (e.g. a field structure).

2. A total ordering.

For simplicity, and because other choices might be inapplicable for survey data processing, we shall assume that the variables involved in polyhedral edits take values in the real numbers or subsets thereof.

2. TESTING THE LOGICAL STRUCTURE OF A QUESTIONNAIRE

2.1 Introduction

After a questionnaire has been constructed on a computer it should be thoroughly tested before it is used in a survey. Of course the first thing to test is whether the questions are correctly specified, e.g. that for each possible answers category a follow-up question is defined, except for the last one, of course. Then the logical structure of the questionnaire should be tested. For q-type questionnaires it is only necessary to check whether the routing structure is described by a routing graph. Testing the correctness of the routing structure is a fairly simple matter. It is more difficult, however, to test a q&e-type questionnaire, as a result of the interplay between the routing and the edit structure. Testing such a q&e-type questionnaire with respect to its logical structure is the subject of the present chapter. We are especially interested in the worst-case computational complexity of several test procedures. As a general reference for matters related to computational complexity, Garey and Johnson (1979) is recommended. The reader is assumed to be, at least casually, acquainted with this theory.

In section 2.2 we consider these test procedures. Testing the formal correctness of the routing structure is a rather simple matter. More difficult is the testing of the correctness of both the routing and edit structure, i.e. the logical structure. As a sample of tests for a q&e-type questionnaire considered in section 2.2 we mention the following: Can all edits be activated? Is the routing structure compatible with the edit structure, i.e. does for any path in the routing graph a record exist which does not violate the activated edits? Are there redundant edits? Is there a path in the routing graph which does not activate any edit? It turns out that most of these problems are computationally intractable. A lesson to be learned from this is that it is vain to believe that the logical structure of any (theoretically possible) questionnaire can be established with 100% certainty.

In section 2.3 we give some general comments on how to cope with the rather formidable computational problems associated with most of the tests presented in section 2.2. This section is certainly not conclusive. In fact, it could be a starting point for a quest for good approximative test procedures, and conditions on the logical structure of questionnaires under which tests become tractable.

2.2 Test procedures

The first check we consider is whether an edit e is activatable, i.e. whether there is a path π in the routing graph which cuts the variables involved in e. Although edits which are not activatable can do no harm in a questionnaire during an interview or when performing data editing, it is nevertheless appropriate that they should be signalled, because they are likely to be a result of a misspecification by the questionnaire designer. In any case they should be either repaired or removed from the questionnaire. To check the activatability of an edit is a fairly simple matter, as the following algorithm shows.

Algorithm 2.2.1 (Activatability check of an edit)

Let G be the routing graph of a q&e-type questionnaire and let A denote its adjacency matrix. Let e be an edit in this questionnaire and let $w_1<\ldots<w_k$ denote the variables involved in e.

Whether e is activated or not is checked as follows.

1. Calculate the transitive closure A^* of A (see appendix A).

2. If $A^*_{w_1,w_2} = \ldots = A^*_{w_{k-1},w_k}=1$ then e is activatable; otherwise it is not. ///

Remark 2.2.1 Algorithm 2.2.1 uses the transitive closure of the adjacency matrix to determine the activatability of an edit. This is clearly not optimal. However, we may assume that the transitive closure A^* is

available, because it is also used to determine other properties or entities pertaining to the logical structure of a questionnaire. ///

A questionnaire designer should want to know whether the combined routing and edit structure in a q&e-type questionnaire is compatible, i.e. whether it obeys the property 1.4.1. We shall call the corresponding testing problem the *logical consistency problem*. A special case of this problem is the *edit cluster problem*. In this problem it is assumed that the routing graph is of an especially simple type, namely a linear digraph.

A questionnaire designer might also be interested to know whether there exist paths in the questionnaire which do not activate any edit. If this is the case he might consider introducing extra edits to 'safeguard' those paths as well. The corresponding problem will be referred to as the *Ker(γ)-problem* (see below for the motivation of this name).

Another problem which a questionnaire designer might raise, is whether there are redundant edits, i.e. edits which can be removed from the original set of edits because they do not introduce any extra constraints on the values of the variables in the questionnaire. We shall refer to this as the *redundancy problem*.

In the remainder of the present section we consider these problems and state a conjecture with respect to one other problem, viz. the determination of the number of edit clusters. In order to facilitate the discussion below, we first introduce some notation. Let G be a routing graph of a q&e-type questionnaire Q&E, Π the set of paths in G and F the collection of edits in Q&E. Let 2^F denote the power set of F, i.e. the set of subsets of F. Let $\gamma:\Pi\rightarrow 2^F$ be the *activation map* which assigns to each path π in Π the cluster of edits activated by π, i.e.

$$\gamma(\pi) = \{e\in F \mid W_e \subset \pi\}, \qquad (2.2.1)$$

where W_e is the set of variables involved in e.

A natural question is: what is the number if edit clusters in a

questionnaire, i.e. $|\gamma(\Pi)|$? The following example shows that this number can be large.

Example 2.2.1 (Size of $\gamma(\Pi)$)

Let G=(V,E) with $|V|=n$ be a complete routing graph, i.e. with a maximum number of n(n-1)/2 edges. Suppose that for every edge $(v,w)\in E$ an edit has been specified, involving the variables v and w. Then the possible number of edit combinations that can be simultaneously activated (i.e. $|\gamma(\Pi)|$) equals the number of paths in G. It is easy to show that the number of paths in G equals

$$(I - A)^{-1}_{1,n} = 2^{n-2}, \qquad (2.2.2)$$

where A is the adjacency matrix of G (see also appendix B). In this case the strict upper triangle of A consists of entries equal to 1 and all remaining entries are 0. So (2.2.2) can be astronomically large for even a moderately large number n of variables (or n(n-1)/2 of edits). ///

So before an attempt is initiated to calculate $\gamma(\Pi)$ explicitly, it is important to know its size. Unfortunately the computational complexity of the determination of $|\gamma(\Pi)|$, in case of a polynomially bounded number of edits, is not known to the present author. We issue the following conjecture.

Conjecture 2.2.1: The determination of $|\gamma(\Pi)|$, with a polynomially bounded number of edits, is a #P-complete problem. ///

Remark 2.2.2 As Shmoys (1988) showed the conjecture can be proved to hold true if there is no restriction on the number of edits in the questionnaire. ///

We now consider the above mentioned problems in separate sections.

2.2.1 Logical consistency problem

As example 2.2.1 shows it is an illusion to assume that $\gamma(\Pi)$ can be calculated for an arbitrary q&e-type questionnaire. So we have to be prepared to settle for less, and consider only a (small) sample of edit clusters, each of which is to be checked to satisfy property 1.4.1. (Such a sample can easily be obtained by generating a random path in the routing graph and by considering the edits it activates. A random path can be obtained by defining a Markov chain on the routing graph which starts in the source and proceeds by consecutively selecting a transition at random until the sink is reached.) But even verifying that a single edit cluster, consisting of CP-edits, is satisfiable is NP-complete, as will presently be shown. In fact this means that verifying the logical consistency of the edit and routing structure for a questionnaire with a trivial, i.e. linear, routing graph is already intractable, if P≠NP.

Edit cluster problem for CP-edits

Instance: Let a questionnaire Q&E be given, consisting of n questions, and containing a collection E of CP-edits. Let, as in section 1.4, the collection of records generated by Q&E with a correct routing structure be denoted by REC. Furthermore, let e be an edit cluster in E, consisting of the edits $e_1, \ldots, e_k$.

Question: Is there a record r∈REC which satisfies the edits in e, i.e. such that $\neg e_1(r) \wedge \ldots \wedge \neg e_k(r)$ holds ?

Then the following theorem can be formulated.

Theorem 2.2.1 The edit cluster problem for CP-edits is NP-complete.

Proof It is possible to verify in time polynomial in the number n of questions in Q&E that a record r∈REC satisfies the edits in E. Hence the problem is in NP. We show that it is at least as difficult as the satisfiability problem (SAT), which is known to be NP-complete (see Garey and Johnson, 1979, pp. 39 ff.).

Let the following Boolean expression C in conjunctive normal form be given

$$C = C_1 \wedge \ldots \wedge C_k \ , \tag{2.2.3}$$

where the clauses C_i are of the form

$$C_i = c_{i1} \vee \ldots \vee c_{ik_i} \ , \tag{2.2.4}$$

for i=1,...,k, and where the c_{ij} are elements from the set $\{v_1,\ldots,v_n, \bar{v}_1,\ldots,\bar{v}_n\}$. Suppose that with each pair $\{v_i,\bar{v}_i\}$ a Boolean variable u_i is associated. We assume that the literal v_i is true if and only if u_i is true, and that the literal $\bar{v}_i$ is true if and only if u_i is false (i=1,...,n). The Boolean expression (2.2.3) is satisfied if and only if there is a truth assignment for each u_i, i.e. an assignment of a truth-value (true or false) for each u_i, such that (2.2.3) evaluates to true.

We translate this instance of SAT into an edit cluster problem as follows. Let w_i be a question with domain $R_i=\{0,1\}$ for i=1,..,n. Suppose that the questions are from a questionnaire in which there are only transitions from w_i to w_{i+1}, for i=1,...,n-1. Hence the corresponding routing graph is linear, with source w_1 and sink w_n. For each clause C_i define an edit e_i as follows. With each literal $c_{ij}\in\{v_p,\bar{v}_p\}$ in C_i associate the condition $\delta(c_{ij})$ defined as

$$\delta(c_{ij}) = \begin{cases} w_p = 1 \text{ if } c_{ij} = \bar{v}_p \\ w_p = 0 \text{ if } c_{ij} = v_p \ . \end{cases} \tag{2.2.5}$$

Now associate the following edit e_i with C_i:

$$e_i : \delta(c_{i1}) \wedge \ldots \wedge \delta(c_{ik_i}) \ , \tag{2.2.6}$$

for i=1,...,k. Hence, for any clause C_i there is an edit, viz. e_i, which prohibits $\bar{C}_i$. It can be easily verified that the edits indeed form an edit cluster. Furthermore there exists a record $(w_1,a_1)\ldots(w_n,a_n)$ with

$a_i \in R_i$ that satisfies these edits if and only if C in (2.2.3) is satisfiable. ///

Example 2.2.2 To illustrate the proof of theorem 2.2.1 consider the following Boolean sentence:

$$(\bar{v}_1 \vee v_2) \wedge (\bar{v}_1 \vee v_2 \vee v_3) \wedge (v_1 \vee \bar{v}_2 \vee v_4) \wedge (\bar{v}_2 \vee \bar{v}_3 \vee \bar{v}_4). \quad (2.2.7)$$

Associate a q&e-type questionnaire Q&E with this sentence (2.2.7) as follows. The questions in Q&E are denoted by w_1, w_2, w_3 and w_4. Assume that they all have the domain {0,1}. Suppose furthermore that the only transitions in Q&E are from w_i to w_{i+1} for i=1,2,3. The edits in Q&E are defined as follows:

$$\begin{aligned} e_1 &: w_1=1 \wedge w_2=0, \\ e_2 &: w_1=1 \wedge w_2=0 \wedge w_3=0, \\ e_3 &: w_1=0 \wedge w_2=1 \wedge w_4=0, \\ e_4 &: w_2=1 \wedge w_3=1 \wedge w_4=1. \end{aligned} \quad (2.2.8)$$

Every record generated by Q&E activates the edits in (2.2.8). These edits are satisfied by a record if and only if (2.2.7) is satisfiable. It is easy to verify that the records $(w_1,0)(w_2,0)(w_3,1)(w_4,0)$ and $(w_1,0)(w_2,0)(w_3,1)(w_4,1)$ satisfy (2.2.8). ///

The edit cluster problem for polyhedral edits is defined similarly to this problem for CP-edits. In fact the former problem is a generalization of the latter one. It is easy to see that the edit cluster problem for polyhedral edits is in NP: the verification that a record satisfies these edits can be carried out in time polynomial in the number of variables in the record. From the fact that the edit cluster problem for CP-edits is NP-complete, it follows that the edit cluster problem for polyhedral edits is also NP-complete. So the following theorem can be formulated.

Theorem 2.2.2 The edit cluster problem for polyhedral edits is NP-complete. ///

Because it is not known that the logical consistency problem is in NP, for either CP-edits or polyhedral edits, it follows from theorems 2.2.1 and 2.2.2, and the observation that the edit cluster problem is a special case, that the logical consistency problem is NP-hard for both CP-edits and polyhedral edits. So the following theorem can be formulated.

Theorem 2.2.3 The logical consistency problem for CP-edits and polyhedral edits is NP-hard. ///

2.2.2 Ker(γ)-problem

We shall now consider the problem how to decide whether there is at least one path which does not activate any edit in a questionnaire. In terms of the activation map, defined in (2.2.1), we can also formulate this as the problem to test whether the *kernel* of γ is non-empty, i.e. whether $\mathrm{Ker}(\gamma)=\gamma^{-1}(\emptyset)\neq\emptyset$. This explains the name of the problem ('the Ker(γ)-problem') introduced above. In fact the Ker(γ)-problem is a decision problem of the following type.

Ker(γ)-Problem
Instance: $G=(V,E)$ is a routing graph and $W=\{W_1,\ldots,W_p\}$ is a collection of subsets of V, such that each W_i lies on at least one path in G.
Question: Is there a path in G which does not cut any W_i?

Then the following theorem holds.

Theorem 2.2.4 The Ker(γ)-problem is NP-complete.

Proof For a given instance of the Ker(γ)-problem it is possible to verify in time polynomial in the number $|V|$ of questions that a path in G is in Ker(γ). Hence the problem is in NP. In the remainder of the proof we show that there is a transformation from the satisfiability problem SAT to the Ker(γ)-problem.

Let a Boolean expression C in conjunctive normal form be given as in the

proof of theorem 2.2.1, from which we also borrow the notation and interpretation of the symbols. We construct a routing graph G which corresponds to the Boolean expression C, as follows. With each clause C_i we associate a layer of vertices L_i in G, i=1,...,k. With each literal c_{ij} appearing in C_i we associate a vertex v_{ij}. Furthermore we add a source which is connected to every vertex in layer L_1 (and to no other vertices), and also a sink to which every vertex of layer L_k is connected (and to no other vertices). The vertices in each layer are connected by edges to each vertex in the layer with the next-higher index (if any). These are the only edges in G.

Let a hypothetical questionnaire with G as its routing graph be given. Assume that the edits in this questionnaire have the following property. For every pair of vertices in G which

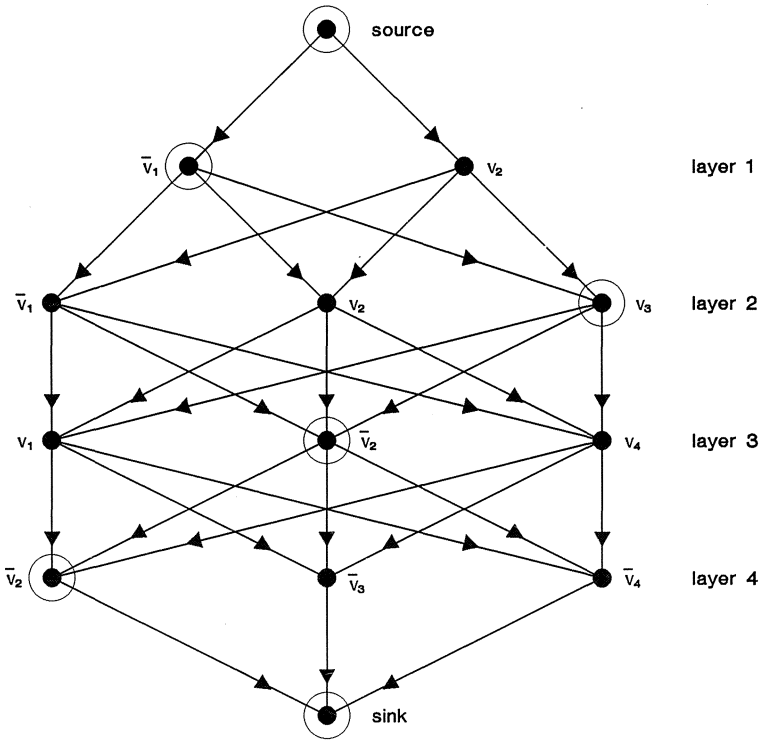
1. belong to different layers, and

2. are labeled by literals corresponding to the same Boolean variable u_i, but with opposite truth values, i.e. v_i and $\bar{v}_i$,

an edit is defined, and there are no other edits. Then it is easy to verify that there is a path π in G which does not activate any edit, i.e. $\pi \in Ker(\gamma)$, if and only if C is satisfiable. ///

Example 2.2.3 To illustrate the proof of theorem 2.2.4 consider the Boolean sentence (2.2.7), and associate the routing graph in exhibit 2.2.1 with this sentence.

Exhibit 2.2.1 A routing graph with a path in Ker(γ)



Assume that this routing graph is from a q&e-type questionnaire in which edits are defined for each pair $\{v_i, \bar{v}_i\}$ of questions from different layers. Then a solution to (2.2.7) corresponds to a path in this routing graph which does not activate any edit, and vice versa. The encircled vertices in the routing graph in exhibit 2.2.2 define such a path. ///

2.2.3 Redundancy problem

The final problem we consider in section 2.2 is the redundancy problem, which we shall first describe more formally for CP-edits. Let Q&E be a q&e-type questionnaire with a set $E=\{e_1,\ldots,e_p\}$ of edits. An edit $e \in E$ is *redundant* if there exist edits $e_{j_1},\ldots,e_{j_k} \in E\backslash\{e\}$, such that e, $e_{j_1},\ldots,e_{j_k}$ are <u>always</u> simultaneously activated and furthermore such that holds

$$e(r) \Rightarrow e_{j_1}(r) \vee \ldots \vee e_{j_k}(r) \ , \tag{2.2.9}$$

for all records r∈REC which activate e, $e_{j_1},\ldots,e_{j_k}$ simultaneously. We can now formally define the redundancy problem for CP-edits.

Redundancy problem for CP-edits

Instance: Let Q&E be a q&e-type questionnaire in which a set $E=\{e_1,\ldots,e_p\}$ of CP-edits is defined.

Question: Does E contain a redundant edit?

Remark 2.2.3 A redundant edit can, as its name already suggests, be removed from a set of edits defined in a questionnaire, without affecting the logical structure. Of course it can also be absorbed into the composite edit to the right of the implication arrow in (2.2.9). ///

The redundancy problem can be split into two subproblems. The first one is to find, in a given set of edits, those edits, which are always simultaneously activated. The second one is to check whether there is a subset of these edits, for which the edits contained in it obey a Boolean expression such as (2.2.9). We shall study these subproblems more closely now, starting with the subproblem on simultaneous activatability.

On the set of edits of a q&e-type questionnaire we can define a digraph structure which indicates which edits always activate each other. This digraph structure will be called a *dominator graph*. It is defined as follows. Let W_1 and W_2 be sets of variables involved in edits e_1 and e_2 respectively. We shall say that e_1 *dominates* e_2, in notation $e_1 \rightarrow e_2$, if any path π cutting W_1 also cuts W_2, i.e.

$$\begin{aligned} e_1 \rightarrow e_2 &\equiv \text{for all paths } \pi\in\Pi\ [W_1 \subset \pi \Rightarrow W_2 \subset \pi] \\ &\equiv \text{for all paths } \pi\in\Pi\ [e_1 \in \gamma(\pi) \Rightarrow e_2 \in \gamma(\pi)]. \end{aligned} \tag{2.2.10}$$

Alternatively expressed, $e_1 \rightarrow e_2$ means that if e_1 is activated by a record

r∈REC then so is e_2. If e_1 dominates e_2 then (e_1,e_2) is an edge in the dominator graph, and there are no other edges in it. Note that the domination relation is reflexive and transitive but not (necessarily) symmetric. The following algorithm can be used to calculate a dominator graph.

Algorithm 2.2.2 (Calculation of a dominator graph)

Let G be the routing graph of a q&e-type questionnaire Q&E, with adjacency matrix A. Let A^* denote the transitive closure of A. Let $E=\{e_1,\ldots,e_p\}$ be the set of edits in Q&E and let $W_i=\{v_{i_1},\ldots,v_{i_{k_i}}\}$ with $v_{i_1}<\ldots<v_{i_{k_i}}$ denote the set of variables involved in e_i $(i=1,\ldots,p)$.

The edges of the dominator graph for E are calculated by carrying out the following steps for each pair W_i, W_j:

1. Consider the set S_i of slices $\{([1,v_{i_1}]), ([v_{i_1},v_{i_2}]),\ldots, ([v_{i_{k_i-1}},v_{i_{k_i}}]),([v_{i_{k_i}},n])\}$ generated by W_i, and determine the subset S_{ij} of slices which contain at least one element of W_j.

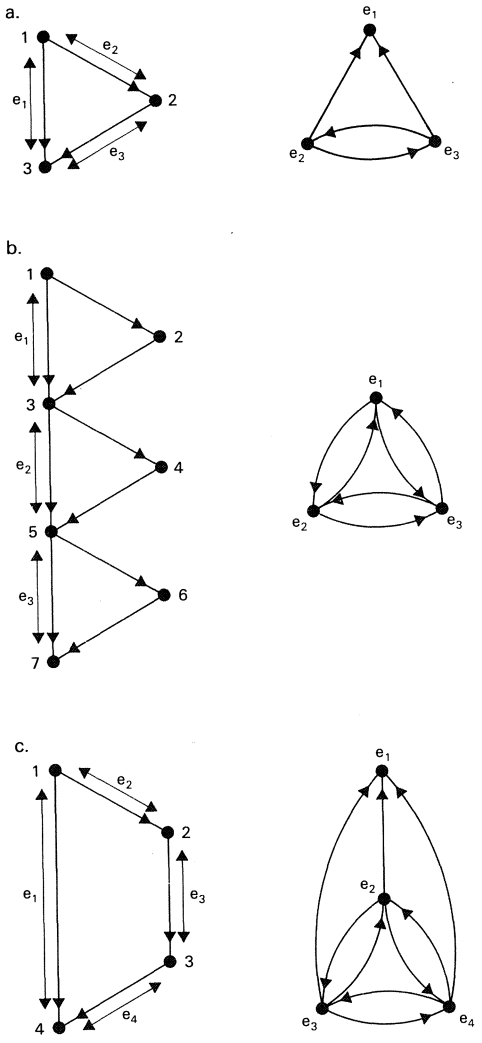
2. Compute the segments of the slices in S_{ij} from A^* as indicated in section 1.3.

3. Check for each slice $\sigma\in S_{ij}$ whether all elements of W_j contained in it are cut-points of the segment corresponding to σ (see appendix A for an algorithm to calculate cut-points in a routing graph). If this holds then $e_i\rightarrow e_j$; otherwise this relation does not hold.

4. To show whether $e_j\rightarrow e_i$ or not, repeat all steps above, after having reversed the roles of W_i and W_j. ///

To illustrate the concept of a dominator graph we provide some examples in exhibit 2.2.2.

Exhibit 2.2.2 Examples of routing graphs plus edits and their dominator graphs



A dominator graph D can be used to find all sets of edits in a given q&e-type questionnaire which are always simultaneously activated. This can be done by calculating its so-called strong components. A *strong component* S in D is a maximal set of vertices in D such that for any two vertices v,w∈S there is a path in S from v to w. The strong components in D can be determined by applying depth-first searches to D (see appendix A for

details). Note that the strong components in a dominator graph consist precisely of the edits which are always simultaneously activated.

The sets of edits in a questionnaire that are always simultaneously activated, can be calculated in time polynomial in the problem size, i.e. the number of edits in the questionnaire.

We now come to the second subproblem, viz. the problem to check whether there are edits which are always simultaneously activated and which satisfy an expression such as (2.2.9). It is clear that we can restrict our attention to the edits in a strong component of the dominator graph corresponding to the questionnaire. Instead of considering the redundancy problem, we consider the opposite problem, viz. the irredundancy problem for these edits. The reason is that this problem is evidently a member of NP, because a possible solution can be verified to be a veritable solution in time polynomial in the problem size. More specifically, we consider the following problem.

Restricted irredundancy problem for CP-edits

Instance: Let Q&E be a q&e-type questionnaire, and let a set $E=\{e_1,\ldots,e_p\}$ of CP-edits form a strong component in the corresponding dominator graph.

Question: Is e_1 irredundant with respect to $\{e_2,\ldots,e_p\}$, i.e. is there at least one $r \in REC$ which activates the edits in E and is such that

$$e_1(r) \Rightarrow e_2(r) \vee \ldots \vee e_p(r) \ , \qquad (2.2.11)$$

does not hold? ///

The following theorem can be proved.

Theorem 2.2.5 The restricted irredundancy problem for CP-edits is NP-complete.

Proof Note that the negation of (2.2.11) is equivalent to

$$e_1(r) \wedge \neg e_2(r) \wedge \ldots \wedge \neg e_p(r). \qquad (2.2.12)$$

By a polynomial transformation from SAT, similar to that employed in the proof of theorem 2.2.1, the NP-completeness of the restricted irredundancy problem is established. The only difference is that e_1 is defined as $\neg e_1$ in theorem 2.2.1. (Note that this is also a CP-edit, although not a normal CP-edit.) ///

It is clear that the decision problem as to whether any edit in a set E of CP-edits, forming a strong component in a dominator graph, is irredundant (in the sense of the restricted irredundancy problem) is also NP-complete. It is only necessary to check for each edit in E whether it is irredundant with respect to the remaining edits in E.

Remark 2.2.4 Let E be a set of CP-edits which all involve the same two variables. Clearly the edits in E are always simultaneously activated. It follows from Lodi et al. (1979, Algorithm A1 in section 5) that for E the restricted irredundancy problem can be solved in time polynomial in the number of edits in E. In addition, this algorithm A1 yields a set of edits equivalent to E, but without any redundant edits. ///

In view of the preceding discussion the following theorem can be formulated.

Theorem 2.2.6 The redundancy problem for CP-edits is co-NP-complete. ///

The restricted irredundancy problem for polyhedral edits is similarly formulated as that for CP-edits. It is straightforward to see that the former problem is in NP. Furthermore since the restricted irredundancy problem for CP-edits is NP-complete, it follows that the corresponding problem for polyhedral edits is also NP-complete. The search for strong components is independent of the form of the edits. Hence we can formulate the following theorem, which also concludes this section.

Theorem 2.2.7 The redundancy problem for polyhedral edits is co-NP complete. ///

2.3 Testing in practice

In view of the results obtained in the preceding section, we consider in this section some of their implications with respect to the testing of questionnaires in practice. It should be stressed in the first place that the computational complexity of the decision problems considered there are worst-case results. It is very well possible that there exist algorithms to solve them, which perform rather well in practice, although they are known to be inefficient in some instances. The simplex algorithm used in linear programming provides an outstanding example of this situation. The moral is that, in order to judge the efficiency of an algorithm, it is not realistic to consider worst-case results only. The performance of an algorithm on an 'average' problem instance of a certain problem type is of importance as well. However, a worst-case result is a guarantee for the performance of an algorithm, whereas an 'average case' result is not.

Many results obtained in the previous section teach us that we should not expect to be able to test the logical structure of a questionnaire for 100%. Maybe we should view this testing problem more as a problem in statistical quality control applied to a complex apparatus. Instead of demanding a 100% guarantee as to its flawless behaviour, we should be more modest and settle for probabilistic reassurance rather than absolute certainty.

For instance, if it is necessary to test whether a property holds for all paths in a routing graph, it may be practically impossible to carry out this test for every path, because their number is exceedingly large. Instead one could generate a finite sample of such paths first, and then carry out the test for each path in the sample. (Assuming that this is itself not an intractable problem. If it is, then one might resort to a probabilistic testing procedure in this case as well.)

The results in the preceding section hold for general questionnaires, because there are no restrictions on the logical structure other than the Markovity of the routing structure and the compatibility of the routing and edit structure. It is interesting to study the logical structure of actually used questionnaires in search of such characteristics. It is then perhaps possible to prove that (some of) the test procedures discussed in the preceding section, when limited to such questionnaires, are tractable.

3. DATA EDITING

3.1 Introduction

In the present chapter we consider the problem of checking records and identifying possible errors. An error is understood here as 'not in agreement with the logical structure of the corresponding questionnaire'. The intention is to replace the incorrect values by others which yield a correct record, i.e. one that is in agreement with the logical structure of the questionnaire. The objective in this data editing process is to retain as much information in the original record as possible, and to change the values of those variables which are most suspicious of being in error.

We assume that the error localization process is carried out in three steps, as was already briefly explained in section 0.2: range checks, a routing check and finally edit checks. It should be stressed that this is an approximation method and not an exact one. The method can be described as follows. Let the *admissible range* of a variable consist of its domain plus a special missing value, called the 'routing skip' (cf. section 4.2). (A variable which has been assigned a routing skip is not on the path through the questionnaire that has been taken in the interview.) If a variable in a record is found to have a value outside the admissible range a special missing value is substituted ('missing due to out of range'; cf. section 4.2). If a record is found to violate either the routing or the edit structure, an attempt is made to localize possible errors. After the routing errors have been localized, the record is partially corrected. This means that for each suspect value (including a 'missing due to out of range') an appropriate transition set is substituted, which is chosen in such a way that a record with a correct routing structure is generated.

Because the range checking is a fairly simple matter, we shall not dwell on it any longer. We start our discussion on data editing in section 3.2 with a consideration of the problem of localizing and correcting routing errors. It is noted that the paths in a routing graph of a questionnaire can be viewed as strings recognized by a finite state automaton. This allows us to employ an algorithm due to Wagner (1974) to localize and correct routing errors in records.

In section 3.3 we switch to a topic which is preparatory to the localization of edit errors, viz. error models. Such models provide the tools to calculate probabilities that certain variables in a record have incorrect values, given that this record violates certain edits. These error probabilities can then in turn be used to identify a set of variables with the highest (or more modestly: a high) probability of being in error. Some authors (e.g. Liepins et al., 1982) have formulated the objective of error localization problem somewhat differently. For this formulation a set of nonnegative weights, so-called error weights, on the set variables involved in a collection of edits is required. They call the resulting optimization problem the Minimum Weighted Fields to Impute (MWFI) problem. The MWFI problem is formulated in section 3.4. In section 3.3 some plausible error weights are given. Furthermore it is shown that under certain conditions the formulation of edit error localization using error probabilities is equivalent to error localization using error weights.

In section 3.4 we discuss the problem of localizing edit errors. First we consider the formulation of edit error localization as an optimization problem. In fact we shall consider several formulations of this problem. Although it is clear from this discussion that edit error localization is NP-hard, we nevertheless consider two approaches to solve this problem for questionnaires with CP-edits and such that all answer categories of questions are finite. These approaches stem from Fellegi and Holt (1976), and Garfinkel (1979), and they are slightly adapted to fit into the framework developed in the present book. We conclude section 3.4 with some general observations on the edit error localization problem.

3.2 Localizing and correcting routing errors

Let a questionnaire Q be given containing n questions $v_1,\ldots,v_n$. Let the routing graph of Q be denoted by G=(V,E). Π and REC are the collection of paths in G and the collection of records with a correct routing structure.

Assume that a record $r=(v_{i_1},\alpha_{i_1})....(v_{i_k},\alpha_{i_k})$ generated by Q is subjected to a routing test, where α_{i_j} is either a regular value in the domain of v_{i_j} or a missing value (e.g. a 'missing due to out of range'), which we assume to be represented by the domain R_{i_j}. If r fails the routing test, it is understood that the possible errors are to be localized and corrected, such that an incomplete record r' results which has a correct routing structure, i.e. $r'\in REC$. Then r is said to be *partially corrected*, and the resulting record r' is a *partially corrected record*. The question is: How could we proceed to calculate such a record r'?

First of all it should be recalled from section 1.3 when a record r has a correct routing structure. Recalling the conditions given in section 1.3, we can say that r has a correct routing structure if the first pair in r corresponds to the source in G, the last pair to the sink in G and every value (or transition set) associated with a variable implies the correct transition. In other words, $r=(v_{i_1},\alpha_{i_1})...(v_{i_k},\alpha_{i_k})$ is a record with a correct routing structure, if the following conditions hold:

1. v_{i_1} is the first question in the questionnaire.
2. v_{i_k} is the last question in the questionnaire.
3. Either $\alpha_{i_j}=R_{i_j,i_{j+1}}$ or $\alpha_{i_j}\in R_{i_j,i_{j+1}}$, for i=1,...,k-1 if k>0. (3.2.1)
4. Either $\alpha_{i_k}\in R_{i_k}$ or $\alpha_{i_k}=R_{i_k}$

If in addition every value in r is regular then r is a complete record.

Next we observe that for the routing structure the exact value of a regular value α_i appearing in a pair (v_i,α_i) is unimportant. What is important is the transition set R_{ij} to which α_i belongs. Therefore we map r into the uniquely defined object τ_r, which we shall call a *string*, and which is obtained from the incomplete record r by replacing each regular value appearing in r by its corresponding transition set. Values which are domains remain unchanged.

Next, we can make the observation that the path set Π in G can be viewed as a so-called regular language (cf. Hopcroft and Ullman, 1969, section 3 for a discussion of regular languages). Characteristic for a regular

language is that its elements, consisting of certain strings of symbols from a finite alphabet, can be recognized by a finite state automaton (FSA). Starting in its initial state FSA reads the symbols in a string one-by-one, from left to right, and determines a state where to jump to, immediately after a symbol has been read. FSA may either remain in the same state or jump to another one. Whether a jump to another state occurs depends on the symbol last read and the state in which FSA is at that moment. (The jump behaviour of FSA can be concisely described by a so-called state diagram.) If by the time FSA has read the whole string it is in a special state, called a final state, then the string is accepted as a member of the regular language which FSA is supposed to recognize. Otherwise, FSA is not in a final state and therefore the string is not accepted as an element of this language.

The observation that Π can be viewed as a regular language is in itself not remarkable: any finite set can be viewed as a regular language (cf. Hopcroft and Ullman, 1969, p. 36 theorem 3.7). In our case, however, there is a very natural interpretation of the routing graph G as the state diagram of an FSA recognizing the elements of Π. The interpretation is as follows. The variables $v_1,\ldots,v_n$ in Q (i.e. the vertices in G) are the states in FSA, the source v_1 of G is the initial state of FSA, the sink v_n of G is the unique final state of FSA, the edges in G indicate the possible transitions between states in FSA, and the pairs (v_i,R_{ij}) of variables and transition sets and the pairs (v_i,R_i) of variables and domains form the symbols in the input alphabet I of FSA. Note that $|I|=|V|+|E|$. The jump behaviour of FSA is described as follows. Suppose FSA is in state v_i, and it reads a symbol (v_i,R_{ij}) for some j. In that case it jumps to state v_j. For all other cases, FSA remains in state v_i. Initially, FSA is in state v_1. If FSA has read an input string and it is in the final state v_n, then (and only then) the string is an element of Π, as one easily verifies.

With this observation in mind, we can apply an algorithm due to Wagner (1974), which changes an incorrect string into a string of the regular language, i.e. a path in G, with as few changes as possible. These changes can be classified into the following three types, which are called *elementary repairs*.

1. changing : (v_i, α_i), with $\alpha_i = R_{ij}$ for some j or $\alpha_i = R_i$, is replaced by (v_i, R_{ik}) for some k; if $\alpha_i = R_{ij}$ then $k \neq j$. (3.2.2)
2. inserting: (v_i, R_{ij}) is added for some j.
3. deleting : (v_i, α_i), with $\alpha_i = R_{ij}$ for some j or $\alpha_i = R_i$, is deleted.

Wagner's (1974) algorithm calculates the least number of elementary repairs necessary to transform a record r with an incorrect routing structure into a record $r' \in REC$. It is based on dynamic programming. Below we give a brief description of this algorithm, somewhat adapted to our purposes. For details and proofs the reader is referred to Wagner (1974). In the discussion to follow we shall use a mixture of formal language 'jargon' and of terminology pertaining to questionnaires.

Let F(j,s) denote the minimum number of elementary repairs which have to be applied to the first j symbols in τ_r, in order to obtain a string $\tau_{r'}$ which forces FSA into state s. (Note that it is only necessary to know the first j symbols of $\tau_{r'}$; the remaining symbols are immaterial.) Furthermore let T(t,s,c) be the smallest number of elementary repairs necessary to change the symbol c into a sequence of symbols which forces FSA from state t into state s. Then the following recursion holds for F:

$$F(j,s) = \min_t \left(F(j-1,t) + T(t,s,\tau_r\langle j\rangle)\right), \tag{3.2.3}$$

for $j \geq 1$, where $\tau_r\langle j\rangle$ denotes the j-th symbol in τ_r, and

$$F(0,s) = \begin{cases} 0 & \text{if } s = v_1, \text{ the source of FSA} \\ \infty & \text{otherwise.} \end{cases} \tag{3.2.4}$$

Let $|\tau_r|$ denote the length of the string τ_r, i.e. the number of symbols in τ_r. The idea of Wagner's algorithm is to calculate $F(|\tau_r|, v_n)$ and then, by tracing back, to find the state t which was used to calculate $F(|\tau_r|, v_n)$ from $F(|\tau_r|-1, t)$ in (3.2.3), and then the state u which was used to calculate $F(|\tau_r|-1, t)$ from $F(|\tau_r|-2, u)$ in (3.2.3); etc., until the source

of FSA is reached. In this way a string/path τ_r in the routing graph is constructed backwards.

We obtain a partially corrected record r' from τ_r and r by applying the following replacement policy to the pairs in τ_r: if $(v_i, R_{i,j}) \in \tau_r$ and $(v_i, a_i) \in r$ and $a_i \in R_{i,j}$ then replace $(v_i, R_{i,j})$ by (v_i, a_i) in τ_r. No other replacements are allowed. Let r' be the result of applying all these replacements to τ_r. It is clear that $r' \in REC$. By comparing r' to r it is easy to find the elementary repairs that have been applied to r in order to obtain r'.

In order to calculate F, consider T first. Let P(t,s) denote the length of the shortest sequence of symbols that forces FSA from state t to state s. Define L(t,s,c) for states t and s of FSA and a symbol c as follows:

$$L(t,s,c) = \begin{cases} 1 \text{ if c appears in at least one of the shortest sequences} \\ \quad \text{of symbols that force FSA from state t to state s,} \\ 0 \text{ otherwise.} \end{cases} \qquad (3.2.5)$$

Then we can express T(t,s,c) as follows

$$T(t,s,c) = \begin{cases} 1 \text{ if } P(t,s)=0 \\ P(t,s) - L(t,s,c) \text{ otherwise.} \end{cases} \qquad (3.2.6)$$

In (3.2.5) and (3.2.6) we need only consider states t, s such that there is at least one path from t to s.

Now P(t,s) and L(t,s,c) can for instance be determined by a slightly adapted algorithm of Floyd, which calculates the shortest distances between all pairs of vertices in G (cf. appendix A). Another algorithm due to Wagner (1976) can also be useful in case G has relatively few edges.

It should be noted that the information on the function T needs to be calculated and stored in the memory of a computer only once. In fact we should calculate and store the following information with respect to T. For every pair of variables/states t and s, such that there is a path from t to s, and for every transition set $R_{t,u} \subset R_t$ as well as for each domain R_t,

we should calculate and retain in random-access memory the T-table consisting of quadruples $(t,s,c',T(t,s,(t,c'))$, for $(t,s)\in E^*$ and $c'=R_{t,u}$ or $c'=R_t$. This amounts to at most $4*|E^*|*|I|$ numbers, where $|E^*|$ is the number of edges in the transitive closure of G. Because $|E^*|\leq n(n-1)/2$ and $|I|=|V|+|E|=n+|E|$, we find that this number does not exceed $2*n(n-1)*(n+|E|)$. Hence in the worst case, when $|E|=n(n-1)/2$, we find that there are at most of the order n^4 numbers required. Likewise, one can find that the time required for building the T-table is at most of the order n^4 time units. Application of Wagner's algorithm to a string τ_r requires

1. The calculation of $F(|\tau_r|,v_n)$. This can be done by consulting the T-table (which is assumed to reside in random-access memory) at most $|E^*|$ times for any symbol in τ_r. Hence the total amount of time for the calculation of $F(|\tau_r|,v_n)$ is at most of the order $|\tau_r|*|E^*|$ time units, which, in turn, is at most of the order n^3 time units.

2. The repairing of the string, using the above-mentioned backtracing procedure. This does not exceed of the order $|\tau_r|*|V|$ time units, which is at most of the order n^2 time units.

Wagner's algorithm applied to a string τ_r requires the F-values to be stored, i.e. an amount of numbers at most in the order of $|\tau_r|*|V|$; this is at most in the order of n^2 numbers. Hence we can conclude that Wagner's algorithm has a polynomial space and time complexity.

Remark 3.2.1 The routing graph G may have non-trivial cut-points, i.e. cut-points which are neither equal to its source nor its sink. In that case it is possible to decompose the routing graph into several routing sub-graphs. Accordingly, a record r can be split into several substrings, each of which can then be independently processed, using the procedure described above. If the necessary equipment were available, this processing might even be carried out in parallel. ///

3.3 Edit error models

In this section we consider error models. Such models can be used to derive the probabilities that variables, each of which is involved in at least one edit from a collection of violated edits, are in error. These error probabilities can be used to find the most probable set of variables with incorrect values, for a given faulty record. Instead of working with error probabilities, it is also possible to use error weights. In a sense these weights are inversely proportional to error probabilities: the larger the weight associated with a variable, the smaller the probability that it is in error. The reason for the introduction of error weights is that they are sometimes more convenient in formulating the error localization problem for edits.

Fellegi and Holt (1976) assume that all variables have the same error weights although they hint at the possibility of using unequal error weights (Fellegi and Holt, 1976, section 7). We remark at the outset that these error models are not limited to the Fellegi-Holt approach, but are useful in any approach to data editing.

Initial work on error models for data editing (not to be confused with outlier detection) stems from Naus et al. (1972). Further developments can be found in e.g. Liepins and Pack (1980, 1981), Liepins (1980), Liepins et al. (1982). In the latter two works it is shown that the assumption of independency of variabes being in error, leads to the MWFI formulation of error localization, which uses error weights.

3.3.1 Error probabilities

We start our discussion with an example of Naus et al. (1972), after having introduced some necessary notation. Let $r=(v_1,a_1)\ldots(v_n,a_n)$ be a record with values in the code space $R_1\times\ldots\times R_n$. In the example and subsequent discussion we shall assume that r is a realization of a multidimensional random record $\underline{r} = (v_1,\underline{a}_1)\ldots(v_n,\underline{a}_n)$. We define the random variable $\underline{\varepsilon}_j$ as follows

$$\underline{\varepsilon}_j = \begin{cases} 1 \text{ if variable } v_j \text{ is in error} \\ 0 \text{ otherwise .} \end{cases} \tag{3.3.1}$$

Then define the following events

$$A_j = [\underline{\varepsilon}_j = 1] \quad \text{and} \quad B_j = [\underline{\varepsilon}_j = 0] \ . \tag{3.3.2}$$

Example 3.3.1 (Comparing $P[A_1 | A_1 \cup A_2]$ and $P[A_1 | \underline{r} \in E]$)

Assume that we have records with only two variables, i.e. $n=2$, and that $R_1 = R_2 = [0,1]$. A record $\underline{r} = (v_1, \underline{a}_1)(v_2, \underline{a}_2)$ can be viewed as a random variable with values in $R_1 \times R_2$. Let e denote an edit and $E \subset R_1 \times R_2$ the corresponding edit set. Suppose that A_1 and A_2 are independent events, with $P[A_i] = p_i$ and $P[B_i] = 1-p_i = q_i$, for $i=1,2$. It trivially holds that

$$P[A_1 \cup A_2 | \underline{r} \in E] = 1 \tag{3.3.3}$$

One might perhaps be tempted to think that the conditional probabilities $P[A_1 | A_1 \cup A_2]$ and $P[A_1 | \underline{r} \in E]$ are equal. But this is in general not the case as the following reasoning shows. It is easy to verify that holds

$$P[A_1 | A_1 \cup A_2] = \frac{p_1}{p_1 + q_1 p_2} \ . \tag{3.3.4}$$

Now suppose that the following conditional probabilities hold

$$\begin{aligned} &P[\underline{r} \in E | B_1 \cap B_2] = 0 \ , \\ &P[\underline{r} \in E | A_1 \cap B_2] = 1/3 \ , \\ &P[\underline{r} \in E | B_1 \cap A_2] = 1/3 \ , \\ &P[\underline{r} \in E | A_1 \cap A_2] = 1/2 \ . \end{aligned} \tag{3.3.5}$$

From (3.3.5) we can calculate with the help of Bayes' theorem, using the independence of A_1 and A_2, that

$$P[A_1 | \underline{r} \in E] = \frac{(1/2)p_1p_2 + (1/3)p_1q_2}{(1/2)p_1p_2 + (1/3)p_1q_2 + (1/3)q_1p_2} \quad . \tag{3.3.6}$$

Hence in this case $P[A_1 | A_1 \cup A_2] \neq P[A_1 | \underline{r} \in E]$ in general. ///

Although we may assume that (3.3.4) and (3.3.6) are different in general, Naus et al. (1972) use conditional distributions like (3.3.4) to approximate (3.3.6). They motivate this choice in the appendix of their article.

The idea behind the approach of Naus et al. (1972) is to express certain conditional distributions in terms of so-called *error rates*, i.e. joint distributions of certain events A_i. In practice it may be impossible to estimate all required error rates because they are too numerous, in view of the sample from which they have to be estimated.

Assume that we have a record r which violates two edits e and f. Let V and W denote the respective sets of variables involved in e and f. Assume that variable u is in $V \cup W$. We shall give an expression for the probability that u is in error, given the circumstance that at least one variable in V is in error and at least one variable in W is in error, i.e for

$$P[A_u | (\cup_{i \in V} A_i) \cap (\cup_{j \in W} A_j)] \quad . \tag{3.3.7}$$

It is easy to verify that (3.3.7) can be expressed as

$$\frac{P[A_u] + P[S] + P[T] - P[S \cup T] - P[A_u \cup S] - P[A_u \cup T] + P[A_u \cup S \cup T]}{P[S] + P[T] - P[S \cup T]} \quad , \tag{3.3.8}$$

where $S = \cup_{i \in V} A_i$ and $T = \cup_{i \in W} A_i$. Each of the terms occurring in (3.3.8) can be expressed as sums of error rates by using the method of inclusion and exclusion. In this case, however, it would be more convenient to work with 'correctness rates' instead of with error rates, where we define correctness rates as simultaneous densities of the events B_i, which are given in (3.3.2). In case $u \in V \cap W$, (3.3.8) reduces to

$$\frac{P[A_u]}{P[S] + P[T] - P[S \cup T]}\ , \tag{3.3.9}$$

because $A_u \cup S = S$ and $A_u \cup T = T$. In case $u \in V \setminus W$, we can express (3.3.8) as

$$\frac{PA_u + PT - P[A_u \cup T]}{PS + PT - P[S \cup T]}\ , \tag{3.3.10}$$

because $A_u \cup S = S$.

In Naus et al.(1972) the situation where two edits are violated is called the *two-test case*. It can be generalized to the k-test case, where k edits are assumed to be violated. The equivalent expressions of (3.3.8) look cumbersome for the k-test case. Matters simplify considerably, however, if the events A_i are assumed to be mutually independent. It is, however, doubtful whether this is a realistic assumption. The error models under the independence assumption are called *independence models* by Naus et al. (1972).

3.3.2 Error weights

Instead of error probabilities it is also possible to introduce *error weights* to indicate the error proneness of values in a record being in error. In fact these weight can be introduced independently of any error model. However, as Liepins and Pack (1980, 1981), Liepins (1980) and Liepins et al. (1982) have shown, under some conditions independence models are equivalent to a model based on an MWFI formulation of error localization. In fact this result is easy to derive. Let V denote the set of variables in a record with regular (i.e. nonmissing) values and let W be an arbitrary subset thereof. Assuming an independence model, the probability p_W that precisely the variables in W are incorrect, is given by

$$p_W = \prod_{i \in W} p_i \prod_{i \in V \setminus W} (1 - p_i)$$

$$= \prod_{i \in V} (1 - p_i) \prod_{i \in W} \frac{p_i}{1 - p_i} \, . \qquad (3.3.11)$$

This shows that trying to find a set W which maximizes p_W is equivalent to searching for a set W which minimizes

$$\sum_{i \in W} \{\log (1 - p_i) - \log p_i\} \, , \qquad (3.3.12)$$

because the expression in (3.3.12) is the negative logarithm (with respect to some base) of the second expression on the right-hand side of (3.3.11), without the first product, which is independent of W. In order to obtain an MWFI formulation, each term in (3.3.12) should be nonnegative, which is equivalent to the requirement that $p_i < 1/2$ for $i \in V$.

In Liepins and Pack (1980) three alternative series of error weights c_i ($i \in V$) are suggested. Their performances in imputation processes were investigated in a simulation study carried out by the authors. The alternatives considered are

1) $c_i = 1$.

2) $c_i = 1/(1 + \#$ times variable i is involved in violated edits) .

3) $$c_i = \frac{(1 + \#\text{ times variable i is involved in satisfied edits})}{(1 + \#\text{ times variable i is involved in violated edits})} \, .$$

(3.3.13)

As a general conclusion from their simulation study it is reported that method 3 was an improvement over both methods 1 and 2, both of which performed similarly. The authors hasten to add, nevertheless, that these results can hardly be interpreted as conclusive.

In Liepins and Pack (1981) it is remarked that simulations have shown that if the error rate is low, an MWFI approach can be expected to perform

successfully. In Liepins and Pack (1980, 1981) there are some results presented, derived from simulations, showing the inverse dependency of the average error probability and the average number of 'correct' imputations ('correct' in the sense that the original record and the imputed record are identical).

3.4 Localizing edit errors

In section 3.4.1 we discuss several formulations of the edit error localization problem, and consider their respective computational complexities. In fact it is shown that edit error localization is probably intractable, i.e. NP-complete or NP-hard, under any of the formulations presented there.

In sections 3.4.2 and 3.4.3 we present two algorithms for CP-edit error localization, in case all domains of the variables are finite. These algorithms have been proposed by Fellegi and Holt (1976) and Garfinkel (1979), but are adapted so as to fit into the framework of the present book. In view of the results in section 3.4.1 these algorithms cannot be efficient but they are interesting enough to be considered.

In section 3.4.4 we discuss a probabilistic method to verify whether an incomplete record is *repairable*, i.e. can be replaced by a complete record by substituting regular values for the missing ones. Furthermore we introduce a 'logical' method to localize edit errors in a faulty record. This latter approach can be viewed as the 'logical' counterpart of the method of Naus et al. (1972) to construct error models.

3.4.1 Formulations of the problem

The localization of edit errors in records can be defined as an optimization problem in several ways. We shall present two basic formulations, one based on error weights and the other on error probabilities. From these basic formulations it is possible to derive some other ones, which yield only approximative solutions when viewed as

solutions of a problem in one of the basic formulations.

The intuition behind both basic formulations is the same. They only differ in the use of either error weights or error probabilities. This basic intuition is the following. Let a (possibly partially corrected) record r be given that activates a subset E_r of the set E of edits. Remember that r can only activate an edit e in E if the variables involved in e are present in r. Suppose furthermore that r violates a subset E_{rv} of E_r. We should like to replace r by a partially corrected record r' such that

1. r' idles all edits in E_{rv}.
2. r' is repairable.
3. r' has the same routing structure as r.
4. r' differs from r in as few variables as possible.
5. The variables for which r and r' have different values, are suspicious to have incorrect values.

We briefly comment on these points. The first requirement is very natural because otherwise there would still be an inacceptable combination of values in r'. In view of point 3 it is sufficient to replace the (regular) values of suspicious variables in r by the transition set to which they belong. Such a transition set can be interpreted as a missing value ('missing due to edit violation'). Points 2 and 3 are conditions appealing to the logical structure of the corresponding questionnaire. Recall that property 1.4.1 guarantees the existence of such an r': replacing all regular values of variables appearing in the record by missing ones, i.e. by the transition sets to which the corresponding regular values belong yields a path, and there is at least one record in this path which satisfies all edits in E_r. (In view of the results in chapter 2 the required property 1.4.1 may be impossible to verify for a particular questionnaire, because the number of edit clusters can be exceedingly large.) To find such an r' is another matter (cf. theorem 3.4.1 and theorem 3.4.2 below). Point 4 expresses the intention to retain as much information in r as possible. This seems a natural requirement. Point 5 assumes that error probabilities or error weights have been calculated. If this is not

the case, however, then it is sometimes implicitly assumed that all variables in r which are also involved in the edits in E_r, have an equal probability of being in error. For some formulations of edit error localization it is necessary to combine conditions 4 and 5 into one requirement.

Before we proceed, we give an example which shows that condition 2 above, i.e. with respect to the repairability of r', is a condition which cannot be discarded.

Example 3.4.1 (Non-repairable record)

Let a record contain 6 variables, denoted by $v_1,\ldots,v_6$. We assume that the corresponding questions occur in this order in a questionnaire. Let R_i be the domain of v_i, $1\leq i\leq 6$. Suppose we have $R_1=R_3=\{0,1\}$, $R_2=R_5=\{0,1,2\}$, $R_4=R_6=\{0,1,2,3\}$. Suppose furthermore that five edits e_i have been defined, to which the following edit sets E_i correspond:

$$\begin{array}{llllll} E_1=R_1 & \times\{0,1\} & \times\{0\} & \times R_4 & \times\{0,1\} & \times R_6 \\ E_2=\{1\} & \times R_2 & \times\{1\} & \times\{0,1\} & \times R_5 & \times\{2,3\} \\ E_3=\{0\} & \times\{1,2\} & \times R_3 & \times\{1,2,3\} & \times R_5 & \times R_6 \\ E_4=R_1 & \times\{0,2\} & \times R_3 & \times R_4 & \times R_5 & \times\{0,1\} \\ E_5=\{1\} & \times R_2 & \times R_3 & \times\{0\} & \times\{1,2\} & \times R_6 \end{array} \qquad (3.4.1)$$

Suppose that we have the record $r=(v_1,1)(v_2,0)(v_3,0)(v_4,0)(v_5,1)(v_6,0)$. This record activates all five edits in (3.4.1). An inspection shows that it violates edits e_1, e_4 and e_5, or, alternatively expressed, $r\in E_1\cap E_4\cap E_5$. Evidently there is no single variable which idles all violated edits. It is easy to verify that the variables v_2 or v_5 are involved in the violated edits. However, it is impossible to impute values from R_2 and R_5 for v_2 and v_5 respectively in order to obtain a correct record. ///

In order to give the first basic formulation of edit error localization, we first introduce some notation. Let r be a record which has activated a set E_r of edits and which violates the edits in a subset E_{rv} thereof. Let $e_{r,i}$ denote an edit in E_r, for $i=1,\ldots,k$, and $e_{rv,i}$ an edit in E_{rv}, for

$i=1,\ldots,m$. Furthermore assume that $W_{r,i}$ and $W_{rv,i}$ are the sets of variables involved in edits $e_{r,i}$ and $e_{rv,i}$ respectively. Let $W_r=\cup_{i=1}^{k} W_{r,i}$ and $W_{rv}=\cup_{i=1}^{m} W_{rv,i}$ denote the sets of variables involved in the edits in E_r and E_{rv} respectively. Let F_{rv} denote the edit set corresponding to the edits in E_{rv}. Using the notation established in section 3.3, we can formulate edit error localization as the problem to find a subset $K\subset W_{rv}$ such that $K\cap W_{rv,i}\neq\emptyset$, for $i=1,\ldots,m$, and such that

$$P[\cap_{j\in K} A_j \cap \cap_{j\in W_r\setminus K} B_j \,|\, r\in F_{rv}] \qquad (3.4.2)$$

has a maximum value (interpreting r as a randomly drawn record), and such that r', which is the partially corrected record, obtained from r by replacing the values of the variables in K by missing values (i.e. appropriate transition sets) is repairable. In order to facilitate computations, however, we apply the same approximation to (3.4.2) as was given by Naus et al. (1972), and which was also used in section 3.3. We then find that, instead of (3.4.2), we should identify a subset K as above, such that the conditional probability

$$P[\cap_{j\in K} A_j \cap \cap_{j\in W_r\setminus K} B_j \,|\, \cup_{j\in W_{rv}} A_j] \ , \qquad (3.4.3)$$

is maximized, subject to the additional constraint with respect to the repairability of r'.

Note that checking whether a partially corrected record is repairable for CP-edits is similar to asking whether a Boolean expression is satisfiable. Hence checking the repairability of a partially corrected record is NP-complete (cf. also theorem 2.2.1). It is also true that the decision versions of the optimization problems formulated above for CP-edits are NP-complete. The corresponding optimization problems themselves, however, are NP-hard. It also holds that the decision versions of the optimization problems for polyhedral edits are NP-complete, whereas these optimization problems themselves are NP-hard.

Theorem 3.4.1 Edit error localization formulated in either of two ways above is NP-hard for CP-edits as well as for polyhedral edits. ///

Another basic formulation of the edit error localization problem is the Minimum Weighted Fields to Impute (*MWFI*) formulation, which runs as follows. Assume that with each variable v_i in r an error weight $c_i \in R^+$ is associated. Let c denote the column vector of these values. Let z be the 0-1 column vector with as many components as there are variables in W_{rv}, i.e. $|W_{rv}|$, such that $z_i=1$ if the values of v_i in r and r' differ, and $z_i=0$ otherwise. In case $z_i=1$ the value of v_i in r is regular and it is a transition set in r'. Then MWFI requires to find a vector z (and hence a record r') such that c'z is minimal and such that the record r' corresponding to z obeys conditions 1 and 2 above (condition 3 is automatically satisfied).

We can express this also as follows. Let the m edits in E_{rv}, i.e. the edits which are violated by r, be ordered according to the (induced) natural order on E_{rv} (cf. section 1.4). Let M be the 0-1 matrix of the order $m \times |W_{rv}|$, the *violated edit matrix*, such that $M_{ij}=1$ if variable j is involved in edit i, and $M_{ij}=0$ otherwise. The MWFI problem can then be formulated as follows. Find a vector z such that

c'z is minimal

subject to: 1. $Mz \geq 1$ (3.4.4)
2. r' is repairable

Without this second condition MWFI would be a set covering problem (see e.g. Garfinkel and Nemhauser, 1972, chapter 8). In view of example 3.4.1 this condition cannot be dropped, however. Nevertheless we shall borrow some terminology from this integer programming problem. A feasible solution to (3.4.4) will also be called a *cover*. A *prime cover* is a cover with a minimal number of components equal to 1 (i.e. a local minimum). The set of variables corresponding to the components in a cover equal to 1, are said to *cover off* the violated edits.

The following theorem can be formulated. It can be proved by noting that the MINIMUM COVER problem (cf. Garey and Johnson, 1979, p. 222) is NP-complete.

Theorem 3.4.2 In the MWFI formulation, edit error localization is NP-hard, for both CP-edits and polyhedral edits. ///

It is clear that in the edit error localization problem the repairability of r' (i.e. condition 2 above) is a bottleneck which cannot be circumvented. So any formulation of edit error localization as an optimization problem is NP-hard.

A variant of either formulation of error localization as given above is the following. Assume that the variables in an incorrect record r have been ordered in descending order of their error proneness, i.e. either in descending order of the corresponding error probabilities or in ascending order of the corresponding error weights. Assume that variables in r are successively assigned missing values, starting with the first variable in the sequence, until for the first time we obtain a repairable incomplete record. Of course such a process will terminate, if the logical structure obeys property 1.4.1, which we assume. It is however not necessary that conditions 4 and 5 will be met. Obviously error localization as formulated here is NP-complete.

In sections 3.4.2 and 3.4.3 we present two other formulations of edit error localization, which (at best) yield solutions which are not necessarily optimal in the sense of either of the basic formulations of the edit error localization problem as given above.

3.4.2 The Fellegi-Holt approach

Suppose that a q&e-type questionnaire Q&E is given which only contains questions/variables with finite domains. Furthermore to localize edit errors in a faulty record it is only necessary to consider the variables involved in the activated edits. One of the aims is to idle the violated edits by assigning missing values to one or more variables involved in these edits. As example 3.4.1 shows, pursuing this aim only may not

succeed in producing a repairable partially corrected record. As Fellegi and Holt (1976) have shown, such a record will be produced, however, if the set of edits is a so-called complete set (see below). The basic idea of the Fellegi-Holt approach is therefore to generate a complete set of edits for a set of edits which are activated by any record in a particular path in the routing graph of Q&E. The problem of edit error localization in a faulty record is then reduced by Fellegi and Holt to a set covering problem, i.e. to solving an MWFI problem (cf. 3.4.4) without bothering about the repairability requirement.

For the Fellegi-Holt approach it is important that each CP-edit is written as a disjunction of normal CP-edits, i.e. for which the corresponding edit set is a cartesian product set. In this section we shall therefore assume that all CP-edits are normal, unless explicitly stated otherwise. Furthermore our attention is restricted to the edits in the set E_r which have been activated by r. Let E_{rv} denote the subset of E_r which consists of the edits violated by r.

We now consider collections of edits which guarantee that a cover of the entire collection yields a repairable record. In order to do this we first introduce the concept of a logically implied edit. An edit e is said to be (*logically*) *implied* by the edits e_1 and e_2, if the corresponding edit set E_e of e is derived as follows from the edit sets E_1 and E_2 of e_1 and e_2 respectively. Let $E_i = A_{i1} \times A_{i2} \times \ldots \times A_{in}$ (i=1,2) for non-empty subsets $A_{ij} \subset R_j$ (i=1,2). Now choose an index j and define an edit e with edit set

$$E_e = (A_{11} \cap A_{21}) \times (A_{12} \cap A_{22}) \times \ldots \times (A_{1,j-1} \cap A_{2,j-1}) \times (A_{1j} \cup A_{2j}) \times (A_{1,j+1} \cap A_{2,j+1}) \times \ldots \times (A_{1n} \cap A_{2n}) \ , \qquad (3.4.5)$$

where $A_{1k} \cap A_{2k} \neq \emptyset$ by assumption, for k=1,...,n and k≠j. It is easy to verify that if a record violates e then it also violates e_1 or e_2. The variable j is called the *generating variable* for e. We shall write

$$e = e_1 *_j e_2 \ , \qquad (3.4.6)$$

to indicate that e is logically implied by e_1 and e_2, using variable j as a

generating variable. It is easy to see that $e *_j e = e$ (i.e. idempotency), $e_1 *_j e_2 = e_2 *_j e_1$ (i.e commutativity) for all j, and that, in general, $e_1 *_j (e_2 *_k e_3) \neq (e_1 *_j e_2) *_k e_3$ (i.e. non-transitivity) for $j \neq k$. But $e_1 *_j (e_2 *_j e_3) = (e_1 *_j e_2) *_j e_3$ (i.e. restricted transitivity) for all j.

An implied CP-edit e is called *essentially new* if the range set $A_{1j} \cup A_{2j}$ of the generating variable j equals the domain R_j, and both A_{1j} and A_{2j} are proper subsets of R_j. This means that the j-th variable is not involved in e. In case an implied edit $e = e_1 *_j e_2$ is not essentially new, we may discard it for localization purposes. The reason is that the set of variables involved in e is the union of the sets of variables in e_1 and e_2, and furthermore it holds that a record violates e if and only if it violates e_1 and e_2. However an implied edit which is not essentially new may be of importance to generate an essentially new implied edit. Therefore we cannot discard it when generating a so-called *complete set* of edits. This is a set of CP-edits such that no essentially new implied edits can be derived from it. Generating a complete set of edits may require a time exponential in $|E|*n$, where $|E|$ is the size of the original set of edits and n the number of variables involved in the edits in E.

In example 3.4.2 the concepts of logically implied edits and of essentially new implied edits is illustrated.

Example 3.4.2 (Graphical illustration of implied edits)

Consider the following pairs of edit sets in $R_1 \times R_2 = [0,10] \times [0,10]$, which are plotted in exhibit 3.4.1:

$E_1 = [3,9] \times [2,6]$; $E_3 = [3,10] \times [0,4]$; $E_5 = [3,10] \times [0,5]$;
$E_2 = [1,4] \times [4,9]$; $E_4 = [1,4] \times [2,10]$; $E_6 = [0,5] \times [4,10]$.

Let e_i be the edit corresponding to E_i, for $i=1,\ldots,6$. In exhibit 3.4.2 some edits which are logically implied by the given ones are presented.

In exhibit 3.4.1 the edit sets corresponding to these edits are drawn.

Note that only the last three implied edits are essentially new. ///

Exhibit 3.4.1 Plots of the edit sets $E_1, \ldots, E_6$

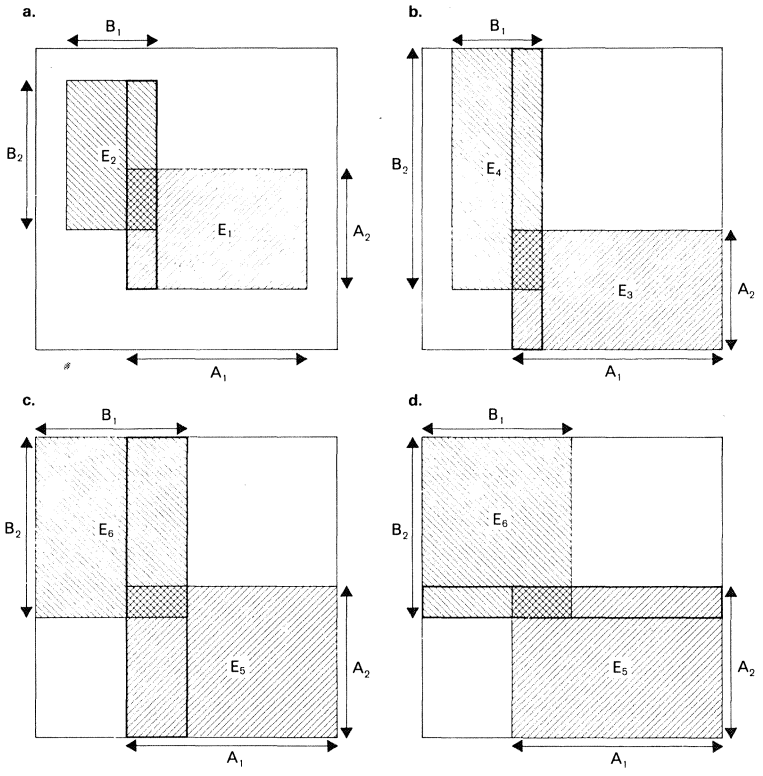


Exhibit 3.4.2 Some logically implied edits

implied edit	edit set	variables involved
$e_1 *_1 e_2$	$[1,9] \times [2,4]$	v_1, v_2
$e_1 *_2 e_2$	$[3,4] \times [2,9]$	v_1, v_2
$e_3 *_1 e_4$	$[1,10] \times [2,4]$	v_1, v_2
$e_3 *_2 e_4$	$[3,4] \times [0,10]$	v_1
$e_5 *_1 e_6$	$[0,10] \times [4,5]$	v_2
$e_5 *_2 e_6$	$[3,5] \times [0,10]$	v_1

Let Ω be a complete set of CP-edits defined for E_r. Furthermore let Ω_k ($1 \leq k \leq n$) be the maximal subset of Ω, containing all edits in Ω in which only the variables $v_1, \ldots, v_k$ are involved. That is, the edit sets corresponding to the edits in Ω_k are of the form: $A_1 \times \ldots \times A_k \times R_{k+1} \times \ldots \times R_n$ where $\emptyset \neq A_i \subset R_i$ ($i=1,\ldots,k$) and R_j is the domain of the j-th variable ($j=1,\ldots,n$). We have $\Omega_1 \subset \ldots \subset \Omega_n = \Omega$. The following theorem holds (cf. Fellegi and Holt, 1976, theorem 1).

Theorem 3.4.3 Let $r=(v_1,a_1)\ldots(v_{k-1},a_{k-1})(v_k,*)\ldots(v_n,*)$, where the *'s are arbitrary values in the appropriate finite domains R_j ($j=k,\ldots,n$), be a record which does not violate the edits in Ω_{k-1}. Then there is a value $a_k \in R_k$ such that $r'(a_k)=(v_1,a_1)\ldots(v_{k-1},a_{k-1})(v_k,a_k)(v_{k+1},*)\ldots(v_n,*)$ does not violate the edits in Ω_k.

Proof Suppose such a value $a_k \in R_k$ does not exist. Then for any value $a_k \in R_k$ there is at least one edit in Ω_k which is violated by $r'(a_k)$. For any $a_k \in R_k$ pick an edit $e(a_k) \in \Omega_k$ which is violated by $r'(a_k)$. A moment's reflection shows that the implied edit $e(a_{k1}) *_k e(a_{k2}) *_k \ldots *_k e(a_{kt})$, where $R_k=\{a_{k1},\ldots,a_{kt}\}$, is well-defined (i.e. the corresponding edit set is nonempty), is violated by $r'(a')$ for any $a' \in R_k$, and is an element in Ω_k. But in fact this implied edit is essentially new, because it does not involve the variable v_k, and therefore it is an element of Ω_{k-1}. But this

implies that the record $(v_1,a_1)\ldots(v_{k-1},a_{k-1})(v_k,*)\ldots(v_n,*)$ violates at least one edit in Ω_{k-1}. This is a contradiction. Therefore such a value $a_k \in R_k$ exists. ///

Remark 3.4.1 The proof of theorem 3.4.3 suggests a sequential approach to edit error localization. Compare this with the approach by Garfinkel as described in section 3.4.3. ///

The following result, an adaption of corollary 2 in Fellegi and Holt (1976, p.24), is basic for the Fellegi-Holt approach.

Corollary to theorem 3.4.3 Let S be a subset of the n variables in a record r. Suppose that the variables in S cover off all edits, in a complete set of edits, which are violated by a record r. Then S yields a repairable record from r, i.e. for each variable in S a regular value can be assigned and substituted so that the resulting record obeys the logical structure of the corresponding questionnaire. ///

Remark 3.4.2 In the formulation of theorem 3.4.3 we assumed that the order in which the variables are considered is the standard order in which they appear in the record. It is, however, possible to assume a different order. For instance if r is an incorrect record violating several edits, an error model could be used to calculate the error probability (or weight) of each variable involved in these edits. By assuming that the variables not involved in any of these edits have error probability 0 (or error weight ∞), we can order the variables in r in descending (ascending) order of their corresponding error probabilities (weights). ///

Remark 3.4.3 The clue of the Fellegi-Holt approach to data editing with CP-edits is to generate a complete set of CP-edits from a set of explicitly given ones. In particular the essentially new implied edits among these are of importance. This idea can also be applied to polyhedral edits.

In Fellegi and Holt (1976, section 5.3) a methodology is given to generate a complete set of implied edits from a given set of polyhedral edits. In order to define the concept of a complete set of polyhedral

edits, we must first define what an essentially new implied polyhedral edit is. To this end, assume we have quantitative variables $x_1,\ldots,x_k$ and two polyhedral edits e_1 and e_2 given by

$$e_1 : \Sigma_i\ a_i x_i \geq 0$$
$$e_2 : \Sigma_i\ b_i x_i \geq 0, \qquad (3.4.7)$$

for a_i, $b_i \in \mathbf{R}$ $(i=1,\ldots,k)$. We shall say that variable x_i is involved in e.g. e_1 if $a_i \neq 0$. From e_1 and e_2 we can derive a polyhedral edit e_3, which we shall call an *essentially new implied polyhedral edit*, if there is a $j \in \{1,\ldots,k\}$ such that $a_j < 0$ and $b_j > 0$. If we define $c_i = a_i b_j - a_j b_i$ for $i = 1,\ldots,k$ then we have that e_3, defined by

$$e_3 : \Sigma_i\ c_i x_i \geq 0, \qquad (3.4.8)$$

is a polyhedral edit which does not involve variable x_j. We shall say that variable x_j is the generating variable for e_3, and we shall write $e_3 = e_1 *_j e_2$. Note that the generation of an essentially new polyhedral edit is more restricted than its CP counterpart.

By repeatedly applying this operation in (3.4.8) to a given set of polyhedral edits, a complete set of polyhedral edits is eventually generated, which is defined in essentially the same way as for CP-edits.

///

Remark 3.4.4 Note that if we really want to apply the Fellegi-Holt approach in practice, a complete set of edits should be generated for every path in the routing graph which activates at least one edit, i.e. which is not in $\mathrm{Ker}(\gamma)$ (cf. section 2.2). This can clearly be infeasible. The sequential method given in section 3.4.3 seems to be more attractive for practical applications in this respect. This holds also for the procedure referred to in remark 3.4.1. ///

3.4.3 Garfinkel's algorithm

In the present section we assume to be in the same position as in section 3.4.2, i.e. having a q&e-type questionnaire with all questions/variables having finite domains. Assume that the edits in this questionnaire are CP-edits.

The Fellegi-Holt procedure is based on generating a complete set of edits from a given set of edits, prior to error localization. Garfinkel (1979) on the other hand presents a sequential algorithm for error localization. Instead of generating a complete set of edits prior to localizing errors, it sequentially generates implied edits, until a repairable partially corrected record is found. The algorithm, which is based on repeatedly solving a set covering problem, tries to find a solution to the MWFI problem as formulated in (3.4.4). In fact the algorithm yields an imputation for the repairable record as well, but this can be discarded. By renumbering the vertices in r we may assume that they are $v_1,\ldots,v_n$. The algorithm is as follows.

Algorithm 3.4.1 (Sequential error localization)

Let an incorrect record r be given, with violated edit matrix M. Let the i-th edit violated by r have edit set $A_{i1}\times\ldots\times A_{in}$, where A_{jk} is a nonempty subset of the transition set to which the value of variable v_j belongs. Let c be the error weight vector. In order to find a solution to the MWFI problem (3.4.4) proceed as follows.

1. Solve (3.4.4) without condition 2, and denote the solution by z^*.

2. Let $J^*=\{j \mid z_j^*=1\}$. For each variable v_j with $j\notin J^*$ retain its value. For each variable v_j with $j\in J^*$ replace the original value a_j by the transition set, R_{jk} say, to which a_j belongs. In order to test that the partially corrected record thus obtained is repairable, check this by letting each v_j, with $j\in J^*$, assume the values in the corresponding transition set, and for each resulting completed record r_l (l denotes the trial number) check whether it satisfies the edits in E_r. If so, then

stop: z^* is a solution and r_l is a completed record. Otherwise continue with step 3.

3. Find a prime cover w (a 0-1 row vector) satisfying

$$wQ \geq 1 \ , \tag{3.4.9}$$

where Q is a 0-1 matrix such that

$$Q_{il} = \begin{cases} 1 & \text{if the l-th record } r_l \text{ in step 2 violates edit i} \\ 0 & \text{otherwise.} \end{cases} \tag{3.4.10}$$

Let w° be a solution of (3.4.9) and put $I^\circ=\{i \mid w_i^\circ=1\}$.

4. Generate the edit $e_{z*}(w^\circ)=\Pi_j A_j$ where

$$A_j = \begin{cases} \cap_{i \in I^\circ} A_{ij} & \text{if } j \notin J^* \\ R_{jl} & \text{if } j \in J^*, \text{ with } R_{jl} \text{ as indicated in step 2.} \end{cases} \tag{3.4.11}$$

Add the 0-1 row vector m at the bottom of M in (3.4.4), where

$$m_j = \begin{cases} 1 & \text{if } v_j \text{ is involved in } e_{z*}(w^\circ) \\ 0 & \text{otherwise.} \end{cases} \tag{3.4.12}$$

5. Go to step 1. ///

Algorithm 3.4.1 terminates because of the following observations:

1. The edit generated in step 4 yields a set covering constraint, which is violated by the current z^*. Hence a new prime cover for (3.4.4), without the repairability condition but with the added constraints, will be found at each iteration of step 1.

2. The number of such prime covers is finite (at most $2^{|W_r|-1}$).

In order to facilitate the understanding of algorithm 3.4.1, we briefly

comment on each step. For an example illustrating the algorithm the reader is referred to Garfinkel (1979).

Discussion of algorithm 3.4.1

Step 1 See Garfinkel and Nemhauser (1972, ch. 8) for a thorough discussion of set covering, which is known to be NP-hard. The corresponding decision problem (MINIMUM COVER) is NP-complete.

Step 2 This step assumes that the number of records to be tested is not too large. Note that the number of records to be tested is of the order $\Pi_{j \in J^*} |R_{jk}|$. After step 2 has been executed a new complete record is generated, which satisfies the edits in E_r. So this algorithm solves the MWFI problem and provides an imputation as well (which may be discarded, of course).

Step 3 Any set I° induced by a prime cover w° of Q, produces a *maximal implied edit* (via (3.4.11)), which is an edit such that the corresponding edit set is not contained in the edit set of any other edit. w° can be found in linear time.

Step 4 This step uses several generating variables to calculate (a generalization of) an implied edit, and is therefore slightly more general than the operation in (3.4.5). A generalization of theorem 3.4.3 proves that $e_{z*}(w^\circ)$ is a valid edit, implying in particular that $A_j \neq \emptyset$ for all j.

As a general remark we may add that the edits generated by different records might be different. Instead of calculating certain implied edits several times, it is plausible to keep the generated edits in a table. Note furthermore that all domains are necessarily finite because of the enumeration in step 2.

It is not clear whether this algorithm performs satisfactorily in practice, with respect to both optimality guarantees and average time and space complexity. To the present author no empirical results about this algorithm are known. It is clear that its worst case behaviour can be bad, essentially due to step 2. ///

3.4.4 Additional remarks

Garfinkel's algorithm uses a 'brute force' method (in step 2) to verify that an incomplete record r' is repairable. Instead we could use a probabilistic method to verify this. We assume that a combination of values is drawn which is substituted for the missing ones appearing in the record r'.

Suppose that, for each incomplete record r', we make at most N independent draws of such values, which are then substituted into r', and the completed record r'' is checked against the edits. As soon as we find a record r'' which satisfies the edits, we stop: r' is repairable.

Assume that the combinations of values are drawn from a uniform distribution. Let p be the probability that a combination is drawn which yields an acceptable record r''. Note that p is the size of the collection of combinations yielding a repairable record relative to the collection of all possible combinations. Let T be the stopping time for the first success, i.e. a combination yielding an acceptable record r''. Then we have for $1 \leq k \leq N$:

$$P[T = k] = q^{k-1}p \ , \qquad (3.4.13)$$

where q=1-p. Hence T has a geometric distribution. As is well known ET=1/p and $Var(T)=1/p^2$. Suppose that at the k_0-th draw a success was scored. We can estimate p from (3.4.13) for $k=k_0$ by applying the maximum likelihood (ML) estimation technique. Then we find that the ML-estimator of p equals $1/k_0$.

Suppose that in N draws no success was obtained, i.e. T>N. We find

$$P[T > N] = q^N \ . \qquad (3.4.14)$$

Of course, the fact that no success was obtained does not justify the conclusion that r' is not repairable. Suppose, however, that we want to say something about p, in order to obtain an idea of the size of the set of solutions compared to the size of the set of all possible combinations. Applying the ML-method to (3.4.14) yields the unsatisfactory result $\hat{p}=0$. In Good (1965, section 3.1) this and similar problems are considered. It is suggested there that a Bayesian method should be applied to find a more sensible estimate.

Let $g:[0,1]\rightarrow\mathbf{R}$ be a prior density for p. Using (3.4.14) we find for the posterior density h of p:

$$h(p) = \frac{(1-p)^N g(p)}{\int_0^1 (1-p)^N g(p)dp} \quad . \tag{3.4.15}$$

As a point estimate for p we can for instance take the expected value of the posterior density. In case we do not have any prior knowledge about p we should use an ignorance prior for g. Some ignorance priors are of the beta form, that is proportional to $p^{\alpha}(1-p)^{\beta}$, where $\alpha,\beta>-1$. Well-known ignorance priors are obtained for the choices $\alpha=0$, $\beta=0$ and $\alpha=-1/2$, $\beta=-1/2$. The mean of the posterior density (3.4.15) for $g(p)\simeq p^{\alpha}(1-p)^{\beta}$ is given by

$$p_{post} = \frac{\alpha + 1}{\alpha + \beta + N + 2} \quad . \tag{3.4.16}$$

For the uniform prior ($\alpha=\beta=0$) this yields $p_{post,1}= 1/(N+2)$, and for the prior with $\alpha=\beta=-1/2$ we find $p_{post,2}=1/(2N+2)$, which is almost twice as small as $p_{post,1}$. So, although we are able to estimate p in this Bayesian approach, the problem is now to find a suitable ignorance prior. (This is of course a problem for Bayesian statistics in general.) We shall not dwell on this topic any further.

We now come to the second point in this section. In order to find edit errors in a record, we might as well proceed as follows. We first try to find a set of variables which are likely to be in error. Then we check whether the resulting incomplete record is repairable. This second step can either be carried out by complete enumeration, if there is only a finite number of possibilities, by a probabilistic method as described above, or

by yet another method. (For CP-edits this is similar to solving a satisfiability problem.) We concentrate on the first step here.

Let r be a record which has activated a set E_r of edits, and violated the edits in $E_{rv} \subset E_r$. Assume that with each variable v_i in r there is a Boolean variable u_i associated, which indicates whether (u_i=true) or not (u_i=false) v_i is assumed to have an incorrect value. We shall associate a Boolean sentence with the triple r, E_r and E_{rv}. This sentence is built using the following principles.

1. For each edit e_i in E_{rv} the variables involved in it are all suspicious of being in error. With such an edit associate a clause $u_{i1} \vee \ldots \vee u_{ik_i}$ if the variables $v_{i1}, \ldots, v_{ik_i}$ are involved in e_i.

2. For each edit e_i in $E_r \setminus E_{rv}$ we can associate either of the following clauses, depending on what seems to be more likely:

   a. No variable involved in e_i is suspicious to have an incorrect value. Hence we associate the clause $\neg u_{i1} \wedge \ldots \wedge \neg u_{ik_i}$ with e_i.

   b. Not all variables involved in e_i are suspicious to have incorrect values. Hence we associate the clause $\neg(u_{i1} \wedge \ldots \wedge u_{ik_i})$ with e_i.

For each edit in E_r a clause is generated, and a Boolean sentence, called a *suspicion sentence*, is formed by taking a conjunction of all these clauses. If there is a solution to this Boolean sentence, then we have a set of candidate variables which might yield a repairable incomplete record, namely the variables associated with the literals u_i which have the value 'true'. Let S denote such a solution set. Note that for any set W_i of variables involved in edit $e_i \in E_{rv}$ $S \cap W_i \neq \emptyset$, for any solution set S. So assigning a missing variable to the variables in S will always idle the edits in E_{rv}. We illustrate the construction of suspicion sentences in the following example.

Example 3.4.3 Consider the situation in example 3.4.1. The record r given there violates the edits e_1, e_4 and e_5. Hence the clauses associated with

these edits are respectively: $u_2 \vee u_3 \vee u_5$, $u_2 \vee u_6$ and $u_1 \vee u_4 \vee u_5$. Suppose that we consider none of the variables in the remaining edits suspect. We then obtain the following clauses for e_2 and e_3: $\neg u_1 \wedge \neg u_3 \wedge \neg u_4 \wedge \neg u_6$ and $\neg u_1 \wedge \neg u_2 \wedge \neg u_4$. Taking the conjunction of these clauses yields the suspicion sentence

$$(u_2 \vee u_3 \vee u_5) \wedge (\neg u_1 \wedge \neg u_3 \wedge \neg u_4 \wedge \neg u_6) \wedge (\neg u_1 \wedge \neg u_2 \wedge \neg u_4) \wedge (u_2 \vee u_6) \wedge (u_1 \vee u_4 \vee u_5) \quad . \tag{3.4.17}$$

It is not difficult to see that (3.4.17) is contradictory. Hence our assumption as to the correctness of the values of all variables in the edits satisfied by r was too optimistic. Going to the other extreme and assuming that not all variables in e_2 and e_3 are suspicious of having incorrect values yields the following suspicion sentence

$$(u_2 \vee u_3 \vee u_5) \wedge (\neg u_1 \vee \neg u_3 \vee \neg u_4 \vee \neg u_6) \wedge (\neg u_1 \vee \neg u_2 \vee \neg u_4) \wedge (u_2 \vee u_6) \wedge (u_1 \vee u_4 \vee u_5) \quad . \tag{3.4.18}$$

The suspicion sentence (3.4.18) is neither a tautology nor a contradiction. A solution to (3.4.18) is for instance $(u_1, u_2, u_3, u_4, u_5, u_6) = (0,1,0,1,0,0)$, where 1 signifies 'true' and 0 'false', as usual. As one easily verifies (3.4.18) does not have a unique solution. It should be attempted, however, to find solutions with as few 1's as possible, implying as few variables as possible having to receive another value in r (provided r'' is repairable).

Of course it is possible, and advisable, to find out whether an 'intermediate' suspicion sentence has a solution, e.g. the sentence which is obtained when assuming that the variables involved in e_2 are not suspicious, whereas those involved in e_3 are not all suspicious. As a guideline in the choice of a suitable suspicion sentence, one could perhaps make use of an error model. Suppose that error weights have been calculated. Assume that an edit which has not been violated and for which the sum of the weights of the variables involved is below a certain level, is considered to be of the second type. This means that for such an edit the variables involved in it are assumed not all to be suspicious. Otherwise, the edit is assumed to be of the first type, i.e. none of the variables involved is assumed to be in error. ///

4. MISSING VALUES

4.1 Introduction

Once we have obtained a partially corrected record we could abandon our correction process. If we know how to live with data files with incomplete records (with a correct routing structure,though) everything is fine. As we have argued in section 0.2, we do not consider it satisfactory to let users dabble with incomplete data files. Nevertheless it is interesting to consider a specific problem with such incomplete files, namely how they could be matched with other files (either complete or incomplete).

In section 4.2 we shall attempt to present a complete taxonomy of missing values we could come across in such incomplete files, or which otherwise play a role in collecting or processing survey data. In section 4.3 we consider the problem how to cope with incomplete records in data bases.

4.2 Various types of missing values

In the previous chapters we have assumed that a record is represented as a sequence consisting of pairs of variables and values. In practice however a record is very often represented by a fixed number of fields, each of which corresponds to a variable in the questionnaire. Such a record representation is of fixed length because it contains a fixed number of fields, each corresponding to a variable in the record (as a sequence of pairs of variables and values), and each field has a specific and fixed length. In such a fixed length record, not only the variables with regular values are represented, but also the variables with missing values, e.g. those which have been skipped as a result from the routing structure in the corresponding questionnaire. Files with fixed length records are called flat. They are to be contrasted with hierarchical files. The advantage of flat files to hierarchical ones is that they are easier to manipulate. An apparant drawback of such files is that they may require much more space

than equivalent hierarchical ones.

In the present section we try to find a taxonomy of missing value types which one might encounter in such fixed length records, which can either be complete or incomplete (but partially corrected). Each missing value type corresponds to a particular reason for missingness. The information which is contained in (certain) missing values is of importance for an insight in the mechanisms which generate them. It is also important to distinguish between missing values when applying a procedure to backtrack in a questionnaire which is implemented on a handheld computer, i.e. in a CAPI survey. Backtracking is applied when answers of a respondent turn out to be in conflict with one or more edits in the questionnaire.

We consider two general classes of missing values: the first class consists of missing values which do not appear in the records as defined in section 1.3. A variable which has been assigned a missing value belonging this class cannot activate edits, but will make them invisible. The second class consists of missing values which should be regular values but have not been observed for various reasons. For obvious reasons we shall call the missings belonging to the first class *invisible missings*, and those belonging to the second class *regular missings*. These are our lists of missing values:

Invisible missings.

1. Initial missing. This missing signifies that a question has not been answered by a respondent nor could have possibly been answered because the interview did not proceed that far. Initially we may consider any field having been assigned this value for any respondent, when we assume that for any respondent which is drawn into the sample (save the unit nonrespondents) a record has been reserved prior to the survey. We can find such values in the records of the incomplete file corresponding to respondents which e.g. finished the interview prematurely. An initial missing value is the only missing value that cannot be assigned in the data editing phase.

2. Routing skip. This missing value indicates that a variable has been skipped due to the routing structure in the questionnaire. It has the property that it does not activate an edit in which a variable is involved to which it has been assigned.

3. Imputed routing skip. A variable in a record which should have been skipped, but has erroneously been assigned a regular value, will receive a value signifying that it should have been skipped, i.e. an 'imputed routing skip'.

4. Missing due to overcompleteness. When the routing is checked it is possible that there are questions which should have been skipped but which have actually been answered. These values are changed into a missing due to overcompleteness. This missing value is treated in exactly the same way as a routing skip but carries extra information in that it indicates that in the original record the variable to which it has been assigned had not been skipped.

Regular missings.

1. Missing due to out of range. This missing value is assigned to a variable when it fails a range check, as a result from the circumstance that a non-missing value was present but out-of-range. It can be identified with the domain of the variable to which it has been assigned. At a later stage in data editing it might be changed into another missing value, which also corresponds to a smaller set of possible values.

2. Missing due to illegitimate skipping. When a record is checked and corrected with respect to its routing structure, it is possible that a variable has been assigned a routing skip (i.e. the corresponding question has been skipped), whereas it should have been assigned a regular value. In this case such a variable will receive a 'missing due to illegitimate skipping'.

3. Missing due to edit violation. If a nonmissing field in a record violates one or more edits its current value is replaced by a missing due

to edit violations when it is decided that this field is responsible for the violation. Although it would in principle be possible to indicate with a special missing value which edits had been violated by the original value of a variable in a record, it seems that this sort of information is of little interest in general and is therefore discarded.

4. Missing due to incorrect transition. A variable may have a value in the wrong transition set, and hence imply a transition which is incorrect. Such a variable should be assigned a 'missing due to incorrect transition'. Clearly with such a missing value a specific transition set is associated.

Each of these missing value types can be considered as being generated by different nonresponse mechanisms. It seems to be advisable to make a distinction between the various types of missing values by using different codes. This information is of value for imputation purposes on the one hand and can be a source for improving the quality of data to be gathered in a future survey.

4.3 Missing values and databases

In the present section we consider some problems related to the processing of incomplete data in databases, in particular in relational databases. It is clear that standard theory for relational databases has to be extended in order to cope with incomplete data.

There are several approaches to the problem of missing values in (relational) databases, four of which we shall discuss here:

1. Codd's (1975) approach, which is based on three-valued logic. It is an extension of ordinary (two-valued) logic.

2. Vassiliou's (1979) approach, which is based on denotational semantics and requires a four-valued logic.

3. An approach in which each missing value is identified with a specific (transition) set.

4. An approach in which each missing value is identified with a random variable.

From these approaches the third is tailored to the framework in the present book, whereas the other three approaches are of a more general character. The fourth approach has a statistical character, whereas the other approaches are purely 'logical'.

Further material on missing values in databases, which is relevant to the problem considered in the present section, is supplied by e.g. Lacroix and Pirotte (1976), Zaniolo (1977), Vassiliou (1980), Lipski (1981) and Date (1983, section 5.5).

Before we present the approaches, we introduce some notation. Apart from the standard Boolean truth-values t (true) and f (false) we shall need truth-values α (unknown) and ω (inconsistent). This latter truth-value ω is only needed in Vassiliou's (1979) approach.

In order to avoid any confusion we shall call generalizations of Boolean expressions, 'truth-valued' expressions, where the set of truth values includes the common truth-values 'true' (or 't') and 'false' (or 'f'), but possibly other values as well.

4.3.1 Codd's approach

Queries and operations in a relational database are based on logic. In order to cope with missing values in such databases one has to extend ordinary two-valued logic to many-valued logics, i.e. with at least three truth-values. Furthermore it should be defined how truth-valued expressions ought to be evaluated. As an extension to the ordinary two-

valued truth-tables for AND, OR and NOT, Codd introduces the following ones:

AND	t	f	α
t	t	f	α
f	f	f	f
α	α	f	α

OR	t	f	α
t	t	t	t
f	t	f	α
α	t	α	α

NOT	
t	f
f	t
α	α

. (4.3.1)

Existential and universal quantifiers behave like iterated OR and AND respectively. Note, by the way, that the AND and OR tables can both be interpreted as multiplication tables for abelian groups of order 3 on the set T = {t,f,α}, with identity t (for AND) and f (for OR). The quadruple (T,t,AND,OR) forms an algebra.

In order to be able to evaluate a truth-valued expression, the following *missing value principle* could be applied. A truth-valued expression S takes the value α if and only if it has the following two properties:

1. each occurrence of α in S can be replaced by a regular value in the domain of the corresponding variable, so as to yield the value t for S,

2. each occurrence of α in S can be replaced by a regular value in the domain of the corresponding variable, so as to yield the value f for S.

Note that in this approach there is just one type of missing value. Furthermore there is no difference between a missing value on the one hand and the truth-value α on the other. Codd's approach is what Vassiliou (1979) calls *truth-functional*: the evaluation $V(S(c_1,c_2,\ldots,c_k))$ of a truth-valued expression $S(c_1,c_2,\ldots,c_k)$, where the c_i are elementary truth-valued expressions, equals $S(V(c_1),\ldots,V(c_k))$. Here V maps the expressions on the set T according to the above rules. For example,

$V(c_1$ AND $(c_2$ OR $c_3))=$ $V(c_1)$ AND $(V(c_2)$ OR $V(c_3))$.

It should be noted that the concepts of tautology, i.e. an expression which always evaluates to t, for any combination of the values of the variables

in the expression, and contradiction, i.e. an expression which always evaluates to f, for any combination of the values of the variables in the expression, have to be reinterpreted. For example, in two-valued logic p OR (NOT p) is a tautology, but not in three-valued logic. But any tautology (contradiction) in three-valued logic is a tautology (contradiction) in two-valued logic.

A further restriction introduced by Codd extends the non-duplicate principle: in a relation (a flat file) there can be no two identical tuples (records, in our case). So if there are two tuples containing missing values, but which agree on the remaining variables, one of these has to be removed from the relation to which they belong. This principle should be viewed in the light of the unidentifiability of missing values.

On the basis of these conventions it is possible to investigate the behaviour of relational algebra in this extended setting. We shall only consider the equi-join as an example. The equi-join is the operation of joining records from two relations on the basis of equality of two variables (cf. Ullman, 1982, p. 155), as an example. In Codd (1975) examples can be found of other operations.

Let v_1, v_2 and v_3 with domains {3,5}, {1,2} and {1,2} respectively. Let R and S be relations, defined as follows:

R		S
v_1	v_2	v_3
3	α	α
α	2	2
5	1	

(4.3.2)

Now the equi-join $R[v_2=v_3]S$ and the *maybe equi-join* $R[v_2=_\alpha v_3]S$ are defined as

$R[v_2=v_3]S$		
v_1	v_2	v_3
α	2	2

$R[v_2=_\alpha v_3]S$		
v_1	v_2	v_3
3	α	α
3	α	2
α	2	α
5	1	α

(4.3.3)

We can view the equi-join $R[v_2=v_3]S$ as a 'lower bound' for any completion of these incomplete relations, and the maybe equi-join $R[v_2=_\alpha v_3]S$ as an 'upper bound'.

A defect in this approach to data bases with incomplete data is the very fact that it is truth-functional, which may give rather undesirable effects. Consider for instance the expression C=(v_1=1) OR NOT (v_1=1), where v_1 is a variable taking values in the domain (1,2,3). If in a record the value of v_1 is unknown, then in the approach of the present section C evaluates to α. However, v_1 has in fact exactly one of the values 1, 2 or 3, and for any of these values for v_1, C evaluates to t.

The phenomenon described in the preceding paragraph is undesirable because we want to use the information as available in the data base, no more but no less either. In the truth-functional approach as described in the present section this latter requirement is not always met. See also Vassiliou (1979, section 2) for a further discussion of this phenomenon. As a matter of fact, precisely this defect in Codd's approach motivated Vassiliou to develop an alternative approach.

4.3.2 Vassiliou's approach

The basic idea of Vassiliou's approach is to extend the domains and codomains of functions. Let $R_1,\ldots,R_{k+1}$ be domains and let $R_i^0 = R_i \cup \{\alpha,\omega\}$ (i=1,...,k+1) be their *extensions*. Let $f:R_1\times\ldots\times R_k \rightarrow R_{k+1}$ be a *partial function*, that is, a function that is not defined on the whole of $R_1\times\ldots\times R_k$. This f can be extended to a *total function* $f^0:R_1^0\times\ldots\times R_k^0 \rightarrow R_{k+1}^0$, as follows.

1. $f^0(a_1,\ldots,a_k) = f(a_1,\ldots,a_k)$ for all $(a_1,\ldots,a_k)\in R_1\times\ldots\times R_k$ for which f is defined.

2. $f^0(a_1,\ldots,a_k) = \omega$ if f is not defined at $(a_1,\ldots,a_k)\in R_1\times\ldots\times R_k$.

3. $f^0(a_1,\ldots,a_k) = \omega$ if $a_i=\omega$ for at least one i.

4. If $a_i=\alpha$ for an $i\in\{1,\ldots,k\}$ and $a_j\in R_j$ for $j\neq i$ then

$$f^0(a_1,\ldots,a_k) = \begin{cases} \omega & \text{if } f(a_1,..,e,..,a_k) \text{ is undefined for all } e \in R_i; \\ c & c \in R_{k+1} \text{ if } f(a_1,..,e,..a_k) \text{ is either undefined or equals c at least once, for all } e \in R_i; \\ \alpha & \text{in any other case.} \end{cases} \qquad (4.3.4)$$

Vassiliou (1979, section 3) presents the truth tables of the and, or and not operators in this four-valued logic, which read

AND^0	t	f	α	ω
t	t	f	α	ω
f	f	f	f	ω
α	α	f	α	ω
ω	ω	ω	ω	ω

OR^0	t	f	α	ω
t	t	t	t	ω
f	t	f	α	ω
α	t	α	α	ω
ω	ω	ω	ω	ω

NOT^0	
t	f
f	t
α	α
ω	ω

(4.3.5)

Note that restricting these tables to the values t, f and α yields Codd's truth tables. It should be stressed, however, that these tables cannot be used in evaluating expressions as in Codd's approach, because Vassiliou's approach is non-truth-functional.

The example considered in Vassiliou (1979) to show this is based on the fact that c OR (NOT c) is not a tautology in Codd's approach. The truth-valued expression $(c_1\wedge c_2)\vee(c_3\wedge\neg c_2)$ evaluates to α if c_1=t, c_3=t and $c_2=\alpha$ in Codd's case, but evaluates to t in Vassiliou's case. ($\wedge$, $\vee$ and $\neg$ to denote 'and', 'or' and 'not' operators, in either interpretation.)

When applying Vassiliou's method in practice, a semantical interpretation of the symbols α and ω should be given. One such

interpretation, in the framework in the present book, is that α is 'a regular missing value', i.e. a missing value which stands for some regular but unknown value, whereas ω is 'an invisible missing value', i.e. a missing value which does not represent a regular value.

Of course, working directly from the definitions in (4.3.4) in a practical situation is rather laborious. Therefore Vassiliou (1979, section 4) gives some results which facilitate the use of the method. The interested reader is referred to his paper to learn about the details.

4.3.3 Set approach

As we have shown in chapter 3, regular missing values can be identified with certain sets, viz. transition sets. We have assumed in this book that records consist of pairs of variables and values. A value is either regular or a regular missing value, i.e. a transition set.

This observation can be used as a basis for another approach to data bases with missing data. We illustrate this by considering an extension of the equi-join concept.

Consider two files, fl1 and fl2, which are to be joined on the basis of equality of the variables, v_1 and v_2, where v_1 is a variable in fl1 and v_2 a variable in fl2, having a common domain R. Assume that both fl1 and fl2 may contain records with missing values for the variables v_1 and v_2. Let $(fl1, r_1, v_1, \alpha_1)$ and $(fl2, r_2, v_2, \alpha_2)$ be data elements from fl1 and fl2 respectively: fli is the file identification, r_i the record identification, v_i the variable identification and α_i a value (i=1,2). This value is a subset of R. (Note that normally a regular value is not represented by a one-element set, but by a value. However the present convention is more convenient in the application we are considering.) Suppose that the result of a successful matching is represented by a quintuple $(fl1*fl2, r_1, r_2, \alpha_3, \beta)$, where fl1*fl2 indicates the matched file to be created, the r_i are the record identifications (i=1,2) and β is a binary variable to be explained below. Consider the following matching rule.

Matching rule There is a matching if and only if $\alpha_1 \cap \alpha_2 \neq \emptyset$, and in that case $\alpha_3 = \alpha_1 \cap \alpha_2$. In case $|\alpha_3| = 1$ and $|\alpha_1|$, $|\alpha_2| > 1$ let $\beta = 1$, otherwise let $\beta = 0$.

This rule is based on the idea that the α_i represent possible values for the variable v_i in record r_i in file fli. So two records from different files can only match if they have some possible values in common. Now we know that a value α_1 with $|\alpha_1| > 1$ is a missing value. If we have that $|\alpha_3| = 1$ and $|\alpha_1|, |\alpha_2| > 1$ we have to indicate that α_3 is not a regular value. This is different from the situation where $\alpha_1 = \alpha_2$ and $|\alpha_1| = 1$, i.e. an exact match. To indicate the difference, we assign the value 1 to β; otherwise β is assigned the value 0.

It is clear that this defines a sensible extension of the ordinary equi-join operation. In fact the idea behind this matching rule can be extended to other operations, such as range queries ('find all records r_1 in fl such that the variable v_1 has a value in [3,10]'). The principle behind this is that there should be at least one element in the value-set associated with a missing value for v_1 that satisfies the condition in a query or whatever.

Note that the elements in the value set corresponding to a missing value all have the same 'likelihood' of being the true value for this missing one. If we allow that the various values in such a set have different probabilities of being the 'true' value, then we obtain our last approach to our problem of missing data in databases. Note that in this approach a missing value is identified with a random variable. An approach based on this idea is discussed in the next section.

4.3.4 Random variable approach

The basic idea of the fourth approach to missing data in databases essentially extends the idea of a value of a variable in a record. Let r be a record in a database and $v_1 = a_0$ the value of the field v_1 in r. Let R_{v_1} be the domain of the variable v_1. Instead of viewing a_0 as the object associated with v_1 in r we can consider the probability density f_{a_0} defined as

$$f_{a_0}(x) = \begin{cases} 0 & \text{if } x \neq a_0 \\ 1 & \text{if } x = a_0, \end{cases} \tag{4.3.6}$$

if R_{v_1} is discrete, and

$$f_{a_0}(x) = \delta(a_0 - x), \tag{4.3.7}$$

if R_{v_1} is continuous (δ is Dirac's function).

Taking this point of view, missing data are easily fit into the framework: a missing value for a variable v in r is associated with a nondegenerated probability density function on R_{v_1}. The density function may be estimated from a sample. We shall furthermore assume that the densities associated with missing values in different records (i.e. for different individuals) are independent.

Now assume that we have two files f_1 and f_2 which may contain missing values. Suppose that v_1 and v_2 are variables in f_1 and f_2 respectively, which have the same finite domain $R=\{1,\ldots,k\}$. Suppose we have a record r_1 in f_1 and a record r_2 in f_2 with associated probability densities $p = (p_1,\ldots,p_k)$ and $q = (q_1,\ldots,q_k)$ for $\underline{a}_0$ and $\underline{b}_0$ respectively. We furthermore assume that $\underline{a}_0$ and $\underline{b}_0$ are independent. Suppose that in an equi-join operation of f_1 and f_2, on the basis of equality of $\underline{a}_0$ and $\underline{b}_0$, i.e. $f_1[v_1=v_2]f_2$, we would associate r_1 with r_2. How large is the probability that matching r_1 and r_2 is correct, in the sense that v_1 and v_2 have the same value for r_1 and r_2 respectively? This probability is easily calculated:

$$\begin{aligned} &P[r_1 \text{ and } r_2 \text{ match}] = \textstyle\sum_i P[\underline{a}_0=i \text{ and } \underline{b}_0=i] = \\ &\textstyle\sum_i P[\underline{a}_0=i]P[\underline{b}_0=i] = \sum_i p_i q_i = (p,q), \end{aligned} \tag{4.3.8}$$

where $(.,.)$ denotes the standard inner product in $\mathbf{R}^k$. It is clear that matching f_1 and f_2 on the basis of equality of v_1 and v_2 should be performed in such a way as to maximize the probability (4.3.8). As a

strategy for calculating the probabilistic equi-join operation, we could require that only those records r_1 in f_1 and r_2 in f_2 are matched, for which the corresponding probability (4.3.8) is larger than a certain bound $\beta>0$. Note that this requirement automatically yields the standard results in case f_1 and f_2 are complete, because in that case r_1 and r_2 are only matched if the (regular) values for v_1 and v_2 are exactly equal. Furthermore it is possible to associate the probability (4.3.8) to a matched pair r_1 and r_2, in order to indicate the strength of the matching.

The idea to view a missing value as a random variable is strongly encouraged by statistical theory. In chapter 6 we shall return to this point of view.

5. ESTIMATING A DENSITY IN THE PRESENCE OF NONRESPONSE

5.1 Introduction

In the present chapter we consider the problem of estimating the density of a discrete *target* or *imputation* variable y in the presence of item nonresponse. Estimating a density is of interest to the approach to imputation in a survey data production process as given in chapter 6.

In this chapter we consider two approaches to the problem of estimating a discrete density using incomplete data. The first approach is called the quasi-randomization approach. Characteristic for this approach is that it is label dependent, i.e. it is based on the inclusion probabilities and response propensities of the respondents on the variable y. Whereas the inclusion probabilities are known, the response propensities are not and have to be estimated with the help of some statistical model. This approach is closest in spirit to the classical randomization approach in survey sampling.

In case one feels that the labels associated with the individuals in the sampled population carry no or little information, and that some likelihood function should be used as a basis for statistical inferences, then one can apply a label independent approach. In the label independent approach adopted below a simple urn model is used to describe the response mechanism with respect to the variable y. Then one can then either use a Bayesian or a maximum likelihood (ML) technique to estimate the density of y, provided the response propensities for each category of y are known. A label independent approach to survey sampling can be found in e.g. Royall (1968) and Hartley and Rao (1968, 1969).

The methods described in this chapter are deliberately kept simple. Our prime intention is to sketch two possible approaches to tackle the estimation of a discrete density, rather than to give a full statistical treatment of this problem. For an overview of a range of techniques dealing with missing data see Madow, Olkin and Rubin (1983), Madow and Olkin (1983) and Rubin (1987).

The present chapter is organized as follows. In section 5.2 some models for response propensities are discussed. These play an essential role in the approaches considered in sections 5.3 and 5.4. In section 5.3 a quasi-randomization approach is sketched and in section 5.4 a label independent one.

Remark 5.1.1 In the remaining sections of this chapter we denote a random variable as a rule by an underscored letter. However, estimators are indicated by a hat ($\hat{}$). Furthermore in some other cases we deviate from the convention, e.g. a sample (a stochastic set) is not represented by an underscored letter. The reason is to avoid too many of these symbols in the text. We also deviate in this chapter from the previous ones by denoting imputation and auxiliary variables by y and x instead of by v and w (these symbols are used in this chapter for other purposes). This is closer to common practice in statistics. ///

5.2 Response propensities

Before we can discuss our approaches we have to introduce some notation and discuss several assumptions made. Let U be a finite population, from which a sample s is drawn without replacement and of fixed size m. Let the set of unordered samples of size m from U be denoted by S_m. A sampling design on S_m is a function $p:S_m \rightarrow [0,1]$ such that $\Sigma_{s \in S_m} p(s) = 1$. Let y denote a qualitative variable with domain $R=\{y_{(1)},\ldots,y_{(p)}\}$. We shall furthermore assume that the sampling design used in the survey we are considering, is noninformative with respect to y, i.e. is independent of y. This is a very common requirement and hardly restrictive in view of practical applications.

Let U_j be the subpopulation of U which consists of all individuals which have a y-value equal to $y_{(j)}$, i.e. $U_j=\{i \in U | y_i=y_{(j)}\}$. Then $U=\cup_j U_j$. In the present chapter we are interested in estimating the density (f_j) of y, i.e.

$$f_j = \frac{|U_j|}{|U|} \quad , \tag{5.2.1}$$

for j=1,...,p, where $|.|$ denotes the size of a set.

Estimating the f_j is equivalent to estimating the sizes $|U_j|$. There are two possibilities to consider in principle: either $|U|$ is known or unknown. We shall, however, restrict our attention to the case that $|U|$ is known and equals N.

Due to unit and item nonresponse, and as a result of the elimination of inconsistent answers in a data editing procedure, y is observed only for the subset $r \subset s$ of respondents, i.e. $r=\{i \in s | y_i \text{ is nonmissing}\}$. Let the first order inclusion probabilities be denoted by π_i, i.e. $\pi_i = P[i \in s]$. It is implicitly assumed that $\pi_i > 0$ for $i \in U$.

We shall furthermore assume that every individual i in the population has a fixed probability, i.e. the *response propensity*, q_i to respond to variable y, which is independent of the sample s into which i is drawn. In fact this is essentially the stochastic response model considered by Bethlehem and Kersten (1986, p. 88), but for unit instead of for item nonresponse.

For each $i \in U$ we can introduce two binary random variables $\underline{a}_i$ and $\underline{r}_i$, indicators for membership of the sample and response to y respectively, and defined as

$$\underline{a}_i = \begin{cases} 1 \text{ if } i \in s \\ 0 \text{ otherwise} \end{cases} \quad , \qquad \underline{r}_i = \begin{cases} 1 \text{ if i responds to y} \\ 0 \text{ otherwise.} \end{cases} \tag{5.2.2}$$

Note that $\underline{a}_i \underline{r}_i$ indicates whether or not individual $i \in s$ responds to y, i.e whether $i \in r$. In view of what was defined above we have $E\underline{a}_i = \pi_i$, $E\underline{r}_i = q_i$. Furthermore we shall assume that the $\underline{r}_i$ are mutually independent random variables, hence $E\underline{r}_i \underline{r}_j = q_i q_j$ for $i,j \in U$ with $i \neq j$. Finally we shall assume that the $\underline{a}_i$ on the one hand, and the $\underline{r}_i$ on the other, are mutually independent random variables.

We shall denote the number of individuals in s that responded $y_{(i)}$ to y by α_j, i.e. $\alpha_j=|U_j\cap r|$, and the (unknown) number of persons in s for whom $y=y_{(i)}$ but who did not respond to y by β_j, i.e. $\beta_j=|U_j\cap s\backslash r|$. The size of r will be denoted by $|r| = \alpha = \sum_i \alpha_i$ and that of $s\backslash r$ by $|s\backslash r| = \beta = \sum_i \beta_i$. We shall henceforth refer to the persons in the sample who responded to y as the *item respondents* and to those who did not as the *item nonrespondents*, i.e. implicitly assuming that the item meant is y.

From the assumptions with respect to the $\underline{r}_i$ we deduce that the probability P[r|s] to realize r within s is given by

$$P[r|s] = \prod_{i\in r} q_i \prod_{i\in s\backslash r} (1-q_i) \quad . \tag{5.2.3}$$

Note that the effect of a nonresponse mechanism is the same as of a two-stage sampling procedure. In the first stage a sample is drawn which is designed by an investigator. In the second stage a Poisson sampling procedure (in the terminology of Hájek, 1981, ch. 6) is applied, which, however, is not controlled by the investigator. In fact the inclusion probabilities q_i in this second stage are unknown to the investigator and have to be estimated, possibly using prior information. These inclusion probabilities are the response propensities.

Note that 5.2.3 defines a joint density from which the m parameters q_i cannot, of course, be identified if no further assumptions are made. If, however, there is an *auxiliary variable*, x say, available, this problem can be solved, at least to some extent. We shall consider three ways to use the availability of such an auxiliary variable:

1. To stratify the sample into classes of assumed constant response propensities and apply *a* randomization approach.

2. To stratify the sample into classes of assumed constant response propensities and apply a Bayesian approach.

3. In a more sophisticated parametric model.

Assume that such an auxiliary variable exists. Employing the first approach, we find that (5.2.3) can be written as

$$P[r|s] = \prod_{k\in strata} \prod_{i\in k\cap r} q_i \prod_{i\in k\cap s\setminus r} (1-q_i)$$

$$= \prod_{k\in strata} w_k^{n_k}(1-w_k)^{m_k}, \qquad (5.2.4)$$

where $n_k=|k\cap r|$, the number of respondents in stratum k, $m_k=|k\cap s\setminus r|$, the number of nonrespondents in stratum k, and w_k the response propensity for stratum k, i.e. $q_i=w_k$ for $i\in k$. If we sum (5.2.4) over all samples $r\subset s$ such that $n_k=|k\cap r|$ and $m_k=|k\cap s\setminus r|$ we find the following probability density

$$l(\{n_k\};\{m_k\}|\{w_k\}) = \prod_{k\in strata} \binom{n_k+m_k}{n_k} w_k^{n_k} (1-w_k)^{m_k}, \qquad (5.2.5)$$

where $\{n_i\}$ is used as a short-hand for $n_1,\ldots,n_m$; $\{m_k\}$ and $\{w_k\}$ are defined likewise.

Estimating the stratum response propensities w_k from (5.2.5) by maximum likelihood yields

$$\hat{w}_{k,ML} = \frac{n_k}{n_k + m_k}, \qquad (5.2.6)$$

which is a rather straightforward estimator. Hence $\hat{q}_i = \hat{w}_{k,ML}$ if i is in stratum k.

We can apply the second approach to estimating the response propensities if prior information is available about the response behaviour with respect to the variable y. In that case we may adopt a Bayesian attitude. Suppose that in an earlier, and similar, survey there were a_k-1 item respondents and b_k-1 item nonrespondents in stratum k. Then we might assume the following prior density for the w_k:

$$\rho[\{w_k\}] = \frac{\prod_{k\in\text{strata}} w_k^{a_k-1}(1-w_k)^{b_k-1}}{\prod_{k\in\text{strata}} B(a_k,b_k)} \quad , \tag{5.2.7}$$

where $B(.,.)$ is the beta-function, i.e. for $a, b \in \mathbf{R}$ we have

$$B(a,b) = \int_0^1 z^{a-1}(1-z)^{b-1}dz = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)} \quad , \tag{5.2.8}$$

with $\Gamma(.)$ the gamma-function. The prior (5.2.7) is conjugate to the family of distributions of the form (5.2.5), a circumstance which facilitates computational work considerably.

Multiplying (5.2.5) and (5.2.7) and normalizing this product (i.e. applying Bayes' theorem) yields the posterior density for $\{w_k\}$:

$$\rho_{\text{post}}[\{w_k\}|\{n_k\};\{m_k\}] = \frac{\prod_{k\in\text{strata}} w_k^{n_k+a_k-1}(1-w_k)^{m_k+b_k-1}}{\prod_{k\in\text{strata}} B(n_k+a_k,m_k+b_k)} \quad , \tag{5.2.9}$$

which is also a product of beta densities, as was the corresponding prior density (5.2.7). If a point estimate for $\{w_k\}$ is desired there are several possibilities, e.g the mode, the median or the mean of (5.2.9), depending on what loss function one is using. Taking the mode for example yields the point estimate

$$\hat{w}_{k,\text{post}} = \frac{n_k+a_k-1}{n_k+m_k+a_k+b_k-2} \quad . \tag{5.2.10}$$

If it is unrealistic to assume that response propensities are constant within strata, identified by an auxiliary variable x, another possibility is to assume a parametric model for the response propensities such as a logit model, i.e.

$$\log \frac{q_i}{1-q_i} = x_i' \gamma \quad , \tag{5.2.11}$$

where γ is a parameter vector. γ can be estimated by substituting the expressions $q_i = 1/(1 + \exp(-x_i'\gamma))$ into (5.2.3) it is straightforward to apply the ML-method to calculate γ. For details and further information on the logit model see e.g. Amemiya (1981, 1985, section 9.2), Gourieroux and Monfort (1981), Cox (1969), McFadden (1984) and Anderson (1982). For the performance of a logit model in a simulation study, using a real data set, see e.g. Abrahamse and Geilenkirchen (1986). For existence theorems see e.g. Albert and Anderson (1984), Santner and Duffy (1986) and Gourieroux and Monfort (1981). For other binary models see e.g. Maddala (1983).

The individual response propensities q_i can be used to define average response propensities v_j for each subpopulation U_j, i.e. consisting of individuals which have an y-value equal to $y_{(j)}$:

$$v_j = \frac{\sum_{i \in U_j} q_i}{|U_j|}, \qquad (5.2.12)$$

for j=1,...,p. These probabilities v_j will be used in section 5.4, where a label independent modelling approach is discussed. The v_j can be estimated as follows, if we assume that for each $i \in s$ the individual response propensities q_i have been estimated by $\hat{q}_i$, with some method suggested above (or any suitable other one):

$$\hat{v}_j = \frac{\sum_{i \in U_j \cap r} \hat{q}_i}{|U_j \cap r|} . \qquad (5.2.13)$$

If $|U_j \cap r|$ is not too small, $\hat{v}_{j,1}$ is approximately normally distributed, according to the central limit theorem.

5.3 Quasi-randomization approach

In the present section we turn to the problem of estimating the density of y. In the discussion in the present section we shall consider modifications of the well-known Horvitz-Thompson estimator for the estimation of the density (f_i) of a categorical variable y. By applying different models to

estimate the response propensities we obtain different modified Horvitz-Thompson estimators. In fact the following item nonresponse models, in ascending order of their 'computational complexity', will be considered below (cf. section 5.2):

1. The response propensities are equal for all individuals in the population U.

2. U can be stratified into strata of constant response propensities, on the basis of auxiliary information, assumed to be available for any element in the sample.

3. A parametric model, such as a logit or probit model, can be used to estimate the response propensities of the elements in the sample.

Let z be a real-valued variable, and z_i the value of z for $i \in U$. The population total $z_t = \Sigma_{i \in U} z_i$ can, in case of a sample with item nonresponse on z, be estimated by

$$\underline{z}_{HT\hat{q}} = \sum_{i \in U} \frac{z_i a_i r_i}{\hat{q}_i \pi_i} = \sum_{i \in U} \frac{\underline{z}_i}{\hat{q}_i \pi_i} , \qquad (5.3.1)$$

where $\underline{z}_i = z_i a_i r_i$ and where the $\hat{q}_i$ are estimators for the response propensities q_i. Because the $\hat{q}_i$ appear in the denominator of the terms of the sum in (5.3.1), $\underline{z}_{HT\hat{q}}$ will in general be biased. It is clear that an estimator such as (5.3.1) can be used to estimate the f_j. To this end consider

$$z_i = \begin{cases} 1 \text{ if } i \in U_j \\ 0 \text{ otherwise} , \end{cases} \qquad (5.3.2)$$

in which case the right-hand side of (5.3.1) reduces to

$$\sum_{i \in U_j} \frac{a_i r_i}{\hat{q}_i \pi_i} = \sum_{i \in U_j \cap r} \frac{1}{\hat{q}_i \pi_i} , \qquad (5.3.3)$$

which is an estimator for $|U_j|$. Dividing (5.3.3) by the population size N of U yields an estimator for f_j, i.e.

$$\hat{f}_j = \frac{1}{N} \sum_{i \in U_j \cap r} \frac{1}{\hat{q}_i \pi_i} \quad . \tag{5.3.4}$$

The estimator (5.3.4) can be used to estimate the f_j under several models for the response propensities. We consider a few of these models here.

1. We assume that $q_i = w$ for all $i \in U$. We can estimate w by e.g. the following estimators:

$$\hat{w}' = \frac{|r|}{|s|} = \frac{\alpha}{n} \quad , \tag{5.3.5}$$

$$\hat{w}'' = \frac{\sum_{i \in r} 1/\pi_i}{\sum_{i \in s} 1/\pi_i} \quad . \tag{5.3.6}$$

Note that the estimator (5.3.6) estimates the fraction of (potential) item nonrespondents in the entire population of estimated size $\sum_{i \in s} 1/\pi_i$.

When the constant response propensity model is assumed we have that the size of r conditional on the size of s has a binomial distribution, i.e.

$$P[|r| = \alpha \mid |s| = n] = \binom{n}{\alpha} w^{\alpha}(1 - w)^{n-\alpha}. \tag{5.3.7}$$

Note that estimator (5.3.5) can be viewed as the ML-estimator obtained from (5.3.7).

2. Assume that there is a stratification of U into strata such that $q_i = w_k$ for $i \in k \cap U$, for stratum k in U. In this case we can proceed in every stratum as in 1. The distribution of the relative sizes of the response parts of the sample per stratum is given by (5.2.6). If prior information is available with respect to the response probabilities in every stratum then (5.2.10) can be applied.

3. Assume that the response propensities can be linked to an auxiliary variable x via a parametric model such as a logit (cf. (5.2.11)) or a

probit model. Estimating the parameters in such models requires considerably more computational effort than in the models proposed in 1. and 2. For instance, estimating the variance of (5.3.4), using such a model for the response propensities is difficult due to the nonlinearity of (5.3.4) and may most conveniently be carried out by employing nonparametric estimation methods such as random grouping, bootstrapping, jackknifing and balanced half-sampling (see e.g. Wolter, 1985). ///

As was already remarked in section 5.1 the quasi-randomization approach is treated very briefly here, because it has been investigated elsewhere; c.f. Bethlehem and Kersten (1986) and Cassel et al. (1983).

5.4 Label independent approach

In the approach of the previous section it was tacitly assumed that the labels of the sample elements carried relevant information for the estimation problem considered. This approach results in estimators which explicitly involve the first order sample inclusion probabilities π_i. If we may assume, however, that the labels carry no or little information it is possible to adopt a different approach to the estimation problem of this chapter. Such an approach is presented in this section.

More specifically, we assume that the individuals in the sample have been independently drawn from a multinomial distribution. We assume that in this multinomial model there are 2p classes: for each value $y_{(j)}$ there is a response and a nonresponse class. The probabilities to be drawn from each class are as follows:

1. For the response class corresponding to $y_{(j)}$: $P[i\in U_j \wedge i\in r] = P[i\in U_j]P[i\in r \mid i\in U_j] = f_j v_j$.

2. For the nonresponse class corresponding to $y_{(j)}$: $P[i\in U_j \wedge i\in s\backslash r] = P[i\in U_j]P[i\in s\backslash r \mid i\in U_j] = f_j(1-v_j)$.

Here the f_j is the density of $y_{(j)}$ as defined in (5.2.1) and v_j is the

average response propensity for elements in U_j as defined in (5.2.12).

Under the assumption of this multinomial sampling model, we find that the probability to draw α_j item respondents and β_j item nonrespondents from U_j in s, for j=1,...,p, is given by

$$f(\{\alpha_j\},\{\beta_j\}|\{v_j\},\{f_j\}) = \frac{(\alpha+\beta)!}{\alpha_1!\ldots\alpha_p!\beta_1!\ldots\beta_p!} \prod_{j=1}^{p} (f_j v_j)^{\alpha_j} [f_j(1-v_j)]^{\beta_j}. \tag{5.4.1}$$

The α_j and β_j are the sufficient statistics in this model. Of course, the β_j are unknown. The likelihood which describes the observed data can be obtained by summing (5.4.1) over all admissible combinations $\beta_1,\ldots,\beta_p$ which sum to β, i.e. the total number of nonrespondents in the sample. We then obtain

$$f_{obs}(\{\alpha_j\},\beta|\{v_j\},\{f_j\}) = \frac{(\alpha+\beta)!}{\alpha_1!\ldots\alpha_p!\,\beta!} [\prod_j (f_j v_j)^{\alpha_j}](1 - \sum_j f_j v_j)^{\beta}. \tag{5.4.2}$$

It is obvious from (5.4.2) that the f_j and v_j are not simultaneously identifiable. However, if either set of parameters is known the other can be identified. We shall assume now that we are able to estimate the response propensities v_j from a previous and similar survey.

Once we have obtained (5.4.2) we can proceed either in a non-Bayesian or in a Bayesian fashion. The latter approach is taken in Chiu and Sedransk (1986). It requires specification of suitable priors for the f_j and v_j. They suggest the use of a Dirichlet distribution for the f_j and beta distributions for the v_j, as natural conjugate priors for the likelihood (5.4.2). (But other priors are possible as well, of course.) We refer the reader to the Chiu and Sedransk paper for further information about this Bayesian approach.

We shall consider an alternative approach, which is based on ML. Estimating the f_j with ML from (5.4.2) under the constraints $\Sigma_j f_j = 1$ and

$f_j \geq 0$, for $j=1,\ldots,p$, can be viewed as an estimation problem for incomplete tables. Such problems are treated in Bishop et al. (1980, chapters 5 and 6). In particular sufficient and necessary conditions are discussed which guarantee the existence and uniqueness of ML-estimators for incomplete tables for various sampling models, among which multinomial models (see Bishop et al., 1980, p. 186). In our case we only have to require that the values of the α_j and β are all greater than 0, in order to assure the existence and uniqueness of the ML-estimator $(f_1^*,\ldots,f_p^*)$ for $(f_1,\ldots,f_p)$. for our multinomial model (5.4.2).

In order to calculate $(f_1^*,\ldots,f_p^*)$ we form a Lagrangean function of the logarithm of f_{obs} and the constraint $\Sigma_j f_j = 1$, and apply the Lagrange multiplier theorem. We find then that the (unique) stationary point satisfies

$$\alpha_j - \frac{\beta f_j^* v_j}{1 - \Sigma_j f_j^* v_j} - \lambda f_j^* = 0 \quad , \tag{5.4.3}$$

for $j=1,\ldots,p$ and a multiplier $\lambda \in \mathbb{R}$. Summing (5.4.3) over j and solving the resulting equation for $\Sigma_j f_j^* v_j$ yields

$$\sum_{j=1}^{p} f_j^* v_j = \frac{\alpha - \lambda}{\alpha + \beta - \lambda} \quad , \tag{5.4.4}$$

and substituting this into (5.4.3) and then solving the resulting equation for f_j^* gives

$$f_j^* = \frac{\alpha_j}{v_j(\alpha + \beta - \lambda) + \lambda} \quad . \tag{5.4.5}$$

Summing (5.4.5) over j and using $\Sigma_j f_j^* = 1$ yields the following polynomial equation for λ:

$$\Sigma_{j=1}^{p} \frac{\alpha_j}{v_j(\alpha + \beta - \lambda) + \lambda} = 1 \; . \tag{5.4.6}$$

We are interested in the unique solution $\lambda \in \mathbb{R}$ of (5.4.6) which yields, when

substituted into (5.4.5), a value $f_j^* \in [0,1]$ for $j=1,\ldots,p$. Because of these constraints, we find the following necessary conditions for this root λ of (5.4.6):

$$\max_{1\le j\le p} \left\{ \frac{\alpha_j - v_j(\alpha+\beta)}{1 - v_j} \right\} \le \lambda \ , \tag{5.4.7}$$

and

$$\min_{1\le j\le p} \left\{ \frac{v_j(\alpha+\beta)}{1 - v_j} \right\} > -\lambda \ . \tag{5.4.8}$$

To illustrate the behaviour of the ML-solutions for (5.4.2), we consider the case that p=2 in the following example.

Example 5.4.1 For p=2 equation (5.4.6) equals

$$1 = \frac{\alpha_1}{v_1(\alpha+\beta-\lambda)+\lambda} + \frac{\alpha_2}{v_2(\alpha+\beta-\lambda)+\lambda} \ , \tag{5.4.9}$$

which can also be expressed as

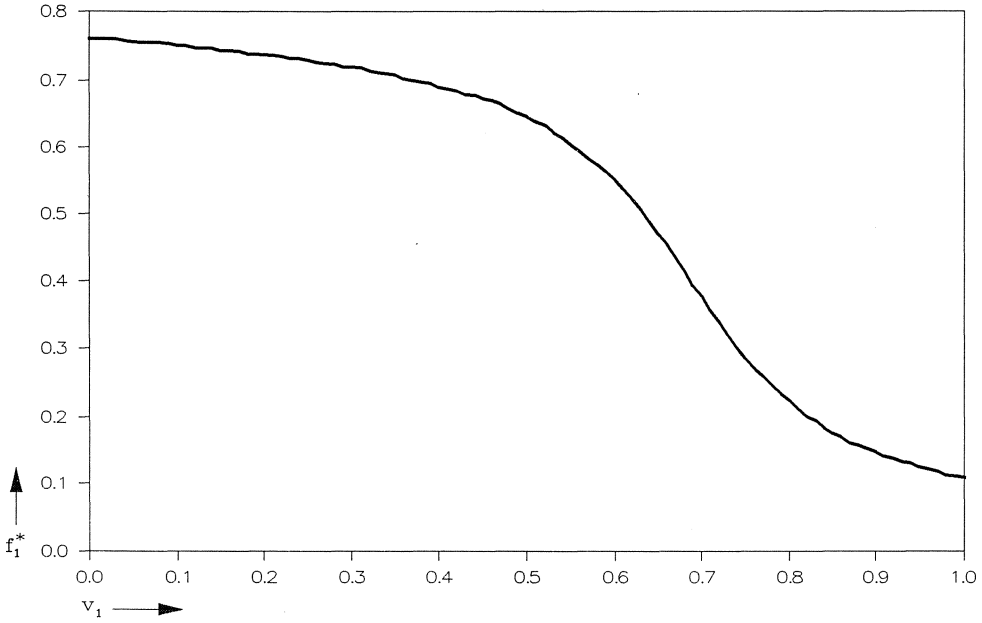
$$w_1w_2\lambda^2 + \{\gamma(v_1w_2+w_1v_2)-\alpha_1w_2-\alpha_2w_1\}\lambda + \gamma^2v_1v_2-\gamma(\alpha_1v_2+\alpha_2v_1) = 0 \ , \tag{5.4.10}$$

where $w_i=1-v_i$ $(i=1,2)$ and $\gamma=\alpha+\beta$. It is easily verified that the discriminant D of (5.4.10) can be written as

$$D = \{(\alpha_1-\gamma v_1)w_2 - (\alpha_2-\gamma v_2)w_1\}^2 + 4w_1w_2\alpha_1\alpha_2 \ , \tag{5.4.11}$$

which is nonnegative for any valid combination of parameter values. Therefore (5.4.10) has only real roots. Denote these roots by λ^- and λ^+ respectively, where $\lambda^- \le \lambda^+$. It can be verified that only λ^+ yields acceptable estimates for (f_1,f_2). In exhibit 5.4.1 it is shown how the value of the ML-estimate f_1^* depends on v_1, for given values of α_1, α_2, β and v_2.

Exhibit 5.4.1 f_1^* as a function of v_1 for α_1=30, α_2=50, β=200 and v_2=0.7



Note that the function considered in exhibit 5.4.1 is of particular interest in order to gain an insight in the stability of the ML-estimate f_1^* as a function of v_1 and v_2. These quantities are in general not exactly known, but have to be estimated (see below), contrary to the other parameters involved. ///

We can also use the EM-algorithm to estimate the f_j^* (see e.g. Dempster et al., 1977). This algorithm alternately determines the expectations of the sufficient statistics β_j for assumed values of the f_j, and calculates the ML values for the f_j, for assumed values of the β_j. In fact we have the following iteration steps:

$$\beta_j^{(n+1)} = \frac{f_j^{(n)} w_j}{\sum_j f_j^{(n)} w_j} \beta \qquad \text{(E-step)} , \tag{5.4.12}$$

where w_j=1-v_j, and

$$f_j^{(n+1)} = \frac{\alpha_j + \beta_j^{(n)}}{\alpha + \beta} \qquad \text{(M-step)} , \qquad (5.4.13)$$

where the superscripts denote the iteration step numbers. The iteration is started by assuming suitable initial $\beta_j^{(0)}$ values. The interpretation of the $\beta_j^{(n)}$ is that of the number of item nonrespondents allocated to subpopulation U_j at the n-th iteration step. Similarly $f_j^{(n)}$ is the density of y at step n. Because of the strict concavity of (5.4.2) this iteration converges to unique points $(f_1^*,\ldots,f_p^*)$ and $(\beta_1^*,\ldots,\beta_p^*)$. Empirical evidence indicates a rather high rate of convergence. Therefore the EM-algorithm offers an attractive method to solve this ML-problem. Note also that the iteration defined by (5.4.12) and (5.4.13) and suitable initial values is very intuitive.

The model (5.4.2) can be readily extended to the following parametric case. Suppose that x is a discrete auxiliary variable which has been observed for all elements in the sample s. We can use x as a stratification variable for the stratification of s in a finite number of strata. Let α_{jk} denote the number of elements in s for which the y-value equals y_j and the x-value equals $x_{(k)}$, and $\beta_{.k}$ the number of elements for which the y-value is missing and the x-value equals $x_{(k)}$. Then, in the same fashion as above, we find that the likelihood describing the observed data is proportional to

$$[\prod_{j,k}(f_{jk}v_{jk})^{\alpha_{jk}}](1 - \sum_{j,k}f_{jk}v_{jk})^{\beta_{.k}} , \qquad (5.4.14)$$

where the f_{jk} and v_{jk} are similarly defined as the f_j and v_j, except that they are conditional on the x-value $x_{(k)}$. The ML-estimators for the f_{jk}, for given v_{jk}, can be obtained using e.g. the EM-algorithm , resulting in a similar iteration as presented above.

6. IMPUTATION IN SURVEY DATA PROCESSING

6.1 Introduction

In the present chapter we consider imputation in the context of survey data processing. The imputation system as we imagine it here receives input in the form of incomplete, but partially corrected, records. It outputs completed records, which also satisfy the constraints imposed by the logical structure of the corresponding questionnaire.

The circumstance that the imputation is assumed to be carried out in a survey data production process implies that a number of practical limitations and requirements are to be considered. These limitations are more severe if the amount of data to be processed is large, or time pressure is high. In order to have a higher chance to complete a data set in time, relatively easy imputation models should be used, which can also be implemented efficiently. It is furthermore wise to develop an imputation system in such a way that external information (densities from which values will be sampled, for example) can be easily plugged in. It should also be arranged that certain information used by the system (such as the densities just mentioned) can be readily inspected by experts.

For a statistical office it might be advantageous to contemplate the development (or purchase) of a software package to generate the 'shells' of imputation programs. (It is perhaps possible to develop a questionnaire design system with such a facility.) The specifications for an imputation program, required for the generation of such a shell, are provided by subject matter experts. After it has been specified, such a shell has to be filled with data (like an expert system) before it is operational. These data consist for instance of certain densities or parameters of distributions. Of course such a system is only useful if it allows the implementation of imputation models which are interesting enough from a statistical point of view.

In section 6.2 we shall consider some statistical aspects of imputation. The intention of that section is not to give any new imputation method: a

sizeable amount of literature discussing such models already exists, and it is unnecessary for our purposes to add yet another one. Our aim is to stress the significance of discriminant analysis as a convenient framework for considering imputation. Furthermore we want to stress, following Rubin (1987), the importance of the application of a *sensitivity analysis*, such as multiple imputation, within an imputation procedure applied in a survey data production process. This seems to be insufficiently appreciated in practice. Such a sensitivity analysis pertains to two aspects:

1. The variability of imputed values under repeatedly applying one and the same imputation model.

2. The influence of the application of different (plausible) imputation models on the resulting imputed values.

In section 6.3 we discuss some computational aspects of imputation. These are related to the testing of the acyclicity of a set of imputations, to the order in which a set of imputations can be carried out, and to the generation of complete records which satisfy the constraints due to the logical structure of the questionnaire.

6.2 Statistical aspects of imputation

Let y be a categorical variable which is an imputation variable. Let x be an auxiliary variable, or a vector of auxiliary variables, to y. We adopt the notation from chapter 5 in this section.

Let x_i denote the value of the auxiliary variable x for item nonrespondent i. A reasonable procedure to classify i into a suitable subpopulation is the following. Classify i into that U_j for which

$$P[i \in U_j \mid x = x_i \wedge i \in s \backslash r] \tag{6.2.1}$$

is maximal. In order to be able to calculate (6.2.1) we have to introduce

some assumptions with respect to the response mechanism. Suppose that it seems reasonable to assume that for each subpopulation U_j it holds that the probability to respond depends only on y, and therefore, in particular, is independent of the covariate x. Hence it follows that for an $i \in U_j$ the events $[x=x_i]$ and $[i \in s \backslash r]$ are independent, conditional on the event $[i \in U_j]$, i.e.

$$P[x=x_i \wedge i \in s \backslash r \,|\, i \in U_j] = P[x=x_i \,|\, i \in U_j]\, P[i \in s \backslash r \,|\, i \in U_j] \ . \qquad (6.2.2)$$

Rewriting (6.2.1) with Bayes' rule and using (6.2.2), we find that

$$P[i \in U_j \,|\, x=x_i \wedge i \in s \backslash r] \simeq P[x=x_i \,|\, i \in U_j] w_j f_j \ , \qquad (6.2.3)$$

where $w_j = P[i \in s \backslash r \,|\, i \in U_j] = 1 - v_j$ and $f_j = P[i \in U_j]$, as have been introduced in chapter 5, and where $\simeq$ denotes proportionality. The v_j and the f_j can be estimated with one of the methods discussed in chapter 5. Furthermore the density $P[x=x_i \,|\, i \in U_j]$ should be known or estimated from the observed data. One can also postulate a parametric model for this conditional probability, for each j, and estimate the parameters from the data.

The classification criterion based on (6.2.1) as given above is deterministic. It can be replaced, however, by a randomized variant. In such a variant i is not allocated to a subpopulation U_j such that $P[i \in U_k \,|\, x=x_i \wedge i \in s \backslash r]$ is maximal. Instead, i is assigned to a subpopulation U_j with probability $P[i \in U_j \,|\, x=x_i \wedge i \in s \backslash r]$. All these classification procedures can proved to be admissible, in the sense that there are no 'better' classification rules (see appendix C for a precise definition of admissibility).

Remark 6.2.1 If a randomized classification rule is used, measures for its quality are provided by the probabilities to correctly classify nonrespondents, or alternatively, to misclassify them. They can be obtained as follows. Suppose that we have a nonrespondent i from U_j. Let $b_j(x_i) = P[x=x_i \,|\, i \in U_j]$. The probability that i will be classified as a member of U_k is given by

$$P[i \to U_k \mid i \in U_j \wedge i \in s \backslash r] = \int_{R_x} \frac{b_j(x) b_k(x) w_k f_k}{\Sigma_l \, b_l(x) w_l f_l} \, d\mu(x) \ , \qquad (6.2.4)$$

where μ denotes Lebesgue measure in case x is continuous, or a counting measure in case x is discrete; R_x is the domain of x. Denote the probabilities in (6.2.4) by φ_{jk} and the p×p matrix of these elements by Φ. A criterion for the quality of the stochastic imputation methodology is the degree in which it, on the average, classifies item nonrespondents correctly, i.e. $tr(\Delta\Phi)$, where Δ is the p×p diagonal matrix with elements $w_i f_i$ on the diagonal (cf. appendix C for background information). ///

In case y is a continuous variable the methodology suggested above can be applied, after y has been discretized. The imputation procedure should then be applied in two steps: in the first step a category selected to which an $i \in s \backslash r$ is likely to belong, using the methodology above. In the next step an y-value y^* in this category is randomly drawn. This value y^* will be imputed for y_i. It is also possible, and more straightforward, to assume a parametric model for y and to estimate its parameters from the data. But this may require a lot of computational effort.

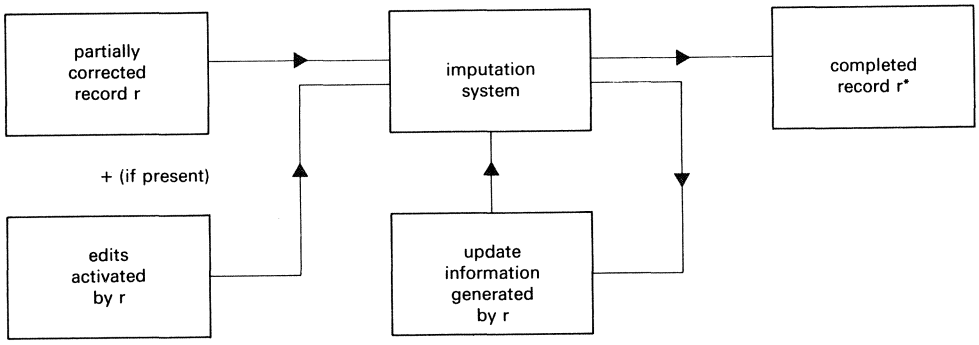
For any imputation procedure which is to be applied, it is assumed that it respects the constraints dictated by the logical structure of the corresponding questionnaire. This matter is considered in the next section.

It should be stressed that applying an imputation method is not without any risk. Important dangers are the overestimation of the precision of estimators and the distortion of relationships between variables. In fact one should be strongly aware of the fact that a missing value is in fact a random variable with an unknown distribution. Therefore it is laudable to investigate the sensitivity of the results of an imputation procedure by applying multiple imputations (cf. Rubin, 1987). This amounts to applying at least two imputation models, and to impute at least two values to each missing value for each imputation model. Ideally, all these imputed values should be retained in the completed file. From a practical point of view this is less attractive because it increases the size of the completed data set.

6.3 Computational aspects of imputation

An imputation system in a survey data processing situation, viewed as a black box, can be imagined to operate as follows (cf. exhibit 6.3.1). As input for such a system we have a partially corrected record r plus a set of edits activated by r. The output of such a system is a completed record r^*, and update information for a monitoring system.

Exhibit 6.3.1 Schematic view of an imputation system



The completion of a partially corrected record is either uniquely determined by r or not. If it is uniquely determined by r, then a series of *deterministic imputations* or *derivations* has been applied. Formally a derivation of v_{k+1} from $v_1,\ldots,v_k$, with $v_1<\ldots<v_k$ ('<' is the strict order derived from a topological sort '$\leq$' of the associated routing graph), is a function

$$g: R_1^* \times \ldots \times R_k^* \rightarrow R_{k+1} \ , \qquad (6.3.1)$$

where the R_i^* are certain subsets (to be explained below) of the domains R_i of v_i, i=1,...,k.

If there are several possible completions of r, such as is usually the case, one or more (*stochastic*) *imputations* should be applied. These stochastic imputations use a statistical model to assign values to missing ones. An example of such a technique is hot-deck imputation (cf. Ford, 1983). Formally, a stochastic imputation of v_{k+1} from $v_1,\ldots,v_k$ with $v_1<\ldots<v_k$ is a function

$$g: R_1^* \times \ldots \times R_k^* \times \Omega \rightarrow R_{k+1} \ , \qquad (6.3.2)$$

where the R_i^* are similar as above, and (Ω,Σ,μ) is a probability space. So for each point $(a_1,\ldots,a_k)\in R_1^*\times\ldots\times R_k^*$, $g(a_1,\ldots,a_k,\cdot)$ is a random variable.

Without loss of generality, we have assumed that the codomains of a stochastic imputation are univariate. In practice this is what is probably simplest to apply, but for the theory itself as it will be presented here, this is of little importance.

As to the R_i^* the following. Let $G=(V,E)$ be the routing graph of the corresponding questionnaire and let $W=\{v_1,\ldots,v_k\}\subset V$ be given, where $v_1<\ldots<v_k$. Let A be the adjacency matrix of G. A set F of pairs $\{(v,R_{v,u})\in E \mid v\in W,\ u\in V,\ R_{v,u}$ transition set$\}$ is called the *vector field* on W in G, if the following conditions are satisfied by the elements in F:

1.a. If $(v_i,R_{v_i,u})\in F$ with $i\leq k-1$, then there is a path from v_i to v_{i+1} which cuts u.

b. If v_k is not the sink of G, then $(v_k,R_{v_k,u})\in F$ for any u such that (v_k,u) is an edge in G.

c. If v_k is the sink of G, then the corresponding transition set is the domain R_{v_k} of v_k.

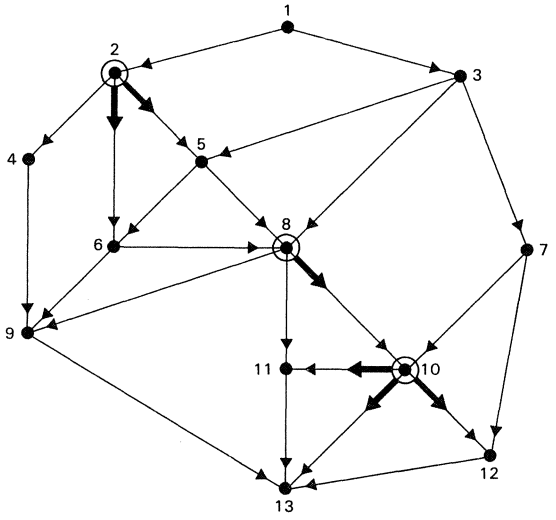
2. F is maximal, i.e. if a pair $(v_i,R_{v_i,u})\in W\times\{$transition sets $R_{i,j}$ associated with G$\}$ satisfies one of the conditions in 1 then $(v_i,R_{v_i,u})\in F$.

In exhibit 6.3.2 an example of a vector field is given. The arrows associated with the vertices in W represent the corresponding transition

sets. Intuitively, the vector field on $W \subset V$ in a routing graph G contains, for each $v \in W$, the largest set of values in the domain R_v which the variable v, lying on a path in Π which cuts W, can take.

For a set of vertices W in a routing graph G consider the vector field F on W in G. For each $v_i \in W$, R_i^*= is the union of all transition sets $R_{v_i,u}$ such that $(v_i, R_{v_i,u}) \in F$. We call $R_1^* \times \ldots \times R_k^*$ the *characteristic set* of F.

Exhibit 6.3.2 Illustration of a vector field in a routing graph



Remark: W={2,8,10}

Algorithm 6.3.1 indicates how to construct the vector field on a set W of vertices in a routing graph G, as well as its characteristic set.

Algorithm 6.3.1. (Calculation of a vector field and its characteristic set)

Let G=(V,E) be a routing graph with n vertices, with adjacency matrix A. Let R_{ij} denote the transition set corresponding to the edge (i,j). Let $W=\{v_1,\ldots,v_k\} \subset V$, with $k>1$ and $v_1<\ldots<v_k$. Let $v_{k+1}=n$ be the sink of G.

The vector field on W in G and its characteristic set are calculated as follows:

1. Calculate the transitive closure A^* of A.

2. For each pair (v_i, v_{i+1}), $1 \leq i \leq k-1$, determine the set of all outgoing edges (v_i, u) from v_i such that v_{i+1} can be reached from u, i.e. such that $A^*_{u, v_{i+1}} = 1$. For $v_k \neq n$, the sink of G, determine all outgoing edges (v_k, u) from v_k. With each edge (v_i, u) found in this way associate the pair $(v_i, R_{v_i, u})$. If $v_k = n$ then associate with with v_k the pair (v_k, R_{v_k}). All these pairs of vertices and transition sets form the vector field F on W in G.

3. By taking the union of the transition sets associated with the outgoing edges of the elements v_i in W one obtains the components R_i^* of the characteristic set of the vector field on W. ///

The deterministic and stochastic imputations introduced above can be related to database theory. In case of deterministic imputation we can say that there is a *functional dependency* between the imputation variable v_{k+1} and the auxiliary variables $v_1, \ldots, v_k$ (cf. Ullman, 1982, pp. 213 ff.). It is assumed that v_{k+1} is not functionally dependent on any other auxiliary variable. A stochastic imputation can be considered to be an extension of a *multivalued dependency* (cf. Ullman, 1982, pp. 243 ff.), in which the auxiliary variables $v_1, \ldots, v_k$ determine a set of values which v_{k+1} can take. In our case this would be the carrier set of the image of the domain $R_1^* \times \ldots \times R_k^* \times \Omega$ under g in (6.3.2), i.e. the set of values in R_{k+1} which can be assumed by v_{k+1} with probability greater than 0. In fact the only thing that is added is a probability to give the 'possible values' of v_{k+1} different weights. The important thing is that in both the deterministic and the stochastic imputation case there is a dependency of the values an imputation variable can take. This value depends in one way or another on the values of the associated auxiliary variables (and on nothing else).

In order to investigate whether a set of imputations is correctly defined, we introduce the following formalism. With each imputation which has been specified we associate an *imputation triple* (g_i, W_i, V_i), in which g_i denotes the i-th imputation, W_i is the set of auxiliary variables for

this imputation, and V_i is the set containing the imputation variable of this imputation. (For our purposes below it is not strictly necessary that such a set V_i contains exactly one element, but we shall assume it in view of the convention adopted above with respect to imputations.)

An evident requirement for an imputation triple is that $W_i \cap V_i = \emptyset$. Furthermore, like edits, imputations should be *activatable*, which means that the variables in $W_i \cup V_i$ should lie on a path in the routing graph. Another requirement, which does not have an equivalent with respect to the edits, is that if there are two imputation triples (f_i, W_i, V) and (f_j, W_j, V) then they should not be simultaneously activatable in order to avoid ambiguities. That means that, in this case, there should be no path in the routing graph which cuts $W_i \cup W_j \cup V$.

We can define a partial order on the set of imputation triples associated with a questionnaire, as follows:

$$(g_i, W_i, V_i) < (g_j, W_j, V_j) , \tag{6.3.3}$$

if and only if

1. (g_i, W_i, V_i) and (g_j, W_j, V_j) are simultaneously activatable.

2. $V_i \cap W_j \neq \emptyset$.

The interpretation of (6.3.3) is that if $(g_i, W_i, V_i) < (g_j, W_j, V_j)$ then g_i should be applied prior to g_j. This order structure defines a directed graph J on the set of imputation triples. Another requirement for a set of imputation triples is that J is acyclic, otherwise certain imputations can never be carried out because the necessary background information is (partly) lacking and will never be supplied by the application of some other imputations. That is, there could be a deadlock situation. Note that this acyclicity requirement for J implies that $W_i \cap V_j = \emptyset$ for any pair of imputation triples satisfying (6.3.3).

Acyclicity of J can be easily checked by applying a topological sort to J (cf. appendix A). If this topological sort cannot be completed, J is cyclic; otherwise J is acyclic.

Remark 6.3.1 This formalism for imputation triples could be adapted so as to obtain a formalism for the routing structure of non-Markovian questionnaires. ///

Now suppose that a partially corrected record with several missing values is entered into the imputation system. The first thing that has to be considered is whether an imputation process can be started, i.e. whether any of the imputations required can indeed be carried out. Therefore it should be verified that the required imputations have indeed been defined for the imputation variables. If not, the record cannot be totally completed by the system. It can still proceed completing as much of the record as possible and then transfer the record to a special file for inspection by some subject matter expert; or it may be dropped altogether, because too much information is lacking.

The order in which the imputations should be applied to a partially corrected record r', is determined by the partial order structure embodied in J. To test whether r' can be totally completed, and if so, in which order which imputations should be applied, proceed as follows. First identify the variables with missing values in the record. Then check that for each of these variables an imputation triple (g_i, W_i, V_i) is available. If not, the record cannot be completed. If so, it can. In this latter case an imputation triple (g_i, W_i, V_i) has one of the following properties.

1. All auxiliary variables in W_i have regular values in r', or
2. There is at least one variable in W_i with a missing value in r'.

If an imputation g_i has the first property it can be applied; in the second case it cannot. Carry out all imputations which have the first property, and check that the resulting values satisfy the constraints imposed by the logical structure. Assuming that this can be carried out without any problems, we have created either a completed record or a new partially

corrected record r''. Repeating the arguments above for r'' instead of for r', and so on, the process will finally yield a completed record. We leave the details to the reader.

7. CONCLUSIONS

In this book a theory was developed for the processing of survey data. The motivation behind this theory is a different conception of a survey data production process. In this conception the questionnaire plays a very important role, viz. as an object containing the relevant defining information on questions, answers, routing, edits, etc.

The outlook in this book is rather theoretical: the purpose in writing it was to develop a theoretical framework for the processing of survey data, rather than a set of ready-for-use recipes. Of course, the final aim is still to solve the practical problems with respect to the processing of survey data. Therefore the best thing to do now seems to be to put some of the ideas that have been suggested in this book to practice. Furthermore there are several points of theoretical interest left for further study. In this final chapter we make some suggestions for future research, both practically and theoretically oriented.

Although we have only considered testing the formal structure of a questionnaire (and showed that this, in a sense, is difficult), it is certainly not the only thing that has to be tested in a questionnaire. Testing pragmatic aspects of a questionnaire, i.e. with respect to its use in a survey, is also important. The same holds for making the logical structure visible for a questionnaire designer. This means that, among other things, a questionnaire design system should contain a module which draws pictures of a routing graph, very much like exhibit 1.3.1 (which, however, is man-made !).

It would be fruitful to try and find special cases for which the test problems considered in chapter 2, and the edit error localization problem permit tractable solutions. This can only result from a closer study of the logical structures of questionnaires which are used in practice. Furthermore it is of considerable significance to search for good approximation or randomization techniques, in order to be able to cope with these edit error localization problems in practice.

It would also be interesting to find out to what extent non-Markovian questionnaires are useful or necessary in practice. Such questionnaires are most likely to be found in CATI or CAPI surveys, but not in PAPI surveys. From a theoretical point of view it is undoubtedlyly interesting to generalize the theory in this book so as to encompass non-Markovian questionnaires. In chapter 6 the attention is drawn to a connection between imputations and non-Markovian questionnaires (cf. remark 6.3.1).

Another point of research could be the investigation of the validity of the assumed property 1.4.1 with respect to the logical structure of questionnaires. This requirement seems very plausible and in the present theory it is convenient, because it allows that the data editing process can be neatly divided into three procedurally independent steps. In particular checking and correcting of the routing structure can be separated from edit error localization and correction. Although this is nice and convenient, it is insufficient to justify its introduction as a requirement for the logical structure of a questionnaire.

Finally we remark that it is interesting to study data bases with missing values more closely, especially when the missing values are interpreted as either sets or random variables (cf. section 4.3). These interpretations of missing data provide intuitive models for many-valued and modal logic respectively.

APPENDIX A. SOME GRAPH ALGORITHMS

In this appendix some fundamental graph algorithms are discussed, which can be considered as basic tools for others appearing in the main text. These basic algorithms are given here for convenience of the reader. Most of them can also be found in a textbook such as Aho et al. (1983). The algorithms are given either in a pseudo-PASCAL or by a rather informal description. The rigour thus sacrificed is compensated by a gain in brevity. For more extensive background information on the algorithms to be discussed, the reader is directed to the reference cited above. It should be remarked that all algorithms discussed in the appendix run in polynomial time. The following (di)graph algorithms are discussed below.

1. Depth-first search (DFS) of a digraph.
2. Topological sort of an acyclic digraph.
3. Calculation of connected components in a graph.
4. Calculation of strong components in a digraph.
5. Transitive closure in a digraph.
6. Calculation of all-pairs shortest paths in a digraph.
7. Calculation of cut-points.

Each of the above-mentioned algorithms will be treated in a separate section. In the algorithms below we shall assume that a (di)graph is represented either as an adjacency matrix or as an adjacency list (=edge list). In this latter representation singly linked lists are used to represent the vertices adjacent to each of the vertices in the (di)graph. For more information we refer the interested reader to a book on data structures, such as Aho et al. (1983, sections 6.2 and 7.1)

We use G=(V,E) to denote a (di)graph, where V denotes the set of vertices in G, containing n elements, and E the set of edges, which are either directed or undirected. Let A denote the adjacency matrix of G of order n, the number of vertices in G. The matrix A can either be considered as a (numeric) 0-1 matrix or as a boolean matrix. An undirected graph can be viewed as a special case of a directed graph, namely as a graph with a symmetric adjacency matrix. L is used to denote an adjacency

list representation of G. For a vertex $v \in V$, L[v] is the list containing all vertices w in G such that (v,w) is an edge in G.

A.1 Depth-first search

Depth first search (DFS) is a method for systematically traversing (di)graphs. It proceeds searching in a (di)graph in a forward direction as long as possible. More precisely, assume that the vertices in G are initially marked unvisited. DFS operates by selecting one vertex v of G as a start vertex, which is marked visited. Then each unvisited vertex adjacent to v is searched in turn, using DFS recursively. Once all vertices that can be reached from v have been visited, the search of v is complete. If some vertices remain unvisited, we select an unvisited vertex as a new start vertex. This process is repeated until all vertices in G have been visited.

We shall give the skeleton algorithm listed in Aho et al. (1983, pp. 215 ff.). This skeleton algorithm is used in many graph algorithms in which systematic traversal is important. Examples are algorithms to calculate a topological sort of an acyclic digraph (cf. section A.2), the connected components of an (undirected) graph (cf. section A.3), and the strong components of a digraph (cf. section A.4).

In the DFS algorithm below, let 'mark' be an array of length n, whose elements are chosen from the set {visited,unvisited}. It can be used to determine whether a vertex has previously been visited. Initially the elements in mark are 'unvisited'.

```
procedure DFS (v: vertex);

var w: vertex;

begin (* DFS *)
  mark[v]:= visited;                                          (A.1)
  for w ∈ L[v]
    do
      if mark[w]= unvisited
      then DFS(w)
end; (* DFS *)
```

A.2 Topological sort

Let G=(V,E) be an acyclic digraph, with $|V|=n$. A topological sort is a bijective map $s:V\rightarrow\{1,\ldots,n\}$ such that $s(i)<s(j)$ if $(i,j)\in E$. The DFS algorithm in section A.1 can be used to yields a topological sort. To this end introduce a 'writeln(v)' statement immediately after the for-loop in algorithm (A.1). Start the algorithm at a source of G, i.e. a vertex with indegree 0. As a result, the vertices in G are printed in a reverse topological order. So to obtain a topological sort s of G, associate n with the first printed vertex, n-1 with the second printed vertex, etc.

A.3 Connected components

Calculating the connected components in a graph G can be done with the help of DFS, as remarked in section A.1. Start a DFS at a vertex of G. If upon termination of the search of G every vertex of G has been visited, G consists of one connected component, i.e. G is connected. If not every vertex has been visited, G is disconnected and consists of at least two components. The DFS process can be started again at an unvisited vertex. When this process has been terminated another component has been found. Repeating this procedure iteratively until all vertices of G have been visited will yield all connected components of G.

A.4 Strong components

Let G be a digraph. Let G^{-1} be the inverted digraph of G. The strong components of G can be found as follows.

1. Perform a DFS of G and number the vertices in order of completion of the recursive calls in algorithm (A.1).

2. Calculate G^{-1} from G.

3. Perform a DFS on G^{-1}, starting to search from the highest numbered vertex according to the numbering obtained in the first step. If this DFS does not reach all vertices, start the next DFS from the highest numbered remaining vertex.

4. Each time a new DFS is to be started in step 3 because there are still some unreached vertices, a new strong component in G has been determined.

A.5 Transitive closure

Warshall's (1962) algorithm can be used to calculate the transitive closure of a digraph. The transitive closure indicates between which vertices i,j in G there exists a path of length one or more. We assume the adjacency matrix A to be boolean. The algorithm is as follows:

```
procedure warshall (var C: array[1..n,1..n] of boolean;
                        A: array[1..n,1..n] of boolean);
(* the transitive closure of A is stored in C *)

var i,j,k: integer;

begin (* warshall *)
  for i:= 1 to n do
    for j:= 1 to n do                                          (A.2)
      C[i,j]:= A[i,j];
  for k:= 1 to n do
    for i:= 1 to n do
      for j:= 1 to n do
        if not C[i,j]
        then C[i,j]:= C[i,k] and C[k,j]
end; (* warshall *)
```

Note that G is acyclic if the main diagonal of C contains 0's only after having applied algorithm (A.2).

A.6 All-pairs shortest paths

Floyd's (1962) algorithm, which is a generalization of Warshall's algorithm (cf. (A.2)), can be used to solve the problem of determining the lengths of the shortest paths between every pair of vertices in a digraph. The algorithm can be adapted so as to recover the shortest path between vertices i and j as well. The algorithm is as follows.

```
procedure floydsp (var C[1..n,1..n] of real;
                       A[1..n,1..n] of real);
                   var P[1..n,1..n] of integer);

(* the lengths of the shortest paths between each pair of vertices
   is stored in C; P can be used to recover the shortest paths *)

var i,j,k: integer;

begin (* floydsp *)
  for i:= 1 to n do
    for j:= 1 to n do
      begin                                                    (A.3)
        C[i,j]:= A[i,j];
        P[i,j]:= 0
      end;
  for i:= 1 to n do
    C[i,i]:= 0;
  for k:= 1 to n do
    for i:= 1 to n do
      for j:= 1 to n do
        if C[i,k] + C[k,j] < C[i,j]
        then
          begin
            C[i,j]:= C[i,k] + C[k,j];
            P[i,j]:= k
          end
end; (* floydsp *)
```

After having calculated C and P we can apply the following short algorithm to actually calculate the shortest path from vertex i to vertex j in G.

```
procedure path (i,j: integer);

label 1;
var k: integer;

begin (* path *)
  k:= P[i,j];
  if k= 0                                                    (A.4)
  then goto 1;
  path(i,k);
  writeln(i,k);
  path(k,j);
1:
end; (* path *)
```

A.7 Cut-points

Let G=(V,E) be a routing graph, with $|V|=n$. Let $s:V\rightarrow\{1,\ldots,n\}$ denote a topological sort of G. Let furthermore $WIDTH|_s:V\rightarrow\mathbf{N}$ be the mapping defined as follows

$$WIDTH|_s(v) = \#\{(u,w)\in E \mid s(u)<s(v)<s(w)\},$$

for $v\in V$. It can be shown that $WIDTH|_s(v)=0$ if and only if v is a cut-point, for any topological sort s of G. (There are routing graphs G such that $WIDTH|_s(v) \neq WIDTH|_t(v)$, for vertices v on G and topological sorts s and t of G. Of course, such vertices v cannot be cut-points.)

So calculating the $WIDTH|_s$ for a certain topological sort s of G and identifying the vertices for which it takes the value 0, yields the cut-points of G. This method is due to Tangelder (1987).

APPENDIX B. CHARACTERIZING THE ROUTING STRUCTURE

In this appendix two characterizations of the routing structure of a questionnaire, i.e. the routing graph, are introduced. The first characterization, the 'balance', is related to the variation in lengths of the various paths through a questionnaire. The second one, called the 'complexity', measures the branchedness of a routing graph. Both measures can be useful to classify questionnaires on the basis of their routing structure. Furthermore both might be useful in assessing aspects of the quality of questionnaires as far as their routing structure is concerned.

Questionnaires with an ill-balanced routing structure might induce interviewers to suggest answers to questions which might result in shorter paths through a questionnaire. This practice is very harmful because it can generate records which are formally correct but do not apply to reality.

Questionnaires with a highly complex routing structure are more difficult to check and also increase the efforts in checking and correcting the records containing the answers of respondents. For pencil and paper questionnaires they increase the probability that an interviewer jumps to a wrong question.

Let G be a routing graph with adjacency matrix A. Let Π denote the collection of all paths in G. Let X be a random variable denoting the length of a randomly drawn path from Π. The *balance* of G is defined as the variation coefficient of X. It is denoted by $\beta(G)$, or β, if the dependence on G is implicitly assumed. That is,

$$\beta = \beta(G) = \frac{\sqrt{\text{Var}(X)}}{EX} = \frac{\sigma}{\mu} , \qquad (B.1)$$

where μ=EX and σ^2=Var(X). The expectation μ and the variance σ^2 of X can be expressed in terms of the adjacency matrix A of G as follows:

$$\mu = \mu(G) = \frac{(I-A)^{-2}_{1,n}}{\pi} - 1 \quad , \tag{B.2}$$

and

$$\sigma^2 = \sigma^2(G) = \frac{2(I-A)^{-3}_{1,n}}{\pi} - \mu^2 - 3\mu - 2 \quad , \tag{B.3}$$

where

$$\pi = \pi(G) = (I - A)^{-1}_{1,n} \tag{B.4}$$

is the total number of paths in G, i.e. the cardinality of Π. (B.2) and (B.3) have been calculated using the generating function of X.

It can be proved (see Willenborg, 1986) that if $G=G_1*G_2$, where * is the glueing operation defined in section 1.3, it holds that

$$\pi(G) = \pi(G_1)\ \pi(G_2) \quad , \tag{B.5}$$

$$\mu(G) = \mu(G_1) + \mu(G_2) \quad , \tag{B.6}$$

$$\sigma^2(G) = \sigma^2(G_1) + \sigma^2(G_2). \tag{B.7}$$

From these results the following identity for the balance follows immediately:

$$\beta = \frac{(\beta_1^2\ \mu_1^2 + \beta_2^2\ \mu_2^2)^{\frac{1}{2}}}{\mu_1 + \mu_2} \quad , \tag{B.8}$$

where $\mu_i=\mu(G_i)$ and $\beta_i=\sigma(G_i)/\mu(G_i)$, for i=1,2. These results are of practical value for the calculation of the balance of a large routing graph. Such a routing graph can be decomposed into smaller ones for which calculations are perhaps feasible. Balances and average path lengths for

each of the components can then be combined by using (B.8) repeatedly, to yield the balance of the original routing graph.

The *complexity* ρ of G is defined as the logarithm (at base 2) of the number of paths in G, i.e.

$$\rho = \log_2 \pi \ . \tag{B.9}$$

The complexity ρ has the following properties (see e.g. Willenborg, 1986):

1. $\rho(G_1 * G_2) = \rho(G_1) + \rho(G_2)$.
2. $\rho(G_1 * G_2) = \rho(G_2 * G_1)$.
3. $\rho(G^{-1}) = \rho(G)$, where G^{-1} denotes the inverse routing graph of G.
4. $\rho(G_1) \leq \rho(G_2)$ if G_1 is a sub-routing graph of G_2.
5. $\rho(G_0) = 0$ if G_0 is the point graph.
6. ρ is invariant under contractions.

The measure ρ is an absolute measure of complexity. It might be convenient to define a relative measure of complexity. This should express the complexity of a routing graph G relative to the routing graph with the highest complexity within a class of routing graphs which are naturally associated with G. One such class consists of all routing graphs with the same number of vertices as G possesses. (Another one is the class of routing graphs with the same number of vertices as the maximal contraction of G. But this one is computationally somewhat inconvenient.) We refer the interested reader to Willenborg (1986) for details and some numerical examples.

Remark B.1 Complexity measures have been introduced in various disciplines in which graph structures are used, such as chemistry (molecular structures), biology (neuronal networks) or software engineering (flow control graphs); see e.g. Karreman (1955), Rashevsky (1955), Trucco (1956), Sabidussi (1959), Moshowitz (1968a, 1968b), Marshall (1971, pp. 235 ff.), McCabe (1976), Henry et al. (1981) and Harrison (1984) for examples.

///

Apart from the balance and complexity there are other characteristics and objects of interest with respect to the routing structure in a questionnaire. We only mention the longest and the shortest path and their respective lengths as examples. Furthermore, a questionnaire designer might for example want to know whether there is a path from question v to question w, and, if there is at least one such path, how many there are in toto. Such information, which is easy to calculate, should be provided by the designing system he is using, upon his request.

APPENDIX C. DISCRIMINANT ANALYSIS

In this appendix we shall discuss some aspects of discriminant analysis (DA). This appendix is meant as a quick reference for the reader. More extensive discussions can be found in e.g. Mardia et al. (1982, ch. 11), Hand (1982) and Titterington et al. (1985, section 5.7). References to more specialized papers on DA are given below.

Let (R,Σ,μ) be a probability space. Let $\mu_1,\ldots,\mu_n$ be probability measures defined on the measurable space (R,Σ), such that the density of μ_i with respect to μ is given by $h_i: R \to \mathbf{R}$, $i=1,\ldots,n$, i.e. $d\mu_i = h_i\, d\mu$. Suppose that each h_i corresponds to a subpopulation U_i of a population $U=\cup_i U_i$. Now suppose that our task would be to classify an $x\in R$ into one of these subpopulations U_i in an 'optimal way' (to be explained below). A natural way to proceed is to propose the following decision criterion to allocate an $x\in R$:

$$\text{if } h_j(x)=\max_i h_i(x) \text{ then allocate } x \text{ to } U_j, \tag{C.1}$$

where we shall assume, here as well as in the sequel without explicitly mentioning it, that we (randomly) choose an index if there are several candidates. We shall call this criterion (C.1) the *maximum likelihood rule* for allocation of $x\in R$. This criterion gives all densities equal weight. Suppose that, additionally, we would possess (estimates of) the mixture weights f_i of the densities , which are such that $\Sigma_i f_i=1$ and $\Sigma_i f_i h_i(.)=h(.)$

for $h: R \to \mathbf{R}$ the population density of U. (So h is a finite mixture of densities h_i with mixture weights f_i.) Then we could formulate the following criterion, which also discounts the relative weights of the densities:

$$\text{if } f_j h_j(x) = \max_i f_i h_i(x) \text{ then allocate } x \text{ to } U_j. \tag{C.2}$$

Formulating this criterion in terms of probabilities we obtain:

$$\text{if } \frac{f_j h_j(x)}{\sum_i f_i h_i(x)} = \max_i \frac{f_i h_i(x)}{\sum_i f_i h_i(x)} \text{ then allocate } x \text{ to } U_j. \tag{C.3}$$

We can interpret such a quotient as a posterior density, obtained by Bayes' theorem from the mixture weights $(f_1,\ldots,f_n)$, interpreted as a prior density, and the likelihood $h(x|i)=h_i(x)$. For this reason we shall call the equivalent criteria (C.2) and (C.3) *Bayes criteria*. All three criteria (C.1) through (C.3) are *deterministic*, in the sense that if x_1, $x_2 \in R$ and $x_1=x_2$ then x_1 and x_2 will be allocated to the same subpopulation almost surely, provided that $\{x\in R|$ there are at least two indices i and j such that $h_i(x)=h_j(x)\}$ or $\{x\in R|$ there are at least two indices i and j such that $f_i h_i(x)=f_i h_j(x)\}$ (whatever applies) are neglegible with respect to the reference measure μ, which we shall assume.

We can generalize our deterministic criteria to *randomized discriminant rules* as follows. Let $\varphi_1,\ldots,\varphi_n$: $R \to \mathbf{R}$ be nonnegative functions which are measurable with respect to μ, and for which holds: $\Sigma_i \varphi_i(x)=1$ for every $x\in R$. In other words $\{\varphi_1,\ldots,\varphi_n\}$ forms a *partition of unity*. We can define the following classification rule:

$$x \text{ is allocated to } U_j \text{ with probability } \varphi_j(x). \tag{C.4}$$

Special cases of this rule are obtained when we take $\varphi_j=f_j h_j$ or $\varphi_j=h_j$. It is furthermore clear that the deterministic criteria above are special cases of (C.4) as well. To obtain e.g. (C.2) define

$$\varphi_j(x) = \begin{cases} 1 \text{ if } f_j h_j(x) = \max_i f_i h_i(x), \\ 0 \text{ otherwise.} \end{cases} \qquad (C.5)$$

Then each φ_j is almost everywhere defined on R, in view of a condition mentioned earlier, and is (R,Σ)-measurable.

We are now in a position to introduce a measure for the quality of a classification criterion. For criterion (C.4) we can calculate the probability φ_{ij} of classifying an element from U_i into subpopulation U_j as follows (note that the integrals exist):

$$\varphi_{ij} = \int_R \varphi_j(x) d\mu_i(x) = \int_R \varphi_j(x) h_i(x) d\mu(x). \qquad (C.6)$$

An element of U_i is correctly classified with probability φ_{ii} and incorrectly with the complementary probability $1-\varphi_{ii}$. The performance of a classification procedure can be summarized in terms of the φ_{ii}. In fact the set $\{\varphi_{11},\ldots,\varphi_{nn}\}$, which exists for each classification rule, can be used to partially order these rules, as follows. Let r and r' be classifications with corresponding sets of correct classification probabilities $\{\varphi_{ii}\}$ and $\{\varphi'_{ii}\}$ respectively. We shall say that r is at least as good as r' if $\varphi_{ii} \geq \varphi'_{ii}$ for i=1,...,n, and r is better than r' if at least one of these inequalities is strict. This leads to an optimality criterion called admissibility: a classification rule r is *admissible* if there is no rule r' better than r. It can be proved that rules such as (C.1) through (C.3) are admissible (see e.g. Mardia et al., 1982, th. 11.2.2). If we have mixture weights f_i available then we can formulate another optimality criterion by considering the average correct classification probability $\Sigma_i f_i \varphi_{ii}$. It can also be proved that rules such as (C.1) through (C.3) are still admissible when applying this criterion (see Mardia et al., 1982, th. 11.2.3).

Denote the average correct classification probability $\Sigma_i f_i \varphi_{ii}$ by ζ and let $\eta=1-\zeta$. If we have a random sample of n individuals (e.g. nonrespondents) from the population we may assume that the number of correctly classified among these is binomially distributed with parameters n and ζ, i.e. according to $Bin(n,\zeta)$. So the expectation of the number of

correctly classified individuals is $n\zeta$ and the variance of this number is $n\zeta(1-\zeta)$.

The discussion above was based on exact probability densities h_i and mixture weights f_i, but in practice these are often not available and have to be estimated from a sample. Especially for continuous h_i this may be a nontrivial matter (see e.g. Tapia and Thompson, 1978). The estimation of the f_i is discussed in chapter 5 of the present book. See also Dempster et al. (1977, sec. 4.3) and Titterington et al. (1985, sec. 4.3.2) for an application of the EM algorithm to estimate the mixture weights f_i. The estimation of the h_i is possible either on the basis of the response in the sample or on the basis of this response plus (a part of) the allocated nonrespondents. Which method should be chosen should depend on the confidence one has in the correctness of the allocation of the nonrespondents.

Another problem is the estimation of the (mis)classification probabilities from a sample. If they are estimated from exactly the same data that have been used to define the classification criteria then the misclassification probabilities are bound to be underestimated. A way out of this problem is by not deriving these estimates from precisely the same data, i.e. by applying a cross-validation or another nonparametric resampling procedure, such as jackknifing or bootstrapping. In Efron (1983) several such estimators are introduced and their performances are investigated in some simulations, the results of which are reported in the article. In this study the variable to be predicted is assumed to be dichotomic. One of the conclusions is that cross-validation gives nearly unbiased answers but often with very high variability, particularly if the training sample is small. The bootstrap estimators considered also have little bias in their answers and furthermore low variability. In this sense these estimators outperform cross-validation. Hand (1982, section 5.2) also gives a presentation of the problem of correctly estimating (mis)classification probabilities.

An important parametric DA technique is based on logistic regression, see e.g. Cox (1969), Anderson (1982) and Schmitz (1986). Nonparametric techniques are discussed in Broffitt (1982).

REFERENCES

(*The expressions within the square brackets refer to sections or appendices*)

Abrahamse, A.P.J. and J.T. Geilenkirchen, 1986, Finite-sample behaviour of logit probability estimators in a real data set (Report, Econometric institute, Erasmus University, Rotterdam). [5.2]

Aho, A.V., J.E. Hopcroft and J.D. Ullman, 1983, Data structures and algorithms (Addison-Wesley, Reading, Mass.). [A]

Albert, A. and J.A. Anderson, 1984, On the existence of maximum likelihood estimates in logistic regression models, Biometrika, vol. 71, 1-10. [5.2]

Amemiya, T., 1981, Qualitative response models: a survey, Journal of Economic Literature, vol. 19, 1483-1536. [5.2]

Amemiya, T., 1985, Advanced econometrics (Basil Blackwell, Oxford). [5.2]

Anderson, J.A., 1982, Logistic discrimination, in: P.R. Krishnaiah and L.N. Kanal (eds.), Handbook of Statistics, vol. 2 (North-Holland, Amsterdam), 139-168. [5.2; C]

Bethlehem, J.G. and H.M.P. Kersten, 1986, Werken met non-response, Statistische onderzoekingen M 30 (Netherlands Central Bureau of Statistics, Voorburg). [5.2; 5.3]

Bishop, Y.M.M, S.E. Fienberg and P.W. Holland, 1980, Discrete multivariate analysis: theory and practice (MIT Press, Cambridge, Mass.). [5.4]

Broffitt, J.D., 1982, Nonparametric classification, in: P.R. Krishnaiah and L.N. Kanal (eds.), Handbook of Statistics, vol. 2 (North-Holland, Amsterdam), 139-168. [C]

Cassel, C.-M., C.-E. Särndal and J.H. Wretman, 1983, Some uses of statistical models in connection with the response problem, in: Madow and Olkin (1983, vol. 3, 143-160). [5.3]

Chiu, H.Y. and J. Sedransk, 1986, A Bayesian procedure for imputing missing values in sample surveys, Journal of the American Statistical Association, vol. 81, 667-676. [5.4]

Codd, E.F., 1975, Understanding relations (instalment #7), FDT (bulletin of ACM SIGMOD), vol. 7, 23-28. [4.3; 4.3.1]

Cox, D.R., 1969, The analysis of binary data (Chapman and Hall, London). [C]

Date, C.J., 1983, An introduction to database systems, volume 2 (Addison-Wesley, Reading, Mass.). [4.3]

Dempster, A.P., N.M. Laird & D.B. Rubin, 1977, Maximum likelihood from incomplete data via the EM algorithm (with discussion), Journal of the Royal Statistical Society, vol. 39, Ser. B, 1-38. [5.4; C]

Efron, B., 1983, Estimating the error rate of a prediction rule: improvemenent on cross-validation, Journal of the American Statistical Association, vol. 78, 316-331. [C]

Fellegi, I.P. & D. Holt, 1976, A systematic approach to automatic edit and imputation, Journal of the American Statistical Association, vol 71, 17-35. [3.1; 3.3; 3.4; 3.4.1; 3.4.2]

Floyd, R.W., 1962, Algorithm 97: shortest path, Communications of the ACM, vol. 6, 345. [A]

Ford, B.L., 1983, An overview of hot-deck procedures, in: Madow, Olkin and Rubin (1983), 185-207. [6.3]

Garey, M.R. and D.S. Johnson, 1979, Computers and intractability : a guide to the theory of NP-completeness (W.H. Freeman & Co, San Francisco). [2.1; 2.2; 3.4.1]

Garfinkel, R.S., 1979, An algorithm for optimal imputation of erroneous data (Working paper, University of Tennessee, Knoxville). [3.1; 3.4; 3.4.1; 3.4.2]

Garfinkel, R.S. and G.L. Nemhauser, 1972, Integer programming (Wiley, New York). [3.4.1; 3.4.3]

Gourieroux, C. and A. Monfort, 1981, Asymptotic properties of the maximum likelihood estimator in dichotomous logit models, Journal of econometrics, vol. 17, 83-97. [5.2]

Good, I.J., 1965, The estimation of probabilities (MIT Press, Cambridge, Mass.). [3.4.3]

Hájek, J., 1981, Sampling from a finite population (Marcel Dekker, New York). [5.2]

Hand, D.J., 1982, Kernel discriminant analysis (Research Studies Press/Wiley, Chichester). [C]

Harrison, W.A., 1984, Applying McCabe's complexity measure to multiple-exit programs, Software-practice and experience, vol. 14, pp. 1004-1007. [B]

Hartley, H.O. and J.N.K. Rao, 1968, A new estimation theory for sample surveys, Biometrika, vol. 55, 547-557. [5.1]

Hartley, H.O. and J.N.K. Rao, 1969, A new estimation theory for sample surveys, II, in: N.L. Johnson and H. Smith (eds.), New developments in survey sampling (Wiley-Interscience, New York), 147-169. [5.1]

Henry, S., D. Kafura and K. Harris, 1981, On the relationship among three software metrics, Performance Evaluation Review, vol. 10, pp. 81-88. [B]

Hopcroft, J.E. and J.D. Ullman, 1969, Formal languages and their relation to automata (Addison-Wesley, Reading, Mass.). [3.2]

Karreman, G., 1955, Topological information content and chemical reactions, Bulletin Mathematical Biophysics, vol. 17, pp. 279-285. [B]

Lacroix, M. and A. Pirotte, 1976, Generalized joins, ACM SIGMOD record, vol. 8, no. 3, 14-15. [4.3]

Liepins, G.E., 1980, Refinements to the Boolean approach to automatic data editing (Internal report, Oak Ridge National Laboratory, Oak Ridge, Tennessee). [3.3; 3.3.2]

Liepins, G.E., R.S. Garfinkel & A.S. Kunnathur, 1982, Error localization for erroneous data: a survey In: S.H. Zanakis and J.S. Rustagi, Optimization in statistics (North-Holland, Amsterdam), 205-219. [3.1; 3.3; 3.3.2]

Liepins, G.E. & D.J. Pack, 1980, An integrated approach to data editing, Proc. of the American Statistical association, Survey Research Section, 777-781. [3.3; 3.3.2]

Liepins, G.E. & D.J. Pack, 1981, Prior probablities, maximal posterior, and minimal field error localization, Proc. of the 13th Symposium on the Interface (Springer, Berlin). [3.3; 3.3.2]

Lipski, W., 1981, On data bases with incomplete information, Journal of the Association of computing machinery, vol. 28, 41-70. [4.3]

Lodi, E., F. Luccio, C. Mugnai and L. Pagli, 1979, On two-dimensional data organization I, Fundamenta Informatica, vol. 2, 245-260. [2.2.3]

Maddala, G.S., 1983, Limited-dependent and qualitative variables in econometrics (Cambridge University Press, Cambridge). [5.2]

Madow, W.G and I. Olkin (eds.), 1983, Incomplete data in sample surveys, vol. 3, Proceedings of the symposium (Academic Press, New York). [5.1]

Madow, W.G., I. Olkin and D.B. Rubin (eds.), 1983, Incomplete data in sample surveys, vol. 2, Theory and bibliography (Academic Press, New York). [5.1]

Mardia K.V., J.T. Kent and J.M. Bibby, 1982, Multivariate analysis (Academic Press, London). [C]

Marshall, C.W., 1971, Applied graph theory (Wiley-Interscience, New York). [B]

McCabe, T.J., 1976, A complexity measure, IEEE Transactions on software engineering, vol. 2, 308-320. [B]

McFadden, D., 1984, Econometric analysis of qualitative response models, in: Z. Griliches and M.D. Intriligator (eds.), Handbook of econometrics, vol. 2, ch. 24 (Elsevier Science Publishers, Amsterdam). [5.2]

Moshowitz, A., 1968a, Entropy and complexity of graphs I, Bulletin Mathematical Biophysics, vol. 30, 175-204. [B]

Moshowitz, A., 1968b, Entropy and complexity of graphs II, Bulletin Mathematical Biophysics, vol. 30, 225-240. [B]

Naus, J.I., T.G. Johnson and R. Montalvo, 1972, A probabilistic model for identifying errors in data editing, Journal of the American Statistical Association, vol. 67, 943-950. [3.3; 3.3.1; 3.4; 3.4.1]

Rashevsky, N., 1955, Life, information theory and topology, Bulletin Mathematical Biophysics, vol. 17, 229-235. [B]

Royall, R., 1968, An old approach to finite population sampling theory, Journal of the American Statistical Association, vol. 63, 1269-1279. [5.1]

Rubin, D.B., 1987, Multiple imputation for nonresponse in surveys (Wiley, New York). [0.2; 6.1; 6.2]

Sabidussi, G., 1959, The composition of graphs, Duke Mathematical Journal, vol. 26, 693-696. [B]

Santner, T.J. and D.E. Duffy, 1986, A note on A. Albert and J.A. Anderson's conditions for the existence of maximum likelihood estimates in logistic regression models, Biometrika, vol. 73, 755-758. [5.2]

Schmitz, P.I.M., 1986, Logistic regression in medical decision making and epidemiology, Ph.D. thesis, Erasmus University, Rotterdam. [C]

Shmoys, D.B., 1988, Private communication. [2.2]

Tangelder, H., 1987, The computation of cut-points in a routing graph (Internal note, Netherlands Central Bureau of Statistics, Heerlen). [A]

Tapia, R.A. and J.R. Thompson, 1978, Nonparametric probability density estimation (The Johns Hopkins University Press, Baltimore). [C]

Titterington, D.M., A.F.M. Smith and U.E. Makov, 1985, Statistical analysis of finite mixture distributions (Wiley, Chichester). [C]

Trucco, E., 1956, A note on the information content of graphs, Bulletin Mathematical Biophysics, vol. 18, 129-135. [B]

Ullman, J.D., 1982, Principles of database systems (2nd ed.) (Computer Science Press, Rockville, Md). [4.3.1; 6.3]

Vassiliou, Y., 1979, Null values in data base management: a denotational semantics approach, ACM SIGMOD International Symposium on Management of Data, 162-169. [4.3; 4.3.1; 4.3.2]

Vassiliou, Y., 1980, Functional dependencies and incomplete information, International conference on very large data bases, Proceedings IEEE '80, 260-269. [4.3]

Wagner, R.A., 1974, Order-n correction for regular languages, Communications of the ACM, vol. 17, 265-268. [3,1; 3.2]

Wagner, R.A., 1976, A shortest-path algorithm for edge-sparse graphs, Journal of the ACM, vol. 23, 50-57. [3.2]

Warshall, S., 1962, A theorem on Boolean matrices, Journal of the ACM, vol. 9, no. 1, pp. 11-12. [A]

Willenborg, L.C.R.J., 1986, Two characterizations of the routing structure in a questionnaire: balance and complexity (Internal report, Netherlands Central Bureau of Statistics, Heerlen). [B]

Wolter, K.M., 1985, Introduction to variance estimation (Springer, New York). [5.3]

Zaniolo, C., 1977, Relational views in a data base system support for queries, Computer Software and Applications Conference, Proceedings IEEE '77, 267-275. [4.3]

INDEX

(The entries refer to the sections or appendices where the corresponding concepts are defined or discussed)

MATHEMATICAL CENTRE TRACTS

1 T. van der Walt. *Fixed and almost fixed points.* 1963.

2 A.R. Bloemena. *Sampling from a graph.* 1964.

3 G. de Leve. *Generalized Markovian decision processes, part I: model and method.* 1964.

4 G. de Leve. *Generalized Markovian decision processes, part II: probabilistic background.* 1964.

5 G. de Leve, H.C. Tijms, P.J. Weeda. *Generalized Markovian decision processes, applications.* 1970.

6 M.A. Maurice. *Compact ordered spaces.* 1964.

7 W.R. van Zwet. *Convex transformations of random variables.* 1964.

8 J.A. Zonneveld. *Automatic numerical integration.* 1964.

9 P.C. Baayen. *Universal morphisms.* 1964.

10 E.M. de Jager. *Applications of distributions in mathematical physics.* 1964.

11 A.B. Paalman-de Miranda. *Topological semigroups.* 1964.

12 J.A.Th.M. van Berckel, H. Brandt Corstius, R.J. Mokken, A. van Wijngaarden. *Formal properties of newspaper Dutch.* 1965.

13 H.A. Lauwerier. *Asymptotic expansions.* 1966, out of print; replaced by MCT 54.

14 H.A. Lauwerier. *Calculus of variations in mathematical physics.* 1966.

15 R. Doornbos. *Slippage tests.* 1966.

16 J.W. de Bakker. *Formal definition of programming languages with an application to the definition of ALGOL 60.* 1967.

17 R.P. van de Riet. *Formula manipulation in ALGOL 60, part 1.* 1968.

18 R.P. van de Riet. *Formula manipulation in ALGOL 60, part 2.* 1968.

19 J. van der Slot. *Some properties related to compactness.* 1968.

20 P.J. van der Houwen. *Finite difference methods for solving partial differential equations.* 1968.

21 E. Wattel. *The compactness operator in set theory and topology.* 1968.

22 T.J. Dekker. *ALGOL 60 procedures in numerical algebra, part 1.* 1968.

23 T.J. Dekker, W. Hoffmann. *ALGOL 60 procedures in numerical algebra, part 2.* 1968.

24 J.W. de Bakker. *Recursive procedures.* 1971.

25 E.R. Paërl. *Representations of the Lorentz group and projective geometry.* 1969.

26 European Meeting 1968. *Selected statistical papers, part I.* 1968.

27 European Meeting 1968. *Selected statistical papers, part II.* 1968.

28 J. Oosterhoff. *Combination of one-sided statistical tests.* 1969.

29 J. Verhoeff. *Error detecting decimal codes.* 1969.

30 H. Brandt Corstius. *Exercises in computational linguistics.* 1970.

31 W. Molenaar. *Approximations to the Poisson, binomial and hypergeometric distribution functions.* 1970.

32 L. de Haan. *On regular variation and its application to the weak convergence of sample extremes.* 1970.

33 F.W. Steutel. *Preservation of infinite divisibility under mixing and related topics.* 1970.

34 I. Juhász, A. Verbeek, N.S. Kroonenberg. *Cardinal functions in topology.* 1971.

35 M.H. van Emden. *An analysis of complexity.* 1971.

36 J. Grasman. *On the birth of boundary layers.* 1971.

37 J.W. de Bakker, G.A. Blaauw, A.J.W. Duijvestijn, E.W. Dijkstra, P.J. van der Houwen, G.A.M. Kamsteeg-Kemper, F.E.J. Kruseman Aretz, W.L. van der Poel, J.P. Schaap-Kruseman, M.V. Wilkes, G. Zoutendijk. *MC-25 Informatica Symposium.* 1971.

38 W.A. Verloren van Themaat. *Automatic analysis of Dutch compound words.* 1972.

39 H. Bavinck. *Jacobi series and approximation.* 1972.

40 H.C. Tijms. *Analysis of (s,S) inventory models.* 1972.

41 A. Verbeek. *Superextensions of topological spaces.* 1972.

42 W. Vervaat. *Success epochs in Bernoulli trials (with applications in number theory).* 1972.

43 F.H. Ruymgaart. *Asymptotic theory of rank tests for independence.* 1973.

44 H. Bart. *Meromorphic operator valued functions.* 1973.

45 A.A. Balkema. *Monotone transformations and limit laws.* 1973.

46 R.P. van de Riet. *ABC ALGOL, a portable language for formula manipulation systems, part 1: the language.* 1973.

47 R.P. van de Riet. *ABC ALGOL, a portable language for formula manipulation systems, part 2: the compiler.* 1973.

48 F.E.J. Kruseman Aretz, P.J.W. ten Hagen, H.L. Oudshoorn. *An ALGOL 60 compiler in ALGOL 60, text of the MC-compiler for the EL-X8.* 1973.

49 H. Kok. *Connected orderable spaces.* 1974.

50 A. van Wijngaarden, B.J. Mailloux, J.E.L. Peck, C.H.A. Koster, M. Sintzoff, C.H. Lindsey, L.G.L.T. Meertens, R.G. Fisker (eds.). *Revised report on the algorithmic language ALGOL 68.* 1976.

51 A. Hordijk. *Dynamic programming and Markov potential theory.* 1974.

52 P.C. Baayen (ed.). *Topological structures.* 1974.

53 M.J. Faber. *Metrizability in generalized ordered spaces.* 1974.

54 H.A. Lauwerier. *Asymptotic analysis, part 1.* 1974.

55 M. Hall, Jr., J.H. van Lint (eds.). *Combinatorics, part 1: theory of designs, finite geometry and coding theory.* 1974.

56 M. Hall, Jr., J.H. van Lint (eds.). *Combinatorics, part 2: graph theory, foundations, partitions and combinatorial geometry.* 1974.

57 M. Hall, Jr., J.H. van Lint (eds.). *Combinatorics, part 3: combinatorial group theory.* 1974.

58 W. Albers. *Asymptotic expansions and the deficiency concept in statistics.* 1975.

59 J.L. Mijnheer. *Sample path properties of stable processes.* 1975.

60 F. Göbel. *Queueing models involving buffers.* 1975.

63 J.W. de Bakker (ed.). *Foundations of computer science.* 1975.

64 W.J. de Schipper. *Symmetric closed categories.* 1975.

65 J. de Vries. *Topological transformation groups, 1: a categorical approach.* 1975.

66 H.G.J. Pijls. *Logically convex algebras in spectral theory and eigenfunction expansions.* 1976.

68 P.P.N. de Groen. *Singularly perturbed differential operators of second order.* 1976.

69 J.K. Lenstra. *Sequencing by enumerative methods.* 1977.

70 W.P. de Roever, Jr. *Recursive program schemes: semantics and proof theory.* 1976.

71 J.A.E.E. van Nunen. *Contracting Markov decision processes.* 1976.

72 J.K.M. Jansen. *Simple periodic and non-periodic Lamé functions and their applications in the theory of conical waveguides.* 1977.

73 D.M.R. Leivant. *Absoluteness of intuitionistic logic.* 1979.

74 H.J.J. te Riele. *A theoretical and computational study of generalized aliquot sequences.* 1976.

75 A.E. Brouwer. *Treelike spaces and related connected topological spaces.* 1977.

76 M. Rem. *Associons and the closure statement.* 1976.

77 W.C.M. Kallenberg. *Asymptotic optimality of likelihood ratio tests in exponential families.* 1978.

78 E. de Jonge, A.C.M. van Rooij. *Introduction to Riesz spaces.* 1977.

79 M.C.A. van Zuijlen. *Emperical distributions and rank statistics.* 1977.

80 P.W. Hemker. *A numerical study of stiff two-point boundary problems.* 1977.

81 K.R. Apt, J.W. de Bakker (eds.). *Foundations of computer science II, part 1.* 1976.

82 K.R. Apt, J.W. de Bakker (eds.). *Foundations of computer science II, part 2.* 1976.

83 L.S. van Benthem Jutting. *Checking Landau's "Grundlagen" in the AUTOMATH system.* 1979.

84 H.L.L. Busard. *The translation of the elements of Euclid from the Arabic into Latin by Hermann of Carinthia (?), books vii-xii.* 1977.

85 J. van Mill. *Supercompactness and Wallman spaces.* 1977.

86 S.G. van der Meulen, M. Veldhorst. *Torrix I, a programming system for operations on vectors and matrices over arbitrary fields and of variable size.* 1978.

88 A. Schrijver. *Matroids and linking systems.* 1977.

89 J.W. de Roever. *Complex Fourier transformation and analytic functionals with unbounded carriers.* 1978.

90 L.P.J. Groenewegen. *Characterization of optimal strategies in dynamic games.* 1981.

91 J.M. Geysel. *Transcendence in fields of positive characteristic.* 1979.

92 P.J. Weeda. *Finite generalized Markov programming.* 1979.

93 H.C. Tijms, J. Wessels (eds.). *Markov decision theory.* 1977.

94 A. Bijlsma. *Simultaneous approximations in transcendental number theory.* 1978.

95 K.M. van Hee. *Bayesian control of Markov chains.* 1978.

96 P.M.B. Vitányi. *Lindenmayer systems: structure, languages, and growth functions.* 1980.

97 A. Federgruen. *Markovian control problems; functional equations and algorithms.* 1984.

98 R. Geel. *Singular perturbations of hyperbolic type.* 1978.

99 J.K. Lenstra, A.H.G. Rinnooy Kan, P. van Emde Boas (eds.). *Interfaces between computer science and operations research.* 1978.

100 P.C. Baayen, D. van Dulst, J. Oosterhoff (eds.). *Proceedings bicentennial congress of the Wiskundig Genootschap, part 1.* 1979.

101 P.C. Baayen, D. van Dulst, J. Oosterhoff (eds.). *Proceedings bicentennial congress of the Wiskundig Genootschap, part 2.* 1979.

102 D. van Dulst. *Reflexive and superreflexive Banach spaces.* 1978.

103 K. van Harn. *Classifying infinitely divisible distributions by functional equations.* 1978.

104 J.M. van Wouwe. *Go-spaces and generalizations of metrizability.* 1979.

105 R. Helmers. *Edgeworth expansions for linear combinations of order statistics.* 1982.

106 A. Schrijver (ed.). *Packing and covering in combinatorics.* 1979.

107 C. den Heijer. *The numerical solution of nonlinear operator equations by imbedding methods.* 1979.

108 J.W. de Bakker, J. van Leeuwen (eds.). *Foundations of computer science III, part 1.* 1979.

109 J.W. de Bakker, J. van Leeuwen (eds.). *Foundations of computer science III, part 2.* 1979.

110 J.C. van Vliet. *ALGOL 68 transput, part I: historical review and discussion of the implementation model.* 1979.

111 J.C. van Vliet. *ALGOL 68 transput, part II: an implementation model.* 1979.

112 H.C.P. Berbee. *Random walks with stationary increments and renewal theory.* 1979.

113 T.A.B. Snijders. *Asymptotic optimality theory for testing problems with restricted alternatives.* 1979.

114 A.J.E.M. Janssen. *Application of the Wigner distribution to harmonic analysis of generalized stochastic processes.* 1979.

115 P.C. Baayen, J. van Mill (eds.). *Topological structures II, part 1.* 1979.

116 P.C. Baayen, J. van Mill (eds.). *Topological structures II, part 2.* 1979.

117 P.J.M. Kallenberg. *Branching processes with continuous state space.* 1979.

118 P. Groeneboom. *Large deviations and asymptotic efficiencies.* 1980.

119 F.J. Peters. *Sparse matrices and substructures, with a novel implementation of finite element algorithms.* 1980.

120 W.P.M. de Ruyter. *On the asymptotic analysis of large-scale ocean circulation.* 1980.

121 W.H. Haemers. *Eigenvalue techniques in design and graph theory.* 1980.

122 J.C.P. Bus. *Numerical solution of systems of nonlinear equations.* 1980.

123 I. Yuhász. *Cardinal functions in topology - ten years later.* 1980.

124 R.D. Gill. *Censoring and stochastic integrals.* 1980.

125 R. Eising. *2-D systems, an algebraic approach.* 1980.

126 G. van der Hoek. *Reduction methods in nonlinear programming.* 1980.

127 J.W. Klop. *Combinatory reduction systems.* 1980.

128 A.J.J. Talman. *Variable dimension fixed point algorithms and triangulations.* 1980.

129 G. van der Laan. *Simplicial fixed point algorithms.* 1980.

130 P.J.W. ten Hagen, T. Hagen, P. Klint, H. Noot, H.J. Sint, A.H. Veen. *ILP: intermediate language for pictures.* 1980.

131 R.J.R. Back. *Correctness preserving program refinements: proof theory and applications.* 1980.

132 H.M. Mulder. *The interval function of a graph.* 1980.

133 C.A.J. Klaassen. *Statistical performance of location estimators.* 1981.

134 J.C. van Vliet, H. Wupper (eds.). *Proceedings international conference on ALGOL 68.* 1981.

135 J.A.G. Groenendijk, T.M.V. Janssen, M.J.B. Stokhof (eds.). *Formal methods in the study of language, part I.* 1981.

136 J.A.G. Groenendijk, T.M.V. Janssen, M.J.B. Stokhof (eds.). *Formal methods in the study of language, part II.* 1981.

137 J. Telgen. *Redundancy and linear programs.* 1981.

138 H.A. Lauwerier. *Mathematical models of epidemics.* 1981.

139 J. van der Wal. *Stochastic dynamic programming, successive approximations and nearly optimal strategies for Markov decision processes and Markov games.* 1981.

140 J.H. van Geldrop. *A mathematical theory of pure exchange economies without the no-critical-point hypothesis.* 1981.

141 G.E. Welters. *Abel-Jacobi isogenies for certain types of Fano threefolds.* 1981.

142 H.R. Bennett, D.J. Lutzer (eds.). *Topology and order structures, part 1.* 1981.

143 J.M. Schumacher. *Dynamic feedback in finite- and infinite-dimensional linear systems.* 1981.

144 P. Eijgenraam. *The solution of initial value problems using interval arithmetic; formulation and analysis of an algorithm.* 1981.

145 A.J. Brentjes. *Multi-dimensional continued fraction algorithms.* 1981.

146 C.V.M. van der Mee. *Semigroup and factorization methods in transport theory.* 1981.

147 H.H. Tigelaar. *Identification and informative sample size.* 1982.

148 L.C.M. Kallenberg. *Linear programming and finite Markovian control problems.* 1983.

149 C.B. Huijsmans, M.A. Kaashoek, W.A.J. Luxemburg, W.K. Vietsch (eds.). *From A to Z, proceedings of a symposium in honour of A.C. Zaanen.* 1982.

150 M. Veldhorst. *An analysis of sparse matrix storage schemes.* 1982.

151 R.J.M.M. Does. *Higher order asymptotics for simple linear rank statistics.* 1982.

152 G.F. van der Hoeven. *Projections of lawless sequences.* 1982.

153 J.P.C. Blanc. *Application of the theory of boundary value problems in the analysis of a queueing model with paired services.* 1982.

154 H.W. Lenstra, Jr., R. Tijdeman (eds.). *Computational methods in number theory, part I.* 1982.

155 H.W. Lenstra, Jr., R. Tijdeman (eds.). *Computational methods in number theory, part II.* 1982.

156 P.M.G. Apers. *Query processing and data allocation in distributed database systems.* 1983.

157 H.A.W.M. Kneppers. *The covariant classification of two-dimensional smooth commutative formal groups over an algebraically closed field of positive characteristic.* 1983.

158 J.W. de Bakker, J. van Leeuwen (eds.). *Foundations of computer science IV, distributed systems, part 1.* 1983.

159 J.W. de Bakker, J. van Leeuwen (eds.). *Foundations of computer science IV, distributed systems, part 2.* 1983.

160 A. Rezus. *Abstract AUTOMATH.* 1983.

161 G.F. Helminck. *Eisenstein series on the metaplectic group, an algebraic approach.* 1983.

162 J.J. Dik. *Tests for preference.* 1983.

163 H. Schippers. *Multiple grid methods for equations of the second kind with applications in fluid mechanics.* 1983.

164 F.A. van der Duyn Schouten. *Markov decision processes with continuous time parameter.* 1983.

165 P.C.T. van der Hoeven. *On point processes.* 1983.

166 H.B.M. Jonkers. *Abstraction, specification and implementation techniques, with an application to garbage collection.* 1983.

167 W.H.M. Zijm. *Nonnegative matrices in dynamic programming.* 1983.

168 J.H. Evertse. *Upper bounds for the numbers of solutions of diophantine equations.* 1983.

169 H.R. Bennett, D.J. Lutzer (eds.). *Topology and order structures, part 2.* 1983.

CWI TRACTS

1 D.H.J. Epema. *Surfaces with canonical hyperplane sections.* 1984.

2 J.J. Dijkstra. *Fake topological Hilbert spaces and characterizations of dimension in terms of negligibility.* 1984.

3 A.J. van der Schaft. *System theoretic descriptions of physical systems.* 1984.

4 J. Koene. *Minimal cost flow in processing networks, a primal approach.* 1984.

5 B. Hoogenboom. *Intertwining functions on compact Lie groups.* 1984.

6 A.P.W. Böhm. *Dataflow computation.* 1984.

7 A. Blokhuis. *Few-distance sets.* 1984.

8 M.H. van Hoorn. *Algorithms and approximations for queueing systems.* 1984.

9 C.P.J. Koymans. *Models of the lambda calculus.* 1984.

10 C.G. van der Laan, N.M. Temme. *Calculation of special functions: the gamma function, the exponential integrals and error-like functions.* 1984.

11 N.M. van Dijk. *Controlled Markov processes; time-discretization.* 1984.

12 W.H. Hundsdorfer. *The numerical solution of nonlinear stiff initial value problems: an analysis of one step methods.* 1985.

13 D. Grune. *On the design of ALEPH.* 1985.

14 J.G.F. Thiemann. *Analytic spaces and dynamic programming: a measure theoretic approach.* 1985.

15 F.J. van der Linden. *Euclidean rings with two infinite primes.* 1985.

16 R.J.P. Groothuizen. *Mixed elliptic-hyperbolic partial differential operators: a case-study in Fourier integral operators.* 1985.

17 H.M.M. ten Eikelder. *Symmetries for dynamical and Hamiltonian systems.* 1985.

18 A.D.M. Kester. *Some large deviation results in statistics.* 1985.

19 T.M.V. Janssen. *Foundations and applications of Montague grammar, part 1: Philosophy, framework, computer science.* 1986.

20 B.F. Schriever. *Order dependence.* 1986.

21 D.P. van der Vecht. *Inequalities for stopped Brownian motion.* 1986.

22 J.C.S.P. van der Woude. *Topological dynamix.* 1986.

23 A.F. Monna. *Methods, concepts and ideas in mathematics: aspects of an evolution.* 1986.

24 J.C.M. Baeten. *Filters and ultrafilters over definable subsets of admissible ordinals.* 1986.

25 A.W.J. Kolen. *Tree network and planar rectilinear location theory.* 1986.

26 A.H. Veen. *The misconstrued semicolon: Reconciling imperative languages and dataflow machines.* 1986.

27 A.J.M. van Engelen. *Homogeneous zero-dimensional absolute Borel sets.* 1986.

28 T.M.V. Janssen. *Foundations and applications of Montague grammar, part 2: Applications to natural language.* 1986.

29 H.L. Trentelman. *Almost invariant subspaces and high gain feedback.* 1986.

30 A.G. de Kok. *Production-inventory control models: approximations and algorithms.* 1987.

31 E.E.M. van Berkum. *Optimal paired comparison designs for factorial experiments.* 1987.

32 J.H.J. Einmahl. *Multivariate empirical processes.* 1987.

33 O.J. Vrieze. *Stochastic games with finite state and action spaces.* 1987.

34 P.H.M. Kersten. *Infinitesimal symmetries: a computational approach.* 1987.

35 M.L. Eaton. *Lectures on topics in probability inequalities.* 1987.

36 A.H.P. van der Burgh, R.M.M. Mattheij (eds.). *Proceedings of the first international conference on industrial and applied mathematics (ICIAM 87).* 1987.

37 L. Stougie. *Design and analysis of algorithms for stochastic integer programming.* 1987.

38 J.B.G. Frenk. *On Banach algebras, renewal measures and regenerative processes.* 1987.

39 H.J.M. Peters, O.J. Vrieze (eds.). *Surveys in game theory and related topics.* 1987.

40 J.L. Geluk, L. de Haan. *Regular variation, extensions and Tauberian theorems.* 1987.

41 Sape J. Mullender (ed.). *The Amoeba distributed operating system: Selected papers 1984-1987.* 1987.

42 P.R.J. Asveld, A. Nijholt (eds.). *Essays on concepts, formalisms, and tools.* 1987.

43 H.L. Bodlaender. *Distributed computing: structure and complexity.* 1987.

44 A.W. van der Vaart. *Statistical estimation in large parameter spaces.* 1988.

45 S.A. van de Geer. *Regression analysis and empirical processes.* 1988.

46 S.P. Spekreijse. *Multigrid solution of the steady Euler equations.* 1988.

47 J.B. Dijkstra. *Analysis of means in some non-standard situations.* 1988.

48 F.C. Drost. *Asymptotics for generalized chi-square goodness-of-fit tests.* 1988.

49 F.W. Wubs. *Numerical solution of the shallow-water equations.* 1988.

50 F. de Kerf. *Asymptotic analysis of a class of perturbed Korteweg-de Vries initial value problems.* 1988.

51 P.J.M. van Laarhoven. *Theoretical and computational aspects of simulated annealing.* 1988.

52 P.M. van Loon. *Continuous decoupling transformations for linear boundary value problems.* 1988.

53 K.C.P. Machielsen. *Numerical solution of optimal control problems with state constraints by sequential quadratic programming in function space.* 1988.