Ser. 1003/29
eks. 2
Utlåneksemplar
Nasjonalbiblioteket avdeling Oslo

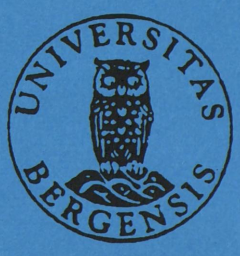
STATISTICAL REPORT

Interfaces in a knowledge-based statistical system, as exemplified by Express

by

Jan H. Aarseth and Ivar Heuch

Report no. 29
July 1996



Department of Mathematics
UNIVERSITY OF BERGEN
Bergen, Norway

Nasjonalbiblioteket avdeling Oslo
Ser. 1003

Department of Mathematics
University of Bergen
N-5007 Bergen
Norway

ISSN 0333-1865

Interfaces in a knowledge-based statistical system, as exemplified by Express

Jan H. Aarseth and Ivar Heuch

Abstract: Express is a tool for constructing knowledge-based systems for statistical data analysis, using already existing statistical software. Such systems operate through an interconnected set of interfaces to knowledge, data, users, knowledge engineers and other software components. A special feature of Express is the interface adapted to external statistical packages, which enables the system to reach decisions about future analyses on the basis of results already obtained. However, the application to data analytic problems imposes restrictions on all interfaces involved. Considering the development of Express and similar tools, it appears that an adequate interface to knowledge will form the most critical component in the construction of a system of considerable practical value.

Statistical Report No. 29
July 1996

1. INTRODUCTION

Express is a general tool for constructing knowledge-based systems in statistics. It runs on IBM-compatible personal computers under MS-DOS version 3.3 or higher (Aarseth & Heuch, 1996a). By defining a suitable knowledge base in Express, a separate system for solving practical problems in data analysis can be developed using already existing statistical software. A previous mainframe version of Express (Carlsen & Heuch, 1986), which included a simple two sample knowledge base with rules built into the program structure itself, belonged to the early systems applying data analytic techniques (Gale *et al.*, 1993). The basic ideas have been retained in subsequent versions adapted to MS-DOS and in a rather limited version running under the X Window System. The structural development of the system has focused on separating the different parts of the knowledge-based system (Heuch *et al.*, 1990).

The knowledge base is now organized as an independent component, which makes it easier to construct new systems relating to any specific statistical domain. Thus Express constitutes a shell, with a separate interface to knowledge. Another important feature is the interface to external statistical software. Particular applications also require well adapted interfaces to the knowledge engineers coding statistical strategies and to the end users taking advantage of a complete system.

In this paper we discuss the implementation of a statistical knowledge-based system considering interfaces between internal and external components. We show how the interface concept makes it easier to formulate suitable requirements to the separate components. The discussion is exemplified by implementations selected in our work with Express. We also draw on our experience with Logistrule, a knowledge-based system for model building in logistic regression (Aarseth & Heuch, 1996b), constructed within the framework of Express.

2. INTERFACES IN SHELLS FOR KNOWLEDGE-BASED STATISTICAL SYSTEMS

A knowledge-based system typically consists of three main parts (Bench-Capon, 1990): the knowledge base, the inference engine with a working memory, and the interface to the end user with an explanation module. A learning module may also be connected to the knowledge base (Patterson, 1990). The knowledge base incorporates general knowledge about the area considered, in a suitable representation. The inference engine manipulates this knowledge to produce conclusions which are conveyed to the user. A considerable amount of time can be saved using a shell as a tool for constructing new knowledge-based systems, despite the loss of some flexibility. Small prototypes are relatively easy to construct in this manner but not always large scale systems (Gillies, 1991).

Implementations based on general shells should be of particular value in statistical inference. Although widely different methods are applied in separate areas, most data analyses have many basic features in common. The analysis usually forms an iterative and cyclic process, quite distinct from the recommendations in textbooks (Hand, 1987). Most problems involve different kinds of computation, frequently performed by specialized statistical software, with subsequent interpretation and examination of aspects which may influence the results. Thus the pioneering system REX for regression analysis (Gale, 1986a) utilized powerful numerical algorithms in S to increase productivity and reduce training requirements. Hence a useful general shell must include a flexible *interface to statistical software*. As the general usage of major packages is often quite similar, this makes it possible to base the implementation of a strategy on several external programs. The *interface to data* must also be designed specifically for statistical analysis. In contrast, both an *interface to knowledge* and an *interface to the users* are needed in all kinds of knowledge-based systems, although the actual design must be adapted to data analysis.

User interfaces are essentially of two kinds. The *interface to the end user* must be informative

and easy to handle. An experienced end user may prefer a more complex, integrated interface, in particular for trying out new ideas (Nelder, 1988), but ease of operation is still important. In contrast, the *interface to the knowledge engineer* must be adapted to the structure selected for other interfaces. The representation of knowledge, in terms of definitions of statistical strategies, and the use of available external software impose constraints on the interaction with the knowledge engineer. In this case, flexibility may be more important than userfriendliness.

Problems involving a "knowledge gap" (Gillies, 1991) may arise if the person possessing relevant statistical insight is not the same as the knowledge engineer. To overcome this problem, a close cooperation is needed between the knowledge engineer and the statistical expert, going beyond what is possible with a simple dialogue (Bell & Watts, 1988). In general shells for knowledge-based systems, one often tries to avoid the knowledge gap by imposing rather simple rules for generating a knowledge base. However, in view of the potential complexity of knowledge, it is more important that flexible tools are available. To minimize the effects of the knowledge gap, a considerable effort should be spent on developing a suitable interface to the knowledge engineer. This work must also take into account implications for the structure of other interfaces.

In our subsequent discussion, we assume that the knowledge incorporated into the system is available as a generally accepted strategy. This is certainly not true for many practical problems in data analysis, as different statisticians may propose divergent strategies (Van den Berg & Visser, 1990; Tung & Schuenmeyer, 1991), especially when they are working in separate areas. Although many basic ideas enjoy widespread acceptance, it may still be advisable to take particular user groups into consideration when knowledge concerning statistical strategies is collected. At the same time, changes to an implemented strategy should be easy to carry out, as statistics is a young science where new techniques are frequently developed (Hand, 1985). This mainly concerns the interfaces to knowledge and the knowledge engineer. In our design of Logistrule, the dilemma of selecting between strategies was largely

avoided by concentrating on procedures recommended by Hosmer & Lemeshow (1989) in their book on applied logistic regression, with minor modifications.

3. THE INTERFACE TO KNOWLEDGE

When a strategy is implemented for a certain domain of applications, general knowledge about the domain must be incorporated into the system. The choice of knowledge representation has important implications for the subsequent handling of practical problems through the interface to knowledge. A representation must be found which is adapted to the domain considered, to the transfer of knowledge and to the user (Bench-Capon, 1990).

To illustrate the difference between general expert systems and systems for statistical analysis, we consider a simplified strategy for a general one-way analysis of variance. We wish to investigate whether observations from different samples may have identical underlying mean values. Our strategy is based on applying the ordinary F test if the samples appear to be normally distributed with equal variances. The decisions concerning normality and equality of variances will be based on the observations, using the Shapiro-Wilk test (Shapiro & Wilk, 1965) and the Levene test (Levene, 1960), respectively. If the assumption regarding normality is accepted but not the assumption about equal variances, the Brown-Forsythe test (Brown & Forsythe, 1974) will be applied. Finally, if the normality assumption fails, the non-parametric Kruskal-Wallis test (Kruskal & Wallis, 1952) is used. Although the strategy mapped in Figure 1 is extremely simple, it is probably adopted in practice by many users of statistics in comparisons of several samples. It forms a natural approach in particular for those familiar with the BMDP package where all tests are readily available. We will not discuss whether such a simple general strategy is actually justified.

It appears from Figure 1 that the strategy is merely based on deciding which one of the three

tests for one-way analysis of variance should be used in order to produce the final conclusion. Thus we have a classification problem which could be coded by simple rules in the form **IF** <conditions> **THEN** <actions/conclusions>. Production systems based on such rules are used extensively in other areas. Typical examples are the well known system MYCIN (Buchanan & Shortliffe, 1984) for diagnosing diseases of the blood and selecting antimicrobial treatment, and toy classification systems considered in textbooks (Winston, 1992). In addition to the rules, such systems need a working memory to store facts about the current situation and goals that should be attained (Bench-Capon, 1990; Winston, 1992). It is easy to write production rules in our simple example with one-way analysis of variance, and the solution of a practical problem by backward chaining will be readily understood by the user.

It has been pointed out, however (Hand, 1985, 1986; Pregibon, 1986; Klösgen & Wenzel, 1989; Augendre & Hatabian, 1992), that data analysis is in general a complex process which cannot easily by reduced to a classification problem. It should rather be regarded as a structured process where each step may depend on preceding steps (Oldford, 1990). Hand (1985) distinguished between knowledge-based systems in statistics for design and selection of strategies and systems for conducting an analysis. He pointed out that systems of the first kind lend themselves to a representation by production rules, as the objective is similar to that of classical expert systems dealing with classification. For the second kind of systems, Hand suggested the adoption of a different architecture, based on a stepwise approach, possibly including cyclic components.

In this context, a representation by hierarchical tasks, as proposed by Pregibon (1986), may be called for. This is because data analysis can usually be decomposed into subtasks, each of which can be further decomposed. Figure 2 shows a hierarchical strategy map for our knowledge-based system Logistrule. This is a simple first map where we follow the recommendation of Pregibon not to include too many details. The figure describes the main steps taken in model building for logistic regression, with tasks separated into two natural

groups. The first group represents selection of relevant independent variables, establishing a correct scaling and deciding whether interaction terms are needed. The second group includes assessment of the model and interpretation of coefficients. If the goodness of fit is not satisfactory, one option is to exclude particular influential observations and go through the tasks in the first group once more. Our analysis of variance example is also consistent with these principles for knowledge representation. In this simple situation, we can move almost immediately from describing the tasks declaratively (what to do) to describing them procedurally (how to do it).

The actual implementation of the strategy for analysis of variance is adapted to the conventions of Express. A more detailed strategy map will show which statistics, plots and intermediate conclusions must be found in order to reach the final conclusion. Relevant examples are the decision about normality and the p-value for the F test. Each such result is regarded as a "slot" in the knowledge base of Express. This term is frequently used in connection with systems based on "frames" (Bench-Capon, 1990; Patterson, 1990; Winston, 1992). Despite the different structure of Express, the same term is used as it defines knowledge which may apply to each separate variable considered in the analysis. The definition of a slot in Express comprises a slot name, an explanation, a specification of legal values and a reference to a procedure (an internal rule or the execution of external software) that can determine its value. The slot has a value part which is filled in during the analysis. Typically, each slot is associated either with a particular variable or a collection of variables. Thus in analysis of variance, the normality decision is associated with each sample whereas the all-normality attribute, the test result for equal variances and the final conclusion about underlying means are all connected with the entire data set consisting of r samples. We may regard these as two kinds of objects, a sample and an r-sample, with knowledge generated in the same way for similar objects. It is sufficient to implement the knowledge with corresponding slot definitions only once for a particular kind of object, and the definition is then inherited by all similar objects. Express keeps track of which object is handled in each

case during execution. A data base records the current situation at any time, with all slot values found thus far.

Another part of the knowledge base consists of Fortran subroutines, referred to as "rules", which execute the analysis in accordance with the defined strategy. These subroutines are primarily built using a standard library of utility routines supplied by Express for communication with knowledge and data base, external software, data storage and the user. Particular routines make it possible to manipulate the contents of the data base to record new conclusions. Thus the inference engine in Express differs substantially from that in productions systems, where a search for triggered rules (rules whose conditions are satisfied) is performed in each step, with a resolution of conflict if several rules may be fired at the same time (Patterson, 1990; Winston, 1992). Using the Express library routines, it is easy to write the necessary rules once a detailed strategy map has been worked out. In comparison to systems with a fixed coding pattern for rules, the implementation is very flexible as special problems can be handled by additional Fortran code. Moreover, it is easy to incorporate numerical processing in the rules. Although Express relies on external software for extensive computations, it is often more efficient to carry out basic calculations within the shell.

When the user indicates to Express that a specific problem should be addressed, a chaining of rules is initiated, beginning with the rule associated with the problem itself. If this rule requires knowledge about a slot value which has not yet been determined, the knowledge base is consulted, indicating which rule can find the value. This may be another rule in the strict sense or the execution of an external package. In the latter case, the system automatically generates necessary commands, executes the software and extracts relevant results from the output. When another rule is invoked, Express makes use of a particular stack. The new rule is added to the stack on top of the first one before it is activated. The second rule may in turn activate even more rules which are put on the stack. Each time a rule has been carried out completely and all slot values referred to in the rule have been determined, the rule is

removed from the stack, and the rule now positioned at the top is reactivated. This process goes on until the stack is empty, which happens when the original problem has been solved.

A cyclic analysis is easily carried out within this framework. Thus, in Logistrule, the selection of essential independent variables is performed by a cyclic refitting scheme. First, non-significant variables are removed from the model and a new model is fitted. Logistrule operates with a study variable of particular interest. If the variable elimination affects the regression coefficient of the study variable markedly (or the set of coefficients when the variable is categorical), some of the variables deleted are reintroduced and the same rule is restarted. In addition, if a model shows a poor goodness of fit, the complete analysis may be restarted following data cleaning.

The implementation in Express allows a one-to-one correspondence between rules and nodes in a detailed strategy map. This map can be shown to the user as a simple illustration of the analysis conducted. Logistrule includes a total of 28 rules, and it is still feasible to design a map which conveys the basic ideas behind the strategy, although the map is considerably more complicated than that shown in Figure 2. As rules in Express represent separate Fortran subroutines, each rule will often serve the same purpose as several rules formulated in systems with a more rigid pattern of definitions.

Although most developers of statistical systems have agreed that a representation involving production rules is not particularly useful, no standard has been adopted for implementation of knowledge. Some systems combine rules with other structures. Thus PROTOSHELL (Adér, 1992) has a rule base structured around steps representing different phases in the analysis, and each step is associated with a group of rules. The order of the steps executed depends on priors attached to each group. In the shell TAXSY (Darius, 1990), designed within the SAS environment, rules and strategies are both defined as SAS data sets. The strategy data set is a superstructure relating rules to particular subgoals of the total analysis. REX (Gale, 1986a)

uses a hierarchy of frames to define the strategy. A queue of hypotheses is recorded, and rules may generate new hypotheses as well as provide answers.

Certain statistical systems do not use production rules at all. In the "fact-based" shell ESIA (Augendre & Hatabian, 1992), the strategy is defined as a set of methods or actions used in a particular order. Conditions attached to each action determine when it is executed. LMG (Linear Models Guide; Hand, 1990) includes some rules, activated when the user is unable to answer questions. However, an analysis is structured as a flowchart and the system works sequentially, on the basis of hypertext rather than rules.

4. THE INTERFACE TO EXTERNAL SOFTWARE

No technical standard has been established for generation and presentation of results among the variety of statistical packages in common use. Whereas some packages are similar to programming languages in this respect (S+ and LISPSTAT), more traditional statistical software leaves the user to select techniques using a specialized control language and produces a standard output including all relevant statistics (SAS, BMDP and SPSS). Thus, the interface to software must be able to cope with widely different types of programs.

When an external package is about to be activated, the inference engine must pass the control to the module for starting the package. Before the actual execution, control language adapted to the particular package must be generated. When the execution has finished, relevant information should be be extracted from the output produced. Finally, the main module must regain control so that the chaining of rules may proceed.

Before the control passes from the rules in Express to the command processing module, the working memory must be updated with detailed information about the particular model

considered, e.g. concerning main effects or interaction terms. This is essential for the command generating module as well as the extraction module. Thus, in Logistrule, several auxiliary slots were introduced in the data base to describe whether the variables included in the model currently being fitted are regarded as categorical or interval scaled. During the system development, it was found that this kind of auxiliary information could be handled most efficiently by writing short Fortran subroutines, called from the rules.

Using software based on a specialized command language, the instructions needed in a particular situation will often depend on the circumstances. The number and names of variables included constitute an obvious example. Furthermore, common methods such as regression analysis can usually be carried out by standard packages under a variety of different options, and the relevant combination of options must be specified each time a regression problem is executed. To create sufficient flexibility in the generation of commands submitted to external packages, it is possible in Express to define a general set of "incomplete commands" in the knowledge base. For any particular execution of an external package, commands will be selected automatically from this set, with the additional information needed to complete any commands being extracted from the data base. It is also possible to specify in advance that certain commands or parts of commands can be deleted from the command sequence actually submitted, depending on information in the data base.

To execute the external statistical package, a simple interface to the operating system is needed. This is handled in Express through a particular assembler routine. When the external package has finished, Express extracts relevant information from the output produced. Depending on the situation, a search is made for particular key words written by the package, and the information printed at this location or at a location defined by its position relative to the key words is extracted and stored in the appropriate slot in the data base. Our experience with Express indicates that the specification of the search process must be flexible, with different alternative procedures for defining where the essential information is stored, relative

to search keys. A flexible definition of search keys is also needed, for example when the program should search for any name among a prescribed collection of variable names. Search keys are defined in a particular section of the knowledge base.

In most cases, coding of commands and search keys is relatively simple. Logistrule includes about 600 possible commands to BMDP and 1100 search keys, forming 81 different combinations of commands. As the variables considered may differ, about 1000 command streams are possible in Logistrule. Although the coding work is time consuming, it is essential that all relevant command combinations should be available.

The interface in Express to external software is capable of handling such diverse packages as SAS, BMDP, Minitab, StatXact, S+ and the NAG Fortran library. Although most shells for knowledge-based statistical systems have incorporated an interface to external software based on other principles, there are also systems using internal computation modules only, such as ESTES (Hietala, 1990) and DINDE (Gale *et al.*, 1993). Frequently, the interface has been designed for a specific back-end only, as with REX adapted to S (Gale, 1986a). Because of the different structure of software more related to general programming languages, the interface can in such cases be much simpler. Other systems with an interface to a particular package include PRINCE using PRINCALS (Duijsens *et al.*, 1988) and TAXSY, with computations in SAS (Darius, 1990). TAXSY is based on a somewhat different approach as the knowledge base is actually defined within the SAS system. In view of the general availability of SAS, this leads to an extremely portable system. However, for certain strategies, there may not be any single package available which is able to carry out all the calculations needed. To achieve portability for a system as Express, which offers a general interface to external software, the design must be as independent of the back-end as possible. We may then be able to switch from one particular statistical package to another without rebuilding the knowledge base. This was also the goal for the FOCUS system (Prat *et al.*, 1992; 1993), which includes a separate back-end manager.

5. THE INTERFACE TO DATA

The interface to data may be divided into two parts, that between the user and data storage, and that between the data storage, knowledge base and external packages. Because data management and data analysis are closely connected, it has been argued that data managers should be integrated into systems for statistical analysis (Haux & Joeckel, 1989). The facilities offered will be particularly useful in knowledge-based systems intended for experienced users. If a successful integration is achieved between a data base manager and statistical software, an expert system shell can easily serve as a front-end to the combined package and thus offer the user a richer computational environment.

The data in Express are read from ordinary text files, with particular codes for missing values. Variables are stored as vectors, with only basic additional information about names, number of observations and missing values. Data management is restricted to listing variables on the screen. This simple approach contrasts with a more object oriented data representation (Oldford, 1990) which benefits from knowledge sharing. This is attained by subdividing the information into classes to store different aspects of the data. In TAXSY (Darius, 1990), a separate SAS data set is used to store knowledge, with a permanent link to the data. This information is used by the inference engine during a chaining of rules. In THESEUS (Bell *et al.*, 1989), a data entry module takes care of editing in addition to conducting basic descriptive analysis. A dialogue is carried out to ensure that the user is satisfied with the data representation. In Express it is presumed that important meta data will be determined by rules, or by questioning the user during the analysis.

The data interface in Express generates separate files used in execution of external packages. Because of the different requirements of the packages, this process must also be flexible. If the standard procedures provided by the data interface are unable to generate files with a particularly complex structure, it is always possible to include additional procedures for file

generation among the rules. More generally, an optimal solution to the problem of data transfer might be based on dynamic data exchange, with a direct link between the package and the data storage. Recent versions of statistical packages offer such facilities to connect with other software.

6. THE INTERFACE TO THE END USER

Creating a suitable interface to both experienced and inexperienced users of a knowledge-based system constitutes a major task (McGraw, 1992). To the users, the system should appear to mimic a human expert. Not only should the inference engine be able to reach substantial conclusions taking the available knowledge into account, but the user should also be provided with explanations of the steps followed. Such explanations must partly deal with the general usage of the system and partly provide guidance and allow for user interaction.

Basic technical assistance is essential in a knowledge-based system. Simple on-line help is available in Express at any time, indicating which options can be selected next. Future statistical systems will most likely take advantage of more recently developed general tools for creating user interfaces, such as hypertext adopted in the ESTES system (Hietala, 1993). However, the choice of a technical solution to describe the strategy must to some extent also depend on the structure of the knowledge interface.

Gale (1986a) argued that the essential characteristic of knowledge-based systems is their ability to explain the underlying reasoning. Statistical systems of this kind have frequently included separate modules with the capability of indicating why a certain action is taken or how a particular procedure is carried out. When unknown terms are referred to by the system, a dictionary module may be consulted (Gale, 1986a; Duijsens *et al.*, 1988; Nelder, 1988; Bell

et al., 1989; Hietala, 1990). Express uses a slightly different approach, with information given at different levels. An on-line dictionary provides an explanation to each slot included in the knowledge base. The information is normally presented on the screen whenever Express attempts to find a slot value. During a chaining of rules, the current stack consisting of rules which have not yet been completed, is shown at any time. Each rule is also assigned a brief explanation. Using the correspondence between rules and nodes in a strategy map, it is possible to follow the execution of the complete strategy.

When a slot value is retrieved from the data base in Express, the operation is nearly always carried out because the value is needed in determining another slot value. A tree structure describing the logical relationship between slots, as recorded during the chaining of rules, can be viewed on the screen and provides an additional tool for understanding why and how slot values are found. This structure also allows backtracking, enabling the user to modify the conclusions reached previously (Nelder, 1988). Such facilities were included, for example, in PRINCE and GLIMPSE. When the user interrupts the analysis in Express and changes a particular slot value, all other slots depending logically on this slot will be assigned a missing value code. The analysis can then proceed from the new situation.

As the knowledge in Express depends essentially on exploiting external packages, the user is given the opportunity to watch how commands are generated and how results are extracted from the output produced by the external software. This information flow promotes user participation. Generally, the extent to which the user will be requested to provide information during the analysis, will vary between sets of rules. System settings control the amount of information given by the program. Express does not include an adaptive user interface as described by O'Brien (1994), but the needs of different user groups may be addressed by writing separate sets of rules adapted to various levels of user skills.

When a particular data analysis has been completed, the details are recorded in the Express

log file. The program can move automatically to the first reference to any particular slot specified by the user. In combination with the dictionary and the structure describing logical relationships between slots, this facility offers extensive support for understanding the analysis performed.

7. THE INTERFACE TO THE KNOWLEDGE ENGINEER

The knowledge engineer must transfer knowledge from the domain expert to the system, concerning the steps needed in a certain strategy. The knowledge must also include information about slots and procedures for determining slot values on the basis of rules, user intervention or external software. The formulation of knowledge, given by the domain expert, should be as simple as possible to increase the chances of a successful implementation. Thus the problem of the knowledge gap is mainly connected with the definition of the strategy. Experience with Logistrule indicates that after refinement of the initial strategy map, rules based on the Express utility library may be readily implemented as Fortran subroutines in a one-to-one correspondence with the nodes of the map. This approach should minimize the knowledge gap.

Slots are defined in Express using a particular editor, with special fields for slot name, type, possible restrictions and instructions on how to determine the slot value. Commands for external software and search keys are also specified in this way. To facilitate this work, on-line help is available on admissible codes.

Separate editors or tools for defining knowledge have often been used in statistical systems, as in PRINCE (Duijsens *et al.*, 1988) and THESEUS (Bell *et al.*, 1989). FOCUS (Prat *et al.*, 1992) includes several toolkits intended to help the knowledge engineer to define different types of knowledge. In PROTOSHELL (Adér, 1992) it is even possible to generate new rules

during execution. Student (Gale, 1986b) aimed at avoiding the knowledge gap completely, by creating knowledge bases through learning by induction. Our experience from the construction of Logistrule suggests that it is difficult to formulate a well-defined strategy by providing examples, at least with the implementation facilities offered by Express. Designing tools for easier knowledge acquisition may be especially important for systems such as Express which provide a great freedom in the specification of knowledge. Optimally, a compiler should be available, accepting simple input connected to a strategy map and generating suitable program code. Flexibility should be preserved by allowing editing of the program code. The TAXSY system (Darius, 1990) represents an intermediate solution, with definition of the knowledge base in the same environment as the computations (SAS). Since this is a familiar environment to most statisticians, it reduces the need for a knowledge engineer.

Complex statistical strategies frequently involve combinations of many separate minor decisions. Threshold values prescribed for intermediate tests are often chosen by the domain expert without regard to the complete strategy. In order to develop an optimal overall performance, it is necessary to explore the statistical properties of the strategy, and possibly carry out an adaptive adjustment of the component procedures. Learning about the strategy in this way is possible by simulation, but only few systems have attempted to do so (Carlsen & Heuch, 1986; Dorda *et al.*, 1990). A recent version of Express running under the X Window System includes a particular simulation module, with an additional interface to random number generators (Aarseth & Heuch, 1996c). If the simulation indicates that major structural changes are needed in the strategy, it is essential for the knowledge engineer that modifications should be easy to carry out. Such a facility will also make it possible to adapt the system to different user groups.

8. CONCLUSION

Regardless of area of application, construction of a knowledge-based system is a complex undertaking. In statistics the first and often most difficult step is to establish a commonly accepted strategy. The possibilty of constructing a useful system still depends essentially on the tool selected for implementation. General shells seem particularly promising in data analysis if they offer a suitable interface to available software. To examine the overall performance of such systems, we have investigated the properties of the separate parts involved in the data analytic process. This makes it easier to see how improvements can be made to the tools, and how modifications to one part will affect the remaining parts.

The most critical issue is whether the chosen tool offers an adequate interface to knowledge. This interface should make it easy for the knowledge engineer to build the knowledge base, and it should allow an implementation without any serious distortion of the knowledge as initially specified. Shells often satisfy the first condition, but at least for complex problems, they hardly fulfil the second one. Hence it is important to increase the flexibility of the knowledge representation. Ordinary production rules do not seem to provide an adequate solution. We have described the adaptation of a standard third generation language such as Fortran with a suitable extension. This design results in a great flexibility, and as demonstrated by our system for model building in logistic regression, a fairly complex problem may find an acceptable solution (Aarseth & Heuch, 1996b).

REFERENCES

Aarseth, J. H. and Heuch, I. (1996a) User's guide to Express: A tool for building knowledge-based systems for statistical data analysis. *Statistical Report* no. 27, Department of Mathematics, University of Bergen, Bergen.

Aarseth, J. H. and Heuch, I. (1996b) Logistrule: A knowledge-based system for logistic regression, *Statistical Report* no. 28, Department of Mathematics, University of Bergen, Bergen.

Aarseth, J. H. and Heuch, I. (1996c) Assessing uncertainty in knowledge-based systems for data analysis by simulation. *Statistical Report* no. 30, Department of Mathematics, University of Bergen, Bergen.

Adér, H. J. (1992) PROTOSHELL: An empty shell to develop statistical knowledge based systems. In *COMPSTAT. Proceedings in Computational Statistics*, Vol. 3, Koenig, S. (ed.), Press Academiques, Neuchatel, pp. 6-15.

Augendre, H. and Hatabian, G. (1992) Inside ESIA: An open and self-consistent knowledge base system. In *COMPSTAT. Proceedings in Computational Statistics*, Vol. 2, Dodge, Y. and Whittaker, J. (eds.), Physica-Verlag, Heidelberg, pp. 33-38.

Bell, E. and Watts, P. (1988) Building a statistical knowledge base: A discussion of the approach used in the development of THESEUS, a statistical expert system. In *COMPSTAT. Proceedings in Computational Statistics*, Edwards, D. and Raun, N. E. (eds.), Physica-Verlag, Heidelberg, pp. 143-148.

Bell, E., Watts, P. and Alexander, J. (1989) THESEUS: An expert statistical consultant.

American Journal of Mathematical and Management Sciences, **9**, 361-370.

Bench-Capon, T. J. M. (1990) *Knowledge Representation. An Approach to Artificial Intelligence*, Academic Press, London.

Brown, M. B. and Forsythe, A.B. (1974) The small sample behavior of some statistics which test the equality of several means. *Technometrics*, **16**, 385-389.

Buchanan, B. G. and Shortliffe, E. H. (1984) *Rule-based Expert Systems: the MYCIN Experiments of the Stanford Heuristic Programming Project.* Addison-Wesley, Reading, Mass.

Carlsen, F. and Heuch, I. (1986). Express - An expert system utilizing standard statistical packages. In *COMPSTAT. Proceedings in Computational Statistics*, de Antoni, F., Lauro, N. and Rizzi, A. (eds.), Physica-Verlag, Heidelberg, pp. 265-270.

Darius, P.L. (1990) A toolbox for adding knowledge-based modules to existing statistical software. *Annals of Mathematics and Artificial Intelligence*, **2**, 109-116.

Dorda, W., Froeschl, K. A. and Grossmann, W. (1990) WAMASTEX - Heuristic guidance for statistical analysis. In *COMPSTAT. Proceedings in Computational Statistics*, Momirović, K. and Mildner, V. (eds.), Physica-Verlag, Heidelberg, pp. 93-98.

Duijsens I. J., Duijkers, T. J., van den Berg, D. and van den Berg, G. M. (1988) PRINCE: An expert system for nonlinear principal component analysis. In *COMPSTAT. Proceedings in Computational Statistics*, Edwards, D. and Raun, N. E. (eds.), Physica-Verlag, Heidelberg, pp. 149-153.

Gale, W. A. (1986a) REX review. In *Artifical Intelligence and Statistics*, Gale, W. A. (ed.), Addison-Wesley, Reading, Mass., pp. 173-227.

Gale, W. A. (1986b) Student phase 1. A report on work in progress. In *Artifical Intelligence and Statistics*, Gale, W. A. (ed.), Addison-Wesley, Reading, Mass., pp. 239-265.

Gale, W. A., Hand, D. J. and Kelly, A. E. (1993) Statistical applications of artificial intelligence. In *Computational Statistics, Handbook of statistics*, Vol. 9, Rao, R. (ed.), North-Holland, Amsterdam, pp. 535-576.

Gillies, A. C. (1991) *The Integration of Expert Systems into Mainstream Software*, Chapman & Hall, London.

Hand, D. J. (1985) Statistical expert systems: necessary attributes. *Journal of Applied Statistics* **12**, 19-27.

Hand, D. J. (1986) Patterns in statistical strategy. In *Artifical Intelligence and Statistics*, Gale, W. A. (ed.), Addison-Wesley, Reading, Mass., pp. 355-387.

Hand, D. J. (1987) The application of expert systems in statistics. In *Interactions in Artifical Intelligence and Statistical Methods*, Phelps, B. (ed.), Gower Technical Press, Aldershot, pp. 3-17.

Hand, D. J. (1990) Practical experience in developing statistical knowledge enhancement systems. *Annals of Mathematics and Artificial Intelligence,* **2**, 197-208.

Haux, R. and Joeckel, K. H. (1989) Database managment and statistical data analysis: The need for integration and for becoming more intelligent. In *Eurostat. Development of Statistical*

Expert Systems, ECSC - EEC - EAEC, Brussels, pp. 246-254.

Heuch, I., Aarseth, J. H., Ottersen, G. and Carlsen, F. (1990) Adaptation of Express to the IBM PC: A tool for building knowlegde-based statistical system using existing packages. In *COMPSTAT Software Catalogue*, Dubrovnik, pp. 13-14.

Hietala, P. (1990) ESTES: A statistical expert system for time series analysis. *Annals of Mathematics and Artificial Intelligence,* **2**, 221-235.

Hietala, P. (1993) Enhancing explanation capabilities of statistical expert systems through hypertext. In *Artifical Intelligence Frontiers in Statistics*, Hand, D. J. (ed.), Chapman & Hall, London, pp. 46-53.

Hosmer, D. W. and Lemeshow, S. (1989) *Applied Logistic Regression*, Wiley, New York.

Klösgen, W. and Wenzel, G. (1989) On the representation of expert-knowledge in data analysis systems. In *Eurostat. Development of Statistical Expert Systems*. ECSC - EEC - EAEC, Brussels, pp. 317-334.

Kruskal, W. H. and Wallis, W. A. (1952) Use of ranks in one-criterion variance analysis. *Journal of the American Statistical Association*, **47**, 583-621.

Levene, H. (1960) Robust test for equality of variances. In *Contributions to Probability and Statistics*, Olkin, I. (ed.), Stanford University Press, Palo Alto.

McGraw, K. (1992) *Designing and Evaluating User Interfaces for Knowledge-based Systems.* Ellis Horwood, Chichester.

Nelder, J. A. (1988) How should the statistical expert system and its user see each other? In *COMPSTAT. Proceedings in Computational Statistics*, Edwards, D. and Raun, N. E. (eds.), Physica-Verlag, Heidelberg, pp. 107-116.

O'Brien, C. M. (1994) Are there any lessons to be learnt from the building of GLIMPSE? In *AI and Computer Power*, Hand D. J. (ed.), Chapman & Hall, London.

Oldford, R. W. (1990) Software abstraction of elements of statistical strategy. *Annals of Mathematics and Artificial Intelligence*, **2**, 291-307.

Patterson, D. W. (1990) *Introduction to Artificial Intelligence and Expert Systems*. Prentice-Hall, Englewood.

Prat, A., Catot, J. M., Lores, J., Fletcher, P., Galmes, J. and Sanjeevan, K. (1992) A separable architecture for the construction of knowledge based front ends. *AICOM*, **5**, 184-190.

Prat, A., Edmonds, E., Catot, J. M., Lores, J., Galmes, J. and Fletcher, P. (1993) An architecture for knowledge-based statistical support systems. In *Artifical Intelligence Frontiers in Statistics*, Hand, D. J. (ed.), Chapman & Hall, London, pp. 39-45.

Pregibon, D. (1986) A DIY guide to statistical strategy. In *Artifical Intelligence and Statistics*, Gale, W. A. (ed.), Addison-Wesley, Reading, Mass., pp. 389-400.

Shapiro, S. S. and Wilk, M. B. (1965). An analysis of variance test for normality (complete samples). *Biometrika*, **52**, 591-611.

Tung, S. T. Y. and Schuenmeyer, J. H. (1991) An expert system for statistical consulting, *Journal of Applied Statistics*, **18**, 35-47.

Van den Berg, G. M. and Visser, R. A. (1990) Knowledge modelling for statistical consultation systems. Two empirical studies. In *COMPSTAT. Proceedings in Computational Statistics*, Momirović, K. and Mildner, V. (eds.), Physica-Verlag, Heidelberg, pp. 75-80.

Winston, P. H. (1992). *Artifical Intelligence*. Addison-Wesley, Reading, Mass.

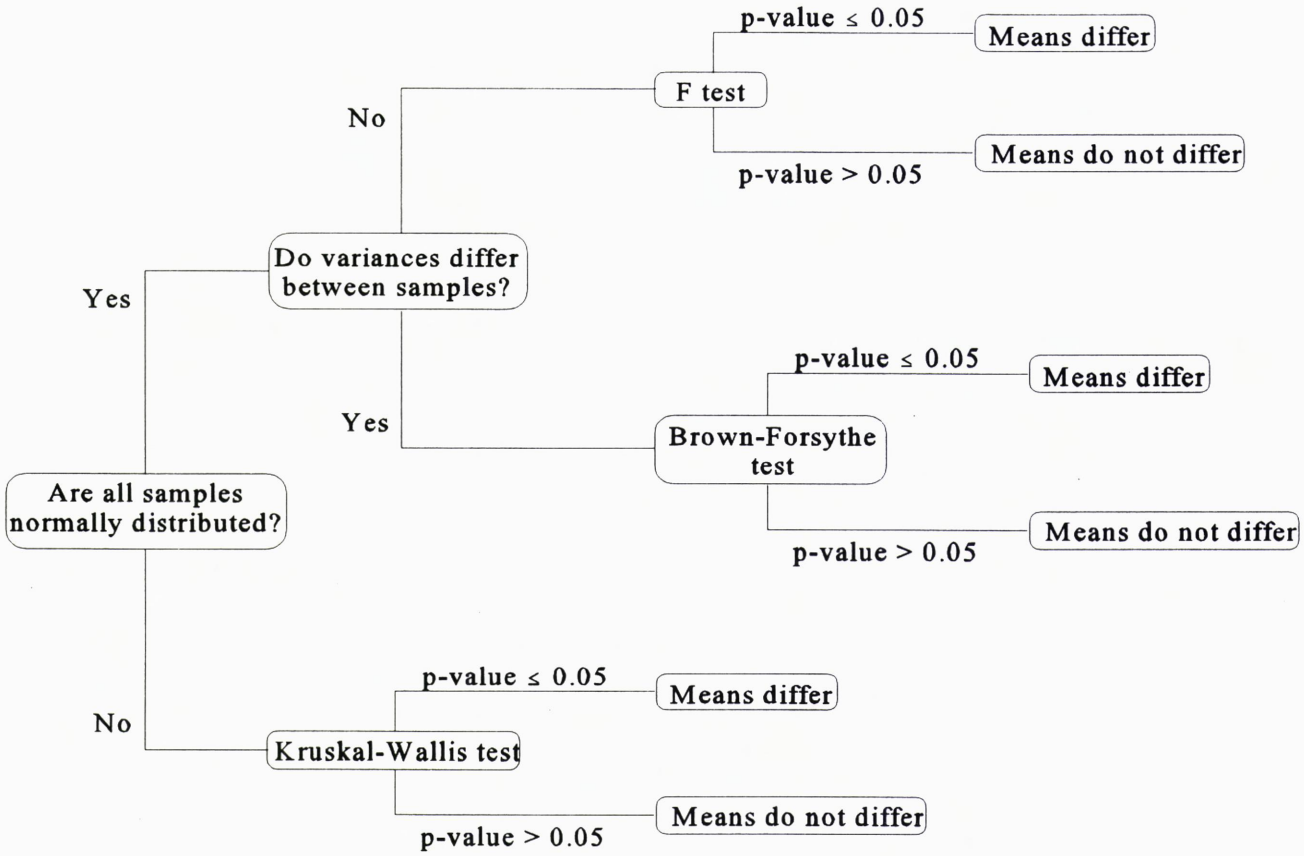


Figure 1. Strategy map for the example involving one-way analysis of variance.

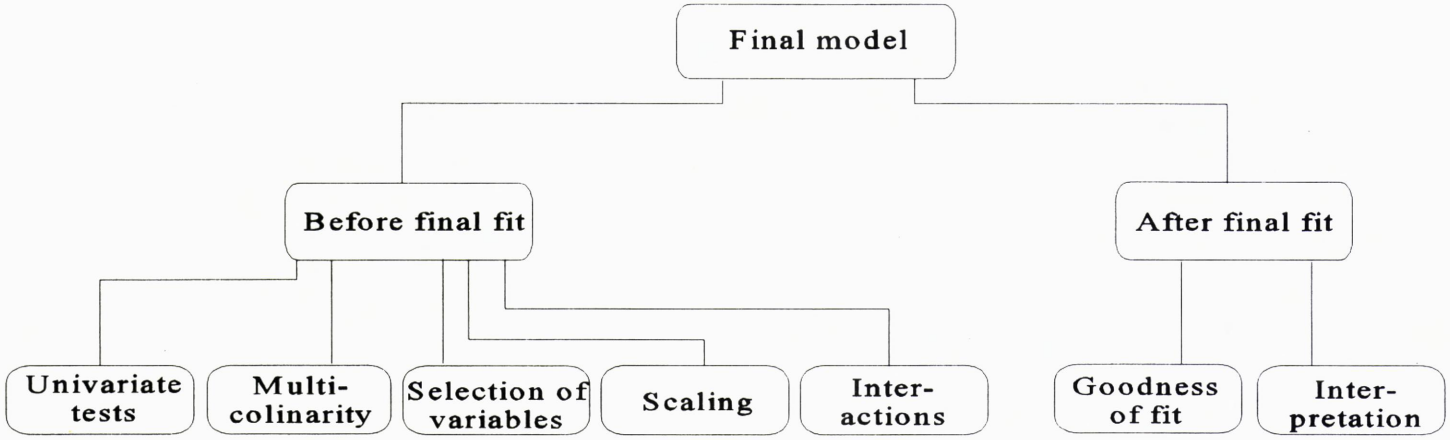


Figure 2. Simplified strategy map for Logistrule, for model building in logistic regression.

Depotbiblioteket
76g0 83 645