The Knowledge Engineering Review

Statistical applications of artificial intelligence and knowledge engineering

WILLIAM A. GALE

AT&T Bell Laboratories, 20278, 600 Mountain Avenue, Murray Hill

Abstract

Knowledge engineering (KE) has now provided some effective techniques for formalization of knowledge about goals and actions. These techniques could open new areas of research to statisticians. Experimental systems designed to assist users of statistics have been constructed in experiment design, data analysis, technique application, and technique selection. Knowledge formalization has also been used in experimental programs to assist statisticians in doing data analysis and in building consultation systems. The best-explored application of KE techniques is building consultation systems. It is now a promising area for development. Analogies with successful artificial intelligence AI applications in other fields suggest other statistical applications worth exploring. Opening new areas to research and providing new tools to users would make considerable changes in the use and production of statistical techniques. However, applying currently available KE techniques will lead to more work for statisticians, not less.

1 Opening new areas for research

Knowledge engineering (KE), the applied branch of the science of artificial intelligence (AI), is responsible for the *techniques and tools* used to implement the *ideas and concepts* of AI. As KE techniques are applied in various disciplines, it is becoming clear that a major contribution is new ways of building formal theories. Formalization of theories in a discipline provides a clear basis for agreement or disagreement, and thus supports combination of efforts in research. The exciting prospect from applications of KE in statistics is thus the prospect for opening new areas to systematic research.

In statistics, the knowledge that is being formalized has been called *statistical strategy*. The term covers higher level decision making than has previously been formalized: how to translate from subject matter goals to statistical models, how to select a data analytic technique, and how to apply a technique validly.

1.a The formal theory level

In 1980, Allen Newell, then president of the American Association for Artificial Intelligence, suggested that AI was creating a "knowledge level." The term referred to a higher level in the well-known hierarchy of computer systems levels. While the knowledge level that he suggested does not fit well in the computer systems hierarchy, the hierarchy and its extrapolation to a higher level do provide a useful insight into the AI enterprise. Table 1 shows a hierarchy familiar to computer scientists.

A level has many characteristics, of which the *medium*, the *components*, and *composition laws* are shown. The medium is processed by the components. The components provide primitive processing. The components can be combined with results predicted by a theory, the composition laws.

Each level can be defined autonomously, without reference to any other level. Programmers do

Table 1 Computer systems levels.

Level	*Medium*	*Components*	*Composition laws*
Program	Symbols	Memory Comparison Arithmetic	Sequential interpretation
Operations	Bit vector	Register Adder Multiplexer	Transfer
Logic circuit	Bits	And gate Or gate	Boolean expressions
Circuit	Voltage Current	Transistor Resistor Capacitor	Electrical engineering
Device	Electrons Magnetic domains	N-layer Cathode	Physics

not need to understand logic circuits, logic designers do not need to be electrical engineers. Yet each level can be implemented, that is *constructively* defined, in terms provided by the level below. This process is the substance of computer architecture courses.

Newell (1981) identified several features common to all these levels.

1 The behaviour of a system defined at any level is deterministic.
2 Total system behaviour results from local processing by each component.
3 Immense variety of behaviour is obtained by an immense variety of ways of assembling relatively few types of components.
4 The medium is realized through stable states of matter, such as electrons in a capacitor.

He also pointed out that each level is implemented through a great restriction on the components which can be constructed using the means provided by the level below it.

Newell then proposed a "knowledge" level above these levels. As he discussed, his proposed level violated all four of the points noted in the previous paragraph. There may be several levels above the program level, and a knowledge level may be one of them. But the most fruitful next level would seem to be one that extended these same properties which have worked so well at lower levels. Much of current AI work can be seen as work towards a level above the programming level that *does* continue all the properties noted above.

The level under construction might be called a "formal theory" level. This level is not complete, and is not as well defined as the lower levels, but in its current status it has begun to be useful. The medium of the formal theory is the statement, a sequence of symbols well formed according to some syntax. The composition rules are logics, such as first order predicate calculus or modal logic. The components most suitable for this level are not yet clear (or perhaps this level will give up this property). In their place we currently find a set of *concepts* and a set of *techniques*. The concepts include belief, knowledge, actions, goals, problems, and reasons. The techniques include rules, frames, and objects.

A number of formal theories, of varying degrees of completeness, have been built using programs as a medium of implementation. The examples given later show what has been done in statistics. As the components for building such theories are not yet standardized, current practice is to focus on the concepts, and to use the techniques available as seems most appropriate.

The contribution of AI research is KE techniques and tools for building formal theories. Formal theories have long been built using mathematical tools, and the progress made using them suggests why new tools for formalization are exciting. What distinguishes current AI programming is the

attempt to build programs that formalize such concepts as goals, problems, and actions. These concepts occur in statistics, as in any rational activity, and in data analysis have come to be called statistical strategy.

1.b Opportunities for formal theories in data analysis

Two rather similar views of the data analysis process have been proposed by Hand (1986a) and Oldford and Peters (1986a). Hand discussed four *stages* of analysis, while Oldford and Peters distinguished four *levels* of strategy. That is, Hand was concerned with entities which actually take place at different times, while Oldford and Peters' description is more of a classification. Still the views are similar, and a comparison may give some feeling for what in data analysis needs to be represented.

Hand's four stages are (1) formulate aims, (2) translate into formal terms, (3) numerical processing, (4) interpretation. These stages were given specifically as stages in a multiple analysis of variance (manova). The first stage is concerned with what dependent and independent variables are involved, how they are related, and what questions the researcher wants to explore. It occupies a large part of the time in actual consultations. The second stage results in the translation from a problem statement in the ground discipline to a problem statement in statistics terms. The third stage consists of estimation, testing, data cleaning, and transformation. This stage functions within the statistician's language. The fourth stage consists of translating back to the ground domain. As Hand point out, there will be various loops in an actual analysis, returning to earlier stages to alter decisions. While given as stages in manova, I believe they present one reasonable view of data analysis.

Oldford and Peters suggest "operational level" as a scale for thinking about procedures. They illustrate the idea rather than define it, but it seems to be related to a possible hierarchical organization of procedures. At the lowest level are standard numerical procedures of statistics, such as least squares fitting or robust fitting. Selections from this level constitute the minimal components of a statistical package. Just above this level are such sub-procedures as collinearity analysis and influential data diagnosis. Each of these presupposes the existence of procedures in the layer below it. Above this layer lies a layer of techniques, such as regression analysis, spectrum analysis, or analysis of variance. The top-most identifiable level has strategies for analysis and for design. This is another reasonable view of data analysis.

The levels idea rests on a notion of a procedure using other procedures as building blocks to carry out its goals. The notion of stages is that of what is done first. The relationship between them is that the high level strategies are used first and more frequently. The low level strategies are used later if at all. Thus the higher levels of a hierarchy of techniques will correspond to the preliminary stages of a study.

In the next section I discuss three computer programs that explicitly formalize statistical strategy under headings based on a combination of the above views. The first heading is translating a research goal into a specific data analytic agenda. This level is represented by a program, RX (Blum, 1982), that takes a research question posed in medical terms and produces a description of the statistical study that needs to be done to answer the question. This level corresponds most closely to Hand's "translate into formal terms". It is the first point that statistical knowledge enters the study. It is characterized by need for knowledge of both a ground domain and statistics.

At this point, we know that some statistical analysis needs to be done. The second heading is choice of technique, which assumes that some analysis needs to be done, but that a technique has not been selected. The program MUSE (Dambroise, 1987) is the best developed at this level. This level is characterized by a lack of assumptions or restrictions on the study, leading to a small role for formal statistical calculations, and a large dependence on information which cannot be gathered by examining the data.

The third heading is analysis given the technique, represented by REX (Gale, 1985b). This level is

Table 2 Levels of statistical strategy

Level	*Characteristic*	*Key problems*	*Example*
Formalization	Translation into statistics	Represent knowledge in two domains	RX
Technique selection	Need information not in data	Get information from naive user	MUSE
Technique	Test many assumptions, make corrections	Represent all assumptions, corrections uniformly Deal with interacting violations Order of corrections	REX

characterized by active use of statistical tests, plots, and transformations to detect violated assumptions and take corrective action.

It should be emphasized that none of the computer programs cited here has progressed beyond the feasibility demonstration stage. They are not even prototypes (suitable for friendly users), much less products (suitable for market and busy users).

Table 2 summarizes the above discussion of the levels of statistical strategy used to organize this paper. The key problems included in the table are discussed later.

2 Examples of knowledge engineering applications in statistics

This section reviews the examples of KE applications in statistics that I now consider to be the most important. Other reviews of work in this area have been published by Hahn (1985), Gale (1986e), Hand (1986b), and Chowdhury (1987).

2.a Translating research goals to statistics

Translating a research goal into a specific data analytic agenda is a high level and difficult task. The one system built at this level shows that some progress can be made with current techniques, but suggests also the difficulty of an open ended system at this level.

The system is RX, built by Robert Blum (1982). It was intended to discover causal relationships in medicine automatically. As developed, it was limited to relationships derivable from a single data base. This limitation allowed substantial research on automatic study design, but suggests that considerable generalization remains to be done.

The data base RX used was a subset of the American Rheumatism Association Medical Information System rheumatology data base. This data base was developed for doctors to record symptoms, laboratory values, and therapies of patients seen in 17 rheumatology clinics. RX used the subset of records consisting of the most active 500 patients with lupus erythematosus at the Stanford Rheumatology Clinic. The patients chosen averaged 50 visits over 4 to 8 year periods. These patients are not a random sample, but a convenient one on which to build a feasibility study. They were seriously ill, with multiple diseases and multiple therapies. The number of concurrent diseases and therapies makes the determination of causal relations challenging. RX was intended to emulate the scientist at work, with three major modules: a discovery module to suggest relationships to study, a study module to refine the suggested relationship to a data analysis problem, and a statistical module to carry out the data analysis required. All three used the knowledge base which is described below. The discovery module and the data analysis module were only minimally developed.

The data analysis module, for instance, simply performed what statisticians call an Ordinary Least Squares regression rather than doing a complete analysis. Ordinary Least Squares (OLS), is a procedure developed by Karl Gauss for representing the trend of a set of data points with a single

line. The line selected by this procedure minimizes the sum of the squares of the deviations from the points. It is a simple and useful procedure, but it can easily misrepresent the data if any of a number of assumptions fail. For instance, if there is a single outlying point removed from the bulk of the data, the OLS line is certain to pass near that one point. Or, if there is a curvature to the data, a line is simply not a useful summary. A regression *analysis* considers these and many other possible problems, and may select some technique other than OLS regression.

The study module, which tackled the problems posed by the multiplicity of diseases and therapies, was developed to the feasibility demonstration stage. In performing its task, the study module uses three main structures for its knowledge base, representing medical concepts, causal relations between the concepts, and the study design.

RX represents medical and statistical concepts in classification trees with inheritance. For instance, prednisone is a steroid, a steroid is a drug, a drug is an action; cholesterol is a chemistry, a chemistry is a laboratory value, a laboratory value is a state. A few hundred medical concepts were included in RX. All are categorized as a state of the patient or an action that can be taken. In the statistics tree, the terminal nodes represented the few tens of methods available to RX by calling on the IDL statistics package. However, it appears that this portion of the knowledge base was not widely explored, as regression was the only technique used.

The causal hypotheses portion of the knowledge base was designed to be the portion that would grow as the automated scientific process was carried out. To start with, it was a representation of relevant results from the medical literature. The knowledge was formalized as a labelled directed graph. In the graph, causes and effects from the medical concepts tree were nodes, causal effects were the arcs. The arcs were annotated with considerable information about the relationship. The nodes and some arcs within this network were to be specified by users of the system. The discovery module would then suggest which additional arcs appeared the most interesting to study. After data analysis, the information in a studied arc would be filled in.

The information included in the arcs represented characteristics of the causal relationship such as setting, intensity, and frequency. The setting is an arbitrary boolean predicate representing the conditions under which it is known that a causal relation exists. Typically, the relationships studied by RX were studied by longitudinal regression models on some or all of the patients in the sample. The distribution across patients of the regression coefficients for the causal variable is stored to show the frequency and intensity with which the causal relation holds. The average regression coefficient is compared to a scale of importance previously input for each effect to measure the importance of the effect. While information on whether the effect is an increase or a decrease is implicit in the signs of the coefficients, it is explicitly represented also. The functional form of the transformation used, including time delays, completes the information summarizing results of a study from the data base.

The arcs also include representations for validity and evidence. These refer to the state of proof of the relationship rather than its apparent strength. The validity of a causal relation is considered to be highest if it has been confirmed repeatedly in prospective randomized studies. A single study (such as done by RX) could rank midway at best on the scale used to rate validity. Evidence includes literature citations for initially input relations, or a list of patients used for a study by RX.

In performing its tasks, the study module fills out a frame of information to pass on to the analysis module. The contents of this frame represent important information for the design of an experimental test. The selection of which things to include in the frame represents important knowledge included in the program by its designer. The first step is to parse the hypothesis and verify that each concept is operationally defined in terms of the underlying data base. The parse is stored. The second step is to identify confounding variables. These are any variables that might affect either of the two principal variables of the hypothesis. These are determined from the causal network and stored. A method for controlling confounders is then selected from (1) dropping patient records, (2) eliminating affected time intervals, (3) incorporation in multiple regression. This choice is made using rules.

Production rules are also used to choose between cross-sectional and longitudinal designs. The

production rules used are an additional knowledge representation technique in RX, but not, it appears, a major one. The statistical method is selected, and the database access calls are constructed and stored.

A key issue at this level of strategy is representing knowledge from two domains, the *ground* domain and statistics. The ground domain is the domain in which statistics is being applied, medicine in the case of RX. RX shows that for a selected ground domain, knowledge from two domains can be represented and used. As statisticians are used to computer systems that do not depend on the ground domain in which they are applied, they may find this limiting. However, the state of the art in knowledge representation will not support knowledge in all domains to which statistics might be applied. The best that can be hoped for in the near future is a system with all the requisite statistical knowledge, to which ground domain knowledge for one domain can be added by a ground domain expert.

The RX system is a state of the art AI system, and deserves study to show current limits and capabilities. It appears that within the limits of the concepts in a single data base, those concepts could be defined sufficiently well to permit design of experiments. However, the resulting system would need considerable human input to deal with additional data bases. Dealing directly with humans in experiment planning would be even more difficult. The causal network established would seem to be useful for other programs, perhaps a clinical consultant.

2.b Selection of data analytic technique

RX, just discussed, has an automation of method selection, based on its statistical technique tree. Each node of the tree has represented objectives, prerequisites, and assumptions. Prerequisites are properties that must hold for mechanical applicability of the technique. Assumptions are properties that must hold for validity of the results. Selection of technique is made by matching the study requirements to the goals and prerequisites. The program was not developed so far as to use the assumptions. Also, since regression was always chosen by RX, this portion of the knowledge base was not well tested. For a study that was not formulated by a machine, RX does not have suggestions of how to acquire the description of the study to match against technique descriptions.

STATPATH was described by Portier and Lai (1983). The program used carefully worded questions to perform a binary tree search of the techniques known to it. This structure was represented by a set of production rules. Thus STATPATH proposed to determine the crucial aspects of the description of the study by asking the user. For several years, this system represented the best approach to selection of a data analytic technique.

However, the key issue is that when the user does not know enough statistics to make a technique choice, he may well not know enough statistics to answer questions well. Portier and Lai were aware of this issue, writing "how the question is asked determines the validity of the answer". Besides careful wording of questions, they provided several other help mechanisms. Additional information on the question could be obtained by replying "?" instead of "yes" or "no". Information was provided on each procedure the system knew about, and this information could be browsed. The user could reply "unknown" to any question and get *both* further lines of questioning.

While these approaches all look useful, it is not clear that they are sufficient. Indeed, Portier and Lai reported "we have spent considerable time on the wording of statements and questions and we still feel that we do not have statements which would be completely understandable to the non-statistician". The exploration of multiple branches may be useful when only a few questions are not understood, but the techniques to consider increase exponentially in the fraction of questions not understood. None of these techniques address the problem of *mis*understanding—the user thinks he understands, but does not. It will be difficult to give the user confidence that he is using a good technique, and that he can defend the choice of that technique.

The formalization of knowledge in this program was not sufficient to bridge the gap between the user's knowledge and the machine's. It now appears that the program applied KE techniques

without sufficient focus on formalization of the knowledge required for the program. The program did highlight the knowledge that needed to be formalized.

"A Statistics Advisor" (ASA) by O'Keefe (1985) provides a sharp contrast in approach. ASA formalizes knowledge about measurements—their sorts, control, and sequence—and about the relationship between measurements and useful analyses. While ASA also did not address the key problem of interacting with a statistically naive user, the approach looks far more promising than the STATPATH approach for developing useful systems. The possibility it opens is that of communicating with the user about data gathering protocols, which could be done in the language of the user's specialty.

ASA formalizes descriptions of value spaces and experiment structures. O'Keefe first tried using a traditional classification of value spaces familiar to most statisticians: nominal, partially ordered, ordinal, interval, and ratio scales. He found as he began to try using this classification in a formal setting that it had several deficiencies.

For instance, counts do not fit into this scheme. They are stronger than a ratio scale in one regard, that their unit is fixed, and yet they are not closed under all operations that can be applied to a ratio scale. The classification fails to distinguish strictly positive scales from differences of such scales. That this was an important distinction became clear when he considered rules for transformations of variables. The classification fails to distinguish linear scales from periodic ones. It does not include permutations. It fails to distinguish the unbounded from the practically bounded (such as lifetimes or heights) from the formally bounded (such as percentages). O'Keefe thus found (in analogue with most formal theory creation) that he needed to set up a new method of classification from which the important distinctions could be derived.

A flavour of the lattice representation that he built can be gained from the following simple example. Counts of oranges and counts of apples are two different value spaces, neither of which subsumes the other. It is still possible for ASA to add 3 oranges + 2 apples; with the result being 5 fruit, since fruit is the common supertype of both apples and oranges in the object taxonomy. The concepts represented in ASA's value spaces include classifications (such as the object classification), approximations, counts, locations, physical dimensions (such as time, length, and mass), directions, and arithmetically derived scales (which include sum, difference, product, ratio, and proportion).

An example will illustrate the importance of a description of a value space in statistical applications. Velleman and Hoaglin (1981) give an example of describing a univariate set of data representing rainfall. The value space for the data is *length(water)* length(water)* length(water)*. Three rules relevant to this value space are: (1) always consider the raw data; (2) consider logs if the value space is positive; (3) consider the kth root if the value space is a kth power. The third of these suggestions leads to a reasonably symmetric histogram. It is important to note that these rules were not created for this example, but are general rules applying to all transformation goals, and were first added to ASA to handle other examples.

In the text a data based "transformational shotgun" is applied to this example. A transformational shotgun is a brute force statistical technique which just applies several transformational powers to the data, usually the square root, the log, the inverse square root, and the inverse. In this case the shotgun showed that neither a square root nor a logarithm was entirely satisfactory. The point is that a useful transform is suggested by the type and not by the data.

O'Keefe went on to give some primitives and a syntax for a language to describe experimental structures. He described the primitive steps of an experiment in terms of a triple giving the description of the unit operated on, the variables known before the step and the variables known after the step. Primitive steps that he used included: the identity step (after variables equal before variables), an observation step (after variables equals union of before variables and observation variables, *with the order of the observations guaranteed to be immaterial*), and the vague step (after variables equal union of before variables, treatment variables, and observation variables, *with nothing known about relationships*). Several ways of combining primitive steps were identified, including sequential combination, parallel processing of components of the original unit, parallel processing of randomly assigned units, and parallel processing of randomly selected units.

The principle contribution of O'Keefe's work is the formal experimental description language. It can be used to select an appropriate method for analysis of the collected data, and to make a statement about the generality of the resulting conclusions. It will still take considerable work to make practical systems using this approach. The range of descriptions for which ASA could determine analysis methods was certainly not trivial, but just how big it was is not clear. A larger problem was that ASA had no method of building a description by interviewing the user. As I commented above, getting sufficient information from the user to carry out an automated task is a key problem.

MUSE, by Dambroise (1987), is the most advanced system proposed for determining the choice of a technique. MUSE is intended for use at a single industrial location, but by all the departments—personnel, engineering, accounting, etc.—at that location. It has three modules that together select the analysis methods used.

First, the user indicates which data is to be used. The system then uses rules and queries to the user to classify the data. Some of the "data" may actually be identifiers. Otherwise a part of the data may by continuous, ordinal, nominal, or dichotomous. Building further, data may be recognized as a higher level construct, such as a matrix of similarities or dissimilarities, or a contingency table. It appears that much of this can be decided from examining the data, and that the rest can be determined unambiguously from the user. This classification of data is less well formalized than is O'Keefe's classification, but it has been shown that the information required can be transferred to the machine by the intended users.

Second, the objectives module selects a set of elementary objectives required to provide service to the user. To do this, the module formalizes the set of statistical tasks that are performed in the industrial location in terms of elementary objectives available through the known statistical techniques. The statistical tasks are represented as "scripts" (a KE technique for representation of frequently occurring sequences of events) of the elementary objectives. It is easy to add scripts. The module uses meta-rules and plausibility factors to select a set of scripts for final selection by the user. The system provides some help to the user in building new scripts if none of the existing ones are satisfactory.

Third, the methods of analysis are selected. This module has a list of available methods, each characterized as to the inputs required and the elementary outputs provided. The statistical technique is selected by matching the description of the inputs available (prepared by the data module) and the description of the outputs required (prepared by the objectives module) to the list of available techniques. This module is the first functional implementation of technique selection based on a formalization of the inputs and outputs of techniques. This method was sketched by RX, but RX never selected any technique except regression, so it did not test the ideas. Thus the data, objectives, and technique selection modules in MUSE provide a substantial advance in automatic technique selection.

MUSE is the first system to demonstrate a means of getting from the information that a statistically naive user can competently provide to a choice of technique. It has accomplished this by formalizing the relationship between what the user does know (attributes of data and task to be done) and what the user does not know (inputs required and provided by various statistical techniques). The formalization was achieved in part by taking a narrow view of the analysis, and the approach needs to be tried in a more comprehensive setting.

2.c Application of data analytic techniques

After translating a domain problem into statistical terms, and after selecting an appropriate statistical method, that method needs to be applied. But application is not just a calculation. Application requires analysis, because peculiarities of the data may violate some assumptions of the selected method.

The first system to demonstrate use of KE techniques to automate the application of a statistical method was REX. The REX system (Regression EXpert), was built by Daryl Pregibon and me.

Gale (1986a) gives the most extensive description. REX advises a user in the analysis of regression problems. It guides the analysis by testing assumptions of regression, suggesting transformations when assumptions are violated, and justifying its suggestions when requested. It interprets intermediate and final results, and instructs the user in statistical concepts.

As REX begins, the user sees several windows on a bitmap display terminal—one for dialogue, one for plots, others for various information. The users must provide REX with a set of measurements of two variables, and must know that they want to do regression analysis. If a system at this level is not combined with assistance in selecting a technique, these requirements will set the skill level required to use it.

The first interactions in REX are personalization questions, such as how thorough the analysis should be. The session continues as REX checks for assumption violations. REX provides information on the interpretation of tests as it runs. At points that the user must make a decision, REX offers to show relevant graphs. REX is always prepared to define terms through a lexicon, explain what a test does, or state why it is suggesting a particular transformation. REX concludes the analysis by issuing a four-page English report on the data and the results.

The strategy REX uses initially accepts the data as given, and assumes a linear model together with ordinary least squares as the fitting method. REX then checks these assumptions in detail. It first checks for superficial problems in any one variable, then it checks each independent variable for linearity, and finally the residuals are checked. At any point that a problem (a violated assumption) is found, REX considers possible transformations of the data, the model, or the fitting method to alleviate the problem. If a transformation will solve a problem, REX suggests the transformation to the user. REX terminates either by solving all problems or by locating a problem for which it cannot find an effective and acceptable solution.

The inference engine, and KE techniques used in REX were described in detail in section 8.5 of Gale, 1986b. The detail presented there would unbalance this review and is not appropriate here. But briefly:

- The inference engine for REX was modelled after Centaur (Aikins, 1983). Control was handled through an *agenda* and a *hypothesis list*. The agenda is a stack, and it must be emptied before the program is finished. *Tasks* are placed on the agenda from slots in frames representing hypotheses. The hypothesis "list" is a weighted queue. If the agenda is empty, but the program is not finished, the first hypothesis among those with greatest weight is selected to provide further tasks.
- The Centaur architecture was copied because it provided a coherent framework that combined both frames and rules. It turned out that most of the knowledge used for REX was coded in the frames, and that the rules were a clumsy programming appendage for this application. However, I would willingly start with the same architecture again in a new application, because until one knows which will be more convenient, it is useful to have both available. The frames were used primarily as organized storage locations. A small amount of inheritance was used to provide the tasks needed for a specific frame.
- A memory module provided a data base. It supported *n*-ary relations on symbolic and numeric values. The query language could be written to look like English, and was the language in which the rules were written. Since the rules turned out to be little used, the data base was also little used.
- The regression strategy used in REX handles a wide variety of problems in actual data sets. It is not as fully developed as it would need to be for a product, but the techniques used would suffice to extend the strategy to handle all commonly occurring problems. There would remain some uncommon problems that it would not be worth the effort to have REX handle.

REX made two major contributions to subsequent work. The first is a viewpoint for thinking about data analysis as a diagnostic problem. Briefly, one should list model assumptions (analogous to possible disease), test the data set at hand for violations of the assumptions (analogous to symptoms), and if found select a transform of the data (analogous to treatment). The success of this approach depends on the representation of statistical knowledge. This is the second major contribution of REX. REX has a set of statistical primitives including tests, plots, assumptions, and

transforms, which can be implemented as frames with slots containing procedures, or as ôbjects (classes) with instance variables and methods. The hierarchical structure of the network of frames directs the interpretation of the statistical knowledge. The classes of frames used in REX provided us with an initial list of classes of primitives that has remained useful and has been expanded. This conceptual model provided a key input for work on Student, as described below.

Thus, we developed a means of representing strategy, providing an initial means of formalization. We found that the most convincing explanations in statistics were provided by graphs. A report in English and graphs provides a useful permanent record. Definitions are easy to provide. Verbal explanations of why a transform was suggested were possible, but not as convincing as graphs. Based on finding means to solve these two programming challenges, we have called REX a feasibility demonstration. It demonstrates the feasibility of using direct construction with artificial intelligence techniques to provide statistical expertise.

Since the experience in constructing REX was described, several other demonstration systems have been built. These include MUSE, Express (Carlsen and Heuch, 1986), RAO (Drapier, 1987), and unnamed systems by Berzuini and others (1986) and by Darius (1986). These systems intend to model the actions of a human analyst, as did REX.

Lubinsky is exploring another approach in *TESS* (Tree Environment for Statistical Strategy). An application of TESS to regression has been described by Lubinsky and Pregibon (1988), while Gale and Lubinsky (1986) described the knowledge representation used by TESS. Rather than try to capture the intuition of human experts, TESS aims to substitute the machine's ability to compute rapidly.

The knowledge representation used in TESS is again a tree of features and transforms, and each feature or transform is represented by a frame. While REX uses such a representation as a procedural guide, TESS uses it as a representation of a space of descriptions to explore.

TESS seeks descriptions that are both interesting and accurate. There is a trade off between these two, since the most accurate description of a set of data is the data set itself, but this is also the least interesting description. Depth in the tree of features is the measure of interest, because with a hierarchical tree the lower nodes are more specific. Accuracy is estimated by techniques specific to each node, expressed as a number between zero and one. There is no theory for combining accuracy and interest measures, but the resulting values for several descriptions can be plotted to facilitate comparison.

TESS's search strategy is a modified depth first search. The objective is to deal in complete descriptions (which is achieved by reaching a leaf of the tree), but to explore differences likely to be important quickly (which is achieved by branching near the root of the tree). The heuristic search algorithm thus produces a set of descriptions which the user can judge for interest and accuracy.

A system currently under construction, GLIMPSE, is a larger scale project than the demonstration systems described so far. GLIMPSE, a front end for the widely used General LInear Modeling package, GLIM, was described by Wolstenholme and Nelder (1986). GLIMPSE is being built with sigma-Prolog and Augmented Prolog for Expert Systems (APES) on a Vax™. The purpose is to make GLIM accessible to more users.

The work on GLIMPSE to date suggests a way to solve two important and related engineering problems. The first of these is how to make a system that a naive user can learn from, slowly becoming an expert user, without the sytem impeding expert users. One way this might be accomplished would be to have the user always in control of an interactive system, but with the system having a sufficient model of the analysis to assist at any time. While this is another unsolved problem, GLIMPSE suggests a way to approach it.

GLIMPSE provides a task command language to the statistician, and three levels of assistance in using it. A single question mark entered while giving a command is a request for a *reminder*. The system replies with the syntax of the command that has been started. A double question mark is a request for *prompting*. GLIMPSE then takes a more active role, providing keyword options as menus, and asking questions (with explanations available) to determine parameter values. Completely specified commands are shown to the user for approval. On request, GLIMPSE will

enter a *hand holding mode* for a specific task. In this mode, the system will suggest a complete command to use. It is this capability, to provide suggestions at *any* point in the analysis that is new.

The reason it is difficult is not too hard to see: when *I* watch a statistician using an interactive statistical system, I am not always sure what the statistician is up to. Inferring intentions from actions is a difficult problem. The way REX resolved this problem of how the machine would have a model of what was being done was to maintain control. But this way does not provide a system that moves gracefully from novice to expert.

The approach in GLIMPSE is two fold. First, there is a formal model of the analytic process as composed of nine *activities*. These activities include determining attributes of the data, as in MUSE, model selection, and model checking. Only some of the transitions between activities are allowed. The activity determines what the program assumes is the statistician's intention. Within the activity, then, each command that the statistician can use not only does the requested activity—as in a conventional statistical system—but also builds a formal structure representing what has been done. This structure is the basis for the machine's suggestion if it is requested.

Part of the model selection activity was described by Wolstenholme and Nelder (selection of link and error were not discussed). The activity will select a subset of variables to use from those available. The user specifies an initial kernel of variables that should be included based on prior knowledge. The remaining variables are free terms that may be added to the kernel or may be dropped. From any given model, the statistician can form a new model by dropping some of the frree terms or by moving some free terms to the kernel, or both. Models with cross terms can be created. For a given model, the system will provide F statistics for adding any given free variable to the kernel and for dropping it from the complete model with all kernel and free variables. These F statistics, also called forward and backward F statistics, are numbers which increase with the explanatory power of the variable added or deleted, and which have well-known levels when a purely random variable is added or deleted.

The structure maintained by the system to represent this search is a graph whose nodes represent models. When asked for advice, the machine uses rules based on the forward and backward F statistics for proposing additions to the kernel and deletions from the free terms. Some of the rules may provide more than one suggestion. A graph of suggestions is then generated, and is explored depth first.

A novice relying entirely on the help of the system will make a mechanical search, which, indeed, the machine could do by itself. As experience is gained, the user may learn some short-cuts, and this is the reason for pulling a novice through the search. If experts normally select a model in this way, they will be glad to have the bookkeeping assistance provided. If they think in some entirely different way, such as Mallows's (1973) C_p statistic, they may he frustrated. (C_p is a statistic that includes both the residual sum of squares, which will decrease as more explanatory variables are introduced, and a penalty for the expected bias from having too many explanatory variables. One normally uses it by examining a plot of many, or all, possible models simultaneously. Its use is thus not compatible with a step wise approach.)

The activity module thus provides its users with the same primitives as are required for a mechanical approach to model selection. If this formalization of the model selection process is unduly restrictive for the expert, or unduly tedious for the novice, then the approach may not be successful.

2.d Assisting an expert statistician

The three previous sections dealt with examples in which the person assisted was a statistical novice. Another line of research continues the venerable tradition of software built by statisticians for statisticians. For who else knows better what would help a statistician?

A leading example of this line of research is the system DINDE (French for "turkey", not an acronym), described by Oldford and Peters (1986b). DINDE aims to ease a statistician's data

analysis task by providing a visual map of an analysis. To do so, it builds a structure representing the analysis, adding a bit to the structure with each command.

The commands available in the system are defined as producing one step of the analysis. This may be a plot, the fitting of a model, or the calculation of a statistic. Each such step of the analysis is represented by a node. The node will have a number of attributes which are based on the command used in creating it. For instance, it has a set of suggested commands available. The user is free to give the system any command at any time, but the menu of commands available at a node are the commands most frequently used when attention is focused on the result of the given command.

When they are created, nodes are shown visually as small rectangles about two centimeters square. In the rectangle is a small graph showing some result of the step. For a graphical command, such as a histogram, the resulting graph itself is shown in low resolution. For a regression fit, a plot of the residuals is shown. The rectangle also has the name of the command and the name of the step if it has been given one. By clicking a mouse on the rectangle the user can get several useful options. One option, *zooming*, expands the rectangle enough to show the full results of the command. Another provides the menu of common next steps.

Three sets of links are maintained between the nodes. These sets are called the *analysis*, the *data flow*, and the *causal* links. Each of these set of links structures the nodes into a set of directed hierarchical graphs. The causal links are created by the computer whenever a new node is created by selection of a command from the common operations menu of an existing node. These links do not change during the analysis. The analysis links are entirely under the users control, although the computer inserts a default analysis link whenever it creates a causal link. The data flow links show the source of each input to the command represented by the node.

The default graphical presentation of the analysis shows the nodes and the analysis links. On command, one of the other sets of links can be used to structure the display. The user manipulates analysis links to produce a personally meaningful display of the analysis. The default links are usually desired, and analysis links are added to show relations between causal graphs initiated by the user.

As the analysis grows, it becomes impossible to show all the nodes at once. Two similar grouping mechanisms are provided. The user can group any set of nodes into an *analysis map*, which is then shown as a single rectangle on the screen. The links shown to the analysis map are the union of all links from any of the enclosed nodes. When an analysis map is zoomed in on, operations on the nodes can be made, with the resulting nodes incorporated into the map. Any number of levels of analysis maps can be used, so long as the structure remains hierarchical (that is without self referential loops). *Views* are a similar device, but do not allow operations to be made.

DINDE was developed using the Loops ™ software environment. This is an extension of Lisp that provides support for object oriented programming and for rules. DINDE has relied heavily on object oriented programming, with its view of data analysis clearly influenced by the tool used. Object oriented programming means writing programs to deal with hierarchically structured objects. The operations available for any given object are those defined explicitly for it, plus all that it inherits from its superordinate objects. It has proved useful when the concepts that a program needs to deal with have a natural hierarchical structure.

The formalization for description of data used by DINDE, described by Oldford and Peters (1986c), is also quite interesting. An observation is regarded as the *value* of a *variate* for an *individual*. Thus, values, variates, and individuals are the primitives used. Individuals are simple entities, featuring only a name, a description, and comments. Variates contain the same information; in addition, they are categorized as continuous, discrete, or categorical variates. Each variate specifies a range, which can be determined or set externally. The result of a measurement of a variate on an individual is stored in a *datum* object. The object records the number or string obtained from the measurement, the censoring (right, left, or none), and the number of significant digits, if applicable.

For example, suppose a person, George, is known to be taller than 1.71 metres. Then an object is made to represent George if he is not known already, an object is made to represent human height if

the variate is not known already, and a datum is created with value 1.71, right censoring, and 3 significant figures. The units of recording human height will be recorded in the variate, along with methods to convert from other units. Notice that if more than one measurement is used for George, it will be easy to determine that the data are paired, and thus to limit suggestions for the kind of analyses to perform.

These primitives are then associated to form higher level constructs. A datum can be associated with a variate, and a group of such associations in turn associated with an individual as a *case*. Or, a datum can be associated with an individual, and a group of such associations in turn associated with a variate as a *factor* or a *batch* depending on whether the variate is categorical or not. Many of the operations on cases, factors, and batches are the same, such as locating a datum value for a given identifier, or finding how many datum values are censored, or translating the values into the form of a vector for numerical convenience. By carefully constructing superordinate concepts, such common operations are inherited and only need to be programmed once. Other operations are specific to the kind of association, such as producing boxplots or histograms for a batch, but not for a factor or a case.

This representation is, of course, not directly observable by the user. The reason it is important is that it enables new services to be provided to the user.

Another system designed *for* experts rather than *as an* expert is KENS, described by Hand (1987). KENS is not a data analysis system at all, but is termed a "Knowledge ENhancement System". It provides paragraphs of text and extensive cross referencing aids so that statisticians can remind themselves about statistical points, or learn about the particular statistical system. Hand suggests that it would be useful to (1) find out *how* to use a system to do something specific one already knows about, such as a Wilcoxon test, (2) be reminded of the available options for a general task one knows one wants to do, such as to compare two groups, and (3) draw one's attention to a missing and crucial piece of information, as in trying to determine if a chi-squared value of 6.2 were significant.

The basic psychological observations behind KENS is that it is easier to recognize than to remember, and that without constant use, skill and knowledge are forgotten. The intent of KENS is that an expert relearn some fact rather than that a novice learn the fact. KENS is a non-linear structuring of the same kind of knowledge as one would find structured linearly in a text book.

KENS is built as two sets of nodes. One set contains the paragraphs, another set contains indexing terms. Along with each paragraph there is a brief, summarizing title, and a summarizing list of words and phrases. Links from indexing terms to paragraphs may be labelled definitions or examples, or may be unlabelled. KENS has notions of subconcepts and superconcepts, equivalent concepts, and opposites.

The formalization of knowledge in KENS is rudimentary. The labour of making a system useful in the short term is thus shifted to the statistician who would provide the node contents. If systems such as this are built and proven useful, however, they provide a very interesting beginning to an extensive formalization of statistical knowledge. Such an extensive formalization would seek to refine each paragraph node of a KENS-like system into a machine-usable set of propositions. The system might then be able to use natural language generation capabilities to formulate tailored responses to more specific questions.

2.e A consultation system constructor

Another type of system designed for use by statisticians is one that allows a statistician to build a consultation system (that a novice will use). This sort of system differs from all the previous ones considered, which all intended to help the user directly to perform a data analysis task. A consultation system constructor is designed to help users indirectly by letting a statistician provide them with a consultation system. The only example of a consultation system constructor is Student (Gale, 1987).

Student is designed to allow a professional statistician to build a knowledge-based consultation system in a data analysis technique by selecting and working examples and by answering questions. The statistician does not need to know the internal representation of the strategy demonstrated, and does not need to know how to write a knowledge based program. He does need to be fluent in the underlying statistical system, a more natural expectation of a statistician.

Like REX and other data analysis consultation systems, Student is based on an underlying statistical analysis system, and constitutes an interface to that system. Student uses S (Becker and Chambers, 1986; Becker and Chambers, 1984). The first version of S provided an interactive environment for statisticians, providing data management services, highly portable interactive graphics, command interpretation, and number crunching. In the latest version of S, the external syntax and appearance have been largely maintained, while adding tools such as programming, browsing, debugging, and editing capabilities. The design of Student assumes that the statistician using Student to create a consultation system knows how to use S.

A methodological prototype study of Student (Gale 1986c) was built using Lisp and a Symbolics machine. The current version of Student is intended as a product definition study. It is programmed in the language provided by S, as this would be the most likely delivery language for a product. The goals of the S version are to study issues such as speed, usefulness to statisticians, and generality of the conceptual framework used by Student. This version is currently a partially developed system that has only begun to be used by statisticians. It has not yet begun to answer the product issues posed, but shows the knowledge acquisition methods more clearly than the prototype, and has begun to be used to acquire a few different data analysis strategies.

By using S, hardware and software requirements are minimized. S will run in most Unix™ environments. Wherever S runs, Student will run. Student is not a product, but if it were, it would require a machine with Unix, and S software.

What Student adds to the capabilities of REX is the capability to acquire its knowledge base by interview and demonstration. The demonstration approach was proposed by Gale and Pregibon (1984), and tested in the Lisp prototype (Gale 1986c).

The importance of acquiring a strategy by interview and demonstration is considerable. In the current state of building knowledge-based consultation systems, two distinct roles, usually played by two different people, are standard. One is the role of subject matter expert, and the other is the knowledge engineer. In building REX, I played the knowledge engineer, while Daryl Pregibon played the statistical expert. This procedure requires the knowledge engineer to learn a lot about the subject matter, or the subject matter expert to learn a lot about the inference engine and programming, or both.

Student's primary goal is to allow a statistician, who does not know how the inference engine is built, to build a knowledge based consultation system without the involvement of a knowledge engineer. This should support greater efficiency in building consultation systems in data analysis.

There is a substantial secondary benefit as well. A statistical consultation system will be used in many other ground domains, such as physics, psychology, or business analysis. Current AI techniques are not adequate to handle knowledge in multiple domains, so we built REX with the explicit assumption that the user was willing to learn statistics concepts and vocabulary. This assumption will be reasonable for many analysts, but it will be unreasonable for many managers or low frequency users of statistics.

Student provides the means to specialize the knowledge and vocabulary used to guide a consultation in data analysis. Because it can learn by interviewing a statistician using locally relevant examples, it can be provided with strategies shaped to local environments. This will increase the utility for a Student-like product as compared to a REX-like product.

Another significant benefit of removing dependence on a knowledge engineer is the capability to specialize a system to a local environment. When Student is first acquired by a group such as a quality engineering group, a specialist statistician can select examples from the group's files and work them in the Student environment. After this specialization training, the engineering experts would use Student for consultation, returning to the statistician with problems beyond its training.

When such a problem seemed frequent, the statistician would work it as an addition to the strategy. If it seemed infrequent, then it would be worked by hand.

There were three main challenges in building Student. First, the system had to support the acquisition of the *first* example. In a rule based system, the first rules to be acquired are typically different from later rules, because a rule based system uses a core of rules to encode control information. A subject matter expert would not be able to provide control information. Second, Student had to acquire knowledge from a new example that was *consistent* with its previous examples. Consistency means that all the examples that the statistician considered as properly worked, remain so when the additions to the strategy are made. Third, the system had to support deliberately inconsistent changes to strategies over a long period of time. Current technology, such as used for REX, results in a "compiled" strategy, which is difficult to change.

The current version of Student has made clear that the first two of these challenges have been met, and it suggests that the third can be met. These challenges have been met by the development of a technique called *knowledge-based knowledge acquisition* (Gale, 1986d). Knowledge-based knowledge acquisition means restricting the domain of knowledge that can be acquired, and developing a conceptual model of the restricted domain.

Student is restricted to acquiring *data analysis* strategies. It is not a general purpose knowledge acquisition program for a general purpose inference engine. With this restriction, I was able to provide a formal model for strategies of data analysis. For instance, we know we have to deal with data sets, and we have provided representations to deal with them. The formal model specifies that the analysis consists of looking for violated assumptions, and if found, of finding a cure. It specifies that we look for violated assumptions by making tests and by showing the user plots. I derived this conceptual model by inference from REX, and by considering extension of the methods to other data analysis techniques. Having this conceptual model provides enough structure to guide the user through the first analysis of a given kind, and to acquire additional consistent examples. It is still a research question how far this view based on work with REX will generalize, and how well inconsistent changes can be treated.

Student is written in modules that fall into three groups: control, management of frames and other data structures, and learning. Student acquires knowledge by filling in frames, which become a significant part of the system. The control and frame management modules are nearly independent of statistics knowledge. The learning modules are specific to data anlaysis.

Having found in building REX that frames were more useful than rules, Student used only frames. A dozen types of frames (as distinguished by their slots) provide the formal conceptual model. The frame types selected represent abstractions relevant to strategy for data analysis.

The lowest level frame types are plot, test, report, and transform. These frame types represent information on how to make a specific plot, how to make a specific calculation and interpret it, how to generate a fragment of a report, and how to make a specific transform of the input variables. The "feature" frame cites one or more tests, zero or more plots, and zero or more transforms to formalize the notion of a statistical concept such as outlier or skewness. The report frames are attached to the plot, test, and transform frames.

A "strategy" frame represents a procedural view of how features are to be combined. Three frame types provide the usual procedural ideas of sequence, alternative, and repetition. A strategy represents one statistician's approach to a given analysis technique.

The "analysis" frame is the highest level frame type. It represents such activities as regression analysis, or the comparison of the locations of two groups. It represents required and optional inputs via "input" frames. Giving specific data sets for each input variable produces an example of the kind of analysis, which are kept track of at the analysis level via "example" frames. To be used for consultation, it requires one or more strategy. The methods used for representing strategy in Student are discussed in more detail by Gale and Lubinsky (1986).

3 Comments on tools used in statistical systems

For those who want to disseminate expertise, implementation of a consultation system is clearly required. However, those who simply want to develop a formalized strategy to clarify their own thinking need to implement it on a computer. The contribution of KE, as pointed out in section 1, is new tools for implementing formal systems. Without using these tools, no more progress in strategy should be expected than has been seen in the past, that is, very little. The reason for implementing is to be able to test the strategy. Without testing, the strategy cannot be falsified, and without the possibility of falsification there is no science. Therefore, research using AI concepts requires implementing, presumably with the KE tools developed.

In implementing a strategy for data analysis, it is necessary to connect the KE tools to a statistical package. It appears that this necessity has limited the AI tools that have been found useful in statistical application.

Work that has been completed so far has either constructed a new statistical package (for instance, DINDE), or has constructed the required KE tools (for instance, REX and Student). My original reason for building the AI tools used in REX was that among the two or three tools I could actually get in 1981, I did not find any that I could use along with S. As more KE tools become available, some should be suitable for use with any given statistical package. So this approach may no longer be necessary.

However, as attention turns from research to products, an approach that may be useful is building the required KE tools *in, or as an extension of* the target statistical package. In this setting, the statistical package is probably determined by the organization developing the product. The organization may also have the access to the package required to make extensions to support AI programming additions. Darius (1986) reports using SAS, a prominent statistical system, without modifications to implement simple backchaining rules. Student is now another example of this approach.

The tools that have been used successfully at all are still low level tools, such as Loops (used by DINDE) and APES (used by GLIMPSE). These are tools which are firmly embedded in a powerful general purpose language popular for AI style programming (Lisp and Prolog, respectively).

Whatever tools are used, I recommend that they at least all work on one machine. This may seem obvious, but the powerful development environment on a Symbolics tempted me into building the Student prototype on a Symbolics communicating with S on a Vax. Even for research that combination was poor.

4 Speculations on the future of KE and AI in statistics

There are some AI successes in other fields which have as yet no analogue in statistics. The first subsection discusses these briefly and speculates on some statistical analogues.

I am frequently asked what this work means for the future of statisticians. Any answer is, of course, speculative. Yet the uncertainties people have about this seem to influence their view of the work, so some discussion of the issues is appropriate. The final subsection examines these issues.

4.a Other possible applications

A major area of AI activity may be called problem solving. It includes planning and consultation systems. The work in consultation systems is the main KE application which does have an analogue in statistics, as we have seen. AI systems which plan programs (Sussman, 1975), molecular genetics experiments (Stefik, 1980), and medical experimental protocols (Weiner, Horwitz and Bauer, 1987) have been built.

One natural analogue in statistics is experimental design. This is an area well studied by statisticians as a formalized discipline of designing matrices with suitable orthogonality properties. The matrices are based on how many factors and how many interactions of factors the experimenter wants to be able to distinguish by analysis of the results of the experiment.

The system described by Haaland, Yen and Liddle (1986) uses AI techniques, but does not move beyond the performance of non-AI systems. In particular, it does not move beyond the formalized view of experiment design as matrix selection. Perhaps more significant aid would require specialization to an industry or company, so that knowledge about the subject of the experiment could be incorporated into the design.

The language for describing experiments developed by O'Keefe is a good beginning of a more comprehensive view of experiments than matrices of factors. A system might be able to build a good description of an experiment design in interaction with a user. Such a description would be useful both for executing the experiment and for analysing the results. It should be subject to modification to reflect the accidents that happen during an experiment.

Another area of AI activity has been natural language processing. This means dealing with strings of characters, not sounds. A lot of work has been put in, and the results so far are mostly interesting research which falls into comprehension, generation, and dialogue. Comprehension research deals with paragraph-length stories. Generation research has shown ability to use user-oriented vocabularies, and has achieved sentence-level utterances without moving much beyond that. Dialogue research is recent and has just begun to identify the hard problems.

Currently, the most practical application is natural language interfaces for data base queries. The domain of discourse is naturally restricted to the part of the world represented in the data base, which limits the concepts to a manageable number. The requests and replies are each at the sentence level. The systems are easier to use, and easier to train people to use, than formal data base query languages.

The importance of a report of a statistical analysis suggests a careful study of generation techniques. Another possible use for natural language techniques is a longer-range project. Knowledge bases for consultation systems are currently not very readable by people. The current approach is to generate natural language versions of the knowledge from the machine-readable version. Another approach would be to devise a restricted natural language that could be compiled into a machine-readable knowledge base. The problem is to make the knowledge in an AI system as accessible for comment by experts as is the knowledge in a journal article or book.

There has been a long history of belief that computers would be useful in education. The belief has not been borne out by experience, but there is always the hope that the next generation of computer techniques will turn the trick. Accordingly, there has been research into applying AI ideas to tutoring. One hope has been that a model of the student could be built, which would allow the tutorial material to be specialized for the student. For a system to build a model of the student from observing its interactions with the student, it is necessary for the form of the model to be pre-specified. The successful models have been competency models, that is, they show what tasks the student can successfully perform. Models of ways that students can make mistakes have foundered on the multiplicity of ways of going wrong. One of the more successful AI based tutoring systems (Anderson, Farrell and Sauers, 1984) simply does not allow the student to continue after a mistake, but corrects it immediately.

The possible applications in statistical education are clear. At present, any work done on this would best be done as part of a team which included expertise on educational issues as well as statistical issues. Barzilay (1984) described a system for tutoring in Bayesian concepts of probability.

A final area of AI interest, for which I can seen an analogue in statistics, is formal models of common sense reasoning (Hobbs and Moore, 1985). One motivation for studying these is based on the observation that a person does not solve the Navier-Stokes hydrodynamic equations before jumping out of the way of spilling hot coffee. This behaviour suggests that people have rapid and accurate models of liquids. AI research has attempted to build qualitative models using just states (Hayes, 1985), or states and directions of change (de Kleer and Brown, 1985). The models produced so far have only been implemented as research systems; they have not been used in practical systems.

The analogue that I see in statistics is models of common sense understandings of uncertainty and probability. Such models could be based on experimental observations begun by Piaget and Inhelder (1951). Their use would be to allow a system to model its users, and thus to provide

explanations understandable at the user's level of development in concepts of uncertainty. This might include some teaching of incrementally advanced concepts.

It is also worth commenting that I see several opportunities to apply AI techniques in the suggestions of Tukey (1983). "Cognostics," for instance, are diagnostics for interpretation by a machine rather than by a human. To select the most interesting scatter plots from a large set would require a cognostic for the interest of a scatter plot. To even begin, one must attempt to formalize the notion of "interesting."

4.b The impact of consulting systems on statistics practice

The basic goals of consulting systems are lower cost statistical information and higher productivity in using statistical techniques. These goals are both realistic and unlimited. They are realistic because some steps can be taken in the short run. They are unlimited because there is no clear stopping point; even if a human level of performance is achieved, perhaps more could be done.

Given this basic goal, many statisticians have asked me whether these systems will replace statisticians. The current status of consulting systems is that several feasibility systems have been described in the literature. None have become prototypes, that is, systems used by a few friendly users. There are no commercial systems based on formal models of data analysis. There are also many user friendly statistical packages available.

In the next five years, I expect to see perhaps a few dozen research systems, several prototypes, and probably one or more commercial systems. Within ten years I would expect to see several commercial systems incorporating formal models of data analysis techniques. The impacts of these systems will be different for users, consulting statisticians, and research statisticians.

Users will not see a big difference. They will perceive the systems available to them as becoming increasingly friendly. The basis of the greater friendliness will not and should not be obvious to them. The most advanced consulting software will handle most of the users statistical problems, all the routine ones.

In a setting where consulting systems are used, then, the consulting statisticians will not see routine problems. They will see unusual problems. Their activities will thus include more learning as they consult the large knowledge base of statistics represented in books and journals. They will also advise the users on the availability and choice of consulting systems. To the extent that the software achieves its goal of providing greater productivity, there must be less involvement by the consulting statistician with each user. This reduction in demand for statistician's services may be offset, or more than offset, by more users of the statistical software.

Research statisticians should simply see an increase in the options for research. In the first place, there will be techniques that would not be feasible without formal models. Just as the availability of cheaper computing has made computation-intensive techniques such as bootstrapping available, so the availability of smarter computing will make new techniques available. For the foreseeable future, there will remain less common techniques and less common problems of common techniques to formalize. There will need to be many formalizations of common techniques in order to discuss their advantages and disadvantages. Consulting systems will provide the option to embed some research now carried out. I see only more work, not less, for research statisticians.

When a Student-like tool becomes easily available, there will be some additional impacts, primarily expanding consulting statisticians' roles. The research statistician would find building consulting system or strategy easier, and would thus be encouraged to do so. The consulting statistician would have an important new role in specializing the systems to their local environment. This would be a continuing activity as well as an initial one, because they would be able to encode the techniques they learned for problems encountered by their users. Users would find the software more specialized to their environment.

I see three limits to any consulting system software in the foreseeable future. First, the systems will have very limited knowledge compared to the entire corpus of statistical knowledge. A good programmer, working with known algorithms and a specified performance, can write programs with

10^6 bits in a year. With diseconomies of scale, unknown methods, and unclear goals, I would not expect that much productivity in writing consulting systems in the next five to ten years. A crude estimate of the extent of the body of statistics knowledge is 10^{11}: 1,000 library shelves, each with 30 volumes, each of 10^6 bits, with another small factor for closely allied areas. Thus only a fraction of the knowledge will be encodable.

Computer systems will not have access to the body of knowledge as represented in the books and journals for the foreseeable future. The work on natural language understanding has shown the huge knowledge that is necessary in order to learn a little bit more. Writing, even in journal articles, is remarkably informal, with frequent appeals to common experience for motivation. Learning techniques in AI are still rudimentary despite a great interest in the topic since the earliest days of AI; this is a hard problem.

The time required for a person to digest the results of an autonomously running program will remain small. Tukey (1986) pointed out the huge number of cycles available on computers every night (10 hours is 4×10^{10} microseconds). It would be desirable to have systems that could use this time autonomously, providing output for the statistician to examine in the morning. As of now, a program that a statistician never tended could interest a statistician for only a few minutes each morning. I think an interesting autonomous program would necessarily have to be a learning program, so I do not see much progress on this in the near future.

As Artificial Intelligence is now only about 30 years old, I do expect progress over a span of several decades on these harder problems. In the mean time, the tools already available make some exciting new options for research by statisticians. And it will be worth watching AI successes in other domains for possible analogues in statistics.

References

The references are grouped by subjects and annotated to enhance their tutorial value.

Review and general

Chowdhury, S I, 1987. "State of the art in statistical expert systems" in *7th International Workshop: Expert Systems & Their Applications*, Avignon: France. (This paper has half page summaries of 17 previous papers).

Gale, W A, (ed.) 1986a. *Artificial Intelligence and Statistics* Reading, Massachusetts: Addison Wesley. This is the first book published in the field. It is a logical starting point for learning about the field.

Gale, W A, 1986e. "Overview," Chapter 1 of *Artificial Intelligence and Statistics*, Reading, Massachusetts. Addison Wesley. The chapter reviews all papers that I knew about at the time, because they were only available in conference proceedings.

Hahn, G J, 1985. "More intelligent statistical software and statistical expert systems: future directions" *The American Statistician* **39** pp. 1–16. This was the first major review. It is directed to statisticians, and contains suggestions for research directions.

Hand, D J, 1986b. "Expert systems in statistics" *The Knowledge Engineering Review* **1** pp. 2–10. Hand's papers have typically a considerable theoretical framework to which their substance is related. This review paper is no exception, doing a nice job of describing aspects of the statistical domain that the KE should appreciate.

Theoretical papers on knowledge representation and acquisition

Aikins, J S, 1983. "Prototypical knowledge for expert systems," *Artificial Intelligence* **120** pp. 163–210. This paper describes Centaur, Aikins' thesis project. It provided a single framework that combined both rules and frames.

de Kleer, J and Brown, J S, 1985. "A Qualitative physics based on confluences," In: Hobbs and Moore, 1985 pp. 109–194. This paper introduces a calculus of directions, and applies it in a few examples.

Gale, W A, 1986d. "Knowledge-based knowledge acquisition for a statistical consulting system," *International Journal of Man-Machine Studies* **26** pp. 55–64. This paper describes the abstract knowledge engineering principles on which student is based.

Gale, W A and Lubinsky, D, 1986. "A comparison of representations for statistical strategies" In: *Proceedings of the American Statistical Association, Statistical Computing Section*, Arlington, Virginia: American Statistical Association, pp. 88–96. The paper compares the knowledge representations used by Student and TESS.

Hand, D J, 1986a. "Patterns in statistical strategy" In: *Artificial Intelligence and Statistics*, Reading, Massachusetts: Addison Wesley pp. 355–388. This paper gives a theoretical view of the task undertaken by the various systems referenced below under strategy of data analysis.

Hayes, P, 1985. "Naive physics I: ontology for liquids" In: *Formal Theories of the Commonsense World*, Hobbs and Moore 1985. Eds, New Jersey: Ablex, Norwood, pp. 71–108. The paper introduces a formalization for the behaviour of liquids.

Hobbs, J, and Moore, R, 1985. *Formal Theories of the Commonsense World*, New Jersey: Ablex, Norwood. This book is a good introduction to building formal theories in areas not previously formalized.

Newell, A, 1981. "The knowledge level" *AI Magazine* **2** (2) pp. 1–20. His presidential address introduced an idea that has organized further research.

O'Keefe, R, 1985. *Logic and Lattices for A Statistical Advisor*, Thesis, University of Edinburgh. This thesis describes a system that selects statistical analysis techniques but the greater value is a beginning on a formalization of data description and experiment design.

Oldford, W and Peters, S, 1986a. "Implementation and study of statistical strategy" In: *Artificial Intelligence and Statistics*, Gale, N A, ed. Reading, Massachusetts: Addison-Wesley, pp. 335–353. This paper gave a theoretical view of strategy for data analysis that is still useful to read.

Oldford, W and Peters, S, 1986c. "Object oriented data representations for statistical data analysis" In *COMPSTAT 1986: Proceedings in Computational Statistics*, De Antoni, F, Lauro, N and Rizzi, A, (eds.) Vienna: Physica-Verlag pp. 301–308. This paper describes the knowledge representation ideas behind DINDE.

Piaget, J and Inhelder, B, 1951. *La genèse de L'idée de hasard chez l'enfant*, Presses Universitaires de France: translated by Leake, Burrell, and Fishbein, 1975, *The Origin of the Idea of Chance in Children*, New York, Norton. A book that describes experiments showing the developmental stages of ideas of randomness and order.

Systems on planning and choice

Anderson, J, Farrell, R and Sauers, R, 1984. "Learning to program in LISP," *Cognitive Science* **8** pp. 87–129. The paper describes an intelligent tutoring system which taught some basics of programming in LISP.

Barzilay, A, 1984. *An Expert System for Tutoring Probability Theory*, Ph.D. Thesis submitted to Graduate School of Business, University of Pittsburgh. The system SPIRIT was built using OPS5. It tutors in conditional probabilities.

Blum, R L, 1982. *Discovery and Representation of Causal Relationships from a Large Time-Oriented Clinical Database: The RX Project*, New York: Springer-Verlag. The book is based on Blum's thesis. It describes the whole RX system.

Dambroise, E, 1987. *Muse: Multivariate Expertise*. Thesis INRA, 9 Place Pierre Viala, 34060 Montpelier Cedex, France: an abbreviated report in English is available in: Dambroise E and Massotte, P "Muse, an expert system in statistics" In: *COMPSTAT 1986: Proceedings in Computational Statistics*, De Antoni, F, Lauro, N and Rizzi, A, (eds.) Vienna: Physica-Verlag, pp. 271–276. The thesis provides Dambroise with enough space to explain MUSE in detail, but the paper is much easier to get.

Haaland, P D, Yen, D and Liddle, R F, 1986. "An expert system for experimental design" In: *Proceedings of the Statistical Computing Section*, Arlington, Virginia: American Statistical Association, pp. 78–87. The paper describes a system which helps an industrial engineer select an experimental design.

Portier, K M and Lai, P, 1983. "A statistical expert system for analysis determination" *Proceedings of the Statistical Computing Section*, Arlington, Virginia: American Statistical Association, pp. 309–311. This paper describes the behaviour of a system based on a decision tree.

Stefik, M J, 1980. "Planning with constraints" *Report No. 80-784, Computer Science Department, Stanford University, (Doctoral thesis)*. This thesis shows how knowledge about molecular genetics can be used to constrain plans and reduce an intractable search space.

Sussman, G J, 1975. *A Computer Model of Skill Acquisition*, New York: Elsevier. This thesis described a program that would learn to write better programs by analyzing bugs in its programs.

Weiner, J M, Horowitz, R and Bauer, M, 1987. "An expert system for analysis of clinical trial data" In: Heiberger, R, (Ed.) *Computer Science and Statistics, Proceedings of the Nineteenth Symposium on the Interface*, North-Holland. This paper is not so much on *an* expert system as on the person/machine organization of the large body of knowledge required to design ethical, legal, and knowledge enhancing experiments on treatment of diseases.

Systems on strategy of data analysis

Berzuini, C, Ross, G and Larizza, C, 1986. "Developing intelligent software for non-linear model fitting as an expert system" In: *COMPSTAT 1986: Proceedings in Computational Statistics*, De Antoni, F, Lauro, N and

Rizzi, A, (Eds.) Vienna: Physica-Verlag, pp. 259–264. This is a first paper on a project to make an intelligent interface to the "Maximum likelihood program," a program encoding an advanced statistical procedure.

Carlsen, F and Heuch, I, 1986. "Express—an expert system utilizing standard statistical packages" In: *COMPSTAT 1986: Proceedings in Computational Statistics*, De Antoni, F, Lauro, N and Rizzi, A, (Eds.) Vienna: Physica-Verlag, pp. 265–270. This system compared the locations of two samples. It used BMDP for calculations, and introduced simulation as a means of testing the system.

Darius, P, 1986. "Building expert systems with the help of existing statistical software" In: *COMPSTAT 1986: Proceedings in Computational Statistics*, F, De Antoni, Lauro, N and Rizzi, A, (Eds.) Vienna: Physica-Verlag, pp. 277–282. Darius used SAS as the programming language for a rule interpreter, and used that to select a test for comparing two group means.

Drapier, P, 1987. "Le système de régression assistée par ordinateur: RAO" preprint, 20 Rue Rouget de L'Isle, 94100 Saint Maur, France. The system described runs on a microcomputer and tackles multiple linear regression.

Gale, W A, 1986b. "REX Review" In: *Artificial Intelligence and Statistics*, Gale, N A, (Ed.) Reading Massachusetts: Addison Wesley pp. 173–228. This was the last and longest of a series of papers on REX. The papers before this do not add much. It describes what REX looked like when it ran, gives an example of the report generated by REX, and describes the inference engine and knowledge representation in detail. It does not discuss the statistical strategy in detail.

Lubinsky, D and Pregibon, D, "Data analysis as search", *Journal of Econometrics*, June, 1988. TESS, the system described, aims to model the *results* obtained by statisticians in analysing data, not the methods they use. The example studied is regression.

Wolstenholme, D and Nelder, J, 1986. "A front end for GLIM" In: *Expert Systems in Statistics*, Haux, R, ed, Stuttgart and New York: Gustav Fischer. The paper describes early work on GLIMPSE, a front end for the popular statistical system GLIM. The book cited has about ten other papers, but they were written at a planning stage rather than at a stage with systems developed.

Systems to aid statisticians

Gale, W A, 1986c. "Student phase 1—a report on work in progress" In: *Artificial Intelligence and Statistics*, Gale, W A, (Ed.), Reading, Massachusetts: Addison Wesley (1986a) pp. 239–266. This described the first version of Student, implemented on a Symbolics Lisp Machine.

Gale, W A, 1987. "Student: a tool for constructing consultation systems in data analysis" In: *Proceedings of the 46th Session of the International Statistics Institute*, Tokyo. This described the second version of Student, implemented in the new S statistical language.

Gale, W A and Pregibon, D, 1984. "Constructing an expert system for data analysis by working examples" In: *COMPSTAT 1984: Proceedings in Computational Statistics*, Havranck, T, Sidak, Z and Novak, M, (Eds) Vienna: Physica-Verlag, pp. 227–236. This paper described the motivation and plans for building Student.

Hand, D J, 1987. "A statistical knowledge enhancement system" preprint, Institute of Psychiatry, London. KENS aids a statistician as an on-line reference book.

Oldford, W and Peters, S, 1986b. "Data analysis networks in DINDE" *Proceedings of the Statistical Computing Section*, Arlington, Virginia: American Statistical Association, pp. 11–18. This paper describes DINDE's behavior and the knowledge representation behind it.

Statistics

Becker, R A and Chambers, J M, 1984. *S: an Interactive Environment for Data Analysis and Graphics*, California: Wadsworth, Belmont. This is the manual for the use of (old) S, a statistical computing system produced by AT&T Bell Laboratories. It contains information on ordering S.

Becker, R A and Chambers, J M, 1986. "Auditing of data analysis" In: *Proceedings of Statistical Computation Section*, Arlington, Virginia: American Statistical Association. pp. 11–18. The paper describes the use of the (new) S, or QPE, in a task of use to statisticians.

Mallows, C L, 1973. "Some comments on C sub P", *Technometrics* **15** pp. 661–667. This paper defined C_p.

Tukey, J W, 1983. "Another look at the future" Heiner, Sachs, and Wilkinson, (Eds.) *Computer Science and Statistics, Proceedings of the Fourteenth Symposium on the Interface*, New York: Springer-Verlag pp. 1–8. This paper poses a dozen challenging tasks for statistical computing, some of which may be feasible with KE methods.

Tukey, J W, 1986. "An alphabet for statisticians' expert systems" In: *Artificial Intelligence and Statistics*, Gale W A, (Ed.), Reading, Massachusetts: Addison Wesley, pp. 401–409. This paper appraises the strengths and weaknesses of the idea of applying KE techniques in statistics.

Velleman, P F and Hoaglin, D C, 1981. *Applications, Basics and Computing for Exploratory Data Analysis*, North Scituate, Massachusetts: Duxbury Press. This book provides practical computer programs and the basic ideas of exploratory data analysis.