1984 ASA proc. Stat. Comp. Sect

BUILDING A STATISTICAL KNOWLEDGE BASED SYSTEM WITH MINI-MYCIN

R. Wayne Oldford and Stephen C. Peters, Massachusetts Institute of Technology

1. Introduction

The wide availability of statistical software which could be easily misued by naive users, together with the demonstrated success of 'knowledge-based' or 'expert' systems in other domains, prompted several statisticians to explore the possibility of introducing expert systems technology into statistical software. Attempts have included 'intelligent' interfaces to existing software (Hájek [1982], Hájek and Iványek [1982]), software which helps the user identify a standard analysis appropriate for the type of data in hand (Portier and Lai [1983]), and software which, together with the user, actively performs an analysis of the given data set (Gale and Pregibon [1982, 1983ab]).

Nelder [1977], Chambers [1981], Chambers et al. [1981], Campbell and Woodings [1981] and Gale and Pregibon [1982, 1983ab] all discuss the value that a statistical expert system might have. From such discussion two principal reasons for building a statistical knowledge-based system have emerged. They are:

- to provide a computer environment which will guide a statistically unsophisticated user to a "safe" analysis;
- to study the statistical strategies involved in good statistical practice.

The first has been described by Chambers [1981] as a 'moral obligation' of the statistical community, and is the professed goal of the systems produced by Gale and Pregibon [1983a] and by Portier and Lai [1983]. The second has been advocated by Gale and Pregibon [1983b]. Statistical strategy has recently been the subject of serious study (e.g., Cox [1977], Cox and Snell [1983], Mallows and Walley [1980], Mallows and Tukey [1983], and Tukey [1982]) without the aid of expert systems technology. It is hoped that this technology will aid a rigourous study of statistical strategy.

The kind of system developed will depend upon which of the above two reasons is predominant. If the goal is to avert disaster in a statistical analysis, then a system which avoids the most common and costly mistakes will be attempted. The system of Portier and Lai [1983] is of this nature. On the other hand, if the purpose is to study statistical strategy, then focussing attention on a very small statistical problem seems to be a natural way to proceed (e.g., Gale and Pregibon's [1983] REX).

We have settled on the latter purpose and what follows, then, is a report of our progress towards developing a statistical knowledge-based system for multiple linear regression analysis, focussing first on collinearity as a subproblem. We shall describe the software that has been implemented, the strategy that was adopted and some methodology which was developed to better describe and understand statistical strategy. In addition, we describe some of the lessons we think we have learned at this juncture and indicate the direction for our subsequent efforts to implement existing ideas and to explore new avenues of research.

The next section describes the anticipated users, and uses, of the proposed system and the reasons for initially focussing attention on the subproblem of collinearity. Section 3 describes the implementation vehicle for our work: Mini-Mycin. Section 4 describes some of the tactics we have chosen to address the collinearity subproblem in this implementation. In Section 5, we introduce some preliminary methodology which has proved useful to us both as a vehicle for describing the lessons learned so far and as an aid to suggest directions for further research. The lessons we've learned in this initial phase of implementation are set out in Section 6. Finally, Section 7 considers future directions, both short and long term.

2. Users and Uses

To help guide our development of the system we have supposed the user to be a non-statistician who is expert in some other discipline, say some social science, and who wishes to perform a regression analysis on his, or her, data. The researcher accesses the computer system and starts an interactive consultation session with software that is 'expert' in linear regression analysis. The system will begin by asking the user questions to ascertain the purpose of the regression (forecasting, tests of significance on coefficients, etc.). This done, the system will continue asking the user questions, as necessary, it will perform possibly extensive calculations, present graphical information, pinpoint problems (collinearity, nonlinear relationships, influential data, etc.) and suggest possible remedies (including 'consult a real statistician').

We currently perceive the task of linear regression analysis to contain four major areas of expertise. They are:

- data transformations;
- collinearity diagnostics;
- influential data diagnostics;
- selection of variables.

More expertise exists to address each of these areas individually than exists to address them collectively. It is therefore our plan to build small prototype 'expert' systems in two or more of these areas before putting them together into an overall regression strategy. Already, we have built a prototype system expert in collinearity diagnosis. The lessons learned there should facilitate the construction of prototypes for the other areas.

We have initially focussed on the area of collinearity diagnosis for two reasons. First, our colleague, David Belsley, has much practical experience in collinearity analysis and provides us with a major source of expertise. Secondly, the area is one where there is not a lot of agreement on how to proceed. Therefore, a developed, and coded, strategy for this problem which could be subjected to objective testing would be a contribution in itself.

3. Mini-Mycin

Our prototype system has been implemented using an expert-systems-building facility called Mini-Mycin. The structure of Mini-Mycin is based on the medical diagnosis system MYCIN (Shoftliffe [1976]) but includes none of the knowledge specific to the medical domain. It is analogous to EMYCIN (Essential MYCIN, Van Melle [1979]), however, programming details are different. Mini-Mycin is a LISP program of about 2000 lines which has been developed by P. Szolovits, R. Patil, and their students at the M.I.T. Artificial Intelligence Laboratory.

We adopted Mini-Mycin because it was an available, 'off-the-shelf' component, home grown in the M.I.T. AI community. Mini-Mycin was delivered to us with the admonition that we should use it as a learning vehicle, that after six months we'd likely contemplate some major overhaul, and this would be true of almost any tool we might have acquired at this stage. Nevertheless, Mini-Mycin has served our initial purposes quite well, and has strongly impressed us with the value of LISP as the underlying substrate: for this state of development, programs written in traditional languages such as FORTRAN, PL/I, C, or PASCAL would have been very awkward to adjust and reshape in the ways and with the frequency we found necessary.

After some experimentation with Mini-Mycin we did tentatively explore the PROLOG 'logic-programming' language as an alternative vehicle for implementation. PROLOG seemed to offer some advantages over Mini-Mycin, principally in technical areas: the back-tracking mechanism was cleaner and faster in PROLOG, and pattern-matching (unification) appeared more thoroughly developed and powerful. We decided that the actual benefits of PROLOG over Mini-Mycin were uncertain, and that retooling would not be worth the trouble.

To understand how Mini-Mycin works it helps to identify three major components: the rule-base (often called the knowledge-base), the inference mechanism, and the explanation facility. Shortliffe [1976] gives book length treatment of the workings of MYCIN (from which Mini-Mycin is derived), we shall necessarily be more succinct. Each of the three components will be taken up in turn.

The rule-base is a collection of English-like if-then-statements. Each rule specifies a set of actions or assertions which should be taken whenever a set of preconditions holds. As a trivial example with just one antecedent and one consequent:

> 'IF number-of-observations < number-of-parameters THEN advise the user of a singularity blunder.'

Notice that number-of-observations and number-of-parameters are essentially arguments or formal parameters of the rule. In any particular instance, these parameters are bound to appropriate values. In general, the antecedents combine numeric and symbolic parameters with a small set of predicates (less than, greater than, known, unknown, etc.), and may be joined with the logical connectives AND or OR. The consequents of a rule cause values to be asserted for other parameters.

The inference mechanism works to combine rules into valid chains of reasonsing. To accomplish this end, Mini-Mycin uses a backward-chaining scheme: given a goal (e.g., determine all advice which can be offered the user), all rules with conclusions bearing on that goal are indentified, then the hypotheses of these rules are adopted as goals which Mini-Mycin further endeavors to satisfy. This process is clearly recursive, a goal is finally satisfied or abandoned when the values of any parameters it references have been established and the predicates on these values have been evaluated to either true or false. The parameters are assigned values either by satisfaction of some goal or in a transaction with the user.

The explanation facility gives Mini-Mycin a limited capacity for self-explanation. At any juncture in its deliberations, Mini-Mycin can be asked how it reached its current goal, or why it is attempting to learn the value of a parameter.

In our first attempt at the collinearity expert system, our goal was to discover what advantages and/or difficulties were associated both with the rule-based approach in general, and with Mini-Mycin in particular. A partial listing of actual rules and parameter descriptions may be found in Oldford and Peters (1984). We begin consideration of these rules by discussing the 'context-tree'.

Both the antecedents and consequents of a rule are populated with parameters. To begin evaluation of some antecedent clause, its parameters are supplied values from the current context. Almost circularly, a context consists of parameter-value bindings. Contexts are organized hierarchically in a tree-structure (see Figure 1). Listed near the context nodes are some of the parameters associated with that context. Contexts are given generic names (e.g., condition-index), instances of a particular context are uniquely identified by a numeric suffix (e.g., condition-index-1, condition-index-2, etc.). The intent of this mechanism is to permit different values to be bound to a parameter in distinct context instances.

So, for example, the parameter 'eta-value' could take the value 1 in a context instance which captures information about the smallest condition index (i.e., condition-index-1), while in the context instance representing the largest condition index (e.g., condition-index-5), this parameter might take on the value, say, 43. The context mechanism permits a certain generality in the rules, namely, we may refer to a parameter generically (e.g., eta-value) without knowning in how many context instances it may be given a value. Continuing the example, suppose 'eta-value' and 'participating-index' are two parameters associated with the context 'condition-index', then with the parameter values previously mentioned, the rule:

> 'IF eta-value > 30 THEN participating-index := TRUE'

when evaluated in the context instance of the largest condition index will establish the value of participating-index as TRUE in that instance; while in the context instance of the smallest condition index, no value would be asserted for participating-index by this rule.

The inference mechanism walks up and down the context tree as it pursues various chains of reasoning. In fact, the tree is grown during this walk, branches are added as required. The walk begins at the 'context root', here, regression-specification-1. A single rule may refer to parameters in several contexts, subject to the restriction that all

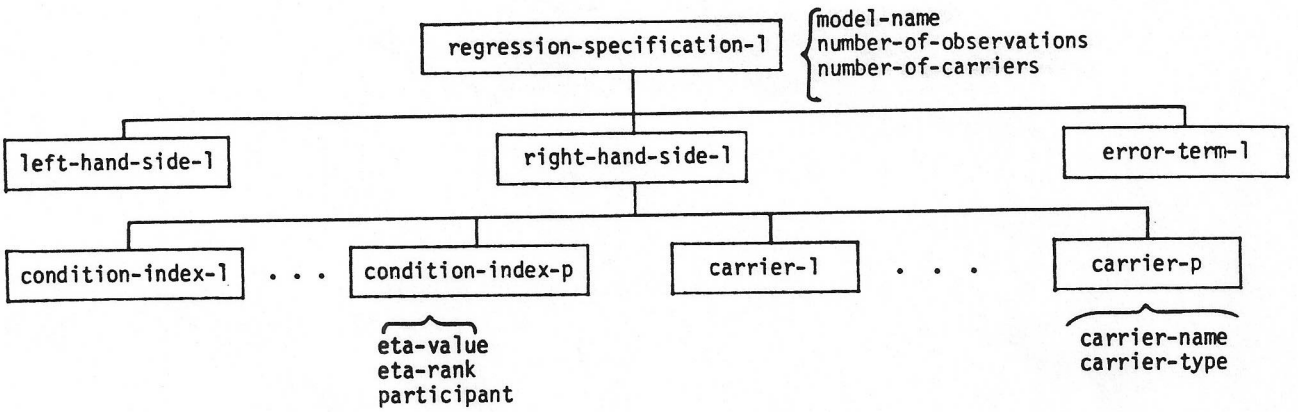

Figure 1

the referenced contexts lie on the single path which returns to the root. This assures unambiguous binding of values to parameters.

We turn next to the definition of parameters. Parameters are symbolic or numeric valued attributes for some aspect of a context. Consider for example the following code, which defines a parameter named 'carrier-type':

```
(gobbleparm  carrier-type single
     (intercept continuous categorical dummy-0-1
                rate-of-change)
carrier labtype
trans  |the type of data represented by this
carrier| )
```

This parameter is declared to be 'single' valued, it can take on exactly one value from the list: {'intercept', 'continuous', 'categorical', 'dummy-0-1', and 'rate-of-change'. This parameter, carrier-type, is associated with the 'carrier' context. Mini-Mycin is to use the 'labtype' method to establish a value for carrier-type, we explain these techniques next. Finally, the meaning of carrier-type is given as 'the type of data represented by this carrier.'

As we received it, Mini-Mycin included two main methods for determining the value of a parameter, we have added a third method. The first of the three is called 'inferred' -- Mini-Mycin attempts to establish the value of the parameter by finding some rule whose consequent provides a value for the parameter, and then tries to satisfy the hypotheses of that rule. The second method is called 'labtype' (for laboratory data type) -- Mini-Mycin required the user to provide a value for the parameter, presumably available from a laboratory test report. As for the method we have added -- Mini-Mycin refers to a cooperating statistical package for the needed value. For technical reasons, this method is joined with the previous 'labtype' method. Originally intended as a means to undertake extensive numerical calculations, the interface to the statistical package also makes available a variety of simple symbolic and numeric information (e.g., names and number of variables, number of observations, etc.). The burden on the user to supply Mini-Mycin these elementary quantities is taken over by the statistical system, which typically records these facts as part of the initial model building and regression stages.

Finally, rules are defined as follows. A rule is given a unique name, has one or more hypotheses, and one or more consequents, these last two are separated in the declaration by the symbol '==>'. Consider the following rule -- EXACT-DEPENDENCY-BLUNDER.

```
(defrule EXACT-DEPENDENCY-BLUNDER
        (same eta-rank 1.)
        (GREATER eta-value 1000000.)
==>     (advice-to-I |
* * * * * * * * * * * * * * * * * * * * * * * * *
*[0.  ] The conditioning of this regression speci-*
*       fication is so poor that I suspect you    *
*       may have blundered by creating an exact   *
*       dependency in the design matrix (e.g.,    *
*       12 monthly dummys with an intercept).     *
*       Check the specification of your model     *
*       carefully.                                *
* * * * * * * * * * * * * * * * * * * * * * * * *
       | 1.0))
```

This rule will be applied in a 'condition-index' context instance. Its meaning is:

> 'IF this is the largest of the condition indices, and the value of this index exceeds one million,
> THEN advise the user of a likely blunder incorporated in the specification of the linear regression model.'

The values of the parameters referenced by the antecedents of this rule: eta-value and eta-rank, are established by means of back-channel communications which Mini-Mycin conducts with the cooperating statistical package. The consequent of this rule asserts a value for the advice-to-I parameter. This value is noted by Mini-Mycin in its knowledge base. All advice so noted is later called out of the knowledge base to construct the consultation report.

Presently, Mini-Mycin runs in Franz Lisp on a VAX 11/730. The statistical package we employ is S.

4. The Collinearity Problem

In this section, our approach to the collinearity subproblem, and the implementation of the strategy with Mini-Mycin is outlined. The present implementation is, of course, not our first (nor likely to be our last); relevant experience will be related. For a general discussion of collinearity, the possibly useful statistics, and some of the issues, see Belsley et al. [1980], Gunst [1983], and Belsley [1984] (with discussion).

For our purposes, the problem may be described as follows. Suppose y and $X_1, X_2, \ldots, X_p$ are vectors of length $n \geq p$. Suppose β is a vector of length p and $X = [X_1, \ldots, X_p]$, an nxp matrix. Linear regression finds β so that $X\beta$ is close to y. If β is estimated with ordinary least squares and the vectors $X_1, \ldots, X_p$ are nearly linearly dependent, then two or more components of β may be poorly determined. This is essentially the collinearity problem. For more formal definitions see Gunst [1983] or Belsley and Oldford [1984].

It is assumed that our researcher, for some reason, is aware of the fact that collinearity may be a problem in linear regression and is curious whether it is in fact a problem in his or her particular case. The system strategy therefore assumes that it will be given some matrix X or a set of vectors $X_1, \ldots, X_p$, with which to work. Then, in conjunction with the user, it attempts to address the following questions:

- Is there a problem at all?
- How many near dependencies need we be concerned about?
- Which variables seem to be involved?
- Does it matter if there is collinearity?
- What can be done to remedy the situation?

Chapter 3 of Belsley et al. [1980] sets out a basic strategy addressing the first three questions. Note that the last two questions depend most critically upon the user. For example, it may be the case that those coefficients which interest the researcher are not affected by the collinearity, or the researcher is interested only in forecasting and then only in a region of the design-space for which the regression plane is well determined. For these purposes, the collinearity problem is irrelevant. Similarly, to remedy the situation the user must be willing to drop variables, change variables, or supply the system with some prior information. For brevity of exposition, the strategy designed and implemented to address only the first two questions will not be outlined. Even in this seemingly simplest case, a number of interesting issues were encountered.

At this level of the strategy, the condition indices, $\eta_1, \ldots, \eta_p$, of the X matrix are the basic units to be examined. Therefore, after some initial questioning of the user to get some basic information about the X-matrix, the first thing the system does is to request from the statistical system the values $\eta_1, \ldots, \eta_p$. The first question above is answered by looking at η_p. If η_p is big there is a problem; if η_p is small there is no problem. Similarly, the second question is answered by counting the number of big η_i's.

This strategy immediately begged the questions 'how big?' and 'how small?'. There is no generally accepted methodology to address these questions for the collinearity problem. However, in a given practical situation an experienced analyst usually does not have much trouble answering them. It is this experience we set out to tap.

Through a series of experiments, Belsley et al. [1980] (see page 153) discovered that when a single near dependency existed, whose "tightness" was determined by the R^2 coefficient of the linear relation involved, the η_p's increased roughly according to the sequence 3, 10, 30, 100, 300, 1000, 3000, ... as the dependency tightened according to R^2 following the sequence .9, .9, .99, .999, .9999, .99999, ... This suggested to us that the base 10 logarithm is a useful scale on which to examine the etas. So we had the system transform the condition indices to $\gamma_i = \log_{10}(\eta_i)$. This done, the first question was partially answered as

- if $\gamma_p \leq 0.5$ (roughly $\eta_p \leq 3$) then there is no problem
- if $\gamma_p \geq 4.0$ ($\eta_p \geq 10{,}000$) then the problem is so bad that the user probably misspecified the X-matrix (e.g., an intercept was included with 12 monthly dummy variables).

Values in between are not so easily dealt with.

Initially, we set out to completely answer both questions by counting the number of etas (or gammas) above 30 (or 1.5). This, however, did not agree very well with what is done in practice. For example, if the condition indices are 1, 3, 5, 29, then there is still evidence that a singular near dependency exists. The cutoff of 30 could not be hard and fast. Indices near 30 need also be considered. One other problem arose with this approach; it became clear that condition indices above 30 should not all be treated alike. For example, if the condition indices are 1, 3, 35, 5000, 7000, 8000, then we are much more interested in investigating the last three dependencies than the one associated with $\eta_3 = 35$.

These issues were addressed by developing a procedure which we call 'multiple hard cuts with softened edges', or MHC for short. The object of this procedure is to divide the condition indices into two groups: those that merit further investigation and those that don't. It achieves this by first arranging the gammas along the non-negative real line and then partitioning this half line strategically into four sets: A_1, A_2, A_3, and A_4 (hence the term "multiple hard cut"). The number, a_i, of gammas in each set, A_i, is then determined. The A_i's were chosen to provide practically meaningful distinctions between the gammas. For example, if $a_4 > 0$, then certainly at least a_4 near dependencies existed. If, additionally, $a_3 = 0$ and $a_2 > 0$, then those near dependencies indicated by the gammas in A_2 would be ignored because of the overriding importance of those indicated in A_4. Some consideration was also given to those γ's which were near the borders of the sets, hence the term 'softened edges'. The MHC procedure was built using many (>30) rules of thumb and was first programmed using Mini-Mycin. This proved to be somewhat awkward and the entire procedure was recoded in Fortran (see Section 5). The procedure itself is now considered a single rule of thumb and is treated as such in the Mini-Mycin code.

5. Some Methodology

In building a statistical knowledge based system, it is convenient to think of a statistical analysis as a collection of procedures applied in some order. The procedures used, and their order of application, will depend on the problem at hand. In this section, we provide some preliminary methodology which we find both useful, in describing the procedures themselves, and helpful, in formulating what it is we have learned so far. The methodology is not meant to be definitive, rather it provides a "useful fiction", in Lenat's [1983] words,

with which we may work. To this end, we find that the following four somewhat autonomous dimensions allow useful descriptions of procedures:

- Statistical strategy level: low vs. high.
- Functional level: mechanical vs. tactical.
- User judgement level: corroborative vs. inferential.
- Contextual level: subject matter dependent vs. context free.

Typically, different applications may require the same procedure to be described differently. Examples are given as we elaborate each of the four dimensions below.

Statistical Strategy Level: Recently, many authors have discussed the notion of statistical strategy in practice, notably Cox and Snell [1981]. These discussions tend to be concerned with what we call "high-level" strategy. That is they address broad matters which are common to most statistical analyses, such as its purpose, the quality of the data, and the amount of prior knowledge. Perhaps the first thing one notices when designing and implementing a statistical knowledge based system is that strategy must be developed at many levels. An example of low-level strategy would be a specific plan to select good tuning constants for a robust-regression estimator. At a higher level, one could be deciding whether to use a robust estimator at all. Two things should be noted. First, the strategy level does not necessarily correspond to the time order in which the procedures are to be executed. The procedures used, and the order of their use, will depend upon many considerations including the quality of the data and the information desired from it. It is possible, however, that the strategic-level position of one procedure will always be higher than another's. Second, this ordering according to strategic level is at most a partial ordering. Within a given strategy, it may be the case that different procedures occupy the same strategic-levels. For example, influential-data diagnostics and collinearity diagnostics can be considered to occupy approximately the same strategic-level (cf. Belsley et al. [1980]). This could be the case even though higher strategic-level procedures may say that one of the two will not be of interest to pursue.

Functional Level: The obvious procedures, like estimation methods, test statistics, and so on are largely *mechanical* in nature. Others, such as deciding which kind of test statistic to use next or interpreting the results of several test statistics, each designed to be powerful against a particular set of alternatives, are by and large *tactical* procedures. These latter procedures are usually based both upon statistical training and upon experience gained from practice. Chambers [1981] and, more strongly, Gale and Pregibon [1983b] have made a similar distinction between *mechanics* and *strategy*. We prefer the term *tactics* over strategy because we have found that, while any given procedure is sometimes best described as tactical and other times as mechanical, in either case it is an integral part of the overall strategy. As an example of the kind of distinction being drawn, consider the MHC procedure briefly described in Section 4. At first, we understood it as a complex tactical procedure which used various heuristics gleaned from practical experience. MHC used the singular value decomposition as a mechanical procedure to provide it with the basic units of information it was to manipulate. With MHC coded in Mini-Mycin, the 'why' facility became less useful in making the overall strategy of the collinearity analysis transparent. So much time was spent explaining how MHC worked that the user was likely to become frustrated and confused when trying to understand the analysis. It became necessary to regard MHC as a 'black box' which took condition indices as input and gave the desired groups and other information as output. The MHC procedure was now regarded as a mechanical one to be manipulated by higher level tactical or mechanical procedures. It was therefore recoded in Fortran, and removed from the Mini-Mycin framework, becoming unexplainable by the 'why' facility. The following observations are helpful in determining whether a procedure is regarded as mechanical or tactical:

- mechanics are the atoms of tactics
- tactics may be controversial
- mechanics need not give the reasoning used to produce a particular output.

In our implementation, the rules which reside in the Mini-Mycin software, its context tree, and its nodal parameters constitute tactical procedures. Calls to a Fortran subroutine, or a statistical package external to the Mini-Mycin code are considered mechanical procedures. In this way, the MHC procedure evolved from a collection of many small tactical procedures to a single mechanical one.

User Judgement Level: Building an expert system always begs the question 'How much can be automated?'. Probably the best answer to this question is 'Not everything'. One soon realizes that for many procedures the user will, at most, be asked to corroborate specific findings. However, there are procedures for which the system cannot, or will not, make inferences. At this point, the analysis depends upon the user for the appropriate inference. We anticipate that the issue of judgement level will arise in a variety of ways, of which three important ones are:

- statistical graphics
- a user selected level of interaction
- required subject matter knowledge.

Statistical graphics have been the subject of much research interest in recent years (see Chambers et al. [1983] and Tufte [1983] for discussion and references) largely for two reasons. The first of these is the recent availability of high resolution computer graphics. The second is the realization that information can often be more efficiently presented with pictures than with numbers. A different kind of information is available with pictures. The eye has the ability to quickly recognize unexpected features of the picture, an ability which would be difficult, if not impossible, to code on a digital computer. In this domain, the corroborative level seems to be equivalent to using a statistical graphic for display purposes alone, while the user inferential level is more akin to using the graphic for exploratory purposes.

Perhaps the most obvious way for the judgement level to change is to allow the user to specify how much of the analysis will be the system's re-

sponsibility. The user can abdicate some responsibility and allow the system to produce an analysis, querying a user at most for corroboration. This might be most appropriate when the user desired a 'quick and dirty' sort of analysis. Today's systems represent the other extreme where the user is expected to make all inferences.

The third point is important enough to merit distinction as a separate dimension.

Contextual Level: Whether certain statistical procedures are to be applied sometimes depends upon the subject matter area as well as the data and the purposes of the analysis (again see Cox and Snell [1981]). For example, in linear regression analysis, regardless of the context, it is generally accepted as good practice to be aware of those data points which are extremely influential in determining the fit. However, when it comes to variable selection, procedures like maximum R^2, which choose the variables to appear on the right hand side of a regression equation, can have a very limited role to play in areas like economics where theory may dictate what variables should or should not be selected. It is therefore important to discern which procedures are basically context free and which depend critically on the context for their reasonableness.

6. What We Have Learned

In this section we summarize what we think we have learned so far in our efforts towards building a statistical knowledge-based system for regression analysis.

- Unlike, say medicine, the diagnostic protocols of the statistician are largely uncharted. Much remains to be learned about the overall organization of a statistical analysis.
- The most difficult issue, by far, is understanding and codifying expert performance. Details count, and they are numerous. Capturing judgements which are based on perception (seeing clusters of numbers or gaps) presents special problems.
- Mini-Mycin provides a useful language environment for programming a knowledge-based system. It is likely that other tools would have done as well. What seems to be important is the flexibility and rapid development cycle offered by these LISP-based systems.
- Mini-Mycin has a 'blind spot' where arithmetic is concerned, but it is easy enough to delegate numerical tasks to the cooperating ststistical package, or to augment Mini-Mycin. We have settled on the former.

7. Further Work

In the coming months we plan to take up the following (in roughly the order given):

- Completion of the collinearity system.
- Evaluation of the completed collinearity system. Probably by allowing users to have a go at it, and looking over their shoulders. Possibly in a more formal setting, by comparing findings with actual experts.
- Extending our system to deal with influential data and other regression diagnostics.

8. Bibliography

Belsley, D.A. (1984), *Amer. Stat.*, 38, pp. 73-77.

Belsley, D.A. et al. (1980), *Regression Diagnostics:* Wiley.

Belsley, D.A. and R.W. Oldford (1984), *Proc. ASA, Bus. Econ. Stat.*

Cambell, N.A. and T.L. Woodings (1981), *43rd I.S.I.*, Buenos Aires.

Chambers, J.M. (1981), *13th Interface.*

Chambers, J.M. et al. (1983), *Graphical Methods for Data Analysis*, Duxbury Press.

Chambers, J.M., D. Pregibon, and E. Zayas (1981), *43rd I.S.I.*, Buenos Aires.

Cox, D.R. (1977), *Proc. 41st I.S.I.*, 47, 1, pp. 552-558.

Cox, D.R. and E.J. Snell (1981), *Applied Statistics*, Chapman and Hall.

Gale, W.A. and Pregibon, D. (1982), *Proc. 14th Interface*, pp. 110-117.

Gale, W.A. and Pregibon, D. (1983a), "Building an Expert Interface," Technical Memorandum, ATT Bell Labs.

Gale, W.A. and D. Pregibon (1983b), "Using Expert Systems for Developing Statistical Strategy," Joint Statistical Meetings, Toronto.

Gunst, R.F. (1983), *Comm. in Stat.*, A12, pp. 2217-2260.

Gunst, R.F. (1984), *Amer. Stat.*, 38, pp. 79-82.

Hájek, P. (1982), *Proc.*, ECAI-82, Orsay, France.

Hájek, P. and J. Ivanek (1982), *COMSTAT-82*, Vienna: pp. 54-60.

Havránek, T. (1981), *J. of Man-Machine Studies, 15*, pp. 253-264.

Lenat, D.B. (1982), *Artificial Intelligence*, 19, pp. 189-249.

Mallows, C.L. and J.W. Tukey (1983), in *Some recent advances in Statistics*, Academic Press.

Mallows, C.L. and P. Walley (1980), *Proc. ASA, Bus. Econ. Stat.*

Nelder, J.A. (1977), in *Recent Developments in Statistics*, North-Holland; pp. 79-86.

Oldford, R.W. and S.C. Peters (1984), MIT CCREMS, TR No. 42.

Portier, J.M. and P. Lai (1983), *Proc. ASA*, Stat-Comp, pp. 309-311.

Shortliffe, E.H. (1976), *Computer-Based Clinical Therapeutics: MYCIN*, American Elsevier.

Tufte, E.R. (1983), *The Visual Display of Quantitative Information*, Graphics Press, Conn.

Tukey, J.W. (1982), in *Modern Data Analysis*, pp. 1-11.

Van Melle, W. (1979), *IJCAI*, pp. 923-925.

Acknowledgements: This is part of a large ongoing research project at MIT involving, in addition to the authors, David Belsley, Edwin Kuh, Alexander Samarov and Roy Welsch. This research was sponsored by NSF grant IST-8218759. Thanks also go to Karen Martel for an excellent typing job.