EQUATION DISCOVERY IN DATABASES FROM ENGINEERING

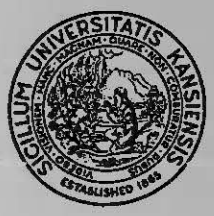
By
Liye Zhang
W. M. Kim Roddis

A Report on Research Sponsored by

THE NATIONAL SCIENCE FOUNDATION
Research Grant No. CDA-9401021 Amendment 4
Proposal No. CDA-9743731

INFORMATION & TELECOMMUNICATION
TECHNOLOGY CENTER
University of Kansas

Structural Engineering and Engineering Materials
SM Report No. 53
April 1999



THE UNIVERSITY OF KANSAS CENTER FOR RESEARCH, INC.
2291 Irving Hill Drive - Campus West, Lawrence, Kansas 66045

EQUATION DISCOVERY IN DATABASES FROM ENGINEERING

By
Liye Zhang
W. M. Kim Roddis

A Report on Research Sponsored by

THE NATIONAL SCIENCE FOUNDATION
Research Grant No. CDA-9401021 Amendment 4
Proposal No. CDA-9743731

INFORMATION AND TELECOMMUNICATION TECHNOLOGY CENTER
University of Kansas

Structural Engineering and Engineering Materials
SM Report No. 53

UNIVERSITY OF KANSAS CENTER FOR RESEARCH, INC.
LAWRENCE, KANSAS
April 1999

ABSTRACT

As the quantity of electronically generated engineering data grows rapidly, building computer systems to analyze data automatically and intelligently becomes increasingly important to engineers. The overall process of extracting useable knowledge from electronically stored data is called knowledge discovery in databases. The part of the process where patterns are extracted or models are built is referred to as data mining.

This dissertation proposes a data mining method that combines machine learning and regression to help engineers in acquiring knowledge which is preferably expressed as equations. A learning algorithm based on the method has been implemented in the computer system EDDE (Equation Discovery in Databases from Engineering). In addition, to obtain useful models that are understandable to engineers, knowledge specific to the particular problem area is incorporated into EDDE to guide the discovery process. The role of this domain knowledge is investigated.

The system EDDE is extensively tested on both synthetic data sets and actual engineering data sets. The tests on synthetic data show that EDDE has some important features, such as not being sensitive to the number of variables in data sets. When compared to other methods (regression tree CART, instances based IBL, multivariate linear regression, model tree M5, neural nets, and combinations of these methods), EDDE generates a smaller size model with lower prediction error. EDDE thus summarizes the data more concisely and describes the data better.

EDDE has been used to analyze actual data sets from civil engineering (duration of construction activities, development/splice length of reinforcing bars, and

CONTENTS

LIST OF TABLES

CHAPTER 1

INTRODUCTION

Engineering is data rich. The data collected for most engineering purposes are generally associated with sets of observations and examples from problem domains. However, the data sets have limited usefulness because they describe the behavior of the domains only at the level of examples, while providing no insight into the domains.

To better understand engineering domains, engineers are often challenged by the need to analyze large databases. Such data analysis has two aspects. On the one hand, analysis of the data offers the opportunities of acquiring useful knowledge implicitly underlying the observed data and using the newly discovered knowledge in conjunction with the existing knowledge. On the other hand, the quantity of data grows and the number of dimensions increases with the easy collection and storage of data by computers. Consequently, manually analyzing the data from engineering domains is time consuming, or even impossible within any practical length of time. Building computer systems to analyze data automatically and intelligently becomes more and more important for engineers to overcome the difficulties of dealing with large quantities of data to acquire useful knowledge on the domains.

This thesis is concerned with building an intelligent computer system to help engineers in acquiring knowledge. This chapter first gives an overview of research in terms of data mining, machine learning, and intended problem characteristics, then closes with a brief description of the organization of this thesis.

Table 1.1 The steps in the KDD process [Fayyard, Piatetsky-Shapiro, Smith, 1996b]

1. Learning the application domain: includes relevant prior knowledge and the goals of the application.
2. Creating a target dataset: includes selecting a dataset or focusing on a subset of variables or data samples on which discovery is to be performed.
3. Data cleaning and preprocessing: includes basic operations, such as removing noise or outliers if appropriate, collecting the necessary information to model or account for noise, deciding on strategies for handling missing data fields, and accounting for time sequence information and known changes, as well as deciding DBMS issues, such as data types, schema, and mapping of missing and unknown values.
4. Data reduction and projection: includes finding useful features to represent the data, depending on the goal of the task, and using dimensionality reduction or transformation methods to reduce the effective number of variables under consideration or to find invariant representations for the data.
5. Choosing the function of data mining: including deciding the purpose of the model derived by the data mining algorithm (e.g., summarization, classification, regression, and clustering)
6. Choosing the data mining algorithm(s): includes selecting method(s) to be used for searching for patterns in the data, such as deciding which models and parameters may be appropriate (e.g., models for categorical data are different from models on vectors over reals) and matching a particular data mining method with the overall criteria of the KDD process (e.g., the user may be more interested in understanding the model than in its predictive capabilities)
7. Data mining: includes searching for patterns of interest in a particular representational form or a set of such representations, including classification rules or trees, regression, clustering, sequence modeling, dependency, and line analysis
8. Interpretation: includes interpreting the discovered patterns and possibly returning to any of the previous steps, as well as possible visualization of the extracted patterns, removing redundant or irrelevant patterns, and translating the useful ones into terms understandable by users
9. Using discovered knowledge: includes incorporating this knowledge into the performance system, taking actions based on the knowledge, or simply documenting it and reporting it to interested parties, as well as checking for and resolving potential conflicts with previously believed (or extracted) knowledge

Mining offers a promising approach to attack the problem.

Based on the consideration of the stated characteristics, the method proposed combines a machine learning technique and regression analysis to automatically and intelligently help in discovery of knowledge hidden in data.

1.2 MACHINE LEARNING

Learning from databases with a set of examples or observations is called empirical learning or inductive learning. This type of learning is one of the most extensively studied in machine learning and statistics, and is the subject of this thesis.

Empirical learning systems can compress and abstract a large data set into a higher level compact model that more concisely describes the data. Based on the degree of supervision a system obtains, there are two techniques that use given examples to derive a concept: *supervised learning* and *unsupervised learning*.

In the activities of supervised learning, a tutor or domain expert provides the information about which variable to focus on during the learning task. In other words, the domain variables are divided as dependent variables and independent variables for supervised learning, in sharp contrast to unsupervised learning activities where such information is absent. However, many methods designed for supervised learning problems can be adapted to unsupervised ones [Langley, 1996].

For most engineering data analysis tasks, the domain knowledge provides sufficient information for engineers to decide which variables to focus on. This is the reason that this thesis aims at supervised learning instead of unsupervised learning. The learning activity scenario of interest in this thesis is one that happens quite often in engineering, and can be stated as follows:

> Given a data set, $\mathbf{L}$, with N examples $\mathbf{L} = \{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_N\}$. Each example $\mathbf{e}_i$ in the data set has one targeted dependent variable y and a fixed number (M) of variables $\mathbf{X} = \{x_1, x_2, \ldots, x_M\}$ in a domain space $\mathbf{D}$, where $\mathbf{X} \in \mathbf{D}$. Therefore, $\mathbf{e}_i$

Generally, an engineering database includes high dimensionality, a mixture of data types, and different relationships holding between variables in different parts of the domain space as mentioned in Section 1.1. Furthermore, engineering problems are various. Building a system that can learn from all kinds of engineering databases does not appear feasible with current computational techniques. Instead, this research presents a method suitable for a particular set of characteristics of the database and acquired equations as discussed in the next section.

1.3 INTENDED PROBLEM CHARACTERISTICS

Frequently in engineering domains, both prediction and description are equally stressed. The functions found from data require not only satisfactory prediction on unseen or new cases, but also understandability to the domain engineers. Therefore, engineers want to introduce a simple, understandable model based on the domain knowledge in guiding data analysis. That is to say, engineers give their expectations about possible forms of the numerical relationships. The introduction of domain knowledge restricts the relationships that can be found. The domain knowledge of the problem type intended for solution by the proposed method includes:

- The problem space is multi-dimensional and nonhomogeneous. Therefore, Eq. (1.1) can be more explicitly expressed as

$$R_i : \; y = f_i \{\mathbf{X}\} \tag{1.2}$$

  where R_i are regional descriptions and f_i are regional equations.

- Independent variables **X** are divided into *description variables* and *prediction variables*. Description variables are used in region descriptions R_i, and prediction variables are used in region equations f_i. The prediction variables have to be numeric while the description variables can either be numeric or symbolic. However, if a numerical variable is used as

affect the degree of the influence of the road length. That is to say, the influence of the road length behaves differently when projects are located in different regions. This is reflected in equation expressions where the same relationship does not hold over the entire problem domain so that different equations hold in different parts of the problem space. In the actual situation, the problem is not so simple because many project attributes (or variables as they are called in general data analysis) are involved in the database with historical records of past construction. The road construction projects in the database can be used as a training set of examples which the data mining method uses to build a model to predict activity durations based on project information including length, tracts, location, and situation. Models could also be built to describe the data in a way that lets the user better understand what the data means, for example, by looking at work done by various contractors and seeing what project types were best done by what contractors.

This planning and scheduling case is one example of the type of engineering learning that is the subject of this thesis.

1.4 THESIS ORGANIZATION

The remainder of this thesis is organized as follows:

Chapter 2 first introduces the concept of learning. The definition by Langley [1996] is given, which the author believes is the most broad and accurate. Then, the chapter reviews the machine learning methods in five paradigms according to knowledge representation. Finally, the chapter discusses in detail the systems designed for function discovery, which are related to the research presented in this thesis. Their learning capabilities and shortcomings are summarized.

Chapter 3 describes EDDE, a learning system for Equation Discovery in Databases from Engineering. The chapter discusses the technical approach specified for EDDE in terms of the learning task, knowledge representation, equation discovery

CHAPTER 2

REVIEW OF RELATED WORK

Learning is viewed as the central feature of intelligent systems. This chapter starts with a brief preview of the definition of learning. Then, it reviews important categories of machine learning techniques grouped by knowledge representation. Finally, it reviews in more detail the machine learning techniques for numerical law discovery, which are closely related to the research presented in this thesis.

2.1 PREVIEW

It would be satisfying to here state a clear, unambiguous definition of learning. Unfortunately, any attempt to draw a fixed boundary around such a broad concept is doomed to failure. Although one can give a precise formal definition, others can always find intuitive examples that fall outside the specified boundaries and counterexamples that fall within them. However, many definitions have been given before and some of them are reviewed in [Reich and Fenves, 1989]. Among the previous definitions, the author believes that the definition given by Lanley [1996] is the broadest and the most accurate:

> *Learning* is the improvement of performance in some environment through the acquisition of knowledge resulting from experience in that environment.

This definition states that learning can only happen with the presence of four

machine learning techniques are classified differently. Here Langley's classification of machine learning techniques is adopted, which classifies the machine learning techniques into five paradigms according to knowledge representations [Langley, 1996].

2.2.1 Neural Networks

Neural networks represent knowledge as a multilayer network of threshold units that spreads activation from input nodes through internal units to output nodes. Weights on the links determine how much activation is passed on in each case. The neural network typically attempts to improve the accuracy of classification or prediction by modifying the weights on the links. A comprehensive presentation of various neural networks is given in [Freeman and Shapura, 1991; Shapura, 1996].

2.2.2 Instance-Based or Case-Based Methods

Instance-based methods, rather than forming some abstract and storing this structure in memory, represent knowledge in terms of specific cases or experiences and rely on flexible matching methods to retrieve these cases and apply them to new situations. Different algorithms are developed based on the matching methods. This paradigm contains methods such as nearest neighbor algorithms [Cover and Hart, 1967; Dasarathy 1991], k-nearest neighbor algorithms [Stanfill and Waltz, 1986], and average-case analysis [Langley and Iba, 1993].

2.2.3 Genetic Algorithms

Genetic algorithms were derived from the evolutionary model of learning [Forsyth, 1989]. They typically represent acquired knowledge as a set of Boolean or binary features, which are sometimes used as the conditions and actions of rules. Genetic algorithms use the Darwinian principle of 'survival of the fittest'. A genetic classifier is comprised of a set of classification elements that replicate and mutate to

As Langley points out, the reasons for the distinct identities of these paradigms are more theoretical than scientific [Langley, 1996]. Recent development in the machine learning community has helped break down these boundaries, and hybrid approaches that cross these boundaries are increasingly common.

2.3 NUMERICAL LAW DISCOVERY

Most methods discussed above construct knowledge structures that classify objects into a finite number of classes. We may refer to them as classification systems. For some applications, however, we would like to predict the value, usually numeric, of an attribute of interest. If we still adopt the classification methods, we can construct decision rules or other forms of knowledge, but some problems arise including the following.

If we construct decision rules that predict the unknown value, we would end up with as many rules as these are different values. A solution to this is to discretize these values by mapping them into a finite number of classes. Methods taking this discretization approach include CART [Breiman, Friedman, Olshen and Stone, 1984] and M5 [Quinlan, 1992]. However, discretization involves a loss of information and prediction accuracy will be influenced. The prediction accuracy depends on how finely the classes are discretized.

If we construct other forms of knowledge such as a neural network, the knowledge underlying the data is not explicitly expressed. Such implicit classification methods are unsatisfying for engineering applications that use models for both prediction and description.

As engineers, we would like the knowledge in the domain to be expressed as numerical functions because they clearly and definitely state the relationship among domain variables qualitatively and quantitatively. Therefore, previous work related to the knowledge acquisition process of numerical law discovery is particularly relevant.

revolution p are related as $\frac{d^3}{p^2} = k$ where k is a constant.

Table 2.1 Data obeying Kepler's third law of planetary motion.

Planet	d	p	Term-1: $\frac{d}{p}$	Term-2 : $\frac{d^2}{p}$	Term-3 : $\frac{d^3}{p^2}$
A	1	1	1.0	1.0	1.0
B	4	8	0.5	2.0	1.0
C	9	27	0.33	3.0	1.0

A set of examples, i.e. the values of d and p for three (strictly hypothetical) planets, are given in the second and third columns of the Table 2.1. Based on the observation that d and p increase monotonically, the term $\frac{d}{p}$ is constructed. This term is not constant, we can see in the fourth column, so term construction continues. Since d and $\frac{d}{p}$ vary inversely, a new term $d\left(\frac{d}{p}\right) = \frac{d^2}{p}$ is constructed. This term varies inversely with $\frac{d}{p}$, so their product $\frac{d^3}{p^2}$ is computed. Computation of this term results in a constant for all examples. The algorithm reports it as the numerical relationship that can describe all the data, and then stops.

BACON has some requirements for input data. It requires enough data to be gathered so it is always possible to observe changing values of two variables while holding the others constant. This requirement can be satisfied if the data comes from controllable experiments. Furthermore, it allows no irrelevant variables in the data provided. That is to say, all variables are relevant. This requirement suggests all variables are present in the function reported by the system, which is quite often not the case in engineering domains.

among the variables.

Abacus does allow irrelevant variables to be present in the input data. The presence of irrelevant variables in data sets exacerbates the problem that combining existing variables to form new ones is inherently exponential. To limit searching, Abacus uses three constraints that prevent mathematically redundant or physically impossible equations from being formed. First, Abacus prohibits redundant terms. For example, if a term $\frac{xy}{z}$ has been created, this strategy will prevent the term formation of $\frac{x}{z}y$. Second, Abacus prohibits combinations that would result in cancellation. For example, this strategy prevents the discovery of tautologies such as the invariance of $\frac{xy}{xy}$. Third, Abacus applies the notion of dimensional analysis. For example, if two terms x and y are measured in different units, this strategy prevents the formation of new terms of $x+y$ and $x-y$.

Another innovation introduced into Abacus is its ability to handle cases in which different functional relationships govern different parts of the input data set. Abacus deals with multiple relationships by finding formulas which are invariant over a significant subset of the data, characterizing and removing this subset and then, recursively, analyzing the remaining data. When multiple equations are discovered for a given set of data, Abacus has a separate module to generate a condition with each equation.

Abacus, like BACON, requires enough data to be collected so that it could always be able to observe changing values of two variables while holding the others constant. It is tested on artificial data sets similar to those tested by BACON. Its ability to analyze real data is also weak.

Other related system are Abacus.2 [Greene, 1988], Coper [Kokar, 1986a and 1986b], Fahrenheit [Zytkow, 1987] and Fortyniner [Zytkow and Baker, 1991].

IDS's search. That is to say, the system selects the function that fits the training data set, not the one that does the best job of predicting a set of unseen data. In this aspect, IDS is also different from the BACON and Abacus type systems discussed above where predictive accuracy is not a concern.

2.3.4 Kepler

Kepler, constructed by Wu and Wang [1991], is a system designed for discovery of functional relationships from observational data. As its authors claimed, it is designed to discover more complicated functions than BACON and Abacus. Like BACON and Abacus, Kepler finds functional relationships by detecting the invariant. However, the relationship is not expressed as a term but as

$$f(x_1, x_2, ...x_n) = k = \text{constant} \tag{2.2}$$

To simplify the problem, Kepler has three assumptions: (1) each variable appears in the formula just once, (2) each operator takes at most two variables as its arguments, and (3) two functions that differ only by a constant (for example, x^2+3 and x^2+7) are considered equivalent.

Unlike BACON and Abacus, which form new terms only using the basic arithmetic operations: addition, subtraction, multiplication and division, Kepler uses primitive functions that are defined as a nondivisible part of a formula. Dividing a formula means putting the formula into different parts, with each variable appearing in only one part. A formula can be discovered by discovering its part. This discovery is possible because the parts are independent of each other. Therefore, discovering a complex multivariable formula is accomplished by finding its primitive functions. Some examples of prototypes of primitive functions are

$$x^{k_1} \times y^{k_2}$$

$$x^{k_1} / y^{k_2}$$

2.3.5 KEDS

The systems reviewed above are intended to be domain independent while KEDS (Knowledge based Equation Discovery System), as its name implies, is intended to be domain dependent [Rao and Lu, 1993; Rao, 1993]. It attempts to discovery comprehensible models in an engineering domain that is multidimensional and nonhomogeneous. These models are expressed as region-equation pairs of the form:

$$R_i : y = f_i(x_1, x_2 \ \ldots) \tag{2.4}$$

where R_i is the description of the region i, y is the predicted (dependent) variable, and $x_1, x_2, \ldots x_n$ are independent variables. f_i is the region related equation, which is a user defined template. The templates represent the domain knowledge. In KEDS, the templates are expressed as families of parameterized models (polynomial equation templates) such as

$$\begin{aligned} y &= c \\ y &= ax_1 + b \\ y &= ax_1^{\,2} + bx_1 + c \\ y &= ax_1 + bx_2 + c \end{aligned} \tag{2.5}$$

where a, b, and c are unknown coefficients.

To accomplish the discovery process, KEDS adopts the method of empirical discovery and concceptual clustering. Its algorithm involves two phases: discovery and partitioning. In the discovery phase, the discovery process is restricted within the boundaries of the regions created by the partitioning. In the partitioning phase, partitioning is model-driven and based on the relationships that are discovered from the data. The discovery and partitioning phases are coupled and are thus done interactively, not sequentially.

$$
\begin{aligned}
&y = k_1 x + k_2 \\
&y = k_1 x^{-2} \\
&y = k_1 x^{-1} \\
&y = k_1 x^{-1/2} \\
&y = k_1 x^{1/2} \\
&y = k_1 x \\
&y = k_1 x^{2}
\end{aligned}
\qquad (2.6)
$$

or the null relation equivalent to "none of the above". Three measures, *significance, distinction*, and *systematic lack of fit*, are used to do the classification. The E* algorithm focuses, in fact, on improving the reliability of direct curve finding but not on function finding [Schaffer, 1991]. For this reason, the E* algorithm is not discussed further here. More details can be found in [Schaffer, 1990].

equation discovery in engineering problems. On the one hand, different engineering problems have their own characteristics and need different methods to solve them; on the other hand, a specific method has its limitations and is not applicable to all kinds of problems.

EDDE is designed for a coherent class of problems with the characteristics and goals discussed in Chapter 1, which attempts to find a relationship between a dependent variable and other independent variables, expressed as region-equation pairs shown in Eq. (1.2). The task is restated here:

> Given a data set, **L**, with N examples $\mathbf{L} = \{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_N\}$. Each example $\mathbf{e}_i$ in the data set has one targeted dependent numerical variable, y, and a fixed number M of variables $\mathbf{X} = \{x_1, x_2, \ldots, x_M\}$ in a domain space **D**, where $\mathbf{X} \in \mathbf{D}$. Therefore, $\mathbf{e}_i = \{y, \mathbf{X}\}$. The data analysis requires finding a model between the dependent variable y and the independent variables **X**, which is expressed as
>
> $$R_i : y = f_i\{\mathbf{X}\} \tag{3.1}$$
>
> where R_i is the region description that sets region boundaries; and f_i is the equation related with region i. The region descriptions are determined by description variables, and the region equations are determined by prediction variables. The information about description variables and prediction variables is provided by domain engineers based on the domain knowledge before the learning begins.

EDDE is targeted to apply to real world engineering problems. Therefore, the given data is almost guaranteed to be noisy with not all of the data points satisfying the model. The only solution is to learn an approximate model. Hence, Eq. (3.1) becomes

$$R_i : y = f_i\{\mathbf{X}\} + Z(0, \sigma^2) \tag{3.2}$$

where the term Z is added to reflect the noise existing in the data. It is assumed that Z

learning partial models and then combining the partial models into a complete model as KEDS does [Rao, 1993]. A partial model is referred to as one region-equation pair in KEDS.

Learning a model represented as region-equation pairs involves learning both region descriptions and region equations. The regional equation f_i may be any function of the prediction variables. It may be a linear or nonlinear function. When only linear functions are taken into account, f_i may be expressed as Eq. (1.3). Note that this equation gives only the general form, not the actual function, because significant variables in the equation are not identified until the model is induced.

In many engineering domains, linear functions can not give satisfactory results. Nonlinear functions are necessary for engineering problems. When the linear function restriction is relaxed, nonlinear functions may appear in regression models. Based on the characteristics appearing in regression models, nonlinear functions can be divided into two types: *intrinsically linear* and *intrinsically nonlinear* [Draper and Smith, 1981]. These two types of nonlinear functions are explained by Draper and Smith as follows:

> Two examples of such models are
>
> $$Y = \exp\left(\theta_1 + \theta_2 t^2 + \varepsilon\right) \quad \text{(a)}$$
>
> $$Y = \frac{\theta_1}{\theta_1 - \theta_2}\left(e^{-\theta_2 t} - e^{-\theta_1 t}\right) + \varepsilon \quad \text{(b)}$$
>
> In these examples the parameters to be estimated are denoted by θ's
>
> . . .
>
> t is the single independent variable, and ε is a random error term with $E(\varepsilon) = 0$, $V(\varepsilon) = \sigma^2$
>
> . . .
>
> The models (a) and (b) are both nonlinear in the sense that they involve θ_1 and θ_2 in a nonlinear way but they are of essentially different character. The model (a) can be transformed, by taking

Nonlinear functions are introduced in the form of function templates. The nonlinear function templates are expressed in the following form

$$y \propto g_i(\mathbf{X}) \tag{3.4}$$

where y is the target or dependent variable, and the g_i are nonlinear function templates. The sign "$\propto$" indicates that the targeted variable y should be *qualitatively* proportional to the nonlinear function of $g_i(\mathbf{X})$ if the template is truly in the final induced model. The followings are some template samples:

$$\begin{aligned} y &\propto x_1^2 \\ y &\propto \sin(x_1) \\ y &\propto x_1 \times x_2 \\ y &\propto e^{x_1} \end{aligned} \tag{3.5}$$

The templates can be any nonlinear functions as long as they are based on and supported by the already existing domain knowledge. However, the system decides, according to the data, whether or not a template appears in the induced model. Therefore, when nonlinearity is considered, the region-equation pairs are expressed as a parameterized model and Eq. (3.2) becomes

$$R_i : y = c_{0i} + c_{1i} g_1(\mathbf{X}) + c_{2i} g_2(\mathbf{X}) + \ldots + c_{mi} g_m(\mathbf{X}) + Z \tag{3.6}$$

where $g_1(\mathbf{X})$, $g_2(\mathbf{X})$, …, $g_m(\mathbf{X})$ are user defined nonlinear function templates that are understandable to the domain engineers and consistent with the existing domain knowledge. To be more general, $g_i(\mathbf{X})$ can be any function template, including linear and nonlinear function templates.

3.1.3 Equation Discovery as Search

To find equations expressed as region-equation pairs, both region description and region equation must be learned. The central problem for this learning is to

somewhere between the top and the bottom of the hierarchy, which is applicable to the kind of learning tasks with the characteristics discussed in Chapter 1.

The nature of the heuristic search process is determined by four basic issues [Langley, 1994; Blum and Langley, 1997].

The first issue is the starting point in the space, which in turn affects the direction of the search. Two extreme approaches are *forward selection* and *backward elimination*. Forward selection starts with no feature and successively adds significant variables, whereas backward elimination starts with all variables and successively removes insignificant variables. Some approaches may be somewhere between the two and start with some features at the beginning. For the intended task, the starting points of searching the variable spaces are different for the variable identifications used in region descriptions and region equations. Forward selection is used to identify the significant variables in region descriptions, and backward elimination is applied to identify the significant variables in region equations.

The second issue is the organization of the search. As seen at the beginning of this section, exhaustive search is not practical for most real world engineering problems. A greedy search method is adopted for the intended learning task, which is a method that considers all possible ways within the allowed constraints at each step, evaluates them in terms of an evaluation function in the step, and then selects the best one for the step.

The third issue concerns the strategy used to evaluate alternative variables and select the best variable. Some induction algorithms incorporate a criterion based on information theory as in C4.5 [Quilan, 1994] for their specific targeted learning task. Others measure the accuracy on the training data set or a separate test data set. A broader issue concerns how the feature selection strategy interacts with the basic induction algorithm. EDDE selects error reduction as the strategy to evaluate the alternative variables as discussed in Section 3.2.

function that fits the training data set, not the one that does the best job of predicting a set of unseen data.

KEDS attempts to discover comprehensible models in an engineering domain. These models are multidimensional and nonhomogeneous. KEDS evaluates generated models along two dimensions: human comprehensibility and accuracy in predicting unknown values.

Like KEDS, EDDE is a system designed to be applied to actual engineering domains. Therefore its evaluation should also be based on both human comprehensibility and predictive strength on the cases that were not accessed to build the model.

Comprehensibility is inversely related to the complexity of a model. The complexity of the model expressed as region-equation pairs depends on the complexity of region descriptions as well as the complexity of associated equations. In general, for a model to have low complexity, it should have as few regions as possible. To further reduce complexity, the equations associated with regions should have as few variables as possible. Therefore, model complexity is measured using two dimensions: *model size* and *leaf size*. Model size (*ms*) is defined and expressed as

$$\begin{aligned} ms &= \text{the number of leaves in a model tree} \\ &= \text{the number of regions in a final model} \end{aligned} \tag{3.7}$$

and leaf size (*ls*) is defined and expressed as

$$ls = \frac{N_V}{N_R} \tag{3.8}$$

where N_V is the total number of variables in all regions and N_R is the total number of regions.

Model size is the primary factor and leaf size is the secondary factor that determines the model complexity. Leaf size should be considered only after the

with higher comprehensibility will be preferable. The parameter ε defines the percentage of predictive error that can be sacrificed to get higher comprehensibility of region equation pairs. In essence, ε is a parameter that measures the trade-off between model complexity and model prediction error.

There is a system design choice as to whether the region-equation pairs are reported to users in all circumstances. BACON and the previously discussed systems like it will not report any equation if the preset criterion is not satisfied. However, EDDE always reports the region-equation pairs in each batch of the learning process. It is believed that, based on the feedback information given by EDDE while reporting the region-equation pairs, users have several choices, including using the reported region-equation pairs as equations adequate to describe the data for the intended purpose, modifying the introduced nonlinear function templates, and adjusting model formation parameters (discussed in Section 3.3.3) to start the learning process again. To obtain useful and satisfactory model discovery, domain engineers must cooperate by not only supplying the required data and available domain knowledge but also paying attention to the findings made by EDDE. Any useful discoveries are likely to occur only through interaction between domain engineers and the system. Chapter 5 illustrates the importance of the interaction between domain engineers and the system.

3.2 EDDE SYSTEM

Based on the consideration of the technical aspects discussed in Section 3.1, EDDE is built up to accomplish the intended learning task: equation discovery in databases form engineering. The algorithm used to find the region-equation pairs capable of describing the data is discussed in this section.

with T_i examples. (For comparison, CART chooses a test to give the greatest expected reduction in either variance or absolute deviation). The algorithm uses a greedy search through the non-ancestor prediction variables one by one to choose the prediction variable that maximizes the expected error reduction. This process is repeated on the subsets until every subset either contains few cases, or no other prediction variable is left for further splitting, or the error measure is less than a user defined threshold Δ. Δ is one of the control parameters for model induction discussed in Section 3.3.3.

Multivariate models (in the form of general linear models) are constructed for the cases at each node of the model tree, using standard regression analysis [Press, Teukolsky, Vetterling and Flannery, 1988]. However, instead of using all templates (linear and nonlinear) in the standard regression analysis of each node, the templates used in a node are restricted to the templates inherited from its parent node.

After each multivariate model is obtained as above, it is simplified by eliminating templates to minimize its weighted standard deviation after the node variable is identified. Weighted standard deviation (*wsd*) of a node is defined as

$$wsd = \sum \frac{T_i}{T} \, sd(T_i) \tag{3.12}$$

The templates are removed one by one through greedy search until no more templates reduce the weighted standard deviation. In the multivariate model the remaining templates at a node will be given to its child nodes.

The multivariate model is further simplified by introducing another model formation parameter δ. For the same data set with noise, the standard deviation will generally decrease with the introduction of more prediction variables. However, the variables that are independent of each other and have no influence on the dependent variable will not result in much decrease in the standard deviation if the variables are added to the equation. These variables should be removed. But, if only the values of standard deviation are taken into account, they will not be removed, even though

The prediction error will decrease or remain essentially the same as a tree is pruned upward, and then will increase rapidly as the tree is overpruned.

For each examined node, the algorithm chooses as the final model for this node either the simplified multivariate model in the node or the model subtree, depending on the prediction error comparison between the simplified multivariate model and the model subtree. If the multivariate model is chosen, the subtree at this node is pruned and the node becomes a leaf.

In the pruning process, if the prediction error of a node is less than that of its subtree, the node is pruned to a leaf. If the prediction errors of a node and its subtree are comparable, the consideration of the trade-off between comprehensibility and predictive strength will determine whether or not the node should be pruned. The prediction errors of a node and its subtree are said to be comparable if

$$\left|\frac{e_n - e_s}{e_n}\right| < \varepsilon \tag{3.14}$$

where e_n and e_s are the prediction errors of the node and its subtree. ε is the same as in Eq. (3.10) and will be discussed Section 3.3.3.

3.2.3 EDDE Algorithm

Given a data set and templates based on the domain knowledge, EDDE will induce region equation pairs based on the following algorithm.

1. Divide the data into training and testing sets
2. Initialize the tree:
3. Grow a tree starting at the root

   while there is a generation {

   while there is a node at the current generation {

   if (sd>Δ or

depends on many factors, such as data set size, and computer speed. On relatively small data sets, such as those discussed in Chapter 5, each batch of learning (Repeat set to default value of 20) can be finished within a minute on Pentium II with a clock speed of 350. For larger size data sets, such as a data set with 50 variables and 800 examples, each batch of learning can be finished within 20 minute on Pentium II with a clock speed of 350.

EDDE learns a complete model from data in a batch scenario. At the beginning of the learning process, users may specify, in addition to the introduction of nonlinear function templates, the model formation parameters for EDDE. In the end, EDDE reports what is found and provides feedback information about the discovery to the users. The users then decide whether to terminate the learning, or to modify the model formation parameters or templates for continuing the learning process. This section discusses the implementation of the algorithm and shows how a model tree is built through an example.

3.3.1 Model Induction

The algorithm of EDDE is presented in Section 3.2.3 and its detailed implementation is shown through the following example. The data set used in this example has 200 data points. It is generated based on Model 1 with $\sigma^2 = 10$ by MATLAB [MATLB, 1992]. The details of Model 1 are given in Table 3.1 and the details of the generated data are given in Table 3.2. The complete data set is listed in Appendix A.

With the given data and the default model formation parameters $t = 30$, $\Delta = 0$, $\delta = 10$, and $\varepsilon = 10$, EDDE reports region-equation pairs that correspond to the induced tree shown in Fig. 3.2. The figure shows that the known function and the induced equations are the same except for small differences in the corresponding coefficients. Now we will show how EDDE induces the equations.

$$sd(\mathbf{T}) = sd(\mathbf{T}; x_6, x_7, x_8, x_9 \text{ and } x_{10})$$

$$= \sqrt{\frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)}{N-M-1}} = \sqrt{\frac{\sum_{i=1}^{140}(y_i - \hat{y}_i)}{140-5-1}} = 29.151$$

where N is the number of the examples in the node, M is the number of the variables (x_6, x_7, x_8, x_9, and x_{10}) in the node, $\hat{y}_i$ is the predicted value of the ith example by the equation that includes x_6, x_7, x_8, x_9, and x_{10}. The root node needs to grow because the standard deviation (29.151) is larger than the threshold Δ which is set at the default value of zero. The selection of a variable used in the root node is done by greedy search as follows. EDDE calculates the error reductions of all possible splits by description variables and selects the split that gives maximum error reduction. The error reductions are

$$\Delta\text{error by selecting } x_1 = sd(\mathbf{T}) - \frac{T_1}{T} sd(\mathbf{T}_1) - \frac{T_2}{T} sd(\mathbf{T}_2)$$

$$= 29.151 - \frac{74}{151} \times 18.694 - \frac{77}{151} \times 3.226$$

$$= 18.344$$

$$\Delta\text{error by selecting } x_2 = 3.569$$

$$\Delta\text{error by selecting } x_3 = -0.582$$

$$\Delta\text{error by selecting } x_4 = 0.863$$

$$\Delta\text{error by selecting } x_5 = -0.325$$

where $\mathbf{T}_1$ is one subset of $\mathbf{T}$ with x_1=Y, $\mathbf{T}_2$ is another subset of $\mathbf{T}$ with x_1=N, and $\mathbf{T}_1+\mathbf{T}_2=\mathbf{T}$. *Sd*'s are taken with regard to x_6, x_7, x_8, x_9, and x_{10}. The variable x_1 gives the maximum error reduction, thus the variable x_1 is used in the root node. The data will be split according to the values of the variable x_1 into two branches <x_1=Y> and <x_1=N> (See Fig. 3.3 branches from root node.)

elimination of x_9 is smaller, thus x_9 is eliminated. This process repeats until no other variables should be removed. For this node, no other variables should be further eliminated from the prediction variables. Therefore, only the prediction variables of x_6, x_7, x_8 and x_{10} will be inherited and used in its child nodes.

Then, the tree will grow to the second generation. For the branch of $<x_1$=Y>, the first node of the second generation, its standard deviation is 18.644, larger than the threshold Δ, and this node needs to grow. EDDE again calculates the error reduction of all remaining description variables (x_2, x_3, x_4 and x_5) and finds the split based on the variable x_2 will give the maximum error reduction. Thus, the variable x_2 is selected to be used in the node. Then, EDDE calculates the weighted standard deviation and finds x_{10} should be eliminated. Therefore, only x_6, x_7 and x_8 will be inherited by its child nodes down below.

For the branch of $<x_1$=N>, the second node of the second generation, its standard deviation is 3.210, larger than the threshold Δ, and this node also needs to grow. EDDE again calculates the error reduction of all remaining description variables (x_2, x_3, x_4 and x_5) and finds the split based on the variable x_4 will give the maximum error reduction. Thus, the variable x_4 is selected to be used in the node. Then EDDE calculates the weighted standard deviation and finds x_6, x_7 and x_{10} should be eliminated. Therefore, only x_8 will be inherited by its child nodes down below.

After EDDE checks all nodes of the second generation, EDDE checks the third generation and so on until the tree stops growing. The tree grows like that in Fig. 3.3 and stops growing because there are not enough examples in the nodes for further splitting thus the nodes become leaves. At this point, the process of growing the tree has been completed.

Before starting its pruning process, EDDE will further simplify the multivariate model in all nodes if the prediction variables do not influence the standard deviation in the nodes significantly (within the threshold δ). For example, at the node $<x_1$=Y>, the first node of the second generation, the standard deviation with

leaf, which contains the following equation

$$y = 2.213+8.546\, x_8$$

Consider another non-leaf node <x_1=N>. The prediction error given by the non-leaf node and its subtree on the testing data are 2.228 and 2.113 respectively. Although the prediction error given by the non-leaf node is larger than that by its subtree, the improvement by keeping the subtree is small and is within the range of ε ($\left|\frac{2.113-2.228}{2.113}\right| = 5.44\% < 10\% = \varepsilon$). The subtree of the node <x_1=N> should be pruned. This pruning process goes on at each non-leaf node of the model tree and the final tree after the pruning is shown in Fig. 3.2. The equations corresponding to the final tree and the information about the induced model are

if x_1 = Y and x_2 = T, $y = 4.489 + 7.949x_6 + 9.889x_7$ <sd = 2.721, COD = 0.991>

if x_1 = Y and x_2 = F, $y = 7.746 + 3.927x_6 + 6.771x_7$ <sd = 2.881, COD = 0.987>

if x_1 = N, $y = 4.492 + 7.979x_8$ <sd = 3.195, COD = 0.983>

Model complexity (ms)	= 3
(ls)	= 1.667
Prediction error	= 2.547
Prediction percentage error	= 28.955%
Prediction accuracy	= 91.837%
1-RE	= 0.993
Ratio mean	= 0.992
Variance of ratio mean	= 1.218

Based on the given information, the user can make the best decision on the induced model. In the following, a description of the feedback information provided by EDDE is given.

Coefficient of determination (COD) [Neter and el. at, 1996] describes the degree of fit between the dependent variable y and the independent variables in the equation. Coefficient of determination indicates the proportionate reduction in the total variation of y associated with the use of the independent variables in each region. Thus, the larger the COD, the more the total variation of y is reduced by introducing the independent variables in the region. The range of COD is [0, 1]. It should be noted that COD is calculated based on training data and is a resubstitution estimate.

The square root of COD

$$COR = \pm\sqrt{COD} \tag{3.15}$$

is called the coefficient of correlation (COR) and is another often used measure. However, the coefficient of correlation is not directly given because it does not have a clear-cut operational interpretation as coefficient of determination does.

3.3.2.2 Model Complexity and Prediction Error

Model complexity and prediction error are the two measures of an induced model as a whole, which are the direct goals the system tries to reach.

Model complexity has two dimensions: model size (*ms*) defined as the number of the regions of the induced model (Eq. (3.7)) and leaf size (*ls*) defined as the average number of variables a region (Eq. (3.8)). Model size should be considered as the primary dimension and leaf size should be considered as the secondary dimension. The complexity and the comprehensibility of the model are inversely related. Therefore, the smaller the model size and leaf size, the less complex and the more comprehensible the equations.

Prediction error is defined as the average error given by an induced model on testing data. It is used to measure the prediction strength of a model. The prediction error and the prediction strength are inversely related. Therefore, the smaller the

where $\hat{y}_i$ is the model predicted value of y and $\bar{y}$ is the mean of y in testing data. It is clear that s_1 represents the mean squared error about the induced model and s_2 is the mean squared error about the mean of y. The relative error (RE) is defined as

$$\text{RE} = \frac{s_1}{s_2} \tag{3.18}$$

Obviously, (1-RE) gives an estimate of percentage of variation of y explained by the introduction of the induced model. The larger the (1-RE), the better the induced model.

3.3.2.5 Ratio Mean and Its Coefficient of Variation

Ratio mean r is defined as

$$r = \frac{1}{\tilde{T}}\sum_{i=1}^{\tilde{T}} r_i = \frac{1}{\tilde{T}}\sum_{i=1}^{\tilde{T}}\left|\frac{\hat{y}_i}{y_i}\right| \tag{3.19}$$

where $\hat{y}_i$ is the predicted value of the ith example by the equation. The coefficient of variation is defined as r's standard deviation as a percentage of its mean [Hogg and Ledolter, 1987]

$$COV = \frac{\sqrt{\frac{1}{\tilde{T}-1}\sum_{i=1}^{\tilde{T}}(r_i - r)^2}}{r} \tag{3.20}$$

where $\tilde{T}$ is the number of testing data. Ratio mean and coefficient of variation are also used by some engineers to evaluate a model. Obviously, the closer to 1 the r and the smaller COV, the better the prediction.

3.3.3 Model Formation Parameters

There are four model formation parameters that control the learning process. They are test percentage t, standard deviation threshold Δ, leaf size trade-off δ and

on unseen cases with unnecessary complexity of the tree structure. The influence of the threshold Δ on the final induced models can be shown using the example below.

Using the same data set in Section 3.3.1, different thresholds (5, 10, 20 and 30) are set while other control parameters remain the same. The induced models under different threshold Δ are different. Their model complexity and their prediction error are shown in Fig. 3.5 and Fig. 3.6.

It is clear that if the threshold is set lower than the actual standard deviation, the overgrown tree can be pruned to the right size tree so that the overfitting can be avoided and the final induced tree is corresponding to the known model used to generate the data. Low standard deviation threshold takes the advantage of the pruning process so that the model can be discovered. But high standard deviation threshold stops tree growth prematurely, and pruning cannot correct this.

To discover real regularities in data, the solution to the problem is to let the tree grow larger than the right size tree and prune it back. However, in most real-world engineering data, the standard deviation of noise may not known. In these cases, it is hard for users to make the decision how low the threshold should be set so that a tree will not stop growing too early. Therefore, the default value of the standard deviation threshold Δ is set to zero. By doing so, the tree will not stop growing too early as long as enough data is provided. However users have easy access to change this threshold.

3.3.3.3 Trade-off Parameters: δ and ε

There are two trade-off parameters. One is the leaf size trade-off parameter δ. The other is the model size trade-off parameter ε. Compared with the formation parameters discussed above, the trade-off parameters have a more critical influence on the final induced tree and its corresponding model region-equation pairs.

The leaf size trade-off parameter δ is the maximum percentage of

However, there are some differences between KEDS and EDDE in the use of function templates.

Although the functional relationship is expressed as region-equation pairs both in KEDS and EDDE, a region equation is one of the templates in KEDS but it may be any possible combination of the introduced templates in EDDE.

- The templates in KEDS are expressed as families of parameterized models (polynomial equation templates), such as those in Eq. (2.5), but the templates in EDDE can be any user defined nonlinear functions based on the available domain knowledge. Some sample templates are shown in Eq. (3.5).
- Region-equation pairs in EDDE are expressed using the tree based model with disjoint regions. In KEDS, the induced region-equation pairs may have overlapping regions with decisions as to which equation to use depending on the ordering of region-equation pairs. From the descriptive point of view, it is confusing that a region may be described by more than one equation. The EDDE approach is thus more descriptively clear.
- The organization of search methods is different. KEDS adopts exhaustive search while EDDE adopts greedy search. Therefore, KEDS has a very low learning speed even on small data sets.

3.5 USE OF DOMAIN KNOWLEDGE IN EDDE

Frequently in engineering domains, data analysis stresses both prediction and description equally. The function found from data requires not only satisfactory prediction on unseen or new cases, but also understandability to the domain engineers. Therefore, engineers want to introduce domain knowledge in guiding data analysis (Section 1.3). The domain knowledge of the problem type intended for solution by EDDE is discussed in Section 1.3.

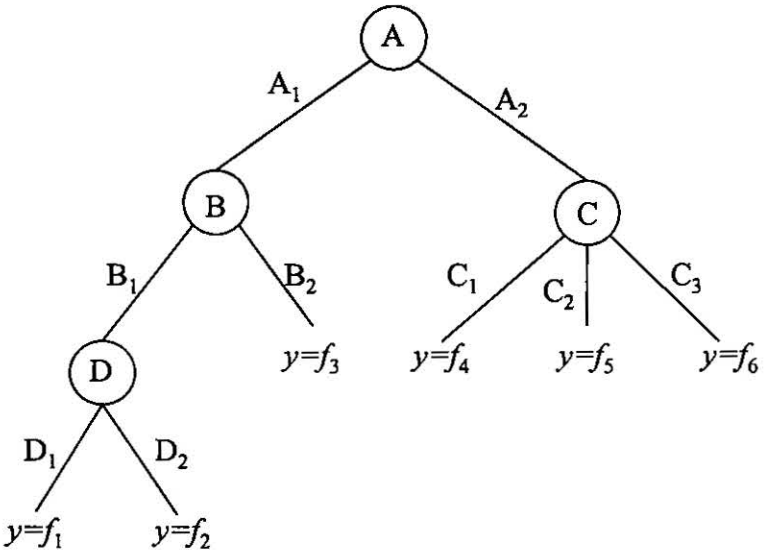


Fig. 3.1 Tree representation of region-equation pairs of Eq.(3.3).

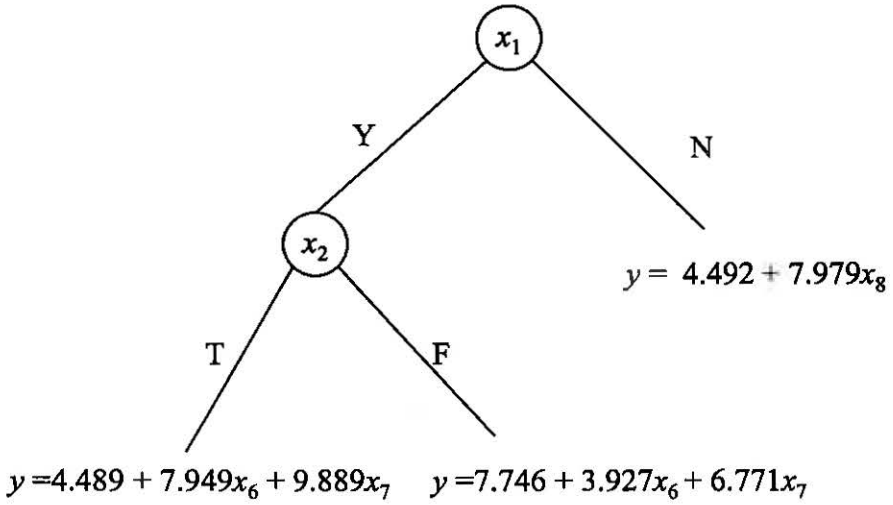


Fig. 3.2 Induced model tree for the example in Section 3.3.1.

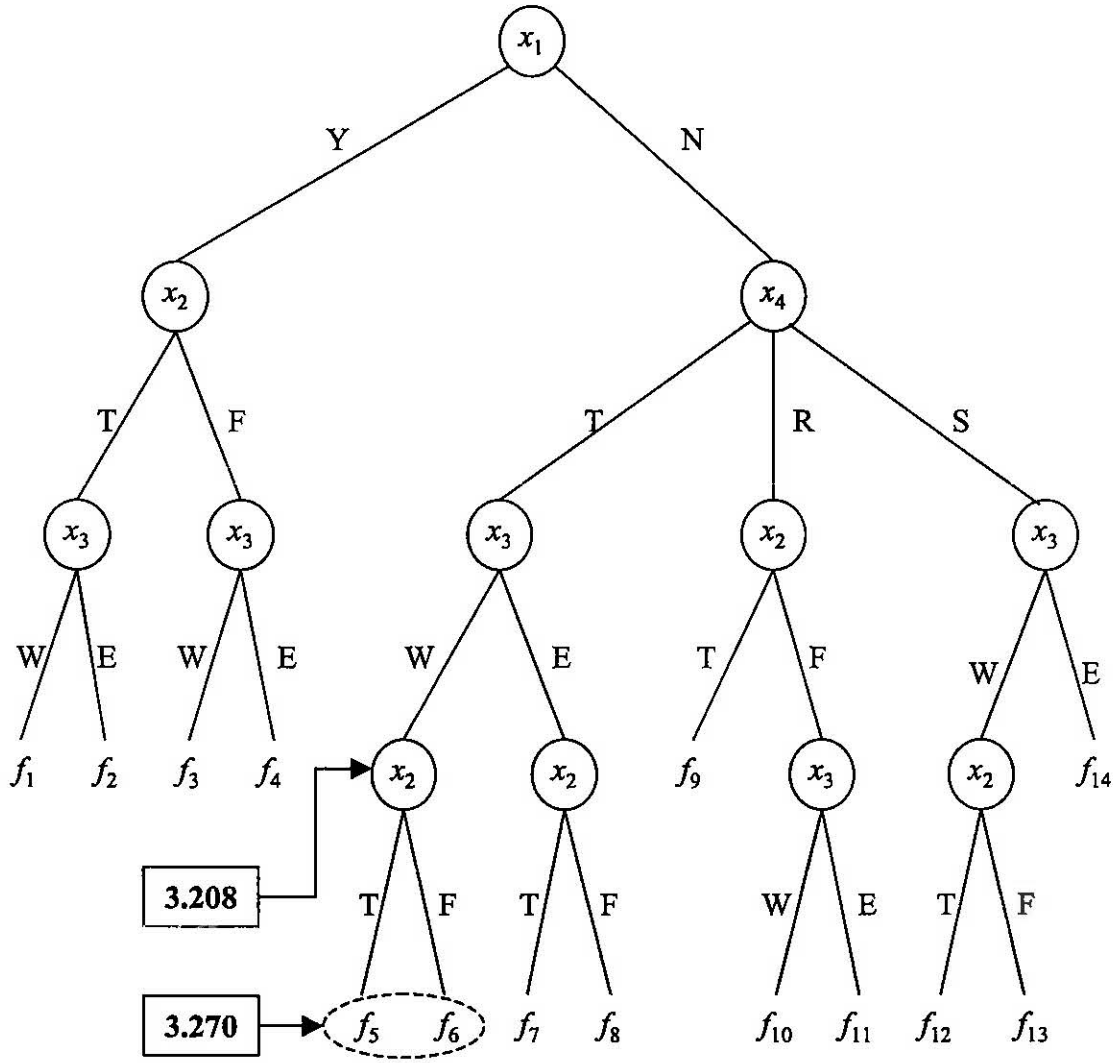


Fig. 3.4 Pruning process for the example in Section 3.3.1.

CHAPTER 4

EVALUATION OF EDDE

Machine learning is one of the most experimentally oriented subfields within artificial intelligence. After a system is built, there is a need to understand the behavior of the system through an experimental study. The experimental study of the system, in other words, testing of the method, should be carried out on both synthetic domains and real world domains [Langley, 1996b]. Experiments with artificial data have important roles in studying the behavior of a system. Such data sets let one systematically vary factors of interest, such as the number of variables, the number of examples, and the amount of noise. By systematically varying the synthetic domain characteristics of the data set, the effect of these factors on the system behavior can be measured and the various aspects of the system are hopefully understood. In this section, an extensive experimental study of EDDE's performance on artificial data is provided.

4.1 TREE INSTABILITY

An experiment is carried out on a series of data sets generated in exactly the same way, but each with a different random number seed. Induced trees and thus their corresponding region-equation pairs may be different under the same model formation parameters. Even for the same data set, when the test sets are selected randomly with different random number seeds, induced trees may also be different.

corresponding region-equation pairs are said to differ from the known model. The induction of false trees is referred to as tree instability.

The amount of tree instability is affected by the noise in the data. The possible causes of instability can be tracked down in the induction process. First, in selecting significant variables for region description, there may be a number of splits on different variables at a given node, some of which give almost the same error reduction. Since data are noisy, the choice between competing splits is almost random. Choosing an alternative split that is almost as good will led to a different evolution of the tree from that node downward. This kind of instability happens quite often in other tree-based systems such as CART [Breiman, Friedman, Olshen and Stone, 1984], but is much less severe in EDDE. Various tests have shown that this kind of instability happens after all significant description variables have been identified, which means different tree structures happen at the lower level of trees (see Fig. 4.1 (b) and (c)). This is attributed to the introduction of domain knowledge, which is used to divide variables into description variables and prediction variables before building a tree. EDDE makes use of domain knowledge to get more stable results whereas CART and the like do not. This shows the importance of introduction of domain knowledge.

Second, in eliminating insignificant variables from a region equation, a variable is eliminated if it does not give an improvement in standard deviation beyond the threshold δ. Because of the clear-cut threshold for elimination, sometimes the significant variable may be eliminated (see Fig. 4.1 (c) f_5) and at other times insignificant variables may not be removed from the equation (see Fig. 4.1 (d) f_1). This will also lead to a false tree.

Third, the instability comes from the pruning process after a tree has been constructed. During the pruning process, a tree is pruned based on the error measure on testing data. If the subtree of a node does not improve the prediction on testing data more than the threshold ε, the subtree will not be pruned. Because of noisy data

4.2.1 The Influence of the Number of Variables

As mentioned in Chapter 1, one of the characteristics of real world engineering problems is that engineers can not always determine the significance of variables *a priori*. In the following experiment, we investigate how insignificant (or irrelevant) variables affect tree stability ratio.

This experiment also uses Model 1 in Section 3.3 to generate data sets, but more insignificant variables are introduced in this experiment. The number of insignificant variables included in the data varies from 6 to 46 in increments of 8. Therefore, the total number of the variables including 4 significant variables in the model varies from 10 to 50 in increments of 8, that is 10, 18, 26, 34, 42, and 50. Of the insignificant variables, half of them are description variables and the other half are prediction variables. Of the insignificant description variables, half of them have two evenly distributed values like x_1 in Model 1; the other half have three evenly distributed values like x_4 in Model 1.

As mentioned, the influence of the number of variables might be interactive with sample size and noise variance. Therefore, tree stability ratio versus the number of variables is studied when both of the sample size and the noise variance are kept at certain levels. The sample size varies from 200 to 800 in increments of 200, that is: 200, 400, 600, and 800. The noise variance varies from 10 to 500, that is 10, 100, 200, 300, 400, and 500. The selection of these variances is explained in Section 4.2.3. The tree stability ratio is investigated under all the combinations of these sample sizes and noise variances.

The tree stability ratios versus the number of variables under various noise variances are shown in Fig. 4.2 to Fig. 4.7. The figures show that tree stability ratio does not change much as the number of variables increases. For example, when the sample size is 100 and noise variance is 10, the stability ratio remains approximately 95 when the number of variables ranges from 10 to 50. Tree stability ratio slightly decreases with respect to increase in number of variables only when noise variance is

600, 800, 1400, and 2000. Data sets of various sample sizes (200, 400, 600, 800, 1400, and 2000) at a fixed noise variance are generated by MATLAB based on the Model 1.

Section 4.2.1 also shows that the influence of sample size is related to noise variance. To more thoroughly study the influence of sample size, the dependence of stability ratio on sample size at various noise variances are also examined. The noise variances selected for study range from 10 to 500, that is 10, 100, 200, 300, 400, and 500. The selection of these variances is explained in Section 4.2.3.

The influence of sample size on stability ratio is shown in Fig. 4.12 when noise variance is also taken into account. The figures show that sample size has influence on tree stability ratio. Under the same level of noise variance, smaller sample sizes result in lower stability ratios. When sample size increases, induced trees become more stable. At the first stage, the stability ratio increases sharply, then its increase gradually slows down. With large sample size, induced trees eventually become stable. However, at what sample size the induced trees become stable depends on the noise variance. When variance is 100, the sample size of 800 makes the induced tree stable. But the sample size of 800 will not make the induced tree stable when noise variance is 500. The influence of noise is discussed in Section 4.2.3.

4.2.3 The Influence of Noise

The influence of introduced noise on stability is studied in this section. Since the number of variables does not influence stability ratio as known from Section 4.2.1, the number of variables is kept the same in the investigation. Model 1 in Table 3.1 is again used to generate data sets with the number of variables equal to 10.

In order to determine the influence of noise, stability ratios of data sets corrupted with various noise variances are studied. Noise variances are selected as 10, 100, 200, 300, 400, and 500. The reason to select these noise variances less than

unnecessary tree structure. Taking the experiment results in Section 4.1 as an example, the true tree (Fig. 4.1 (a)) has higher prediction strength than the false trees (Fig. 4.1 (b), (d) and (e)). Also, the true tree (Fig. 4.1 (a)) has the comparable prediction strength without unnecessary tree structure (Fig. 4.1 (c)). This observation promotes a method to stabilize the final induced tree and the final model.

To deal with the instability of model trees and find the true function, the system EDDE runs the data a user preset number of times (called Repeat in the system) by randomly selecting the test data with different random number seeds. EDDE will induce a model tree each time and choose the one with the consideration of the trade off between the prediction error on test data and model complexity. In fact, if the induced model captures the actual regularity in data, it is entirely possible that the learned model will be the same as the true model.

However, it is hard to find an optimal value of Repeat for building trees because many factors influence the stability of the final reported tree as seen in Section 4.2. Qualitatively speaking, the larger the Repeat, the lower the learning speed and the higher the probability of discovering actual regularity. The default value for repeat is 20. Users have easy access to adjust this value based on sample size and noise strength.

4.4 COMPARISON STUDY

The experiment in this section compares the results given by CART [Breiman, Friedman, Olshen and Stone, 1984] and other algorithms [Quinlan, 1993] and the results by EDDE in order to investigate the model tree's interpretation strength and prediction strength. CART is a regression tree method. The method used in Quinlan's study [1993] consist of one default method, one instance-based method, three model-based methods (regression, model tree, and neural nets), and combinations of the instance-based method with each model-based method. The experiment is carried out on the data set generated in the exactly the same way as that

reported equation is

$$\text{if } x_1 = 1,\ y = 3.170 + 3.044x_2 + 1.800x_3 + 0.891x_4$$

$$\text{<Sd = 1.530, COD = 0.797>}$$

$$\text{if } x_1 = -1,\ y = -2.972 + 2.910x_5 + 1.675x_6 + 1.201x_7$$

$$\text{<Sd = 1.483, COD = 0.814>}$$

with

Model size (*ms*)	= 2
Leaf size (*ls*)	= 3
Prediction error (*e*)	= 1.02
RE	= 0.11

However, on the simulated data by the same known model, CART's regression tree is much larger with 13 leaves. The prediction in each leaf is the dependent variable mean of the cases in that leaf. The induced tree is shown in Fig. 4.14 (b).

Table 4.2. Error measure comparison on the evaluation data.

	EDDE	CART
The number of leaves (model size *ms*)	2	13
Leaf size (*ls*)	3	---
Prediction error (*e*)	1.14	---
RE	0.12	0.17
1-RE	0.88	0.83

CART provides only the estimate of relative error RE on the evaluation data (the additional 5000 cases), which is defined in Eq. (3.18). The comparison of the induced trees by EDDE and CART is listed in Table 4.2.

From the comparison in Table 4.2 and Figure 4.14, it is noticed that while EDDE's function divides the data space into two regions as does the known function,

7. neural nets: a straight forward neural network was constructed and trained using the default conjugate gradient method of the Xerion neural network simulator developed at the University of Toronto,

8. neural nets + instances: the IBL approach is combined with the neural net.

Quinlan studied the prediction strength of these methods on the simulated data using the same known model as Model 2[1]. The performance of these methods is measured by 10-fold cross-validation. To compare the results with EDDE, the 200 cases are divided into 10 subsets. For each run, only 9 subsets are provided to EDDE leaving the other subset as evaluation data. Therefore, there are 10 runs. The average error measures on the evaluation data of the 10 runs are listed in Table 4.3.

Table 4.3 shows that the prediction error (e) and the relative error (RE) are 1.18 and 12% by EDDE respectively. (The percentage error is not compared because the actual values of the dependent variable y include zero). They are comparable to or lower than all the algorithms presented in [Quinlan, 1993] except Quinlan's model tree method. Apparently, EDDE outperforms most of algorithms, with the improvement rate of up to 220% in prediction error and 180% in RE. This is not a surprise because those algorithms do not require any information on the domain and no domain knowledge is provided to the system. In contrast, EDDE was given the additional information that x_1, x_8, x_9, and x_{10} are used for region description and the other variables are used for region prediction. This demonstrates the important role of domain knowledge in knowledge discovery from data. With the introduction of domain knowledge, the system can discover equations with higher prediction strength. This is seen for EDDE's performance on actual data sets, as discussed in the

---

[1] Quinlan's 1993 study uses a total of eight data sets to compare the eight methods. EDDE is here run on the single artificial data set generated by Model 2. In Section 5.5, EDDE is run on the actual data set dealing with automobile fuel consumption. The other 6 actual data sets (housing, cpu, auto-price, servo, lhrh-att, and lhrh-def) are not run on EDDE because domain knowledge is required for EDDE and this knowledge is not included in the data sets.

up to 300. However, actual engineering data sets are often small so that tree instability occurs.

To deal with the instability of model trees and find the tree equations, a method, using trade-off parameters to select the final tree among alternate trees, is promoted to stabilize the final induced tree. Using the method even on a relatively small sample size (200 with 10 variables), the system can find the actual equations from the data, which is shown when the tree interpretation and prediction strength are studied and compared with other methods [Quinlan, 1993] in Section 4.4. The other methods includes regression tree CART, instance based IBL, multivariant linear regression, model tree M5, neural nets, and combinations of these methods.

On the same data set, EDDE generates the equations that divide the data space into two regions as does the known function while CART's equations divides the data space into 13 regions. The results show that EDDE's induced tree summarizes the data more concisely, and its description of the data is more straightforward comparing with CART.

Also on the same data set, the prediction strength is compared with other methods in [Quinlan, 1993]. The results show that the prediction error is 1.18 by EDDE while the prediction error ranges from 1.1 to 3.77 by other methods. Apparently, EDDE outperforms most of the algorithms, with the improvement rate of up to 220% in prediction error and of up to 180% in RE.

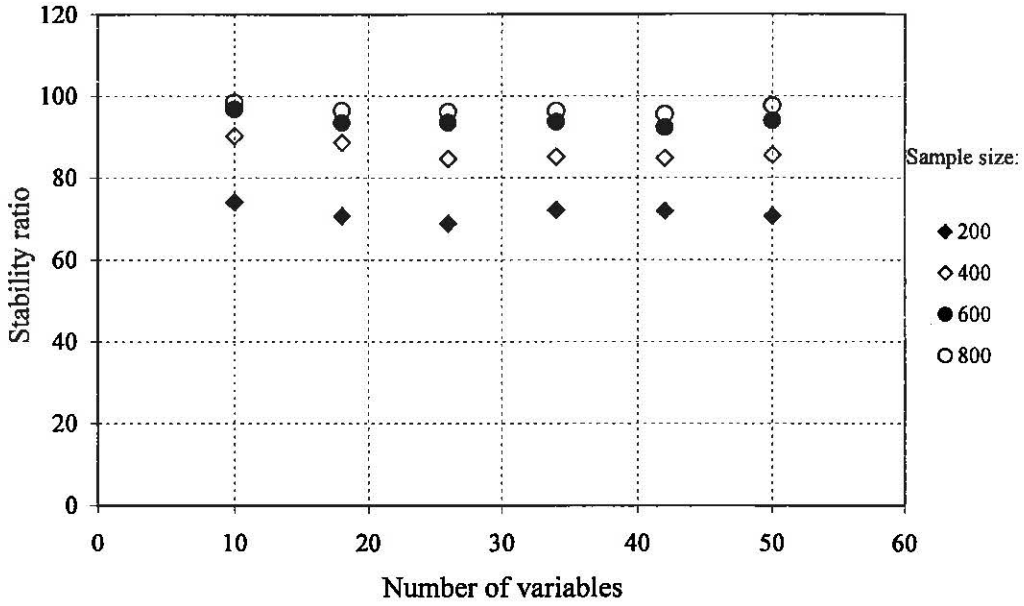


Fig. 4.2 Stability ratio when noise variance is 10.

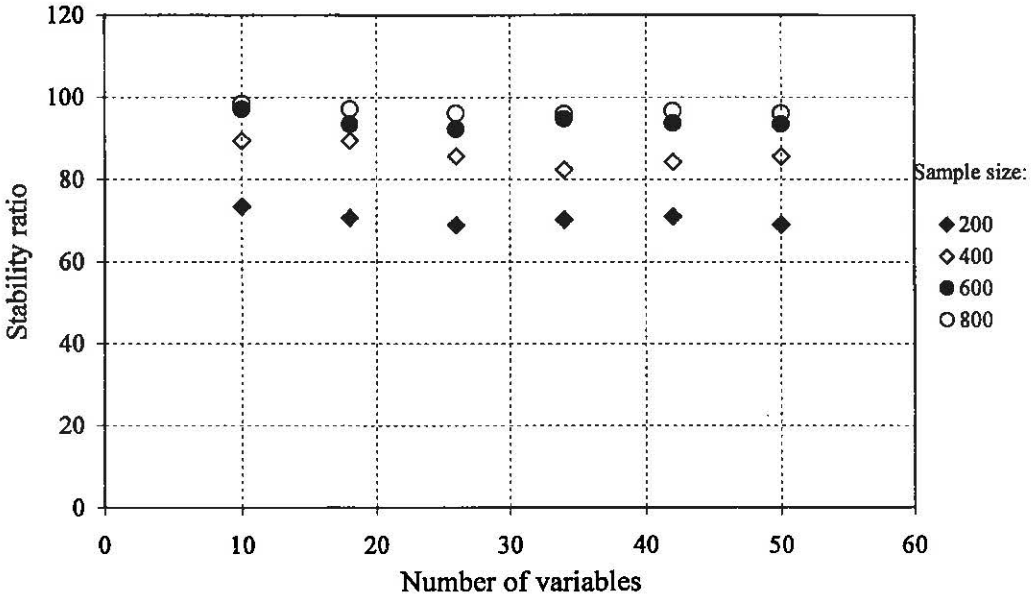


Fig. 4.3 Stability ratio when noise variance is 100.

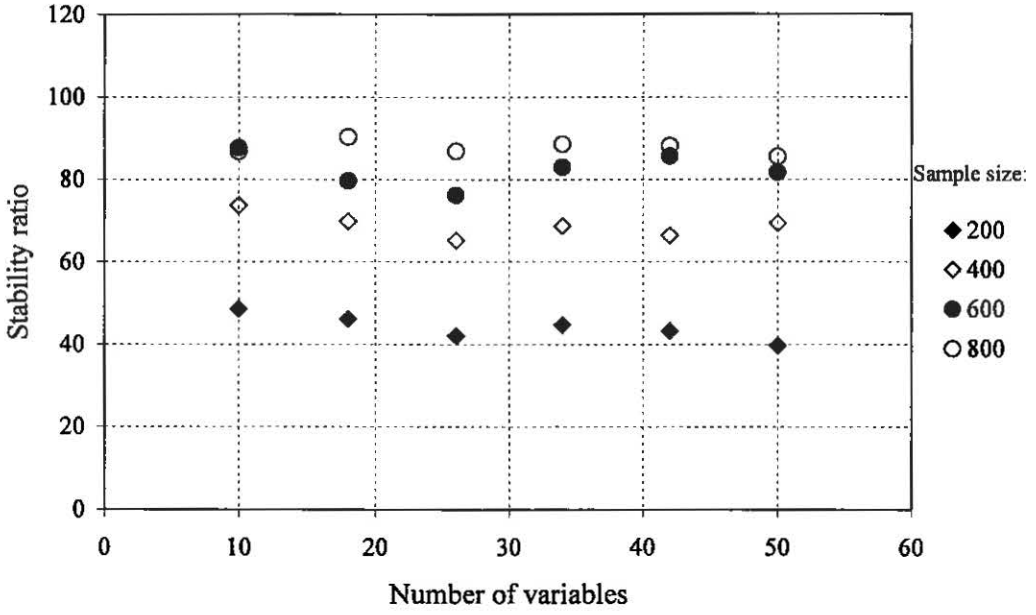


Fig. 4.6 Stability ratio when noise variance is 400.

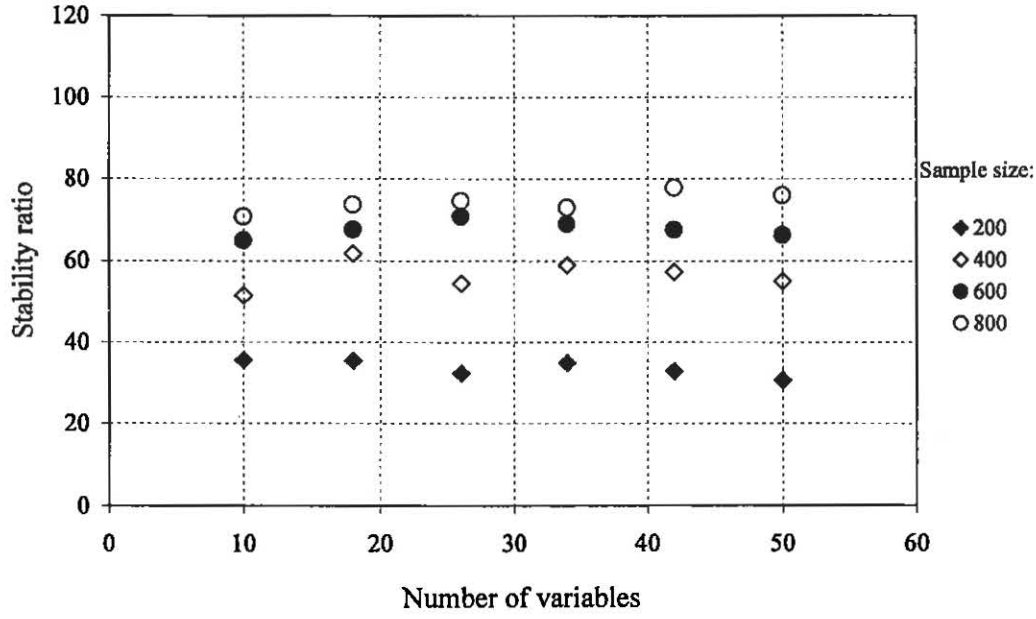


Fig. 4.7 Stability ratio when noise variance is 500.

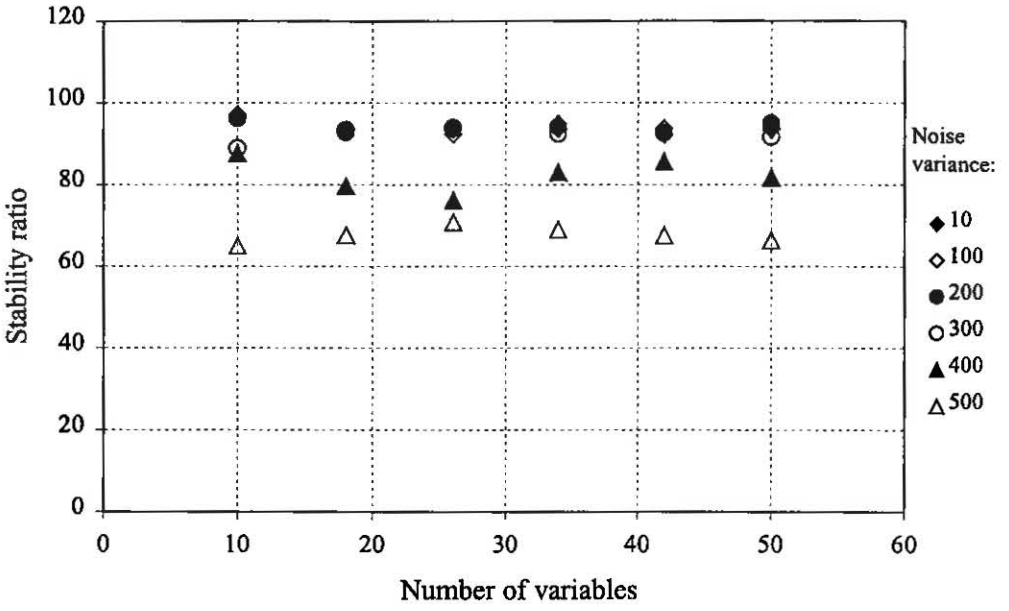


Fig. 4.10 Stability ratio when sample size is 600.

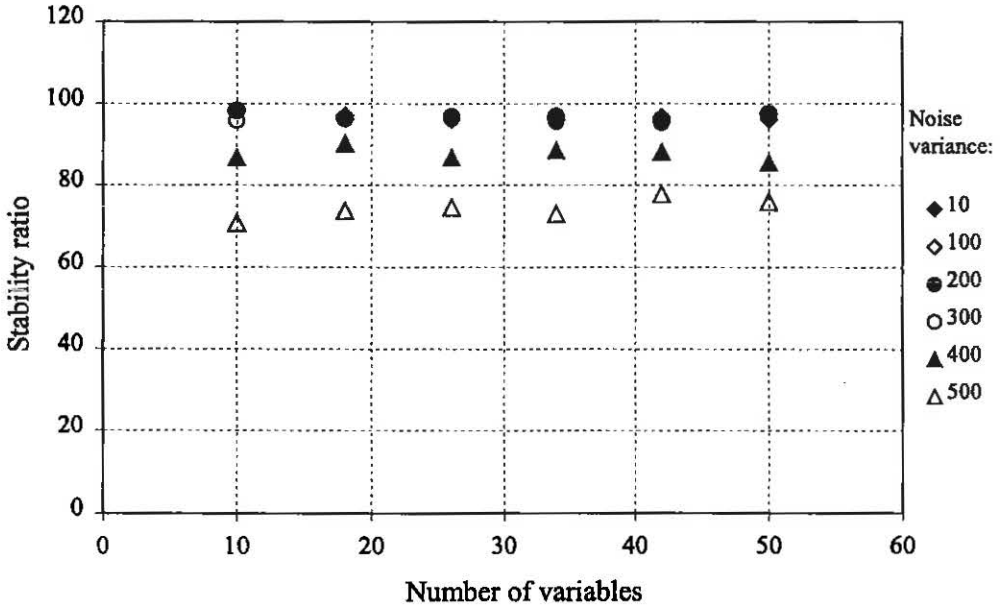
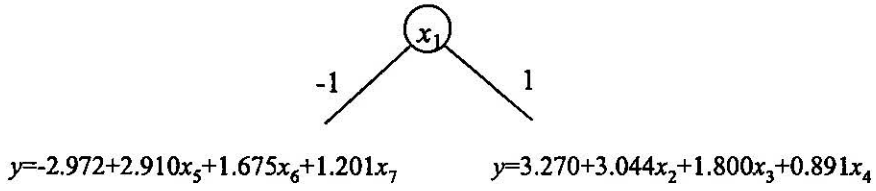
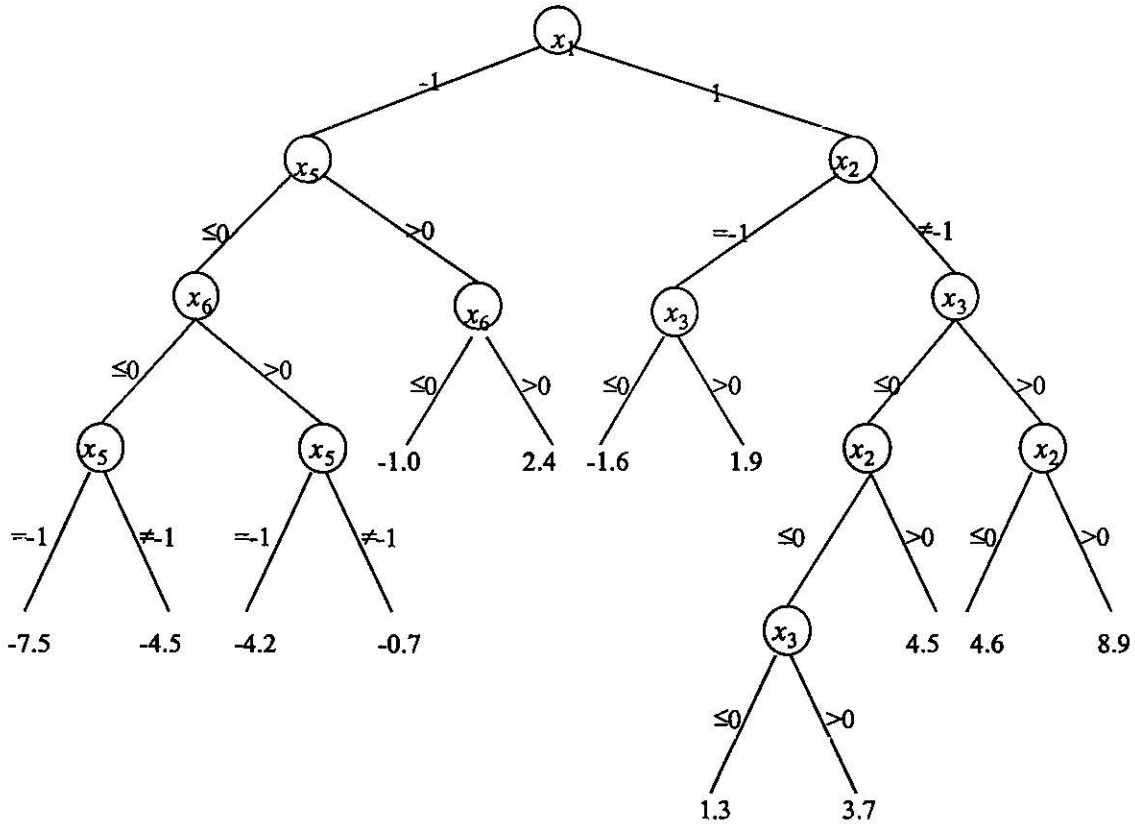


Fig. 4.11 Stability ratio when sample size is 800.



(a) EDDE's induced tree



(b) CART's induced tree

Figure 4.14 Induced trees by EDDE and CART for Model 4.2

project. The planner chooses a generic planning template that most closely matches the project type, establishing an activity network for the project. Next, the duration of each activity in the project is projected according to the predictive models stored in the management system. Finally, a complete plan and schedule is generated either by forward or backward pass calculations.

The effectiveness of planning and scheduling depends on the prediction accuracy of activity duration, which in turn depends on the prediction models. However, the current predictive models were established by experienced engineers based on rules-of-thumb, and their prediction does not accurately reflect the agency's current business practices and requirements. The early phases of EDDE's development were concerned with using the collected historical data to update the predictive models, which are expressed as equations. The updated models result in more effective planning so that costs overruns and schedule slippage can be avoided.

The earlier phases of this research include using the prototype version of EDDE to update the predictive models. Comparing the updated predictive models and the old predictive model, the percentage deviation is 50.7% using the updated predictive models, while the percentage deviation is 67.5% using the old predictive model. The updated predictive model does improve the percentage deviation of the duration prediction by 25.4%. The induced equations are of practical value, which displays the usefulness of the research presented in this thesis. Meanwhile, it was also learned that the quality of the induced equations highly depends on the quality of the provided data set. When data sets are corrupted with strong noise, the data sets have to be cleaned and preprocessed. The issues of data cleaning and preprocessing should be addressed in the data preparation step of the whole process of knowledge discovery in databases, which is beyond the scope of the study presented in this thesis. The earlier phases of research and results were presented at the 1997 Transportation Research Board annual meeting and published in [Zhang and Roddis, 1997]. More extensive description is provided in [Roddis and Zhang, 1997].

and Darwin [1998]. To investigate the design criteria, the study is based on the assumption that the total force in a bar at splice failure, T_b, equals the sum of a concrete contribution, T_c, and a transverse reinforcement (steel) contribution, T_s, which can be written as

$$T_b = T_c + T_s \tag{5.1}$$

In their research, they study both the concrete contribution T_c and the transverse reinforcement (steel) contribution T_s. However, only the concrete contribution T_c is studied by EDDE in this section. Therefore, only their results concerning the concrete contribution T_c are provided below.

Using a database including 171 specimens containing developed or spliced bars not confined by transverse reinforcement, the contribution of T_c is studied. (The data set is listed in Appendix B.) T_c is the product of the bar area, A_b, and the bar stress at failure, f_s, in psi. They study the relationship between $\frac{T_c}{f_c'^p}\left(=\frac{A_b f_s}{f_c'^p}\right)$ and specimen properties of the beam, where p is a constant, and f_c' is concrete compressive strength in psi.

A series of dummy variable analyses are carried out based on bar size and concrete strength when different power values of p and effective c_{si} are considered. The findings of their research include but are not limited to the following:

- When p = 1/4 and effective c_{si} = actual c_{si} + 0.25, the introduction of the term $\left(0.1\frac{c_{max}}{c_{min}}+0.9\right)$ is necessary to describe $\frac{T_c}{f_c'^p}$ and the equation is expressed as

$$\frac{T_c}{f_c'^{1/4}} = \left[59.81 l_d \left(c_{min} + 0.5 d_b\right) + 2350 A_b\right]\left(0.1\frac{c_{max}}{c_{min}}+0.9\right) \tag{5.2}$$

where

c_{min}, c_{max} = minimum and maximum values of c_s and c_b

process. However, to accommodate users' different needs, EDDE provides other information about the discovered equations. The additional information, which is described in Section 3.3.3, includes the ratio mean and COV. But, ratio mean and COV in Zuo's research and in EDDE are calculated differently. Ratio mean and COV in Zuo's research are the resubstitution estimations of all input data, while those in EDDE are the sample estimations of only testing data. Obviously, Zuo's error measures may overestimate the equation performance in comparison to EDDE error measures because Zuo's estimations of ratio mean and COV are computed using the same data used to establish the equation.

5.2.2.2 Data Preparation

To test EDDE on this actual engineering data set including 171 examples, the data set is prepared before the actual learning process starts. A part of data, suggested by a domain engineer and containing 28 examples (from Zuo's experiments [Zuo, 1998]), is held as evaluation data, which will not be given to EDDE for learning. The other part containing 143 examples is given to EDDE to discover equations. After equations are discovered, the evaluation data is used to evaluate the equations.

On the 143 data points, the following is done.

delete 2 data points with $d_b = 0.375$in.
delete 5 data points with $d_b = 0.625$in.
delete 1 data points with $d_b = 1.128$in.
delete 2 data points with $d_b = 1.177$in.
delete 2 data points with $d_b = 1.960$in.
change 2 data points with $d_b = 0.992$in. to $d_b = 1.000$in.

The reason for the changes is that EDDE requires the number of each data subset to be larger than the number of the variables when EDDE tests whether to use a variable to divide the domain space. If those data are kept, the test for variable <bar size> will fail and the influence of <bar size> as a possible variable for splitting the data into subsets will not be checked.

area of the reinforcing bar A_b.

Third, experiments also show that c_{max} and c_{min} affect T_c. T_c becomes larger as the ratio c_{max} / c_{min} increases. This was found by plotting the version of equation 5.2 that does not account for c_{max}, $\frac{T_c}{f_c'^{1/4}} = 59.81 l_d(c_{min}+0.5d_b)+2350A_b$, for various values of c_{max} and observing the upward trend in T_c with increasing c_{max}. To account for this behavior of the data, the term $\left(0.1\frac{c_{max}}{c_{min}}+0.9\right)$ was introduced in Eq. 5.2.

Based on the above considerations, the following basic templates are selected for submittal to EDDE.

a. $l_d(c_{min}+0.5d_b)$

b. A_b

c. $l_d(c_{min}+0.5d_b)\left(0.1\frac{c_{max}}{c_{min}}+0.9\right)$

d. $A_d\left(0.1\frac{c_{max}}{c_{min}}+0.9\right)$

- The descriptive variables are

  a. Bar size, represented by bar diameters

  b. Concrete strength, discretized based on the study by Zuo and Darwin [1998] as

     2500 psi $< f_c' \leq$ 3500 psi

     3500 psi $< f_c' \leq$ 4500 psi

     4500 psi $< f_c' \leq$ 5500 psi

     5500 psi $< f_c' \leq$ 6500 psi

     6500 psi $< f_c' \leq$ 14500 psi

that $\frac{T_c}{f_c'^{1/2}}$ is described by a different equation for each range of f_c'. For example, when 60% of data is used as testing data, the induced equations are

if $f_c' = 2500 \sim 3500$

$$\frac{T_c}{f_c'^{1/2}} = \left(8.545 l_d (c_{\min} + 0.5 d_b) + 287.171 \mathrm{A_d}\right)\left(0.1\frac{c_{\max}}{c_{\min}} + 0.9\right)$$

if $f_c' = 3500 \sim 4500$

$$\frac{T_c}{f_c'^{1/2}} = \left(5.118 l_d (c_{\min} + 0.5 d_b) + 467.105 \mathrm{A_d}\right)\left(0.1\frac{c_{\max}}{c_{\min}} + 0.9\right)$$

if $f_c' = 4500 \sim 5500$

$$\frac{T_c}{f_c'^{1/2}} = \left(8.280 l_d (c_{\min} + 0.5 d_b) + 190.730 \mathrm{A_d}\right)\left(0.1\frac{c_{\max}}{c_{\min}} + 0.9\right)$$

if $f_c' = 5500 \sim 6500$

$$\frac{T_c}{f_c'^{1/2}} = \left(6.871 l_d (c_{\min} + 0.5 d_b) + 266.329 \mathrm{A_d}\right)\left(0.1\frac{c_{\max}}{c_{\min}} + 0.9\right)$$

if $f_c' = 6500 \sim 14500$

$$\frac{T_c}{f_c'^{1/2}} = \left(3.205 l_d (c_{\min} + 0.5 d_b) + 322.297 \mathrm{A_d}\right)\left(0.1\frac{c_{\max}}{c_{\min}} + 0.9\right)$$

if $f_c' = 14500 \sim 16500$

$$\frac{T_c}{f_c'^{1/2}} = \left(572.266 \mathrm{A_d}\right)\left(0.1\frac{c_{\max}}{c_{\min}} + 0.9\right)$$

Zuo [Zuo and Darwin, 1998] concludes that p=1/2 can not represent the effect of concrete strength on the development/splice strength of bars with a single equation. That is to say, the equations describing $\frac{T_c}{f_c'^{1/2}}$ in different ranges of concrete strength should be different. The equations induced by EDDE show this same behavior of different equations for different f_c' regions. But, the equations in the regions can not

because the analysis of this data set is sought to get the lowest COV with a ratio mean closest to 1, 60% should be used in the following analysis.

Table 5.1. Discovered equation when the dependent variable is $\frac{T_c}{f_c'^{1/4}}$

percentage of testing data	K_1	K_2	R^2	e	Ratio Mean	COV
10	51.746	2793.081	0.9718	181.663	0.9872	0.0903
20	52.557	2715.439	0.9730	346.435	1.0282	0.0934
30	50.242	2929.775	0.9728	354.173	0.9679	0.1209
40	50.367	2919.484	0.9705	307.719	0.9845	0.1189
50	54.336	2615.943	0.9717	380.349	1.0261	0.0934
60	52.226	2799.449	0.9683	360.582	0.9856	0.0893
70	50.985	2854.187	0.9694	343.735	0.9929	0.0949
80	50.171	2815.255	0.9723	400.752	1.0123	0.1064
90	51.493	2789.824	0.9648	377.982	0.9938	0.1025

5.2.3.3 Power p in Dependent Variable $\frac{T_c}{f_c'^{p}}$

The Section 5.2.3.1 and 5.2.3.2 discuss the results when p=1/2 and p = ¼ in $\frac{T_c}{f_c'^{p}}$. In this section, the influence of p is more extensively examined.

The power p is varied from 0.22 to 0.50. Because different dependent variables are taken into account, the prediction error e on the testing data can not be used as the criterion to decide which one is better than the others. In this test, the ratio mean and the COV of testing data are taken as the main criteria. The results are

5.4.(b).

To reflect the splitting failure modes for bond due to the effect of concrete cover and bar spacing, effective c_{si} is studied. In ACI 318-95, the effective value of c_{si} is equal to c_{si}. In the Canadian code (CSA Standard A23.3-94), a greater value (4/3 c_{si} + 1/6 d_b) is used as the effective value of c_{si} to give a better match between test development /splice strength and predicted strength than using the actual value of c_{si}. Zuo and Darwin found that using c_{si} + 0.25 (in.) as the effective value of c_{si} gives a better match between test development/splice strength and predicted strength than using the actual value of c_{si}.

To investigate effective c_{si} using EDDE, first, the templates including all effective values of c_{si} are put into the system. These templates are:

$$l_d\left(c_{\min}(k)+0.5d_b\right)\left(0.1\frac{c_{\max}(k)}{c_{\min}(k)}+0.9\right) \tag{5.4a}$$

$$A_d\left(0.1\frac{c_{\max}(k)}{c_{\min}(k)}+0.9\right) \tag{5.4b}$$

where k = 0.0 and 0.2~0.4 in increments of 0.02, $c_{\min}(k)$ and $c_{\max}(k)$ are $c_{\min}$ and $c_{\max}$ with effective c_{si} = actual c_{si} + k. The induced equation is

$$\frac{T_c}{f_c'^{1/4}} = 55.414 l_d\left(c_{\min}(0.36)+0.5d_b\right)\left(0.1\frac{c_{\max}(0.36)}{c_{\min}(0.36)}+0.9\right)$$

$$+ \quad 2619.910 A_d\left(0.1\frac{c_{\max}(0.40)}{c_{\min}(0.40)}+0.9\right)$$

Because the equation includes the templates with k value for defining effective c_{si}, the learning cycle starts again using revised template. Using the templates listed in Eq. (5.4), but, each time only the templates with the same definition of effective c_{si} are introduced, where k = 0, or 0.2 or ... or 0.4. The induced equations have the form of Eq. (5.3) with different effective c_{si} = c_{si} + k

between confined bars, which, in turn, overestimates the concrete contribution to bond strength. Another disadvantage of using the larger effective value of c_{si} means that the assumed splitting cracks change from a horizontal plane to a vertical plane for some specimens in which splitting was actually controlled by the clear spacing. Based on these considerations, effective $c_{si} = c_{si} + 0.25$ is used for both confined bars and bars not confined by transverse reinforcement in Zuo and Darwin's research. Because the test on EDDE is limited to the reinforcing bars without transverse confinement, Table 5.3 can only show that effective $c_{si} = c_{si} + 0.25$ is one of a family of reasonable choices. To maintain comparability to Zuo and Darwin's research, an effective $c_{si} = c_{si} + 0.25$ is used in the templates for generating EDDE's induced equation for final evaluation.

5.2.4 Final Evaluation

The final induced equation is

$$\frac{T_c}{f_c'^{1/4}} = \frac{A_b f_s}{f_c'^{1/4}} = \left[52.226 l_d \left(c_{\min} + 0.5 d_b\right) + 2799.447 A_b\right]\left(0.1\frac{c_{\max}}{c_{\min}} + 0.9\right)$$

$$\text{COD} = 0.968262$$

with the following information

e = 360.582
e% = 7.0068%
1-RE = 0.9818
r = 0.98560
COV = 0.08925

The induced equation is evaluated by its predictive performance on the evaluation data that is not accessed during the learning process. This data set, unseen during learning, contains 28 examples from Zuo's experiments. The comparison is listed in Table 5.4.

used as a black box: give input and get output. Users need to pay attention to the intermediate results and interact with the system so that they can obtain results they can trust.

5.3 FRACTURE TOUGHNESS

Although the total number of structures that have failed by brittle fracture is low, brittle fracture is catastrophic. Fracture mechanics addresses the behavior of materials during brittle fracture. An important measure used in fracture mechanics is the material fracture toughness, which characterizes the fracture behavior of structural materials. The three primary factors that affect the fracture toughness of structural and pressure vessel steels are temperature, loading rate, and constraint. Considerable research has been conducted in the areas of temperature and loading rate effects on fracture toughness. Generally, the fracture toughness of structural steels increases with increasing temperature and decreases with increasing loading rate.

To study the effect of constraint on fracture toughness, experiments were carried out and data were collected at the University of Kansas (KU) [Smith and Rolfe, 1997] and at Oak Ridge National Laboratories (ORNL) [Theiss, Shum and Rolfe, 1994]. Using the collected data, the influence of constraint was studied by Smith and Rolfe [1997]. In this section, EDDE is tested on this actual engineering data set, and the results are discussed and compared with those in their research [Smith and Rolfe, 1997].

5.3.1 Preparation

5.3.1.1 Data Collection

An experimental investigation was conducted to study the relative roles of crack depth (a) and crack-depth to width ratio (a/W) on the fracture toughness of an ASTM A533-B steel. The experiments were carried out on three-point bending specimens. The test set-up is shown in Fig. 5.5, where a is crack depth and W is

only the toughness CTOD is included in the study. There are two reasons for selecting CTOD as the test objective. First, various types of toughness are related with one another. Understanding one of them is of benefit in understanding the others. Meanwhile, their study has already shown that the effect of crack depth and *a*/W ratio on the toughnesses are similar. From the qualitative point of view, they are the same. Therefore, studying one type of toughness will be sufficient for the testing of EDDE. Second, CTOD is one of the most often-used notch toughness measure because the CTOD test method is based on the determination of a critical strain at fracture from a load-displacement record that does not require a stress analysis. Using CTOD, linear-elastic fracture mechanics can be extended into the nonlinear elastic-plastic region.

5.3.1.3 Data Preparation

To test EDDE on this actual engineering data set including 85 examples, the data set is prepared before the actual learning process starts. Keeping the data analysis objectives in mind, the data is prepared so that the results can compare the toughness for specimens with a constant crack depth and varying *a*/W ratios, and also compare the toughness for specimens with varying crack depths and constant *a*/W ratios. Therefore, the data preparation mainly includes the discretization of these two variables *a* and *a*/W.

After consulting a domain engineer (Dr. Rolfe), the discretization is kept the same as in their study. The actual value of crack depth *a* ranges from 0.08 to 2.0 in, and it is discretized into 3 nominal values $a = 0.08$, 0.4, and 2.0 in. The ratio *a*/W ranges form 0.1 to 0.5 and is discretized into 3 nominal values $a/W = 0.1$, 0.32, and 0.5. The results of discretization are also shown in Appendix C.

5.3.1.4 Domain Knowledge

The domain knowledge, acquired from the discussion with the domain

5.3.2 Analysis Results by EDDE

5.3.2.1 Discovered Equation When Both *a* and *a*/W Are Description Variables

In this experiment test, the data set of 85 examples is used and the dependent variable is taken as *ln*(CTOD). In order to decide what percentage of data will be used as testing data, preliminary analysis is run on the data with all default model formation parameters except the percentage of testing data. The percentage of testing data varies from 20% to 80% in increments of 10%, where the two extremes of 10% and 90% are not included because the given data set is small. The selection of the percentage is based on the prediction performance on testing data.

Table 5.5 Test results

Percentage of testing data	Prediction error (*e*)	1-R	Ratio Mean (*r*)	COV
20	0.24975	0.9221	0.80258	1.04281
30	0.25853	0.9285	0.99685	0.17257
40	0.41793	0.8721	1.02542	0.59647
50	0.29773	0.8866	0.56607	5.14652
60	0.40660	0.8229	0.93358	0.50458
70	0.41277	0.8349	1.09716	0.74747
80	0.43492	0.8123	0.85552	1.04247

The results are listed in Table 5.5. From the table, it is noticed that the best results are obtained when the testing data is 30% of total given data, which is the default value. Therefore, 30% of the input data will be used as testing data in later data analysis. Next, we discuss the results given by EDDE when the percentage of testing data is 30%.

The results are shown in Fig. 5.8. Both crack depth and *a*/W ratio are used to

itself. When a is 0.08 in. and 2.0 in., the a/W ratio is constant. All a/W ratios are 0.1 when a is 0.08 in, and all a/W ratios are 0.5 when a is 2.0 in. When the data does not provide information about varying a/W, it is impossible for EDDE for find how the a/W ratio influences specimen toughness when a varies to 0.08 in. or 2.0 in.

5.3.2.3 Toughness with Varying Crack Depth a and Constant a/W Ratio

In order to compare the toughness for specimens with varying crack depths and a constant a/W ratio, only the a/W ratio is taken as a description variable while leaving the crack depth a as a prediction variable. The results are shown in Fig. 5.11 when all model formation parameters are set at default values.

Fig. 5.11 shows that when the a/W ratio are 0.1 and 0.5, crack depth does affect the specimen toughness. But, when a/W is 0.32, the influence of crack depth is not reflected in the induced equation. This does not mean that crack depth has no effect on the specimen toughness. In fact, crack depth may also influence CTOD when a/W is 0.32, which is in between 0.1 and 0.5, because the effect of the a/W ratio should be continuous and gradual. However, the data provides no information on varying crack depth when a/W is 0.32. When a/W is 0.32, all specimen crack depths are 0.4 in.

In the following, we discuss the influence of crack depth on the specimen toughness only when the a/W ratio is 0.1 and 0.5. When the a/W ratio is 0.1, referred to as a shallow crack geometry, ln(CTOD) = 6.2783+0.0377T-5.259779a. When a/W is 0.5, referred to as a deep crack geometry, ln(CTOD) = 2.9124+0.0237T-0.375273a. Obviously, increasing crack depth a will decrease the specimen toughness. However, the strength of the influence is different. Crack depth has less effect on the fracture toughness for deep crack geometry (a/W is 0.5) than for shallow crack geometry (a/W is 0.1). For the same crack depth, assume a is 1.0 in., ln(CTOD) will decrease 5.2594 for shallow crack depth and 0.3753 for dcep crack depth. The magnitude of decrease in ln(CTOD) is significantly larger for shallow crack depth than for deep

5.4.1 Preparation

Before the tests are carried out on EDDE, the following preparations have been done, which include the discussions of data collection and domain knowledge.

5.4.1.1 Data Collection

An experimental investigation was conducted to study the dissolution of ionizable drugs. The experiments were carried out on a laminar flow device. The test set-up is shown in Fig. 5.15. As shown in the figure, a dissolution medium flows into the flow channel, then passes the compressed pellet of a drug where some drug is dissolved into the medium, finally, the medium with the dissolved drug flows out of the flow channel. During the tests, the steady state drug concentration in laminar flow at a single flow rate was measured using equipment. Then, the drug flux, defined as the dissolution rate of a drug per surface area, is calculated based on drug concentration. Drug flux will be the dependent variable in the study.

Four model drugs are used in the experiments, which are shown in Tables 5.6. The data are listed in Appendix D at the end of this dissertation. Compared with other data sets discussed in this chapter, this data set is relatively small with 64 entries.

Table 5.6 Model drugs

Model Drugs	Solubility ($x10^5$M)	pKa
Cinnarizine	0.598	7.47
Naproxen	13.0	4.57
Benzoic	2250	4.03
2-Naphthoic acid	13.7	4.02

account.

The final induced equation is also tested using evaluation data that is not accessible to EDDE. The testing shows the error measures on evaluation data are consistent with the estimated performance predicted by the system. The prediction error, ratio mean, and coefficient of variation are 360.582, 0.9856, and 0.0893 respectively on evaluation data, while the estimates of these measures are 360.232, 1.0420, and 0.0711 respectively by EDDE.

It is learned from this application that the final results are not obtained just by doing one batch of learning. The meaningful results are obtained through the interaction between the system and the user.

The application in Section 5.3 studies the effect of crack length *a* and crack constraint *a*/W ratio on fracture toughness. The learning objectives are to compare toughness for specimens with a constant crack depth *a* and varying *a*/W ratios, and to compare toughness for specimens with varying crack depth *a* and constant *a*/W ratios. These objectives are applied to help the user in classifying the variables into description and prediction variables.

1) When the effect of different crack depth *a* and crack constraint *a*/W is investigated, *a* and *a*/W are both classified as description variables. The results show that *a* and *a*/W have interactive effect on fracture toughness CTOD. 2) To compare toughness for specimens with a constant crack depth *a* and varying *a*/W ratios, *a* is classified as a description variable and *a*/W is classified as a prediction variable. The results show that *a* is used to divide the domain space, and increasing *a*/W ratio will decrease the toughness CTOD when *a* is constant. 3) To compare toughness for specimens with varying crack depth *a* and constant *a*/W ratios, *a*/W is classified as a description variable and *a* is classified as a prediction variable. The results show that *a*/W is used to divide the domain space, and increasing crack depth *a* will decrease the specimen toughness when *a*/W is constant. However, the strength of the

compared with those by Quinlan's study [1993] in Table 5.9. The methods used in Quinlan's study [1993] are described in Section 4.4.2. Table 5.9 shows that the prediction error and the relative error are 2.33 and 17.0% respectively by EDDE. They are lower than or comparable to the algorithms (except neural nests and neural nets + instances) presented in [Quinlan, 1993]. Apparently, EDDE outperforms most of the algorithms, with the improvement rate of up to 180% in prediction error and of up to 35% in RE. This is not a surprise because those algorithms do not require any information on the domain and no domain knowledge is provided to the system. This also demonstrates the important role of domain knowledge in knowledge discovery from data.

Another advantage of the induced equations over the Quinlan's method is that the trend of *mpg* with model year can be studied using the induced equations.

From all 392 examples, the average values of each attribute are calculated, and they are

$$w = 2977.6 \qquad d = 194.4 \qquad p = 104.5$$

Substitute these average values into the induced equations, the *mpg* of each year is obtained. The *mpg* versus year is plotted in Fig. 5.22. Fig. 5.22 shows that *mpg* has an increase trend with year although *mpg* of last year is not always less than that of next year.

5.6 SUMMARY

This chapter tests the performance of EDDE in engineering domains using actual engineering data sets. The results show that EDDE can be applied in a variety of engineering domains, verifying that the problem type characterized in Chapter 1 occurs widely in engineering. They also show that understanding the domain and introducing function templates to the system based on domain knowledge play a very important role in discovering useful and meaningful knowledge. They also demonstrate the importance of the interaction between the system and the users. In

parameters are set at default values except that the percentage of testing data varies from 10% to 90% in increments of 10%.

Table 5.8 Results by different percentages of testing data.

Percentage of testing data	Prediction error (e)	Percentage error (e%)	Accuracy	Ratio Mean (r)	COV
10	1.85297	8.0734	84.21053	0.96493	0.09429
20	1.94019	9.0120	82.85714	1.05255	0.20450
30	1.82198	8.1414	89.69072	1.00200	0.10336
40	2.12580	9.3674	79.06977	1.00758	0.13008
50	2.07802	9.1219	78.84615	1.01711	0.12193
60	2.10521	9.6459	83.79888	1.01284	0.19395
70	2.29164	9.8994	77.89474	1.05087	0.49790
80	2.35314	10.1382	77.33333	1.01116	0.20043
90	3.07342	13.7061	58.84956	1.02713	0.18373

Note: All induced equations have 13 region-equation pairs where the regions are divided by the attribute <model year>.

Table 5.8 shows that the attribute <model year> is always used to divide the domain space, and the equations induced when the percentage of testing data is 30% give best prediction on testing data. The induced equations are

Year 70: $mpg = 54.46 + 0.0754d - 0.1847p - 0.0028w - 11.2880d/p$

Year 71: $mpg = 43.35 - 0.0891p - 6.656144d/p$

Year 72: $mpg = 35.90 - 0.0053w$

Year 73: $mpg = 20.20 + 0.069067p - 0.006682w + 0.399698w/p$

Year 74: $mpg = 31.23 - 0.0030w - 6.1964d/p + 0.4337\ w/p$

Year 75: $mpg = 37.88 + 0.1437d - 0.0145w - 15.0480d/p + 0.9108w/p$

Year 76: $mpg = 41.52 - 0.0065w$

Year 77: $mpg = 78.00 + 0.1792d - 0.4575p - 23.8928\ d/p$

5.5.2.1 Role of Domain Knowledge

First, experiments are carried out on the data set that excludes the examples with the attribute <cylinders> of either 3 or 5 in order to check whether the attribute <cylinders> should be used to divide the domain space. Therefore, the data set contains 385 examples.

All model formation parameters are set at default values. The results show that the attribute <cylinders> has never been used to divide the domain space no matter whether nonlinear templates (discussed in last section) are provided to EDDE. This is consistent with the anticipation based on the domain knowledge. Therefore, the data changes are unnecessary. All 392 examples, without removing the examples whose attribute <cylinders> equal 3 or 5, are given to EDDE in later experiments in order to used more information and acquire better results.

On the data set of 392 examples, experiments are carried out at three stages. At all three stages, the model formation parameters are set at default values.

At the first stage, no domain knowledge is introduced so that all templates used in learning are linearly related with the dependent variable *mpg*. A set of equations is induced by EDDE where the attribute <model year> is used to divide the domain space into 13 regions. Information on the performance of the induced equations on testing data is provided in the second column of Table 5.1. At the second stage, in addition to the templates in the first stage, a nonlinear template *d/p* is introduced to EDDE for learning. A set of equations is induced by EDDE where the attribute <model year> is also used to divide the domain space into 13 regions. Information on the performance of the induced equations on testing data is provided in the third column of Table 5.1. At the third stage, in addition to the templates in the second stage, another nonlinear template *w/p* is introduced to EDDE for learning. A set of equations is induced by EDDE where the attribute <model year> is also used to divide the domain space into 13 regions. Information on the performance of the

Section 5.5.1.2.

5.5.1.1 Data Preparation

The data concerns city-cycle fuel consumption in miles per gallon (*mpg*) to be predicted in terms of three multivalued discrete attributes and four continuous attributes. The three discrete attributes are <cylinders>, <model year>, and <origin>. The four continuous attributes are <displacement>, <horsepower>, <weight>, and <acceleration>. The identifications of description variables and prediction variables are very straightforward. The three discrete attributes are used as description variables and the other four continuous attributes are used as prediction variables. The *mpg* is taken as the dependent variable.

To test EDDE on this actual engineering data set, the data set is prepared before the actual learning process starts. First, there are 6 examples with missing values. They are removed from the data provided to EDDE. The data set without any missing values contains 392 examples and is listed in Appendix E at the end of this dissertation.

The data list shows that there are only 4 examples when the attribute <cylinders> equals 3 and only 3 examples when the attribute <cylinders> equals 5. EDDE requires the number of each data subset to be larger than the number of the variables when EDDE tests whether to use a variable to divide the domain space. If those examples are kept, the tests for variable <cylinders> will fail and the influence of <cylinders> as a possible variable for splitting the data into subset will not be checked. Therefore, those 7 examples with <cylinders> of 3 or 5 are removed from the data provided to EDDE. Such changes make the data set contain 385 examples. In fact, we can see later that the changes are unnecessary because <cylinders> is not used by EDDE to divide the domain space. Therefore, the data set including all 392 examples without the removal of the 7 examples is then provided to EDDE for learning in order to use more given information and obtain better results.

variable to divide the domain space. If all the prediction variables are introduced at the same time, the test for any of the description variables will fail and the influence of a description variable as a possible variable for splitting the data into subsets will not be checked. Therefore, the description variables cannot be introduced at the same time. Based on repetitive tests, it is found that the maximum of two prediction variables can be introduced at that same time so that the influence of description variables can be checked. Because the medium pH is the most significant variable based on the existing domain knowledge, at the beginning the prediction variables are introduced in the following ways:

- <Medium pH> and <drug solubility>
- <Medium pH> and <particle size>
- <Medium pH> and <flow rate>

It is found that the induced equations are always the same no matter what prediction variables are introduced. The discovered equations are

If <pKa1> = 4.02, $\langle flux \rangle^{0.1} = 1.011 + 0.289\, e^{0.1 * \langle medium\ pH \rangle}$

If D_pKa1 = 4.03, $\langle flux \rangle^{0.1} = 2.147 + 0.141\, e^{0.1 * \langle medium\ pH \rangle}$

If D_pKa1 = 4.57, $\langle flux \rangle^{0.1} = 0.943 + 0.318\, e^{0.1 * \langle medium\ pH \rangle}$

If D_pKa1 = 7.47, $\langle flux \rangle^{0.1} = 3.627 - 1.376\, e^{0.1 * \langle medium\ pH \rangle}$

Prediction error (e)	= 0.05868
Prediction percentage error (e%)	= 3.3651%
1-RE	= 0.9679
Ratio mean (r)	= 0.99813
Coefficient of Variation (COV)	= 0.04248

The induced equations are consistent with the existing domain knowledge. First, they confirm that medium pH is the most significant variables. Second, they display that medium pH has a positive effect on acidic drug and a negative effect on basic drugs. That is to say, flux increases with the increase of medium pH for acidic drugs, while flux decreases with the increase of medium pH for basic drugs.

influence is different. The effect is stronger for shallow crack geometry than for deep crack geometry.

It is learned from this application that in addition to the importance of introducing templates based on domain knowledge, it is also important to keep the learning objectives in mind in the learning process.

The application in Section 5.4 studies the dissolution of ionizable drugs using data from chemical engineering. The templates are put into the system based on domain knowledge. The induced equations can predict the $<flux>^{0.1}$ with most of data within 15% prediction error boundary. In addition, the results are carefully studied by domain engineers. It is found that the results by EDDE are consistent with the existing domain knowledge: 1) medium pH is the most significant variables; 2) medium pH has a positive effect on acidic drug and a negative effect on basic drugs. That is to say, flux increases with the increase of medium pH for acidic drugs, while flux decreases with the increase of medium pH for basic drugs.

This thesis presents only the final results for this application. In fact, before the final results, there are many iterations of learning. Each time the user studies the feedback information, and decides whether to modify the existing templates, or to add more domain knowledge to the system.

The application in Section 5.5 studies automobile fuel consumption using the data from data repository maintained at University of California, at Irvine. The results show that the more domain knowledge is introduced to the system, the better the results. This can be shown in prediction error and percentage error. The prediction error without the introduction of nonlinear function templates, with the introduction of one nonlinear function template, and with the introduction of two nonlinear function templates are 2.03, 1.85, and 1.82 respectively, while the percentage error are 10.2%, 8.93%, and 8.14% respectively.

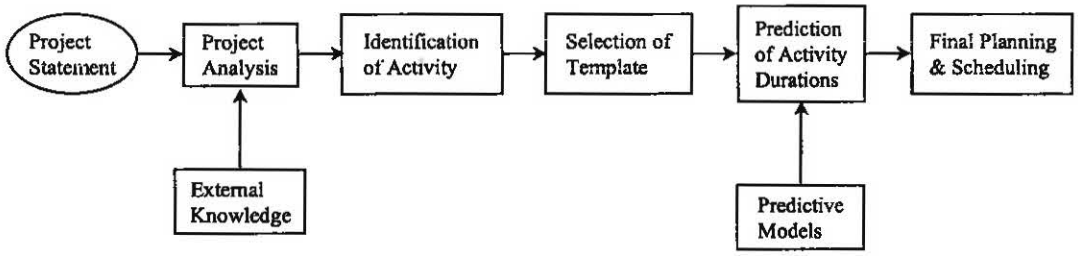


Fig. 5.1 Process of managing a project.

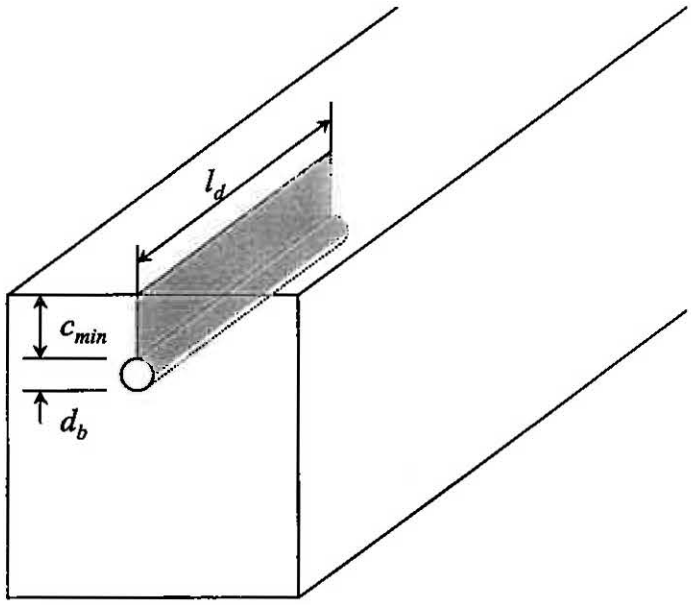


Fig. 5.3 Fracture surface at splice failure.

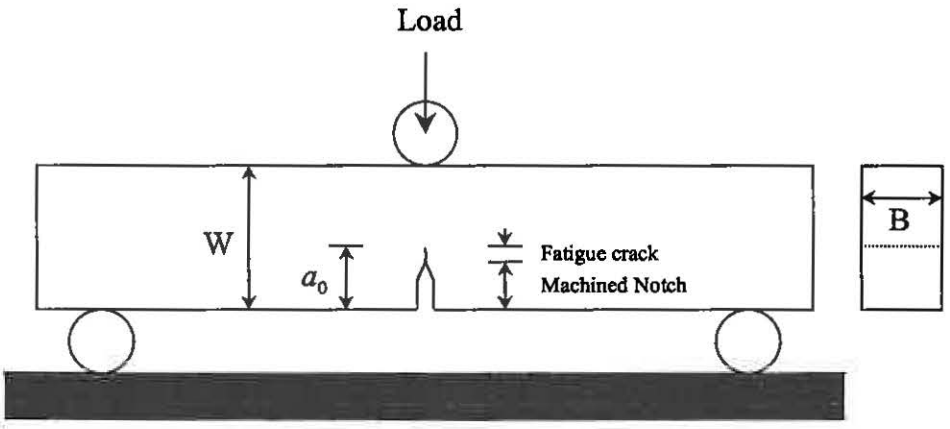


Fig. 5.5 Three point bending test set-up for CTOD.

assist users with making best decisions.

- The system based on the method of combining machine learning and regression analysis is a monostrategy system, which can solve only certain classes of learning problems as we discussed in Chapters 1 and 3. To be able to solve more equation discovery problems, more than one learning algorithm would be incorporated into the system [Michalski and Tecuci, 1994].

- The model is measured at two dimensions: prediction strength and comprehensibility. Choosing between alternate models, trade-off parameters are introduced. It would be interesting to use some other methods, such as MDL (Minimum Description Length). The MDL principle [Rissanen, 1989] is a statistical theory that balances model complexity and model error.

- As mentioned in Section 3.1.2, nonlinear functions are necessary for engineering problems. Although EDDE can find nonlinear functions from data, the nonlinear functions are intrinsically linear. It would be interesting to extend the system so that it can find intrinsically nonlinear functions. However, such extension will greatly increase the learning time because the solution can not be found in closed form when the function to be fit is nonlinear in the unknown parameters.

- The system can learn from data with noise, even when the noise variance is 50% of total variance. However, it is assumed that the noise in data is independent of the variables, and the noise variance is constant in the range of the independent variables (Section 3.3.1). It would be useful to study the performance of EDDE when the noise is related to the independent variable(s), and when the noise variance may change in the range of the independent variables.

to the data either noise free or with very small noise, the tests show that the system can correctly discover the underlying equations even when the noise variance is 50% of the total variance.

The system can find the actual equations from the data when the tree interpretation and prediction strength are studied and compared with other methods in Section 4.4. The methods include: regression tree CART, instance based IBL, multivariant linear regression, model tree M5, neural nets, and combinations of these methods.

On the same data set, EDDE generates the equations that divide the data space into two regions as does the known function while CART's equations divides the data space into 13 regions. The results show that EDDE's induced tree summarizes the data more concisely, and its description of the data is more straightforward comparing with CART. Also on the same data set, EDDE generates equations that give the prediction error of 1.1, while other methods give the prediction error from 1.1 to 3.77. On this past set, EDDE outperforms those algorithms, with an improvement rate of up to 260% in prediction error.

6.1.3 Application in Different Engineering Domains

The system EDDE has been applied on actual data sets from different engineering domains. The actual data sets come from civil engineering to study 1) durations of construction activities, 2) development/splice strength of reinforcing bars, and 3) fracture toughness; from chemical engineering to study dissolution of ionizable drugs; and from mechanical engineering to study automobile fuel consumption.

These applications show that the domain knowledge encoded in the algorithm is very general in engineering, so that EDDE can find many different applications in engineering. Each application has its own domain knowledge that is specific to the domain. This domain knowledge is provided to the system in the form of nonlinear

6.1.1 Methodology

The data analysis carried out in engineering is computationally expensive due to its characteristics. A methodology that combines machine learning and regression technique for equation discovery in databases from engineering has been developed in this thesis to overcome the difficulties in data analysis. This method successfully applies heuristics, well studied in machine learning, to solve the problems, which cannot be solved using only traditional regression technique. The data analysis where many variables, nonhomogeneous equations, and unknown significant variables are involved at the same time is successfully solved using the method to intelligently search the domain space to find the equation that satisfy users' requirements.

The important contribution is that the method successfully incorporates general engineering domain knowledge in all aspects of learning process, from knowledge representation to variable selections to interpretation of the discovered results, and specific domain knowledge in the form of nonlinear function templates when the system is used to solve specific engineering problems. Therefore, the method can be applied in a variety of engineering disciplines.

6.1.2 Learning Algorithm

Based on the proposed method, a learning algorithm has been developed and implemented in a computer system EDDE. The contributions embodied in the learning algorithm are listed as follows.

- For the intended learning task, the induced model is expressed as region-equation pairs, which are well represented using a model tree. Unlike previous representations for equation discovery, this representation is able to organize the knowledge into a hierarchy that can be easily understood.
- Equation discovery is a problem of searching the domain space to find the model tree that satisfies users' requirements. Unlike previous systems, the

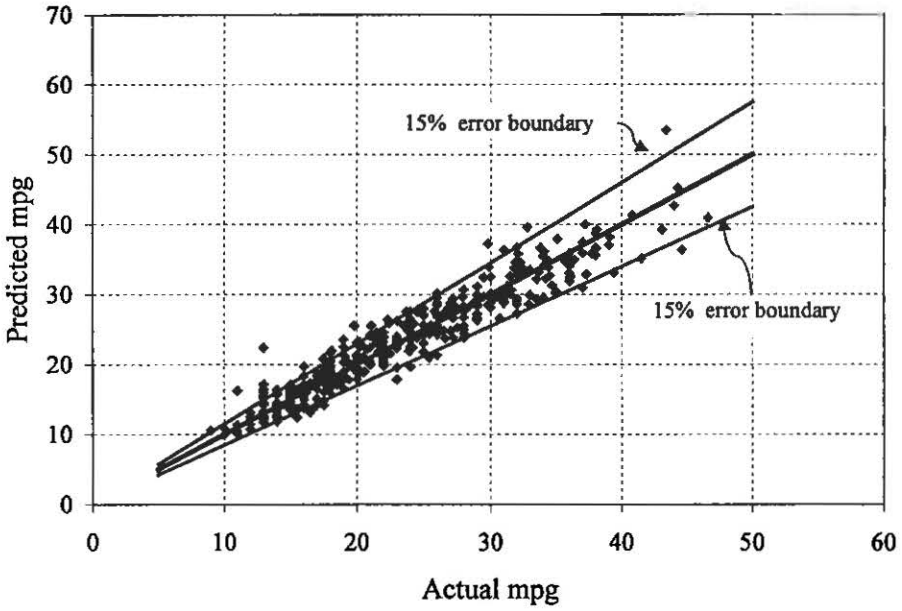


Fig. 5.21 Prediction performance on fuel consumption.

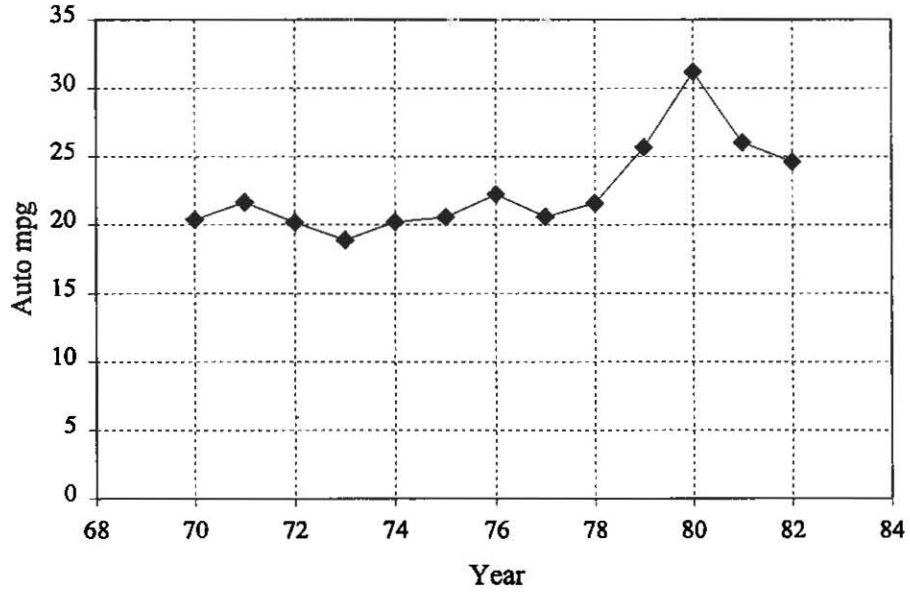


Fig. 5.22 Annual trend of mpg.

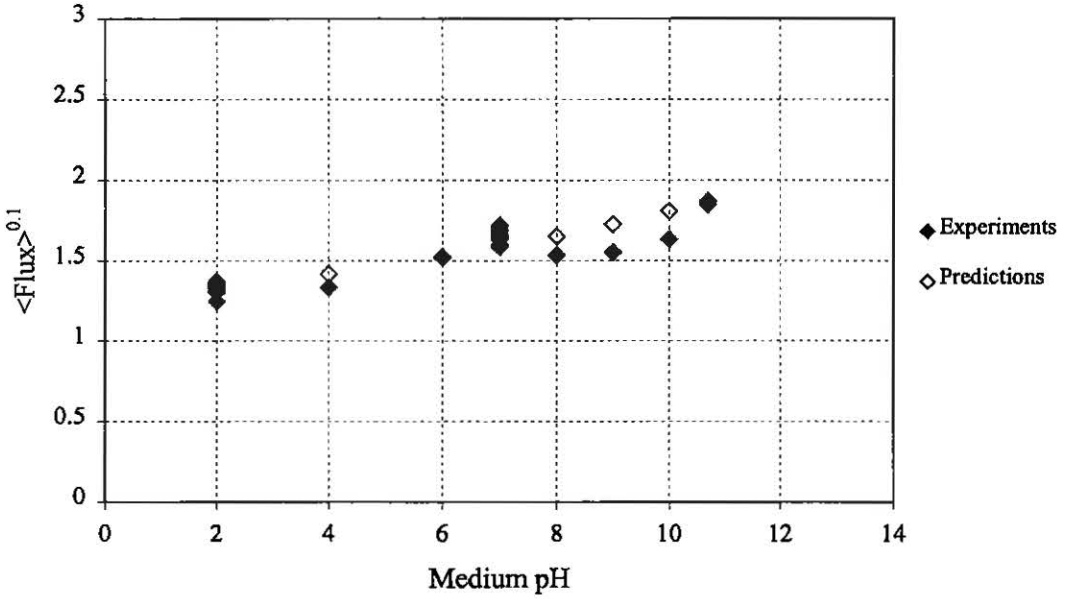


Fig. 5.18 Flux prediction when pKa = 4.57.

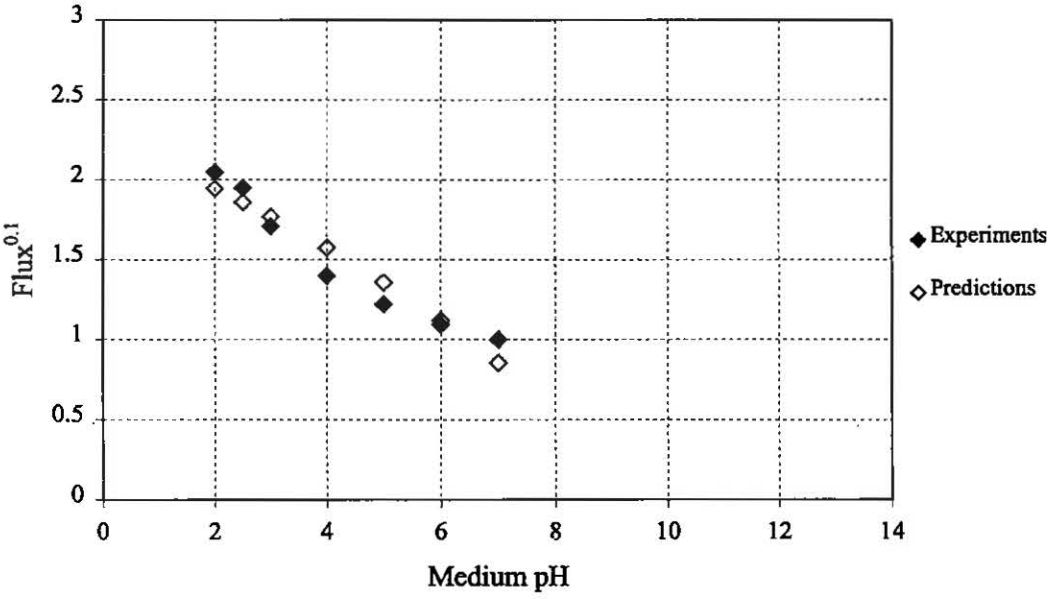


Fig. 5.19 Flux prediction when pKa = 7.47.

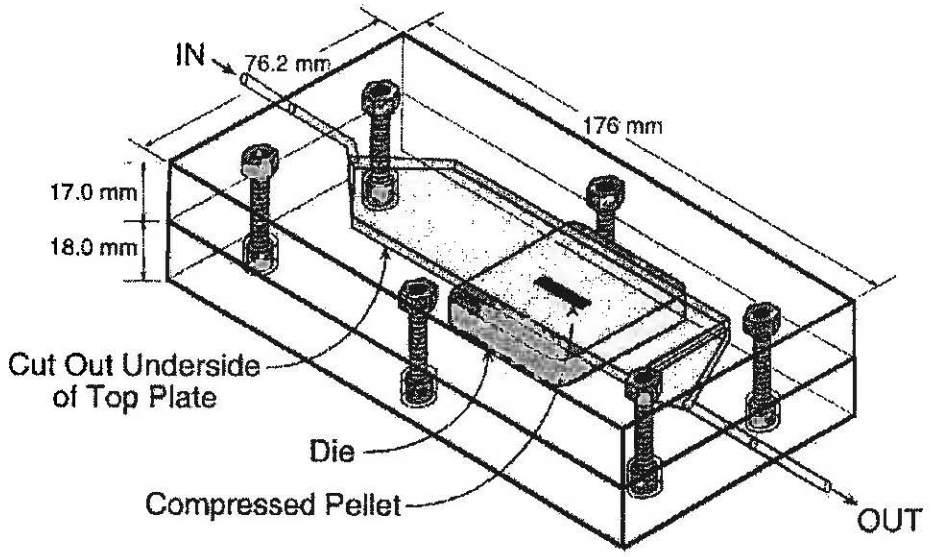


Fig. 5.15 Schematic diagram of the dissolution cell.

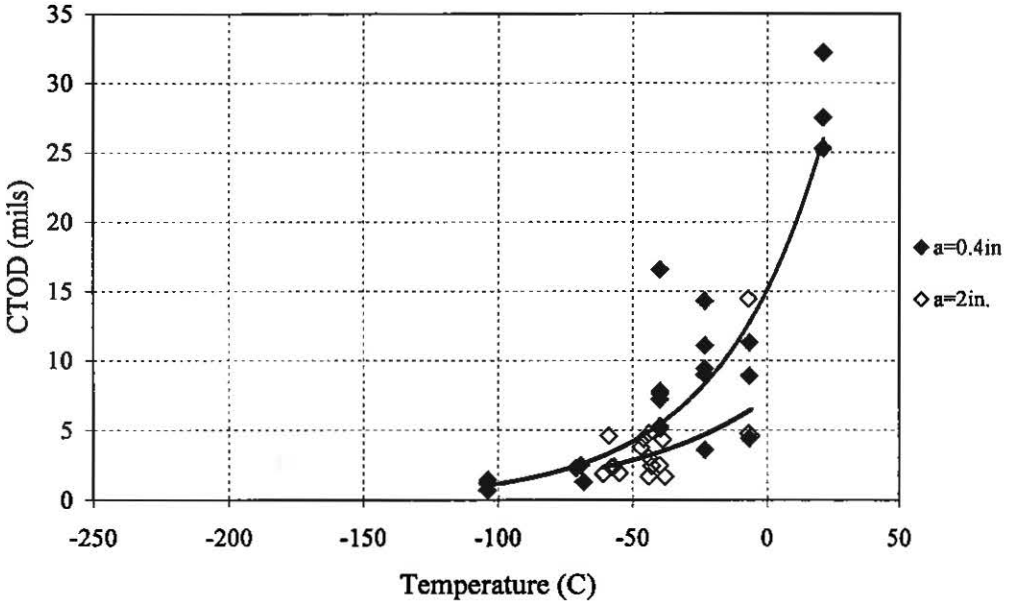


Fig. 5.12 Deep crack geometry

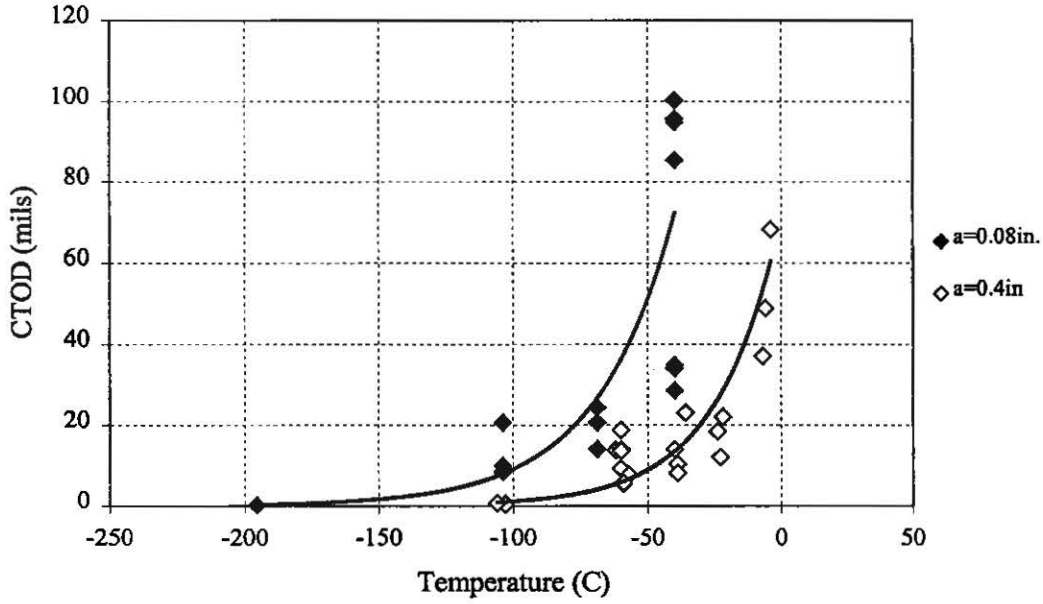
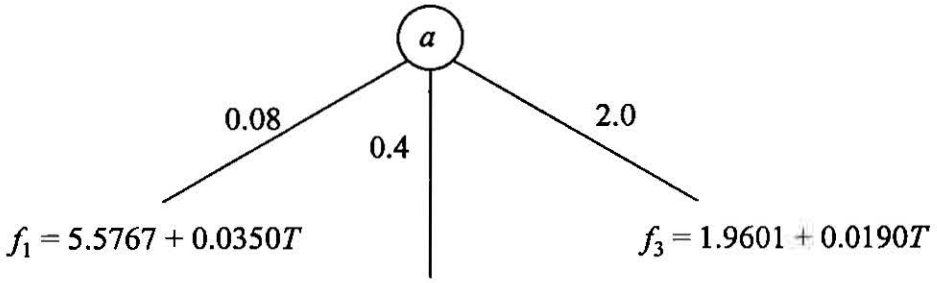


Fig. 5.13 Shallow crack geometry



$$f_2 = 3.8801 + 0.0310T - 1.8284a/W$$

Fig. 5.9 Constant crack depth and varying a/W ratio

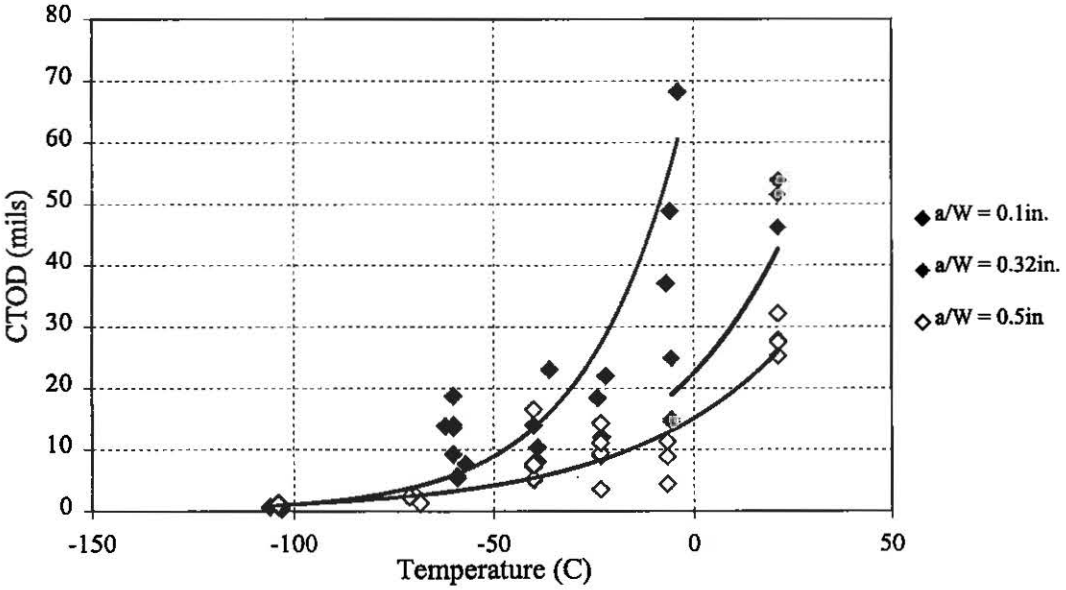


Fig. 5.10 Constant crack depth and varying a /W ratio

6.3 CLOSING REMARKS

The results shown in this thesis demonstrate the importance of research in assisting engineers in mining useful and meaningful knowledge from data. EDDE is a useful addition to the family of systems for equation discovery in databases. It opens up a paradigm of knowledge discovery research applicable to actual engineering domains.

Knowledge Discovery in Databases. In *Advances in Knowledge Discovery and Data Mining*, Eds. Fayyad, U.M., G. Piatetsky-Shapiro, P. Smyth and R. Uthurusamy, AAAI/MIT Press, Cambridge, Mass

Fayyad, U., G. Piatetsky-Shapiro and P. Smyth, (1996b). The KDD Process for Extracting Useful Knowledge from Volumes of Data, *Communications of the ACM*, Nov. 1996, Vol. 39, No. 11.

Forsyth, R. (1989), *Machine Learning, Principles and Techniques*, Chapter 4, Chapman and Hall Company.

Freeman, J. A. and Skapura, D. M. (1991). *Neural Networks: Algorithms, Applications, and Programming Techniques.* Addison-Wesley, Reading, MA.

Green, G. H. (1988), *The Abacus.2 System for Quantitative Discovery*, Master Thesis, Department of Computer Science of the University of Illinois at Urbana-Champaign.

Grzymala-Busse, J. W.(1991), *Managing Uncertainty in Expert Systems*, Kluwer Academic Publishers.

Hogg, R. V. and J. Ledolter (1987). *Engineering Statistics*, Macmillan Publishing Company, New York.

Kibler, D., D. W. Aha, and M. K. Albert, (1988), Instance-based Prediction of Real-Valued Attributes, Technical Report 88-07, ICS, University of California, Irvine.

Kokar, M. M. (1986a), Determining Arguments Of Invariant Functional Descriptions. *Machine Learning*, 1(4), 1986.

Kokar, M. M. (1986b), Discovering Functional Formulas Through Changing Representation Base, In *Proc. of the Fifth National Conference on Artificial Intelligence, 1986.*

Langley, P. (1994). Selection of Relevant Features in Machine Learning, *Proceedings of the AAAI Fall Symposium on Relevance*, New Orleans, LA: AAAI Press.

Langley, P. (1996). *Elements of Machine Learning*, Morgan Kaufmann Publishers Inc., San Francisco, California.

Langley, P. (1996). Relevance and Insight in Experimental Studies, *IEEE Expert*, Oct. 1996.

Proceedings on the Tenth International Conference of Machine Learning, 236-243, University of Massachusetts, Amherst. Morgan Kaufmann.

Quinlan, R. J. (1994), *C4.5 : Programs for Machine Learning*, Morgan Kaufmann Publisher Inc.

Rao, R. B. (1993), *Inverse Engineering: A Machine Learning Approach to Support Engineering Synthesis*, Ph. D. dissertation, University of Illinois at Urbana-Champaign.

Rao, R. B. and S. C-Y. Lu (1993), A Knowledge-Based Equation Discovery System for Engineering Domains, IEEE Expert, August, 1993.

Reich, Y. (1991). *Building and Improving Design Systems: A Machine Learning Approach*, Ph.D. Thesis, Carnegie Mellon University, Pittsburgh, PA.

Reich, Y. and Fenves, S. J. (1989). The Potential of Machine Learning Techniques for Expert Systems. *Artificial Intelligence for Engineering Design, Analysis, and Manufacturing*, 3(3).

Rissanen, J., (1989). Stochastic Complexity in Statistical Inquiry, World Scientific.

Roddis, W. M. K. and L. Zhang (1997a), Development of Project Activity Duration and Resource Requirement Algorithms Based on Historical Data. Report No, K-TRAN:KU-95-7, Civil and Environmental Engineering Department, University of Kansas.

Roddis, W. M. K. and L. Zhang (1997b), Combination of Machine Learning and Regression Analysis in Knowledge Acquisition, Computing in Civil Engineering.

Schaffer, C. (1990). *Domain – Independent Scientific Function Finding*, Ph.D. Thesis, Rutgers University.

Schaffer, C. (1991). On Evaluation of Domain – Independent Scientific Function Finding Systems. in *Knowledge Discovery in Databases*, ed. G. Piatetsky-Shapiro and W.J. Frawley, The MIT Press.

Schlimmer, J. C. and Fisher, D. (1986), A Case Study of Incremental Concept Induction Science, *Proc. of the 5th National Conference on AI*, San Mateo, CA, Morgan Kaufmann.

Engineering, The University of Kansas.

Zytkow, J. M. (1987), Combining many searches in the FAHRENHEIT discovery system. In Proc. of the fourth international workshop on machine learning, San Mateo, California, Morgan Kaufmann.

Zytkow, J. M. and J. Baker (1991), Interactive mining for regularities in databases. In *Knowledge Discovery in Databases*, Eds. G. Piatetsky-Shapiro and W. J. Frawley, AAAI Press, Menlo Park, California.

25	Y	F	W	S	C	8.968	7.064	9.401	2.232	5.746	93.117
26	N	T	W	T	A	0.860	9.109	0.165	0.179	1.874	1.810
27	N	T	W	T	C	8.321	9.327	8.544	3.387	3.416	77.922
28	N	T	E	R	B	6.464	1.966	5.105	9.307	7.879	47.393
29	N	F	W	T	A	8.614	2.309	5.619	6.981	2.297	46.450
30	N	F	W	R	C	7.048	0.863	0.694	1.948	1.810	14.495
31	Y	T	E	R	A	5.253	7.465	1.386	8.110	6.360	120.846
32	N	F	E	T	B	2.483	7.228	0.933	6.754	6.112	12.112
33	N	T	E	S	A	3.259	7.997	6.460	6.699	1.690	57.412
34	Y	T	W	S	A	4.770	8.370	3.376	4.781	8.929	128.694
35	Y	F	E	S	A	0.724	1.963	1.453	7.967	1.174	27.387
36	N	T	E	S	B	0.955	8.816	8.637	8.336	0.314	73.309
37	Y	T	E	S	A	8.680	5.343	6.338	9.556	4.767	125.749
38	N	F	E	T	B	9.803	5.312	1.381	4.961	3.711	14.306
39	Y	T	E	S	A	6.339	1.922	3.249	3.975	7.408	70.450
40	N	F	E	S	C	6.730	5.240	2.328	8.334	0.177	28.010
41	Y	F	W	T	B	0.991	1.090	8.700	1.203	2.220	16.870
42	Y	T	W	S	C	2.406	3.893	8.008	4.252	9.527	59.312
43	Y	T	W	T	A	0.613	5.631	8.272	0.505	5.466	64.947
44	Y	F	W	S	B	7.814	1.581	9.577	2.329	5.393	46.802
45	N	T	W	S	C	5.180	6.756	3.260	2.674	2.417	33.926
46	N	T	E	S	A	2.385	0.563	8.085	4.038	7.482	65.542
47	N	T	W	S	C	8.191	5.114	0.701	8.046	5.629	6.040
48	Y	F	W	S	C	1.214	6.188	2.065	9.969	9.706	47.734
49	Y	F	E	S	C	7.522	4.900	1.803	3.240	3.820	68.079
50	N	T	W	S	B	6.944	7.494	5.394	6.348	4.286	48.437
51	N	F	E	R	C	9.491	1.947	2.845	4.006	7.583	25.395
52	Y	T	W	R	A	9.053	7.561	1.854	6.157	5.084	153.166
53	N	T	E	R	A	8.454	3.768	6.380	8.210	3.774	58.562
54	Y	T	W	S	A	8.744	4.477	2.209	1.897	5.275	120.861
55	Y	T	W	R	B	2.336	2.168	6.249	5.728	6.280	42.701
56	N	F	E	T	C	7.949	4.964	4.558	6.531	2.142	39.508
57	N	T	W	S	A	9.904	3.476	7.477	2.951	9.428	61.619
58	N	T	E	R	B	9.509	0.152	0.489	0.732	6.612	5.634
59	Y	F	E	R	B	2.800	1.337	4.340	5.652	4.869	29.336
60	Y	T	E	T	C	5.812	6.918	3.355	9.814	4.880	117.507
61	Y	F	E	R	A	2.478	7.360	5.273	8.579	5.399	70.566
62	Y	F	W	R	A	9.803	5.818	6.545	0.957	3.674	88.831
63	Y	T	W	T	B	6.035	5.014	6.107	2.488	0.453	104.360
64	N	T	W	S	A	6.487	4.512	8.763	9.149	2.138	74.344

105	N	F	W	T	A	8.850	4.204	7.200	5.909	6.706	68.933
106	N	T	E	S	B	4.814	0.063	7.216	5.829	9.305	63.113
107	Y	F	E	T	B	1.974	2.638	5.847	2.004	8.314	27.549
108	N	T	W	R	B	0.846	3.646	1.264	5.552	6.352	19.865
109	Y	T	E	R	B	1.727	1.590	9.666	0.387	4.455	39.764
110	Y	T	W	T	C	1.474	5.502	0.478	7.080	9.912	67.251
111	N	T	E	R	A	7.669	4.087	1.254	8.410	6.886	19.683
112	Y	T	E	T	B	0.721	3.858	0.384	7.009	5.025	44.285
113	Y	T	W	S	A	4.567	8.038	9.911	5.296	9.337	121.794
114	Y	F	W	T	A	9.217	3.116	9.892	2.745	9.316	61.570
115	Y	T	E	T	B	3.494	6.377	7.073	7.170	7.733	92.248
116	N	T	W	T	B	9.864	3.455	9.748	3.880	1.239	82.064
117	Y	T	W	T	B	4.088	8.175	4.520	4.715	3.611	115.887
118	Y	F	W	R	C	8.292	8.984	2.998	2.374	9.806	101.052
119	Y	F	W	S	A	0.027	9.413	3.035	6.910	3.814	72.240
120	N	F	E	S	A	0.452	3.622	8.897	3.043	7.283	75.330
121	Y	F	W	S	C	8.950	9.037	4.831	9.036	7.951	101.656
122	Y	F	E	S	C	2.794	1.270	3.030	6.250	5.554	28.598
123	Y	F	E	T	C	2.706	1.462	3.548	2.602	2.027	29.244
124	N	T	E	S	C	8.665	3.550	4.474	7.848	0.379	39.369
125	N	T	E	S	C	1.724	3.446	2.754	8.560	4.682	25.100
126	N	T	E	T	A	2.054	1.747	1.677	2.219	9.211	15.292
127	Y	T	E	R	C	9.675	6.160	2.593	8.221	2.035	139.656
128	N	T	E	S	B	5.802	3.857	4.949	5.677	7.085	42.027
129	N	T	E	T	C	9.135	6.047	5.554	7.908	6.004	47.257
130	N	T	W	R	B	9.841	3.770	2.407	8.712	1.611	23.397
131	N	T	E	T	B	1.333	3.426	7.062	6.627	6.071	63.552
132	N	F	E	S	B	5.694	7.863	9.769	8.950	7.579	81.756
133	Y	T	W	R	B	9.256	0.186	3.366	4.972	3.989	80.169
134	N	F	W	S	A	7.551	3.376	0.505	5.726	4.506	8.893
135	Y	F	W	T	C	3.042	8.275	5.070	0.958	6.528	77.830
136	Y	T	W	S	A	8.913	4.866	3.784	0.296	8.481	123.951
137	Y	T	W	R	B	3.880	7.273	4.802	5.193	4.557	108.489
138	N	T	E	R	A	5.497	8.355	0.255	9.448	4.179	9.947
139	N	F	W	R	A	6.423	5.140	1.694	6.907	3.318	11.799
140	N	T	E	S	B	4.101	0.292	3.695	6.820	7.781	37.036
141	N	T	E	S	C	9.259	5.611	3.724	7.652	6.329	38.404
142	Y	F	W	S	B	5.845	7.228	6.373	8.325	5.132	78.609
143	N	F	W	R	C	9.325	3.777	9.723	2.817	0.802	79.254
144	Y	F	E	R	C	0.007	3.987	4.397	7.250	2.129	35.541

185	Y	T	E	R	A	4.370	7.371	7.682	4.120	4.641	111.314
186	Y	T	E	S	B	2.123	4.663	6.802	3.481	2.969	67.981
187	N	T	E	T	A	8.019	8.450	3.974	2.286	6.233	36.318
188	N	T	W	T	B	9.020	4.457	8.800	8.078	6.905	76.510
189	N	T	W	R	A	3.748	2.529	8.446	9.191	2.445	70.718
190	N	F	W	S	B	7.279	0.048	3.629	8.237	4.748	34.100
191	N	F	W	R	A	9.498	3.510	5.847	4.953	3.516	53.709
192	Y	F	E	R	C	4.806	4.834	1.919	1.247	0.535	61.573
193	Y	T	W	R	C	9.440	1.390	7.541	1.884	6.334	94.834
194	Y	F	E	S	B	9.687	4.748	6.639	2.315	3.361	76.547
195	Y	T	W	T	B	7.411	2.653	8.991	7.040	3.118	86.362
196	N	F	W	T	C	4.650	4.024	5.052	1.666	2.496	45.938
197	Y	F	E	T	A	4.735	7.731	7.089	5.023	4.293	79.484
198	N	F	W	R	C	5.499	6.732	3.292	6.911	9.405	22.076
199	Y	F	W	R	B	0.215	6.926	6.952	9.052	3.102	56.424
200	N	T	W	S	C	7.963	8.160	5.720	1.991	6.197	53.406

A.2 TESTING DATA

index	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	y
5	Y	F	E	T	C	3.760	1.730	3.550	5.410	3.240	32.720
13	N	F	W	S	A	4.470	6.160	1.430	6.620	2.100	18.240
15	N	F	W	T	B	8.710	3.200	2.300	0.860	7.490	20.790
16	N	F	W	T	C	0.870	4.240	5.730	7.780	8.190	45.420
17	Y	T	E	S	B	4.600	6.590	5.760	9.880	5.510	106.770
28	N	T	E	R	B	6.460	1.970	5.110	9.310	7.880	47.390
40	N	F	E	S	C	6.730	5.240	2.330	8.330	0.180	28.010
42	Y	T	W	S	C	2.410	3.890	8.010	4.250	9.530	59.310
43	Y	T	W	T	A	0.610	5.630	8.270	0.510	5.470	64.950
48	Y	F	W	S	C	1.210	6.190	2.060	9.970	9.710	47.730
52	Y	T	W	R	A	9.050	7.560	1.850	6.160	5.080	153.170
53	N	T	E	R	A	8.450	3.770	6.380	8.210	3.770	58.560
57	N	T	W	S	A	9.900	3.480	7.480	2.950	9.430	61.620
64	N	T	W	S	A	6.490	4.510	8.760	9.150	2.140	74.340
65	N	T	W	T	C	5.720	0.490	4.480	4.910	8.210	35.070
67	N	F	W	S	B	2.610	3.800	1.280	2.240	5.620	15.010
80	Y	T	E	R	C	8.880	6.060	7.920	9.730	2.120	138.570

APPENDIX B

THE DATA SET USED IN SECTION 5.2

B.1 THE INPUT DATA SET

Test No.	n	l_d (in.)	d_b (in.)	c_{so} (in.)	c_{si} (in.)	c_b (in.)	b (in.)	h (in.)	d (in.)	f'_c (psi)	f_y (ksi)	f_s* (ksi)	$T_c/f'^{1/2}_c$ ($in.^2$)	$T_c/f'^{1/4}_c$ ($in.^2$)
Chinn(1956)														
D31	1	5.50	0.375	1.470		0.830	3.69	-	-	4700	79.00	60.70	98	810
D36	1	5.50	0.375	1.470		0.560	3.69	-	-	4410	79.00	49.21	82	667
D10	1	7.00	0.750	1.060		1.480	3.62	-	-	4370	57.00	26.41	176	1429
D20	1	7.00	0.750	1.125		1.420	3.75	-	-	4230	57.00	27.12	183	1479
D22	1	7.00	0.750	1.095		0.800	3.69	-	-	4480	57.00	23.97	158	1289
D13	1	11.00	0.750	2.905		1.440	7.31	-	-	4820	57.00	49.14	311	2595
D14	1	11.00	0.750	1.095		0.830	3.69	-	-	4820	57.00	32.82	208	1733
D15	1	11.00	0.750	2.875		0.620	7.25	-	-	4290	57.00	42.45	285	2308
D21	1	11.00	0.750	2.905		1.470	7.31	-	-	4480	57.00	43.53	286	2341
D29	1	11.00	0.750	1.095		1.390	3.69	-	-	7480	57.00	44.62	227	2111
D3	2	11.00	0.750	1.500	0.500	1.500	9.00	-	-	4350	57.00	37.15	248	2013
D32	1	11.00	0.750	2.875		1.470	7.25	-	-	4700	57.00	46.24	297	2457
D38	1	11.00	0.750	1.560		1.520	4.62	-	-	3160	57.00	28.50	223	1673
D39	1	11.00	0.750	1.095		1.560	3.69	-	-	3160	57.00	28.05	220	1646
D5	1	11.00	0.750	2.000		1.500	5.50	-	-	4180	57.00	44.76	305	2449
D6	2	11.00	0.750	1.500	0.625	1.160	7.25	-	-	4340	57.00	33.48	224	1815
D7	1	11.00	0.750	1.060		1.270	3.62	-	-	4450	57.00	34.15	225	1840
D8	2	11.00	0.750	1.500	0.625	1.480	7.25	-	-	4570	57.00	36.28	236	1942
D9	1	11.00	0.750	1.060		1.440	3.62	-	-	4380	57.00	35.33	235	1911
D34	1	12.50	0.750	1.060		1.490	3.62	-	-	3800	57.00	37.46	267	2099
D12	1	16.00	0.750	1.125		1.620	3.75	-	-	4530	57.00	46.37	303	2487
D17	1	16.00	0.750	1.095		0.800	3.69	-	-	3580	57.00	40.56	298	2307
D19**	1	16.00	0.750	2.905		1.700	7.31	-	-	4230	57.00	57.60	390	3142
D23	1	16.00	0.750	1.060		0.780	3.62	-	-	4450	57.00	39.70	262	2139

11F36a	2	49.50	1.410	4.590	4.635	1.500	24.09	18.00	15.79	4570	73.00	64.66	1492	12268
11F36b	2	49.50	1.410	4.590	4.605	1.470	24.03	18.00	15.83	3350	65.00	60.09	1620	12321
11F42a	2	57.75	1.410	4.590	4.590	1.480	24.00	18.00	15.82	3530	65.00	64.57	1695	13067
11F48a**	2	66.00	1.410	4.590	4.620	1.530	24.16	18.03	15.80	3140	73.00	73.91	2058	15402
11F48b**	2	66.00	1.410	4.590	4.665	1.580	24.15	18.22	15.93	3330	65.00	72.24	1953	14835
11R48a	2	66.00	1.410	4.590	4.670	1.500	24.16	18.03	15.83	5620	93.00	82.81	1723	14920
11R48b	2	66.00	1.410	4.590	4.700	2.060	24.22	18.19	15.43	3100	93.00	73.20	2051	15303
11F60a**	2	82.50	1.410	4.590	4.575	1.590	23.97	18.09	15.83	2610	73.00	84.80	2589	18508
11F60b**	2	82.50	1.410	4.590	4.590	1.500	24.00	18.09	15.92	4090	65.00	78.02	1903	15219
11R60a	2	82.50	1.410	4.590	4.590	1.410	24.00	18.12	16.01	2690	93.00	77.19	2322	16720
11R60b	2	82.50	1.410	4.590	4.575	1.750	24.00	18.03	15.58	3460	93.00	90.35	2396	18378
Thompson_etal(1975)														
6-12-4/2/2-6/6	6	12.00	0.750	2.000	2.000	2.000	33.00	13.00	10.63	3730	61.70	57.96	418	3263
8-18-4/3/2-6/6	6	18.00	1.000	2.000	2.000	3.000	36.00	13.00	9.50	4710	59.30	57.00	656	5435
8-18-4/3/2.5-4/6	6	18.00	1.000	2.500	2.000	3.000	36.00	13.00	9.50	2920	59.30	50.86	744	5466
8-24-4/2/2-6/6	6	24.00	1.000	2.000	2.000	2.000	36.00	13.00	10.50	3105	59.30	51.89	736	5491
11-25-6/2/3-5/5	5	25.00	1.410	3.000	3.000	2.000	44.06	13.01	10.30	3920	66.30	45.00	1121	8873
11-30-4/2/2-6/6	6	30.00	1.410	2.000	2.000	2.000	40.88	13.01	10.30	2865	60.50	39.56	1153	8436
11-30-4/2/4-6/6	6	30.00	1.410	4.000	2.000	2.000	44.88	13.01	10.30	3350	63.40	45.90	1237	9413
11-30-4/2/2.7-4/6	4	30.00	1.410	2.700	2.000	2.000	44.88	13.01	10.30	4420	63.30	58.48	1372	11189
11-45-4/1/2-6/6	6	45.00	1.410	2.000	2.000	1.000	40.88	13.01	11.30	3520	60.50	46.72	1228	9462
14-60-4/2/2-5/5	5	60.00	1.693	2.000	2.000	2.000	37.50	16.15	13.30	2865	57.70	48.13	2023	14801
14-60-4/2/4-5/5	5	60.00	1.693	4.000	2.000	2.000	41.50	16.00	13.15	3200	57.70	56.64	2253	16944
Zekany(1981)														
9-53-B-N	5	16.00	1.128	2.000	1.423	2.000	27.25	16.00	13.44	5650	62.80	47.77	636	5510
N-N-80B	4	22.00	1.410	2.000	1.849	2.000	27.25	16.01	13.30	3825	60.10	38.53	972	7642
Choi_etal(1990&1991)														
1.1	2	12.00	0.625	2.000	2.000	1.000	10.50	16.00	14.69	5360	63.80	61.51	260	2229
1.2**	3	12.00	0.625	2.000	2.000	1.000	15.75	16.00	14.69	5360	63.80	64.00	271	2319
2.3	2	12.00	0.750	2.000	2.000	1.000	11.00	16.01	14.63	6010	70.90	51.34	291	2566
2.1	2	12.00	0.750	2.000	2.000	1.000	11.00	16.01	14.63	6010	63.80	45.67	259	2282
3.3	2	16.00	1.000	2.000	2.000	1.500	12.00	16.00	14.00	5980	63.80	43.00	439	3863
3.1	2	16.00	1.000	2.000	2.000	1.500	12.00	14.00	12.00	5980	67.00	42.81	437	3846
4.3	2	24.00	1.410	2.000	2.000	2.000	13.65	16.01	13.30	5850	63.10	37.93	774	6765
4.1	2	24.00	1.410	2.000	2.000	2.000	13.65	16.01	13.30	5850	64.60	40.37	823	7201
Hester_etal(1991&1993)														
1.1	3	16.00	1.000	2.000	1.500	2.000	16.00	16.00	13.50	5990	63.80	50.13	512	4501
2.1	3	16.00	1.000	2.000	1.500	1.840	16.00	16.33	13.99	6200	69.00	46.25	464	4118
3.1	3	16.00	1.000	2.000	1.500	2.040	16.09	16.23	13.69	6020	71.10	46.86	477	4202
4.1	3	16.00	1.000	2.000	1.500	2.100	16.08	16.22	13.62	6450	71.10	42.36	417	3734
5.1	3	16.00	1.000	2.000	1.500	2.050	16.09	16.27	13.72	5490	69.00	39.86	425	3658
6.1	3	22.75	1.000	2.000	1.500	2.150	16.06	16.19	13.54	5850	69.00	51.99	537	4696
7.1	2	16.00	1.000	2.000	4.000	2.120	16.03	16.20	13.58	5240	69.00	45.40	495	4215

16	2	40.00	1.410	3.016	2.969	1.895	18.07	16.28	13.64	5180	77.77	52.75	1143	9700

B.2 EVALUATION DATA SET

Test No.	n	l_d (in.)	d_b (in.)	c_{so} (in.)	c_{si} (in.)	c_b (in.)	b (in.)	h (in.)	d (in.)	f'_c (psi)	f_y (ksi)	f_s* (ksi)	$T_c/f'^{1/2}_c$ (in.^{2})	$T_c/f'^{1/4}_c$ (in.^{2})
Zuo (1998)														
25.1**	3	16.50	0.625	1.985	1.023	1.556	12.19	16.27	14.37	4490	62.98	63.72	295	2413
19.1	3	36.00	1.000	1.953	1.930	1.961	18.14	16.16	13.66	4250	80.57	73.51	891	7193
19.2	3	36.00	1.000	2.016	1.883	1.929	18.06	16.13	13.66	4250	80.57	67.85	822	6639
20.6	3	40.00	1.000	1.516	0.672	1.300	12.08	15.60	13.76	5080	80.57	57.15	633	5348
23a.5	2	22.00	1.000	2.000	1.891	1.938	18.19	16.16	13.63	9320	80.57	62.24	509	5005
23a.6	2	29.00	1.000	2.031	1.875	1.919	12.24	16.11	13.67	9320	80.57	75.47	618	6068
23b.3	2	19.50	1.000	3.031	3.859	3.057	18.23	16.32	12.72	8370	80.57	71.64	619	5917
24.1	2	32.00	1.000	2.000	1.875	1.903	12.14	16.12	13.69	4300	79.70	61.91	746	6040
26.3	3	40.00	1.000	1.547	0.652	1.889	12.11	16.19	13.78	4960	79.70	62.52	701	5885
26.5	3	40.00	1.000	1.500	0.684	1.891	12.15	16.17	13.75	4960	77.96	64.36	722	6058
31.5	3	22.00	1.000	1.828	0.508	1.494	12.26	15.58	13.56	12890	79.70	61.43	427	4555
31.6	3	22.00	1.000	1.719	0.539	1.492	12.17	15.49	13.44	12890	69.50	63.42	441	4702
34.1	3	24.00	1.000	2.063	1.938	1.941	18.13	16.12	13.66	5440	79.70	57.88	620	5324
34.2	3	24.00	1.000	2.070	1.945	1.918	18.17	16.05	13.61	5440	79.70	61.97	664	5701
34.3	3	24.00	1.000	2.080	1.844	1.981	18.12	16.02	13.49	5440	69.50	58.94	631	5422
34.4	3	24.00	1.000	2.045	1.883	1.936	18.21	16.02	13.53	5440	69.50	58.49	626	5380
36.3	3	26.00	1.000	2.016	1.836	2.000	18.17	16.10	13.55	5060	69.50	62.78	697	5881
36.4	3	26.00	1.000	2.031	1.828	1.988	18.14	16.10	13.56	5060	69.50	60.17	668	5636
38.1	3	26.00	1.000	1.938	1.953	1.802	18.25	16.10	13.75	5080	69.50	53.96	598	5049
38.2	3	26.00	1.000	2.125	1.844	2.075	18.17	16.14	13.51	5080	69.50	60.30	668	5643
39.6	3	21.00	1.000	1.953	0.516	1.505	12.19	15.41	13.59	14450	67.69	67.38	443	4855
40.5	2	17.00	1.000	2.000	1.875	1.846	12.11	16.04	13.67	15650	77.96	65.81	416	4649
28.5	2	30.00	1.410	1.977	4.031	1.999	18.09	16.20	13.45	12610	77.77	50.89	707	7492
30.5	2	30.00	1.410	2.063	4.016	1.956	18.12	16.15	13.44	13220	77.77	66.95	908	9740
32.1	2	32.00	1.410	2.000	0.984	1.904	12.17	16.17	13.52	14400	77.77	63.33	823	9019
32.2	2	32.00	1.410	2.000	1.063	1.916	12.14	16.16	13.51	14400	66.69	61.49	799	8757
32.3	2	32.00	1.410	1.969	4.016	1.947	18.14	16.15	13.45	14400	77.77	60.64	788	8635
32.4	2	28.00	1.410	2.031	4.047	1.935	18.20	16.17	13.50	14400	66.69	61.01	793	8688

39	-40	20.3	2.18	0.086	0.08	0.108	0.1	0.886	34.88
24	-40	20.3	1.73	0.068	0.08	0.085	0.1	2.411	94.92
5	-40	20.3	1.98	0.078	0.08	0.098	0.1	2.17	85.43
13	-68.9	20.3	1.37	0.054	0.08	0.068	0.1	0.358	14.09
19	-68.9	20.3	2.06	0.081	0.08	0.101	0.1	0.525	20.67
22	-68.9	20.3	1.91	0.075	0.08	0.094	0.1	0.617	24.29
2	-103.9	20.3	1.96	0.077	0.08	0.096	0.1	0.251	9.88
21	-103.9	20.3	2.24	0.088	0.08	0.11	0.1	0.251	9.88
28	-103.9	20.3	1.91	0.075	0.08	0.094	0.1	0.213	8.39
29	-103.9	20.3	1.91	0.075	0.08	0.094	0.1	0.523	20.59
26	-195.6	20.3	2.24	0.088	0.08	0.11	0.1	0.005	0.2
3	-195.6	20.3	2.69	0.106	0.08	0.133	0.1	0.005	0.2
65	21.1	31.8	10.44	0.411	0.4	0.329	0.32	0.708	27.87
66	21.1	31.8	10.36	0.408	0.4	0.326	0.32	1.368	53.86
69	21.1	31.8	9.88	0.389	0.4	0.311	0.32	1.31	51.57
72	21.1	31.8	10.11	0.398	0.4	0.318	0.32	1.173	46.18
67	-5.6	31.8	10.18	0.401	0.4	0.321	0.32	0.368	14.49
68	-5.6	31.8	10.24	0.403	0.4	0.322	0.32	0.632	24.88
71	-5.6	31.8	9.8	0.386	0.4	0.309	0.32	0.383	15.08
Fracture mechanics									
3	-36	100	10	0.394	0.4	0.1	0.1	0.586	23.07
7	-59	94	10.2	0.402	0.4	0.109	0.1	0.137	5.39
8	-60	94	9.6	0.378	0.4	0.102	0.1	0.476	18.74
9	-62	94	9.5	0.374	0.4	0.101	0.1	0.352	13.86
10	-60	94	14	0.551	0.4	0.149	0.1	0.235	9.25
11	-57	94	8.4	0.331	0.4	0.089	0.1	0.196	7.72
13	-60	94	8.8	0.346	0.4	0.094	0.1	0.357	14.06
14	-60	93	8.7	0.343	0.4	0.094	0.1	0.346	13.62
15	-59	94	8.7	0.343	0.4	0.093	0.1	0.146	5.75
18	-24	102	10.6	0.417	0.4	0.104	0.1	0.468	18.43
20	-4	101	10.8	0.425	0.4	0.107	0.1	1.733	68.23
21	-23	102	10.7	0.421	0.4	0.105	0.1	0.306	12.05
22	-7	102	10.9	0.429	0.4	0.107	0.1	0.942	37.09
26	-40	102	11	0.433	0.4	0.108	0.1	0.355	13.98
27	-22	102	10.7	0.421	0.4	0.105	0.1	0.559	22.01
28	-6	102	10.3	0.406	0.4	0.101	0.1	1.242	48.9
32	-103	102	11.1	0.437	0.4	0.109	0.1	0.016	0.63
33	-103	102	10.7	0.421	0.4	0.105	0.1	0.009	0.35
34	-106	102	10.4	0.409	0.4	0.102	0.1	0.017	0.67
37	-39	102	10.8	0.425	0.4	0.106	0.1	0.263	10.35
38	-39	102	10.8	0.425	0.4	0.106	0.1	0.206	8.11

APPENDIX D

THE DATA SET USED IN SECTION 5.4

solubility ($x10^5$M)	pKa1	compound	particle_size	medium pH	flowrate (ml/min)	flux ($x10^{10}$mol/min.cm^2)
13.7	4.02	acidic	120	2	5.1	24.38
13.7	4.02	acidic	120	3	5.1	28.73
13.7	4.02	acidic	120	4	5.1	45.27
13.7	4.02	acidic	120	6	5.1	87.06
13.7	4.02	acidic	120	7	5.1	95.77
13.7	4.02	acidic	120	8	5.1	104.47
13.7	4.02	acidic	120	9	5.1	128.85
13.7	4.02	acidic	120	10	5.1	430.95
2150	4.03	acidic	120	2	1.1	2989.52
2150	4.03	acidic	120	2	3.48	4442.16
2150	4.03	acidic	120	2	5.1	5356.48
2150	4.03	acidic	120	2	8.49	5785.91
2150	4.03	acidic	120	2	13.41	6969.39
2150	4.03	acidic	120	2	5.1	5356.78
2150	4.03	acidic	120	4	5.1	5356.78
2150	4.03	acidic	120	7	5.1	5503.54
2150	4.03	acidic	120	9	5.1	5503.54
2150	4.03	acidic	120	10.6	5.1	6237.34
2150	4.03	acidic	120	11.2	5.1	7117.91
2150	4.03	acidic	120	12	5.1	11887.64
2150	4.03	acidic	120	12.2	5.1	19959.49
2150	4.03	acidic	120	12.5	5.1	33167.98
13.7	4.57	acidic	137	2	5	21.43
13.7	4.57	acidic	137	2	5	23.6
13.7	4.57	acidic	37	7	5	144.8
13.7	4.57	acidic	137	2	5	22.29
13.7	4.57	acidic	37	7	5	138.3
13.7	4.57	acidic	137	7	5	133.5
13.7	4.57	acidic	111.5	7	5	138.3

APPENDIX E

THE DATA SET USED IN SECTION 5.5

mpg	c	d	h	w	a	y	o	Car Name
18	8	307	130	3504	12	70	1	chevrolet chevelle malibu
15	8	350	165	3693	11.5	70	1	buick skylark 320
18	8	318	150	3436	11	70	1	plymouth satellite
16	8	304	150	3433	12	70	1	amc rebel sst
17	8	302	140	3449	10.5	70	1	ford torino
15	8	429	198	4341	10	70	1	ford galaxie 500
14	8	454	220	4354	9	70	1	chevrolet impala
14	8	440	215	4312	8.5	70	1	plymouth fury iii
14	8	455	225	4425	10	70	1	pontiac catalina
15	8	390	190	3850	8.5	70	1	amc ambassador dpl
15	8	383	170	3563	10	70	1	dodge challenger se
14	8	340	160	3609	8	70	1	plymouth 'cuda 340
15	8	400	150	3761	9.5	70	1	chevrolet monte carlo
14	8	455	225	3086	10	70	1	buick estate wagon (sw)
24	4	113	95	2372	15	70	3	toyota corona mark ii
22	6	198	95	2833	15.5	70	1	plymouth duster
18	6	199	97	2774	15.5	70	1	amc hornet
21	6	200	85	2587	16	70	1	ford maverick
27	4	97	88	2130	14.5	70	3	datsun pl510
26	4	97	46	1835	20.5	70	2	volkswagen 1131 deluxe sedan
25	4	110	87	2672	17.5	70	2	peugeot 504
24	4	107	90	2430	14.5	70	2	audi 100 ls
25	4	104	95	2375	17.5	70	2	saab 99e
26	4	121	113	2234	12.5	70	2	bmw 2002
21	6	199	90	2648	15	70	1	amc gremlin
10	8	360	215	4615	14	70	1	ford f250
10	8	307	200	4376	15	70	1	chevy c20
11	8	318	210	4382	13.5	70	1	dodge d200
9	8	304	193	4732	18.5	70	1	hi 1200d

15	8	304	150	3892	12.5	72	1	amc matador (sw)
13	8	307	130	4098	14	72	1	chevrolet chevelle concours (sw)
13	8	302	140	4294	16	72	1	ford gran torino (sw)
14	8	318	150	4077	14	72	1	plymouth satellite custom (sw)
18	4	121	112	2933	14.5	72	2	volvo 145e (sw)
22	4	121	76	2511	18	72	2	volkswagen 411 (sw)
21	4	120	87	2979	19.5	72	2	peugeot 504 (sw)
26	4	96	69	2189	18	72	2	renault 12 (sw)
22	4	122	86	2395	16	72	1	ford pinto (sw)
28	4	97	92	2288	17	72	3	datsun 510 (sw)
23	4	120	97	2506	14.5	72	3	toyouta corona mark ii (sw)
28	4	98	80	2164	15	72	1	dodge colt (sw)
27	4	97	88	2100	16.5	72	3	toyota corolla 1600 (sw)
13	8	350	175	4100	13	73	1	buick century 350
14	8	304	150	3672	11.5	73	1	amc matador
13	8	350	145	3988	13	73	1	chevrolet malibu
14	8	302	137	4042	14.5	73	1	ford gran torino
15	8	318	150	3777	12.5	73	1	dodge coronet custom
12	8	429	198	4952	11.5	73	1	mercury marquis brougham
13	8	400	150	4464	12	73	1	chevrolet caprice classic
13	8	351	158	4363	13	73	1	ford ltd
14	8	318	150	4237	14.5	73	1	plymouth fury gran sedan
13	8	440	215	4735	11	73	1	chrysler new yorker brougham
12	8	455	225	4951	11	73	1	buick electra 225 custom
13	8	360	175	3821	11	73	1	amc ambassador brougham
18	6	225	105	3121	16.5	73	1	plymouth valiant
16	6	250	100	3278	18	73	1	chevrolet nova custom
18	6	232	100	2945	16	73	1	amc hornet
18	6	250	88	3021	16.5	73	1	ford maverick
23	6	198	95	2904	16	73	1	plymouth duster
26	4	97	46	1950	21	73	2	volkswagen super beetle
11	8	400	150	4997	14	73	1	chevrolet impala
12	8	400	167	4906	12.5	73	1	ford country
13	8	360	170	4654	13	73	1	plymouth custom suburb
12	8	350	180	4499	12.5	73	1	oldsmobile vista cruiser
18	6	232	100	2789	15	73	1	amc gremlin
20	4	97	88	2279	19	73	3	toyota carina
21	4	140	72	2401	19.5	73	1	chevrolet vega
22	4	108	94	2379	16.5	73	3	datsun 610
18	3	70	90	2124	13.5	73	3	maxda rx3
19	4	122	85	2310	18.5	73	1	ford pinto
21	6	155	107	2472	14	73	1	mercury capri v6
26	4	98	90	2265	15.5	73	2	fiat 124 sport coupe

16	8	318	150	4498	14.5	75	1	plymouth grand fury
14	8	351	148	4657	13.5	75	1	ford ltd
17	6	231	110	3907	21	75	1	buick century
16	6	250	105	3897	18.5	75	1	chevroelt chevelle malibu
15	6	258	110	3730	19	75	1	amc matador
18	6	225	95	3785	19	75	1	plymouth fury
21	6	231	110	3039	15	75	1	buick skyhawk
20	8	262	110	3221	13.5	75	1	chevrolet monza 2+2
13	8	302	129	3169	12	75	1	ford mustang ii
29	4	97	75	2171	16	75	3	toyota corolla
23	4	140	83	2639	17	75	1	ford pinto
20	6	232	100	2914	16	75	1	amc gremlin
23	4	140	78	2592	18.5	75	1	pontiac astro
24	4	134	96	2702	13.5	75	3	toyota corona
25	4	90	71	2223	16.5	75	2	volkswagen dasher
24	4	119	97	2545	17	75	3	datsun 710
18	6	171	97	2984	14.5	75	1	ford pinto
29	4	90	70	1937	14	75	2	volkswagen rabbit
19	6	232	90	3211	17	75	1	amc pacer
23	4	115	95	2694	15	75	2	audi 100ls
23	4	120	88	2957	17	75	2	peugeot 504
22	4	121	98	2945	14.5	75	2	volvo 244dl
25	4	121	115	2671	13.5	75	2	saab 99le
33	4	91	53	1795	17.5	75	3	honda civic cvcc
28	4	107	86	2464	15.5	76	2	fiat 131
25	4	116	81	2220	16.9	76	2	opel 1900
25	4	140	92	2572	14.9	76	1	capri ii
26	4	98	79	2255	17.7	76	1	dodge colt
27	4	101	83	2202	15.3	76	2	renault 12tl
17.5	8	305	140	4215	13	76	1	chevrolet chevelle malibu classic
16	8	318	150	4190	13	76	1	dodge coronet brougham
15.5	8	304	120	3962	13.9	76	1	amc matador
14.5	8	351	152	4215	12.8	76	1	ford gran torino
22	6	225	100	3233	15.4	76	1	plymouth valiant
22	6	250	105	3353	14.5	76	1	chevrolet nova
24	6	200	81	3012	17.6	76	1	ford maverick
22.5	6	232	90	3085	17.6	76	1	amc hornet
29	4	85	52	2035	22.2	76	1	chevrolet chevette
24.5	4	98	60	2164	22.1	76	1	chevrolet woody
29	4	90	70	1937	14.2	76	2	vw rabbit
33	4	91	53	1795	17.4	76	3	honda civic
20	6	225	100	3651	17.7	76	1	dodge aspen se
18	6	250	78	3574	21	76	1	ford granada ghia

43.1	4	90	48	1985	21.5	78	2	volkswagen rabbit custom diesel
36.1	4	98	66	1800	14.4	78	1	ford fiesta
32.8	4	78	52	1985	19.4	78	3	mazda glc deluxe
39.4	4	85	70	2070	18.6	78	3	datsun b210 gx
36.1	4	91	60	1800	16.4	78	3	honda civic cvcc
19.9	8	260	110	3365	15.5	78	1	oldsmobile cutlass salon brougham
19.4	8	318	140	3735	13.2	78	1	dodge diplomat
20.2	8	302	139	3570	12.8	78	1	mercury monarch ghia
19.2	6	231	105	3535	19.2	78	1	pontiac phoenix lj
20.5	6	200	95	3155	18.2	78	1	chevrolet malibu
20.2	6	200	85	2965	15.8	78	1	ford fairmont (auto)
25.1	4	140	88	2720	15.4	78	1	ford fairmont (man)
20.5	6	225	100	3430	17.2	78	1	plymouth volare
19.4	6	232	90	3210	17.2	78	1	amc concord
20.6	6	231	105	3380	15.8	78	1	buick century special
20.8	6	200	85	3070	16.7	78	1	mercury zephyr
18.6	6	225	110	3620	18.7	78	1	dodge aspen
18.1	6	258	120	3410	15.1	78	1	amc concord d/l
19.2	8	305	145	3425	13.2	78	1	chevrolet monte carlo landau
17.7	6	231	165	3445	13.4	78	1	buick regal sport coupe (turbo)
18.1	8	302	139	3205	11.2	78	1	ford futura
17.5	8	318	140	4080	13.7	78	1	dodge magnum xe
30	4	98	68	2155	16.5	78	1	chevrolet chevette
27.5	4	134	95	2560	14.2	78	3	toyota corona
27.2	4	119	97	2300	14.7	78	3	datsun 510
30.9	4	105	75	2230	14.5	78	1	dodge omni
21.1	4	134	95	2515	14.8	78	3	toyota celica gt liftback
23.2	4	156	105	2745	16.7	78	1	plymouth sapporo
23.8	4	151	85	2855	17.6	78	1	oldsmobile starfire sx
23.9	4	119	97	2405	14.9	78	3	datsun 200-sx
20.3	5	131	103	2830	15.9	78	2	audi 5000
17	6	163	125	3140	13.6	78	2	volvo 264gl
21.6	4	121	115	2795	15.7	78	2	saab 99gle
16.2	6	163	133	3410	15.8	78	2	peugeot 604sl
31.5	4	89	71	1990	14.9	78	2	volkswagen scirocco
29.5	4	98	68	2135	16.6	78	3	honda accord lx
21.5	6	231	115	3245	15.4	79	1	pontiac lemans v6
19.8	6	200	85	2990	18.2	79	1	mercury zephyr 6
22.3	4	140	88	2890	17.3	79	1	ford fairmont 4
20.2	6	232	90	3265	18.2	79	1	amc concord dl 6
20.6	6	225	110	3360	16.6	79	1	dodge aspen 6
17	8	305	130	3840	15.4	79	1	chevrolet caprice classic
17.6	8	302	129	3725	13.4	79	1	ford ltd landau

40.9	4	85	?	1835	17.3	80	2	renault lecar deluxe
33.8	4	97	67	2145	18	80	3	subaru dl
29.8	4	89	62	1845	15.3	80	2	vokswagen rabbit
32.7	6	168	132	2910	11.4	80	3	datsun 280-zx
23.7	3	70	100	2420	12.5	80	3	mazda rx-7 gs
35	4	122	88	2500	15.1	80	2	triumph tr7 coupe
23.6	4	140	?	2905	14.3	80	1	ford mustang cobra
32.4	4	107	72	2290	17	80	3	honda accord
27.2	4	135	84	2490	15.7	81	1	plymouth reliant
26.6	4	151	84	2635	16.4	81	1	buick skylark
25.8	4	156	92	2620	14.4	81	1	dodge aries wagon (sw)
23.5	6	173	110	2725	12.6	81	1	chevrolet citation
30	4	135	84	2385	12.9	81	1	plymouth reliant
39.1	4	79	58	1755	16.9	81	3	toyota starlet
39	4	86	64	1875	16.4	81	1	plymouth champ
35.1	4	81	60	1760	16.1	81	3	honda civic 1300
32.3	4	97	67	2065	17.8	81	3	subaru
37	4	85	65	1975	19.4	81	3	datsun 210 mpg
37.7	4	89	62	2050	17.3	81	3	toyota tercel
34.1	4	91	68	1985	16	81	3	mazda glc 4
34.7	4	105	63	2215	14.9	81	1	plymouth horizon 4
34.4	4	98	65	2045	16.2	81	1	ford escort 4w
29.9	4	98	65	2380	20.7	81	1	ford escort 2h
33	4	105	74	2190	14.2	81	2	volkswagen jetta
34.5	4	100	?	2320	15.8	81	2	renault 18i
33.7	4	107	75	2210	14.4	81	3	honda prelude
32.4	4	108	75	2350	16.8	81	3	toyota corolla
32.9	4	119	100	2615	14.8	81	3	datsun 200sx
31.6	4	120	74	2635	18.3	81	3	mazda 626
28.1	4	141	80	3230	20.4	81	2	peugeot 505s turbo diesel
30.7	6	145	76	3160	19.6	81	2	volvo diesel
25.4	6	168	116	2900	12.6	81	3	toyota cressida
24.2	6	146	120	2930	13.8	81	3	datsun 810 maxima
22.4	6	231	110	3415	15.8	81	1	buick century
26.6	8	350	105	3725	19	81	1	oldsmobile cutlass ls
20.2	6	200	88	3060	17.1	81	1	ford granada gl
17.6	6	225	85	3465	16.6	81	1	chrysler lebaron salon
28	4	112	88	2605	19.6	82	1	chevrolet cavalier
27	4	112	88	2640	18.6	82	1	chevrolet cavalier wagon
34	4	112	88	2395	18	82	1	chevrolet cavalier 2-door
31	4	112	85	2575	16.2	82	1	pontiac j2000 se hatchback
29	4	135	84	2525	16	82	1	dodge aries se
27	4	151	90	2735	18	82	1	pontiac phoenix