AN ABSTRACT OF THE THESIS OF

MARTIN OSBORNE (Name) for the degree DOCTOR OF PHILOSOPHY (Degree)

in COMPUTER SCIENCE (Major Department) presented on November 19, 1974 (Date)

Title: A MODIFICATION OF VETO LOGIC FOR A COMMITTEE OF THRESHOLD LOGIC UNITS AND THE USE OF 2-CLASS CLASSIFIERS FOR FUNCTION ESTIMATION

Abstract approved: Redacted for privacy
Emilio Gagliardo

The well-known local adjustment algorithm for training a threshold logic unit, TLU, is extended to a local adjustment algorithm for training a network of TLUs. Computer simulations show that the extension is unsatisfactory.

A new logic for a committee of TLUs, called modified veto logic, and a local adjustment algorithm for training a modified veto committee are described. Unlike a majority committee, a modified veto committee may have members added during training, and the modified veto committee is free to attain a size needed to solve a problem. Computer simulations show that a modified veto committee can solve difficult pattern recognition problems and, in the instances tested, does so more successfully than a majority committee.

A technique for using a number of 2-class classifiers to perform function estimation is described. The 2-class classifiers are trained on a set of ordered pairs belonging to the function being estimated; no information about the form of the function is needed; the function can have any number of independent variables; and the accuracy of the estimate increases with the number of 2-class classifiers used. Computer simulations on artificially generated data and on "real life" data show that this technique provides accurate estimates of functions.

It is shown that replacing non-binary variables by binary variables can greatly increase the recognition rate of a TLU.

A Modification of Veto Logic for a Committee of Threshold Logic Units and the Use of 2-Class Classifiers for Function Estimation

by

Martin Osborne

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Doctor of Philosophy

Completed November 19, 1974

Commencement June 1975

APPROVED:

Redacted for privacy

Professor of Computer Science

in charge of major

Redacted for privacy

Head of Department of Computer Science

Redacted for privacy

Dean of Graduate School

Date thesis is presented November 19, 1974

Typed by Clover Redfern for Martin Osborne

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

A MODIFICATION OF VETO LOGIC FOR A COMMITTEE OF THRESHOLD LOGIC UNITS AND THE USE OF 2-CLASS CLASSIFIERS FOR FUNCTION ESTIMATION

I. INTRODUCTION

In the late 50's and early 60's many people did research in automatic pattern recognition, mainly investigating methods for building pattern classifiers from combinations of threshold logic units, TLUs, and designing iterative algorithms for training the classifiers (see, for instance, the bibliography at the end of [7]). The most notable success was the discovery of a convergent iterative algorithm for training a single TLU to recognize all the patterns in a linearly separable problem. The classifiers built from combinations of TLUs were able to solve more complex problems than could be solved by a single TLU, but the algorithms for training the classifiers--for the most part extensions of the iterative algorithm for training a single TLU--were not proven to converge to a solution when a solution existed. Nilsson [24] gives a fairly complete and clear summary of this research.

Some research is still being done on the design of classifiers from TLUs, leading occasionally to new techniques but more usually to the modification or extension of known techniques. Kaminuma and Watanabe [15], for instance, have modified and improved the iterative algorithm for training a TLU; Chang [6] has generalized the structure

of piecewise linear discriminant functions; and Mueller [23] has improved Ridgway's [26] algorithm for training a committee of TLUs. Extensions of the local adjustment algorithm for training a single TLU to local adjustment algorithms for training a network of TLUs have been given only under restrictive conditions. Francalangia [9] trains networks at only a single level, and Holderman [12] needs to use a global criterion to determine the adjustments to be made in response to a misclassified pattern.

A single TLU and a network of TLUs are suitable for solving 2-class problems; and piecewise linear discriminant functions, for solving many-class problems. By application of the theory of paired comparisons, the use of 2-class classifiers has been extended to include the solution of many class problems, but the use of classifiers does not seem to have been extended to include the solution of problems of function estimation. There are, of course, many techniques for function estimation, the most well-known being linear and non-linear regression; all are unrelated to classification techniques.

Investigators have noted that the inputs to many pattern recognition problems are binary [10] and that replacing non-binary variables by binary variables can simultaneously increase recognition rates and decrease storage requirements [30]. Replacing non-binary by binary variables can greatly increase the number of variables needed to describe a pattern, but the number can be decreased by

using a method of feature selection, for instance that described in [22]. Apparently no examples have been presented in the literature in which the introduction of binary variables has enormously improved the recognition rate of a classifier.

II. CLASSIFICATION

2.1 The Problem of Classification

A doctor learns to diagnose the diseases of his patients; a meteorologist learns to predict tomorrow's weather; we all learn to recognize the letters of the alphabet, even when written by different people in different scripts. These are examples of what is called pattern recognition, and we would like to imitate the recognition process on a general purpose digital computer.

If we wish to imitate a doctor, we might begin by describing each patient as a vector in R^n. The first component of the vector could be the number of heart beats per minute; the second, the weight in pounds; the third, the temperature in degrees Fahrenheit; etc. We collect the vectors of a large number of patients, place the vectors in classes according to the diseases they represent, and index the classes from one to the number of diseases p. Next we define a family of functions, called classifiers, from R^n to $\{1, \ldots, p\}$. If $X \in R^n$ is a vector representing a patient with the i-th disease and if f is one of the classifiers, then we say that f classifies X correctly if $f(X) = i$. We try to write an algorithm to search the family of classifiers for one that classifies a large percentage of vectors correctly. Usually the family is infinite, the search is not

exhaustive, and there is no guarantee that we will find the best classifier.

We use the classifier we find to diagnose new patients. But if the percentage classified correctly is not sufficiently high, we can re-examine the imitation, looking for ways to improve it. Perhaps our description of a patient included misleading information, contained many highly correlated pieces of information which then assumed a disproportionate importance, or lacked necessary information. Perhaps we encoded the information poorly, made the scale of some components too large and other too small. Perhaps we tried to learn from a collection of vectors that was too small or did not include representative examples of all the diseases likely to be encountered in patients. Perhaps the family of classifiers included no members capable of classifying a large percentage of the vectors correctly, or the search algorithm did not find the good classifiers that were present.

We should realize that there is no single automatic technique for solving all pattern recognition problems, that some problems are trivially easy and others impossibly difficult. In this chapter we restrict our attention to pattern recognition problems in which there are two classes and use what is called a local adjustment algorithm to search families of classifiers called networks of threshold logic units. If we are able to solve 2-class problems, we can use a number of

2-class classifiers simultaneously to solve an N-class problem. The best way to use a number of 2-class classifiers to solve an N-class problem is treated extensively in the statistical theory of paired comparisons.

2.2 Preliminaries

We begin by introducing some terminology.

2.2.1 Pattern, sample. A pattern, also called a sample, is a point in R^n. A set of patterns will be denoted by the letter S.

2.2.2 Classifiers. A classifier is a function whose domain is S and whose range is a finite subset of the integers.

2.2.3 Response. The value of a classifier at a pattern is called the classifier's response to the pattern.

2.2.4 Class. If $C_1, C_2, \ldots, C_p$ is a partition of S, i.e.,

$$S = \bigcup_{i=1}^{p} C_i \quad \text{and} \quad C_i \cap C_j = \phi \quad \text{if} \quad i \neq j,$$

then C_i is called the i-th class. The partition defines a function on S called class, where

$$\text{class}(X) = i \quad \text{if} \quad X \in C_i .$$

When the partition consists of two classes, we denote the classes by A and B and let

$$\text{class}(X) = \begin{cases} -1 & \text{if } X \in A \\ +1 & \text{if } X \in B . \end{cases}$$

2.2.5 Correct, incorrect. The response of a classifier f to a pattern X is correct if

$$f(X) = \text{class}(X),$$

otherwise it is incorrect.

2.2.6 Recognize, misclassify. A classifier recognizes a pattern if the response is correct, otherwise it misclassifies the pattern.

2.2.7 Recognition rate, error rate. The percentage of patterns recognized is called the recognition rate, the percentage misclassified the error rate.

2.2.8 Discriminant function. A discriminant function f is a function whose domain is S and whose range is the real numbers R. If S is partitioned into p classes and if $d_1, d_2, \ldots, d_p$ are discriminant functions, then for $p > 2$, a classifier f can be defined by

$$f(X) = \max\{i : d_i(X) \geq d_j(X) \text{ for all } j \text{ with } 1 \leq i, j \leq p\}.$$

If $p = 2$, then f can be defined by

$$f(X) = \text{sgn}(d_2(X)-d_1(X)),$$

where

$$\text{sgn}(X) = \begin{cases} -1 & \text{if} \quad x < 0 \\ +1 & \text{if} \quad x \geq 0 \end{cases}.$$

As mentioned in Section 2.1 to imitate human pattern recognition we need an algorithm to search the family of proposed classifiers for a classifier with a high recognition rate. We now give a formal definition of a local adjustment search algorithm and of some related concepts.

2.2.9 Training sequence, training set. Let T be a finite subset of S. A training sequence is a sequence

$$T_X = \{(X_i, \text{class}(X_i))\}_{i=0}^{\infty},$$

where each X_i is in T and each pattern in T appears infinitely many times as a first component in the training sequence. T is called a training set.

2.2.10 Local adjustment algorithm. Let F be a family of classifiers defined on S, let $f \in F$, and let $(X_i, \text{class}(X_i))$ be a term of some training sequence. A local adjustment search algorithm, or local adjustment algorithm for short, is a function h of the form

$$h(f, X_i, \text{class}(X_i)) = (f', X_{i+1}, \text{class}(X_{i+1})), \qquad (2.2.1)$$

where $f' \in F$ and $(X_{i+1}, \text{class}(X_{i+1}))$ is the next term in the training sequence.

Typically, the classifiers in the set F differ from each other only in the values of some parameters. If P_f and $P_{f'}$ denote the tuples whose components are the parameters defining the classifiers f and f' respectively, then a common strategy used in designing the local adjustment algorithm of line (2.2.1) is to let

$$P_f' = \begin{cases} P_f & \text{if } f(X_i) = \text{class}(X_i) \\ P_f + \Delta(f, X_i, \text{class}(X_i)) & \text{if } f(X_i) \neq \text{class}(X_i) , \end{cases} \qquad (2.2.2)$$

where $\Delta(f, X_i, \text{class}(X_i))$ is a tuple and is a function of f, X_i and $\text{class}(X_i)$. In other words, f' and f are equal if f classifies X_i correctly, otherwise f' is obtained from f by incrementing the parameters of f. The tuple $\Delta(f, X_i, \text{class}(X_i))$ is chosen so that f' either recognizes X_i or will do so if the norm of $\Delta(f, X_i, \text{class}(X_i))$ is increased sufficiently.

2.2.11 Local adjustment. The tuple $\Delta(f, X_i, \text{class}(X_i))$ is called a local adjustment to f due to X_i.

2.2.12 Classifier search sequence. Let T_X be a training sequence, $f_0 \in F$, and h be a local adjustment algorithm. The sequence

$$\text{CSS}(T_X, f_0) = \{f_i : (f_i, X_i, \text{class}(X_i)) = h(f_{i-1}, X_{i-1}, \text{class}(X_{i-1}))\}_{i=1}^{\infty}$$

is called a classifier search sequence.

2.2.13 Effective local adjustment algorithm. A local adjustment algorithm is effective if for every pair (T_X, f_0) the classifier search sequence $CSS(T_X, f_0)$ converges in a finite number of steps to a classifier recognizing all the patterns in T, provided such a classifier exists. (We say a sequence converges in a finite number of steps if for some N all terms after the N-th are equal.)

2.2.14 Minimal local adjustment algorithm. A local adjustment algorithm is minimal if for every pair (T_X, f_0), where T is a singleton set, $CSS(T_X, f_0)$ converges in a finite number of steps to a classifier recognizing the single pattern in T, provided such a classifier exists.

When the local adjustment algorithm is defined by the equation on line (2.2.2), it is minimal if and only if for every pair (X, f_0), where X is a pattern and f_0 misclassifies X, a finite number of successive local adjustments to f_0 due to X results in a classifier recognizing X, provided such a classifier exists. As successive adjustments due to X are not necessarily identical, the effect of m successive local adjustments due to X is not necessarily the same as a single local adjustment due to X multiplied by m.

A heuristic consideration supporting the use of local adjustment algorithms is that adjusting a classifier, i.e., generating a new classifier, in response to the demands of one pattern at a time results in a classifier that satisfies the demands of a large number of patterns simultaneously, where the demand of a pattern is that it be classified correctly. In general it is very hard to determine whether a local adjustment algorithm is effective; and for most of the numerous local adjustment algorithms presented in the literature, it is not known if they are effective, or it is known that they are not. It is considerably easier to determine if a local adjustment algorithm is minimal; most appear to be so, and we will show that the local adjustment algorithms discussed in this chapter are minimal.

There are several rather similar effective local adjustment algorithms for searching a family of classifiers called threshold logic units, see Section 2.3 . When it is not known whether a local adjustment algorithm is effective, its merits can be demonstrated by simulating it on a digital computer. Even if a local adjustment algorithm is effective, computer simulations can give information on how quickly it converges and on how it performs when no classifier in F is able to recognize all the training patterns. Before the simulation begins, a training sequence and initial classifier are defined. The computer then generates some initial portion of the classifier search sequence and computes the error rate on the training set for some of the

classifiers generated. The error rate on subsets of S other than the training set can also be computed. These other sets are called test sets. Hopefully, the error rate decreases as the index of the classifiers in the search sequence increases.

2.2.15 Capacity, solution. A family of classifiers F has the capacity to recognize a training set T if some classifier in F recognizes all the patterns in T. Such a classifier is called a solution.

2.3 Threshold Logic Units

If a set of patterns S is a subset of R^n, then every hyperplane in R^n defines a 2-class classifier on S as follows: assign all the patterns on one side of the hyperplane to the first class and the remaining patterns to the second class. Analytically, the classifier defined by a hyperplane can be described by the function

$$f(X) = \mathrm{sgn}(X \cdot W - \theta),$$

where X is a pattern, W is in R^n, θ is in R and the hyperplane is defined by the equation $X \cdot W = \theta$. When the components of each pattern are binary, such a classifier is easily realized by a hardware device called a threshold logic unit, abbreviated TLU, see Figure 2.3.1, and it has become a common practice to call the

classifier a threshold logic unit also. The components of the pattern are called inputs to the TLU, W a weight vector and θ a threshold. If each pattern X is augmented by an $(n+1)$-st component equal to one to give a pattern X' and if W is augmented by an $(n+1)$-st component $-\theta$ to give a vector W', then $f'(X') = \text{sgn}(X' \cdot W') = f(X)$, and the classifier determined by a hyperplane in arbitrary position in R^n is equivalent to a classifier determined by a hyperplane through the origin in R^{n+1}. The TLU corresponding to f' has $n+1$ inputs and a threshold of zero. For notational reasons and, as we have just seen, without loss of generality we now adopt the convention that the last component of all patterns is one and that the threshold of all TLU's is zero. Hence, a TLU will be completely defined by specifying its weight vector.

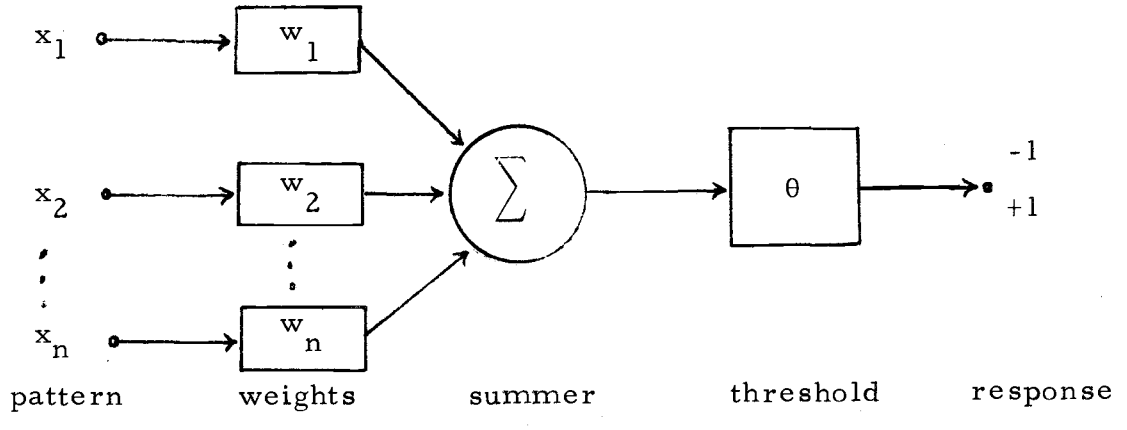


Figure 2.3.1. A threshold logic unit.

A set of patterns is said to be linearly separable if there exists a TLU recognizing every pattern. We now describe a well known local adjustment algorithm for searching a family of TLUs [24]. Let T_X be an arbitrary training sequence defined on S, and let $\{c_i\}_{i=1}^{\infty}$ be a bounded sequence of positive numbers for which the sum of every subsequence diverges. Prior to the first local adjustment let the weight vector W be arbitrary, and let the i-th local adjustment be defined by:

$$W \leftarrow \begin{cases} W - c_i TLU(X_i)X_i & \text{if } TLU(X_i) \neq class(X_i) \\ W & \text{if } TLU(X_i) = class(X_i) , \end{cases} \qquad (2.3.1)$$

where $TLU(X_i)$ denotes the response of the TLU to X_i prior to the adjustment. Block [3] has shown that the algorithm is effective. Previously a number of people had shown in different ways that the algorithm is effective when $c_i = 1$ for all i [2, 21, 27].

2.3.1 Certainty. The certainty of a TLU at X is defined as $|X \cdot W|$, where W is the weight vector defining the TLU.

It is easy to show that the algorithm is minimal. Let W and W' denote the weight vector before and after the i-th adjustment, and suppose that X_i is misclassified before the adjustment. If $class(X_i) = -1$, then

$$X_i \cdot W \geq 0$$

and

$$X_i \cdot W' = X_i \cdot W - c_i X_i \cdot X_i < X_i \cdot W.$$

If $\text{class}(X_i) = +1$, then

$$X_i \cdot W < 0$$

and

$$X_i \cdot W' = X_i \cdot W + c_i X_i \cdot X_i > X_i \cdot W .$$

In both cases the effect of an adjustment is either to change the response of the TLU at X_i or to decrease the certainty of the TLU at X_i. As each c_i is positive and the sum of the c_i's diverges, the response of the TLU to a pattern is changed by a finite number of successive adjustments to the TLU due to the pattern. What is remarkable about the algorithm is that adjusting the TLU in response to the patterns taken in an arbitrary and perhaps changing order is sufficient to insure that the TLU eventually responds correctly to all the patterns, provided of course that the problem is linearly separable and there are a finite number of patterns.

We now outline an alternative method for finding the weight vector of a TLU [10], [19]. Let ϵ be some criterion, called the error, which we wish to minimize. For instance,

$$\epsilon = E\{\phi(X)\} , \qquad (2.3.2)$$

where

$$\phi(X) = \text{class}(X)(|X \cdot W| - \text{class}(X)X \cdot W)^2 / \|\bar{W}\|^2 \qquad (2.3.3)$$

or

$$\phi(X) = \text{class}(X)(\text{class}(X) - \text{sgn}(X \cdot W))^2 , \qquad (2.3.4)$$

E is the expected value taken over all samples, $\bar{W} = (w_1, w_2, \ldots, w_{n-1})$, and w_i is a component of W. Now ϵ is a function of W, and we can use a gradient technique to try to find the value of W which minimizes ϵ. If $\{c_i\}_{i=1}^{\infty}$ is as described above and if the initial value of W is arbitrary, at the i-th iteration let

$$W \leftarrow W - c_i \frac{\partial \epsilon}{\partial W}\Big|_W . \qquad (2.3.5)$$

If $\frac{\partial \epsilon}{\partial W}$ cannot be computed, an approximation can be used. When ϵ is suitably restricted and the patterns are linearly separable, an arbitrary weight vector W converges in a finite number of iterations to a value which minimizes ϵ [10]. Minimizing the criterion defined on lines (2.3.2) and (2.3.3) minimizes the average squared distance between misclassified patterns considered as points in R^{n-1} (remember the n-th component of all patterns is one) and the hyperplane in R^{n-1} defined by the vector $\bar{W}$ and constant $-w_n$. Minimizing the criterion defined on lines (2.3.2) and (2.3.4) minimizes the number of samples misclassified.

Each adjustment to W on line (2.3.5) is determined by all the patterns simultaneously and is called a many-at-a-time adjustment. Application of the local adjustment

$$W \leftarrow W - c_i \frac{\partial \phi(X)}{\partial W}\Big|_W \tag{2.3.6}$$

once for each pattern X in S approximates a single many-at-a-time adjustment, and hopefully convergence to a solution, i.e., to a value of W which minimizes ϵ, is more rapid using local adjustments [19]. If

$$\phi(X) = \text{class}(X)\,|X \cdot W| - X \cdot W, \tag{2.3.7}$$

then the local adjustments on lines (2.3.1) and (2.3.6) are the same [19].

When the patterns are not linearly separable, the methods of adjustment described above may or may not yield a sequence of weight vectors converging to a weight vector which is in some sense optimal, e.g., a weight which minimizes some error criterion. The danger to avoid is that successive adjustments yield weight vectors whose measure of goodness in terms of some criterion oscillates wildly. The size of the oscillations can sometimes be reduced by properly choosing the sequence $\{c_i\}_{i=1}^{\infty}$, for example by choosing a sequence which converges to zero or in which c_i is a function of X_i, $X_i \cdot W$

or $X_i \cdot W / \|\overline{W}\|$, such as $c_i = 1/\|X_i\|$, where $\|\cdot\|$ is the Euclidean norm. The size of the oscillations can sometimes be reduced by periodically multiplying W by a scalar to keep the length of W within some predetermined bounds. While the methods employed to reduce oscillation may yield adjustment algorithms deemed good on the basis of their performance in a number of pattern recognition problems, only some of which are linearly separable, it may not be known if the algorithms are effective. A few of the algorithms proposed for iteratively adjusting a weight vector are described in [8, 14, 15, 17, 28, 29, 31].

Mueller [23] gives a striking example of the effect of the sequence $\{c_i\}_{i=1}^{\infty}$ on the local adjustment algorithm of line (2.3.1). The example, illustrated in Figures 2.3.2 and 2.3.3, is from [23]. In each figure the letters a and b represent samples of classes A and B respectively. The lines labeled $H_p, H_1, H_2, \ldots$ are hyperplanes corresponding to a sequence of weight vectors $W_p, W_1, W_2, \ldots$ with W_p being the initial weight vector. The problem is not linearly separable. Figure 2.3.2 shows the result of using the adjustment of line (2.3.1) with $c_i = 1$ for all i, and Figure 2.3.3 shows the result of using the same adjustment with

$$c_i = (\alpha + \frac{\beta}{\beta\gamma + d_i^2}) \frac{1}{\|X_i\|} ,$$

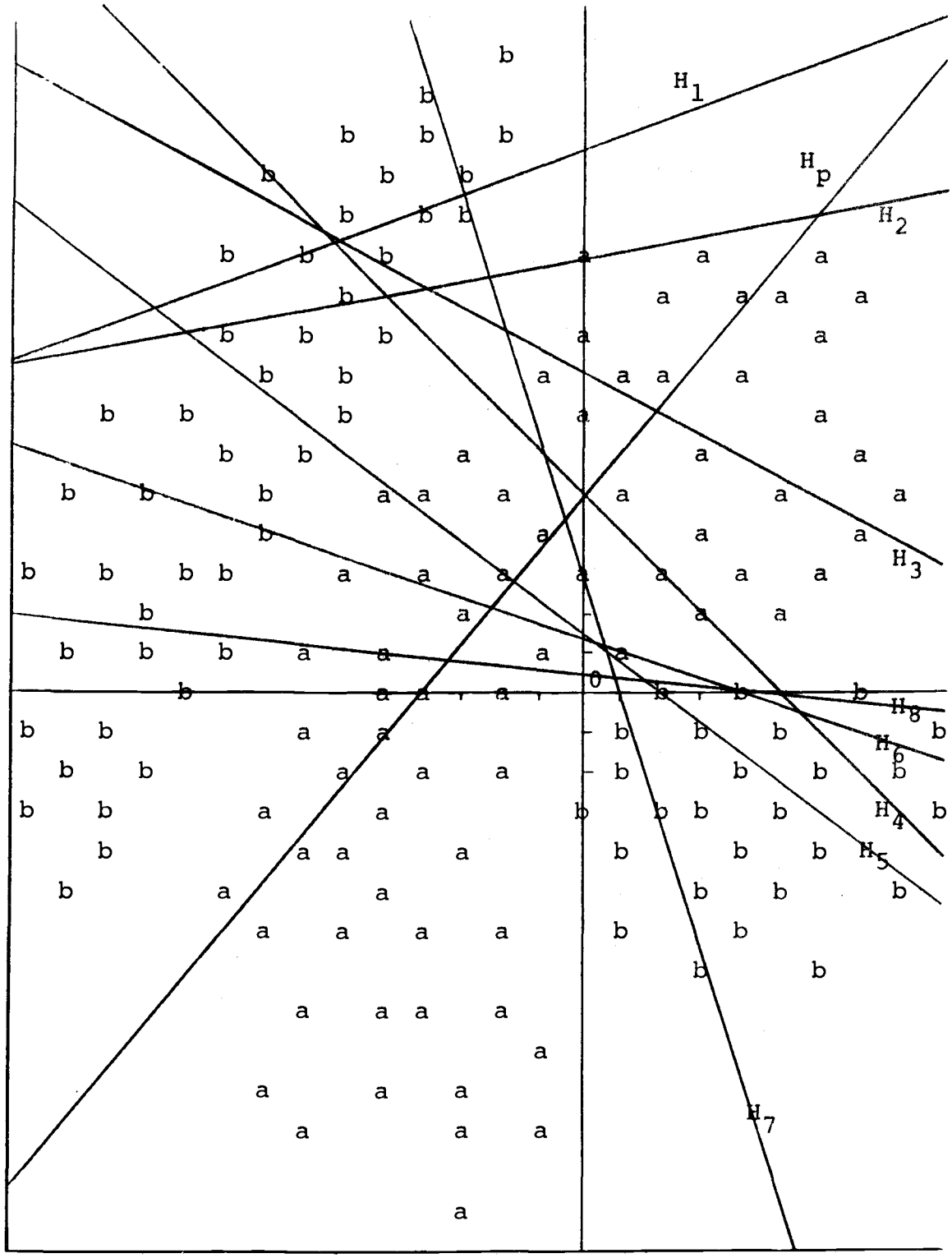


Figure 2.3.2. Local adjustment algorithm with $c_i = 1$.

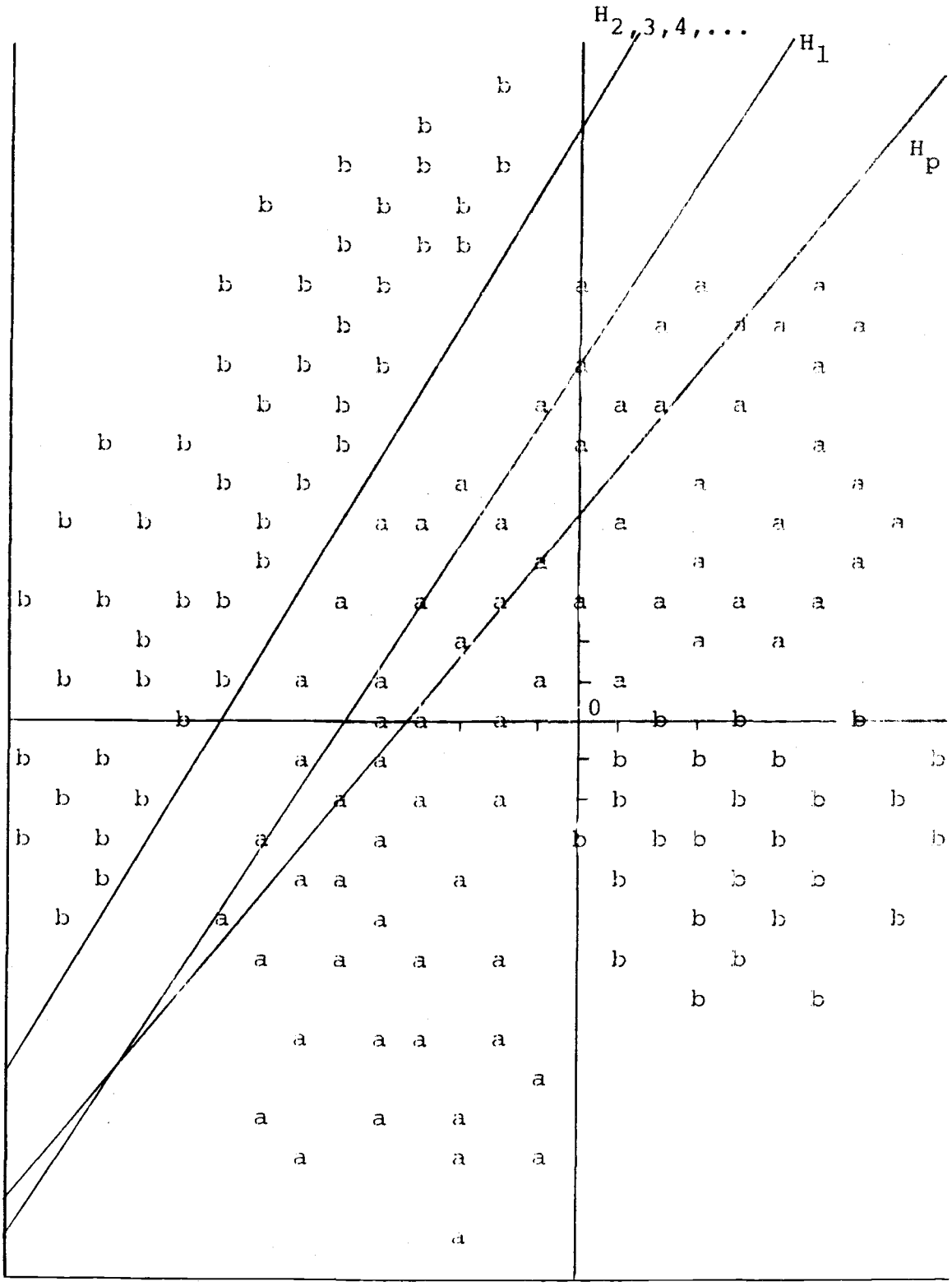


Figure 2.3.3. Local adjustment algorithm with $c_i = (\alpha + \frac{\beta}{\beta\gamma + d_i^2}) \frac{1}{\|X_i\|}$.

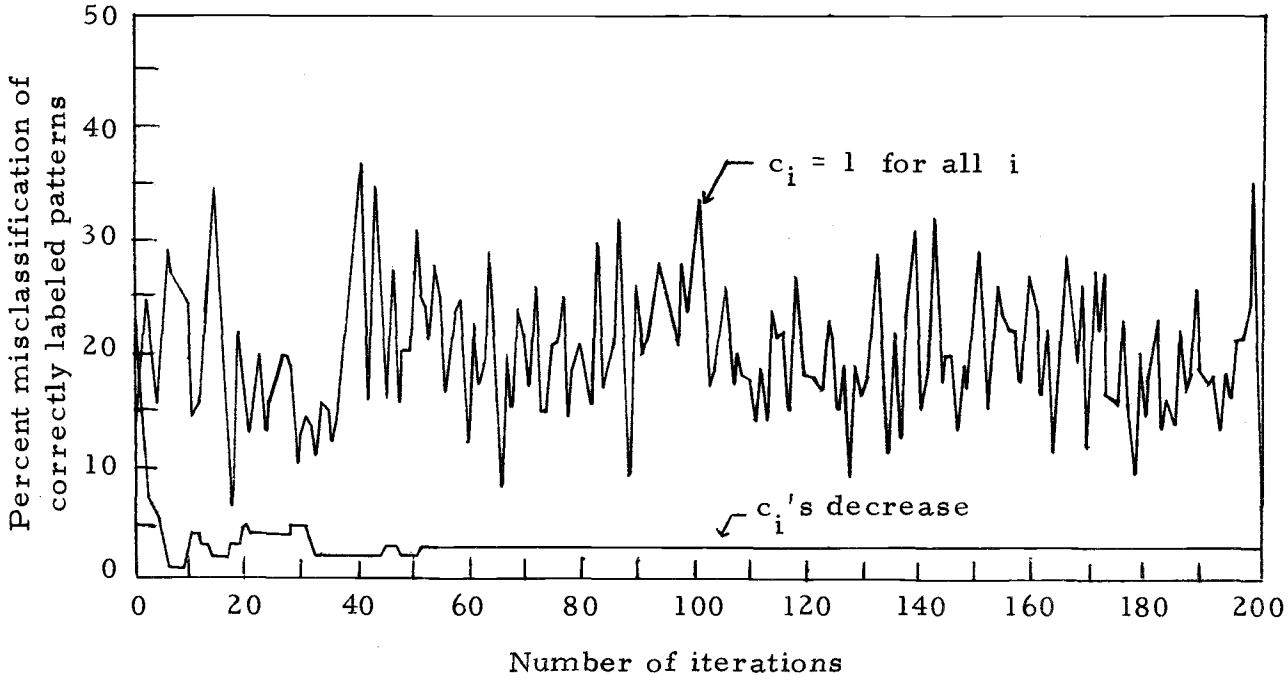


Figure 2.3.4. Training curve of Duda and Singleton.

where α, β and γ are positive constants and $d_i = X_i \cdot W$. Duda and Singleton [8] give another striking example (see Figure 2.3.4). A TLU is trained on a set of linearly separable patterns using the local adjustment algorithm of line (2.3.1). During training, 1/8th of the adjustments are randomly made incorrectly, i.e., $\text{class}(X_i)$ is replaced by $-\text{class}(X_i)$. This causes the recognition rate of the TLU to oscillate wildly when $c_i = 1$ for all i, but if the sequence $\{c_i\}_{i=1}^{\infty}$ decreases, the recognition rate does not oscillate. If $\{W_i\}_{i=1}^{\infty}$ is the sequence of weight vectors obtained by letting $c_i = 1$ for all i and $\{A_i\}_{i=1}^{\infty}$ is a sequence of weight vectors with $A_i = \sum_{j=1}^{i} W_j/i$, then $A_1 = W_1$, and A_i can be obtained by successive adjustments to A_1 with $c_j = (i-j+1)/i$ for $j \leq i$. It is for the sequence of weight vectors $\{A_i\}_{i=1}^{\infty}$ that the recognition rate does not oscillate.

The ease with which a TLU properly classifying every sample of a linearly separable problem can be found is offset by the fact that generally only a small fraction of the possible dichotomies of a set can be realized by a TLU. In particular, let S be a set of p patterns each with binary components and with the last component constantly one. Nilsson [24] has shown that the fraction of all possible dichotomies that are linearly separable is

$$F(p,n) = \begin{cases} 2^{1-p} \sum_{i=0}^{n+1} \binom{p-1}{i} & \text{if } p \geq n \\ 1 & \text{if } p < n \,. \end{cases}$$

As the number of samples is usually much larger than the dimension of the space, $F(p,n)$ is usually about zero. Problems that are not linearly separable can sometimes be made so by changing the system of variables to include not only the original variables but new variables which are functions of the original variables [20].

Despite these limitations, threshold logic units have been widely used. Even when a problem is not linearly separable, a TLU may be used as a classifier, the disadvantage of errors in classification being offset by the simplicity of the device. Rosenblatt [27] investigated networks of TLUs as a model of brain functioning, the adjustment technique of line (2.3.1) being considered a possible model of learning. Further, the ease with which a TLU may be realized as a hardware device has added to its importance historically, and some people have built complex networks of TLUs [5], although most investigators prefer to simulate networks of TLUs on digital computers.

2.4 Networks of TLUs

If a TLU is used as a classifier in a 2-class problem which is not linearly sepable, some patterns are misclassified. As indicated in the last section, the addition of new variables may make the problem linearly separable or at least reduce the number of samples misclassified. Alternatively, a more powerful classifier may be obtained by combining several TLUs in a network similar to that shown in Figure 2.4.1 [24], where each circle represents a TLU, and where there are no loops in the network, i.e., directed paths in the network that begin and end at the same TLU. The number of TLUs and the way in which they are interconnected can be varied. The inputs to each TLU consist of a subset of the original variables, a subset of the responses of the other TLUs in the network, plus the constant one. Only the last TLU in the network needs to have a threshold element.

Given a 2-class problem, the question naturally arises whether there exists a network recognizing all the patterns in the problem. Figure 2.4.2 indicates a network capable of realizing any Boolean function of $n-1$ binary variables [20]. Each of the 2^{n-1} TLUs in the first layer corresponds to one of the distinct $(n-1)$-tuples with components chosen from the set $\{-1, 1\}$. For a pattern X_i whose first $n-1$ components are chosen from the set $\{-1, 1\}$ and whose

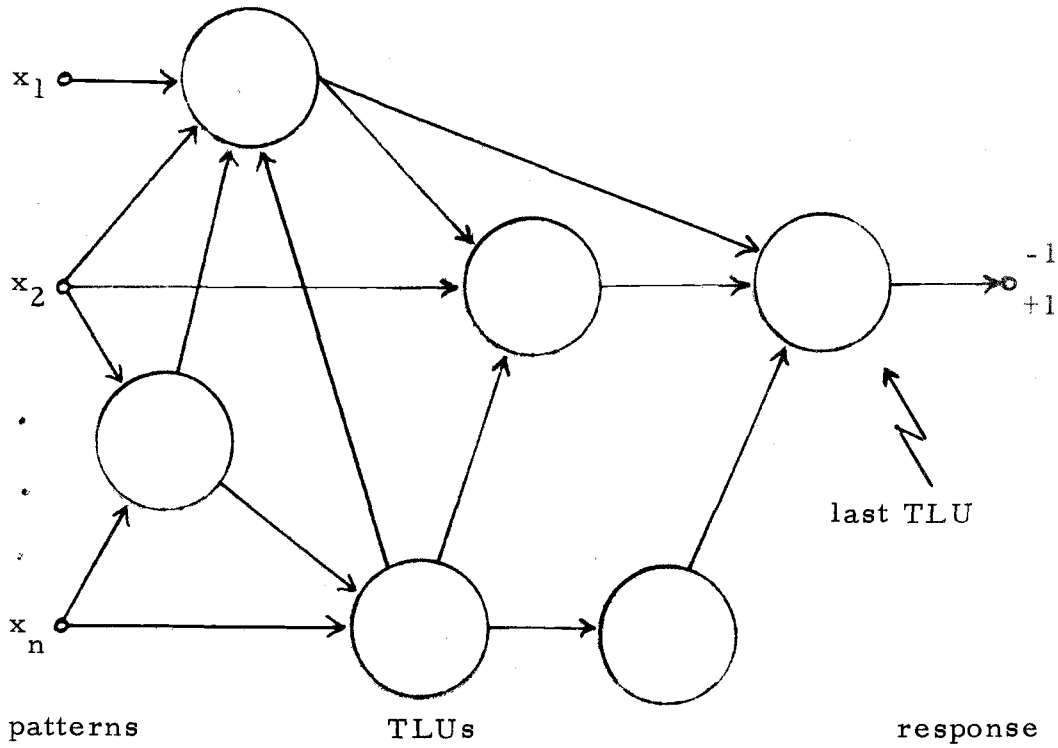


Figure 2.4.1. A network of TLUs.

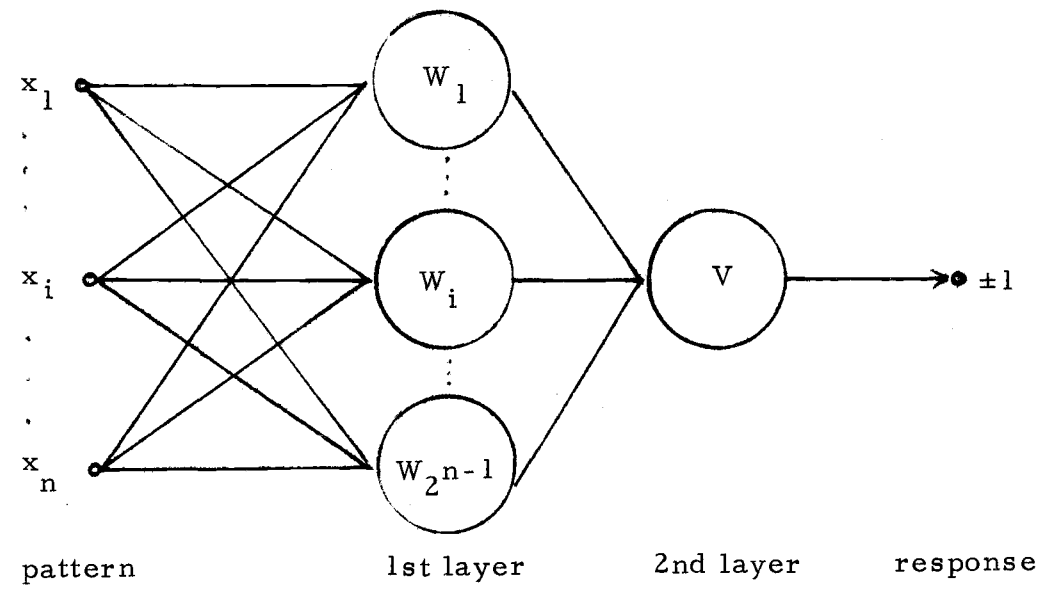


Figure 2.4.2. A two-layered network.

n-th component is the constant one, the weight vector W_i of the i-th threshold logic unit TLU_i is defined by

$$w_{ij} = \begin{cases} x_{ij} & \text{if } j = 1, \ldots, n-1 \\ -n+1 & \text{if } j = n \end{cases}$$

so that

$$TLU_i(X) = \operatorname{sgn}(X \cdot W_i) = \begin{cases} -1 & \text{if } X \neq X_i \\ +1 & \text{if } X = X_i \end{cases}.$$

The components of the weight vector V of the threshold logic unit TLU_V in the second layer depend on the problem, and

$$v_i = \begin{cases} 0 & \text{if } X_i \text{ is in } A \\ 1 & \text{if } X_i \text{ is in } B \\ -1 & \text{if } i = 2^{n-1} + 1 , \end{cases}$$

where the union of the disjoint sets A and B contains the 2^{n-1} possible patterns. The response of the networks to a pattern in A is -1 and to a pattern in B is $+1$, so the network classifies every sample correctly.

For patterns whose components can be any real numbers, Ablow and Kaylor [16] have shown how to construct a two layered network capable of performing an indicated dichotomy of a finite set of patterns S. In the construction both the first and second layer depend on the problem; the first layer may contain as many TLUs as there

are patterns in S, and the second layer contains a single TLU. Each of the TLUs in the first layer has a threshold element, so that each responds either -1 or +1 to a pattern.

The network of Figure 2.4.2 is suggestive of the conjunctive normal form of a Boolean function, and generally is not the optimum realization of a 2-class classifier, where an optimum network is one for which the number of inputs summed over all the TLUs is minimum. For any particular problem there may be many better networks containing more than two layers. There is no known algorithm for finding an optimum network for problems that are not linearly separable, nor are there algorithms that are known to be effective for training a network. Minsky and Papert [20], who prove so many interesting and elegant theorems about a single TLU, consider networks of TLUs to be "monsters of vacuous generality" for which theoretical exposition of behavior is impossible. Nevertheless there has been some success in designing and training useful networks of TLUs (we do not consider the constructions just mentioned to be useful), but the networks and the adjustment algorithms proposed are quite restricted.

Nilsson [24] surveys the techniques known in 1965 for designing and training networks. Research since then has improved and extended these techniques, for instance [6, 9, 12, 23].

Given an arbitrary network and an arbitrary 2-class problem, one scheme for training the network is to find a natural extension of

the local adjustment algorithms used to train a single TLU. Such an extension might converge in a finite number of steps to a solution whenever a solution exists and to a configuration with a low percentage of error whenever a solution does not exist. Here is a rather obvious extension of the local adjustment algorithm used to train a single TLU: If a pattern is classified correctly by the network, leave the weights of the TLUs in the network unchanged. If a pattern is misclassified, either adjust the weight of the last TLU in the network, in the manner described on line (2.3.1) or (2.3.6), or adjust the weights of one or more of the last TLU's predecessor, or do both, where we call TLU_1 a predecessor of TLU_2 if the response of TLU_1 is an input to TLU_2. If a predecessor to the last TLU is to be adjusted, this too is done in the manner described on line (2.3.1) or (2.3.6). Alternatively, instead of adjusting the weight vector of the predecessor, we can adjust the predecessor's predecessors, and so on. There are numerous small variations, but whatever the precise scheme of adjustment, the idea is to change the weights in the network so that a finite number of adjustments to the network due to a pattern changes the response of the network to the pattern, as happens with a single TLU. The fact that this rather obvious scheme for adjusting a network is not described in the literature raises the suspicion that perhaps it does not work well, and as we shall see shortly, it does not.

We now describe in detail a local adjustment algorithm of the type we have been discussing. We begin with some definitions.

2.4.1 Initial TLU. A TLU all of whose inputs are components of a pattern is called an initial TLU.

2.4.2 Response. Let X be the input to some network; let W be the weight of some threshold logic unit TLU_0 in the network; let $Y_X = (y_1, y_2, \ldots, y_m)$ be the inputs to TLU_0, where for $i = 1, \ldots, m-1$, y_i is a component of X or the response of a predecessor of TLU_0 and where $y_m = 1$. The response of TLU_0 to X is

$$TLU_0(X) = \begin{cases} Y_X \cdot W & \text{if } TLU_0 \text{ has no threshold element} \\ \operatorname{sgn}(Y_X \cdot W) & \text{if } TLU_0 \text{ has a threshold element.} \end{cases}$$

The response of the network to X, denoted $NET(X)$, is the response of the last TLU in the network to X.

2.4.3 Certainty. The certainty of TLU_0 at X is $|Y_X \cdot W|$.

2.4.4 Rank. The rank of an initial TLU is one. The rank of any other TLU is one larger than the rank of its highest ranking predecessor.

2.4.5 Local adjustment. Let W be the weight vector of

TLU_0. Let

$$P = \{TLU : TLU \text{ is a predecessor of } TLU_0 \text{ and changing the sign of } TLU(X) \text{ reduces the certainty of } TLU_0 \text{ at } X \text{ or changes the sign of } TLU_0(X)\}.$$

If $P = \phi$, a local adjustment to TLU_0 due to X is said to be of the first type and is a change in W defined by

$$W \leftarrow W - c \cdot \text{sgn}(Y_X \cdot W) Y_X , \tag{2.4.1}$$

where $c > 0$ is a constant. If $P \neq \phi$, then a local adjustment to TLU_0 due to X is said to be of the second type and is a local adjustment to that member of P that is least certain.

A local adjustment to a network due to X leaves all the weights in the network unchanged if $NET(X) = class(X)$, otherwise it consists of a local adjustment, as just described, to the last TLU in the network.

2.4.6 Antecedent. TLU_1 is an antecedent of TLU_2 if it is a predecessor of TLU_2 or if it is the antecedent of a predecessor of TLU_2.

2.4.7 Simple network. A network is simple if no threshold logic unit has predecessors which are antecedents of each other or which have common antecedents.

We now prove that the local adjustment algorithm described in Definition 2.4.5 is minimal when applied to a simple network.

2.4.8 Theorem. Let $\{c_i\}_{i=1}^{\infty}$ be a sequence of positive numbers for which the sum of every subsequence diverges. Then after a finite number of local adjustments to a simple network due to X, with the constant $c = c_i$ on the i-th adjustment, the network recognizes X.

Proof. Let X be given; let TLU_0 be some threshold logic unit in the network; let $\{c_i'\}_{i=1}^{\infty}$ be a subsequence of $\{c_i\}_{i=1}^{\infty}$ with $c = c_i'$ on the i-th adjustment to TLU_0. Let

$Y_X^i = (y_{i1}, y_{i2}, \cdots, y_{im})$ be the input vector to TLU_0,

$W_i = (w_{i1}, w_{i2}, \cdots, w_{im})$ be the weight vector of TLU_0, and P_i be the value of P, see Definition 2.4.5, after the i-th adjustment to TLU_0 due to X.

If the i-th adjustment to TLU_0 is of the first type, then

$$Y_X^i = Y_X^{i-1}, \tag{2.4.2}$$

$$W_i = W^{i-1} - c_i' \operatorname{sgn}(Y_X^{i-1} \cdot W_{i-1}) Y_X^{i-1}, \tag{2.4.3}$$

$$\begin{aligned} Y_X^i \cdot W_i &= Y_X^{i-1} \cdot W_i \\ &= Y_X^{i-1} \cdot W_{i-1} - c_i' \operatorname{sgn}(Y_X^{i-1} \cdot W_{i-1}) Y_X^{i-1} \cdot Y_X^{i-1}, \end{aligned} \tag{2.4.4}$$

and the sign of $TLU_0(X)$ is changed or the certainty is reduced by the amount

$$c_i' Y_X^{i-1} \cdot Y_X^{i-1} \geq c_i' . \tag{2.4.5}$$

We now prove by induction on the rank of TLU_0 that

(i) an adjustment to TLU_0 changes the sign of $TLU_0(X)$ or does not increase the certainty of TLU_0 at X, and

(ii) after some undetermined but finite number of adjustments to TLU_0, the sign of $TLU_0(X)$ changes.

The theorem is an immediate consequence of condition (ii).

Suppose the rank of TLU_0 is one. Then every adjustment is of the first type, and $\sum_{i=1}^{\infty} c_i' = \infty$ and lines (2.4.2) to (2.4.5) imply that conditions (i) and (ii) are true.

Suppose that (i) and (ii) are true whenever the rank of TLU_0 is less than n, and suppose the rank of TLU_0 is n. We now show that (i) is true. If the i-th adjustment is of the second type, then it involves a predecessor of TLU_0 chosen from P_{i-1}, call it TLU_1, and assume WLOG, i.e., without loss of generality, that the response of TLU_1 is the first component of the input vector to TLU_0. As the rank of TLU_1 is less than the rank of TLU_0, condition (i) holds for TLU_1, and after the i-th adjustment to TLU_0,

$$|Y_{i1}| \leq |y_{i-1,1}| \quad \text{or} \quad \operatorname{sgn}(y_{i1}) \neq \operatorname{sgn}(y_{i-1,1}) . \tag{2.4.6}$$

As the network is simple,

$$y_{ij} = y_{i-1,j} \quad \text{for} \quad j = 2, \ldots, m, \tag{2.4.7}$$

and as the adjustment is of the second type,

$$W_i = W_{i-1} \cdot \tag{2.4.8}$$

TLU_1 is in P_{i-1} because changing the sign of the product $y_{i-1,1}\, w_{i-1,1}$ decreases the magnitude or changes the sign of $Y_X^{i-1} \cdot W_{i-1}$. Decreasing the magnitude of the product $y_{i-1,1}\, w_{i-1,1}$ also decreases the magnitude or changes the sign of $Y_X^{i-1} \cdot W_{i-1}$. Hence, by lines (2.4.6) to (2.4.8),

$$|Y_X^i \cdot W_i| \leq |Y_X^{i-1} \cdot W_{i-1}|$$

or (2.4.9)

$$\operatorname{sgn}(Y_X^i \cdot W_i) \neq \operatorname{sgn}(Y_X^{i-1} \cdot W_{i-1}) \cdot$$

Condition (i) now follows from lines (2.4.2) through (2.4.5) and from line (2.4.9).

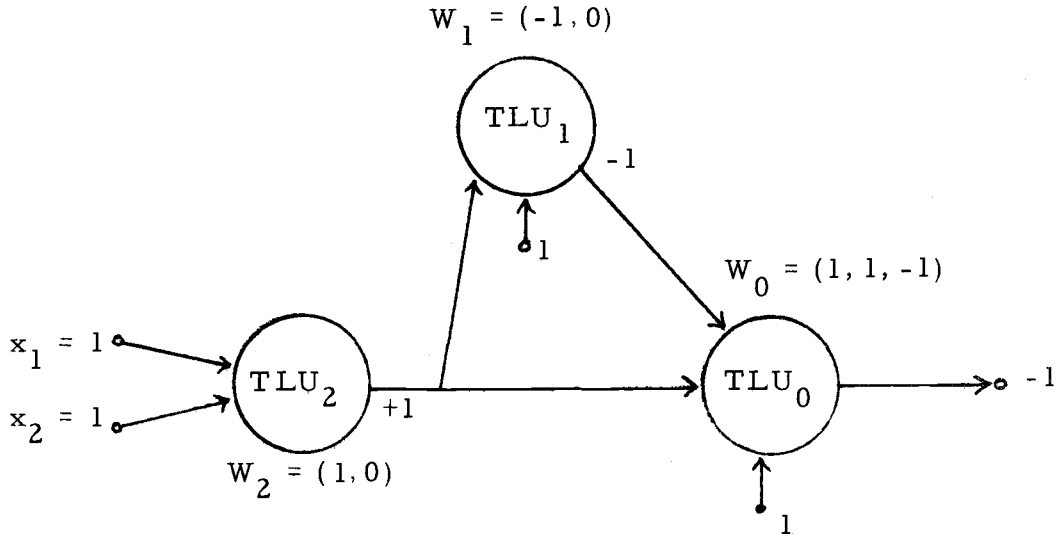
We now show that (ii) is true. If the i-th adjustment is of the second type, involving the predecessor TLU_1, then the adjustment does not increase the certainty of TLU_1 or it changes the sign of $TLU_1(X)$; and because of the simplicity of the network, the adjustment has no effect on the response of any other predecessor of TLU_0.

Hence,

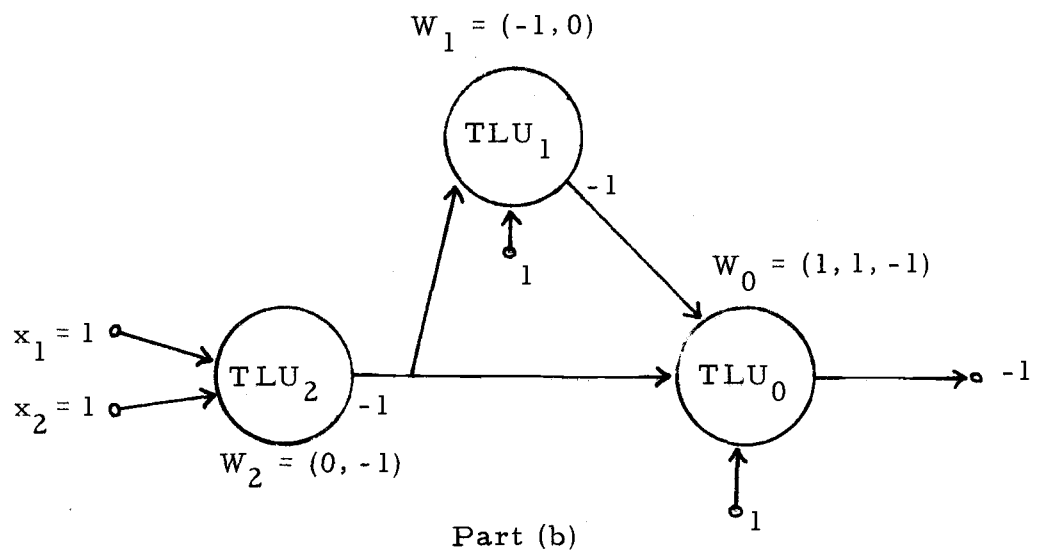
$$P_i = P_{i-1} \quad \text{or} \quad P_i = P_{i-1} \setminus \{TLU_1\} . \tag{2.4.10}$$

As the rank of every predecessor of TLU_0 is less than n, after a finite number of successive adjustments of the second type, (1) the sign of $TLU_0(X)$ will change, or (2) P will be empty, and there will be an adjustment of the first type. Suppose that none of the adjustments to TLU_0, up to and including the i-th adjustment, changes the sign of $TLU_0(X)$. None of these adjustments of the second type increases the certainty of TLU_0, and all of the first type decrease the certainty by an amount at least as large as c'_j, where j is the index of the adjustment, see lines (2.4.2) to (2.4.5). As there can be only a finite number of adjustments of the second type before there is one of the first type and as the sum of every subsequence of $\{c_i\}_{i=1}^{\infty}$ diverges, the sign of $TLU_0(X)$ changes after a finite number of adjustments. This proves (ii) and the theorem. ●

As the following example illustrates, if the network is not simple, we can no longer prove Theorem 2.4.8. Figure 2.4.3 illustrates two states of a network of three threshold logic units, labeled TLU_0, TLU_1 and TLU_2. The input to TLU_2 is a pattern X with two components; the inputs to TLU_1 are $TLU_2(X)$ and the constant one; and the inputs to TLU_0 are in this order $TLU_2(X)$, $TLU_1(X)$



Part (a)



Part (b)

Figure 2.4.3. An oscillating network.

and the constant one. Let W_0, W_1 and W_2 denote the weight vectors of TLU_0, TLU_1 and TLU_2 respectively. Suppose that, in the sequence $\{c_i\}_{i=1}^{\infty}$, $c_i = 1$ for all i and that we wish to train the network to respond with a $+1$ to $X = (1, 1)$. If the initial values of the weight vectors are $W_0 = (1, 1, -1)$, $W_1 = (-1, 0)$ and $W_2 = (1,0)$, the response of the network to X is -1, see part (a) of Figure 2.4.3. An adjustment to the network due to X is an adjustment to TLU_0, is of the second type and involves TLU_1. An adjustment to TLU_1 is also of the second type and involves TLU_2. An adjustment to TLU_2 is of the first type; after the adjustment, $W_2 = (0, -1)$; and the response of the network to X is still -1, see part (b) of Figure 2.4.3. A second adjustment to the network due to X again results in an adjustment of the first type to TLU_2; after the adjustment, $W_2 = (1, 0)$; and the response of the network to X is -1, see part (a) of Figure 2.4.3. So we see that, no matter how many adjustments are made, the response of the network to X will never be $+1$.

Even if a network is simple, the local adjustment algorithm, though appearing reasonable, is not satisfactory. In Section 5.1 we will describe the results of computer simulations of the algorithm on a two-layered network in which there were three TLU's in the first layer. As the performance of the algorithm was unsatisfactory, it seemed pointless to perform simulations on more complex networks.

In a seminar on pattern recognition conducted at Oregon State University in the spring of 1971, two modifications of the local adjustment algorithm were also found to be unsatisfactory. (It is misleading to call the algorithms tested in the spring of 1971 modifications of the algorithm described here as they predated this algorithm.) In both modifications an adjustment to a TLU consisted of:

(1) a change in the TLU's weight vector as defined on line (2.4.1), and

(2) if the response of the TLU was not changed by the adjustment, then the set P, see Definition 2.4.5, was determined and in the case of the first modification the least certain element of P was adjusted while in the case of the second modification all the elements of P were adjusted.

A local adjustment to a network due to X left all the weights in the network unchanged if NET(X) = class(X), otherwise, it consisted of a local adjustment, as just described, to the last TLU in the network. If a network is simple, both modifications are minimal.

2.5 Committee Machines

It is probably because of difficulties like those described in Section 2.4 that a particularly simple type of network, called a committee, has received attention. A committee has an arbitrary

number of initial TLUs, each with a threshold element so that its response to a pattern is -1 or +1, and a final TLU whose inputs are the responses of the initial TLUs plus, of course, the constant one. The network of Figure 2.4.2 is an example of a committee; and as we saw in Section 2.4, every two class problem containing a finite number of patterns can be solved by some committee, although containing perhaps as many initial TLUs as there are patterns. The initial TLUs are called committee members, and the response of each member is called its vote. The final TLU is called the vote-taking TLU, and its weight vector determines what is called the logic of the committee.

Committees can be trained using a local adjustment algorithm. The logic is fixed beforehand, and the weights of the committee members are adjusted during training. The actual scheme of adjustment depends on the logic of the committee. Although a training algorithm would be potentially more powerful if the logic could also be changed in some beneficial way during training, there do not seem to be any algorithms described in the literature for doing this, perhaps because of the difficulties encountered in using the local adjustment algorithm of Definition 2.4.5.

Given a 2-class problem the question arises as to what will be the logic of the committee of minimum size recognizing all the patterns in the problem. Kaylor [16] conjectures that if the components

of each pattern are binary, among the committees of minimum size there is one called a majority committee. The logic of a majority committee is defined by letting the components of the weight vector of the vote-taking TLU be

$$v_i = \begin{cases} 1 & \text{if} \quad i = 1, \ldots, p \\ 0 & \text{if} \quad i = p+1 \end{cases} \tag{2.5.1}$$

where for this and all committees the i-th input to the vote-taking TLU is the response of a member if $i = 1, \ldots, p$ and the constant one if $i = p+1$. The response of a majority committee is $+1$ if and only if the response of at least half of the members is $+1$. Mueller [23] shows that, if the patterns do not have binary components, Kaylor's conjecture is false: he exhibits a problem which can be solved by a three member veto committee, but not by a three member majority committee. On a veto committee, the components of the weight vector V of the vote taking TLU are

$$v_i = \begin{cases} 1 & \text{if} \quad i = 1, \ldots, p \\ -p & \text{if} \quad i = p+1 , \end{cases} \tag{2.5.2}$$

so that the response of a veto committee is $+1$ if and only if the response of each member is $+1$.

Because in later sections we will present a new logic for a committee and a local adjustment algorithm for training it, for

contrast we now describe two local adjustment algorithms due to Ridgway [26], the first for training a majority committee, and the second for training a veto committee. We need only define the local adjustment used in each case. Denote the i-th member of the committee by $MMBR_i$, its weight vector by W_i, and its response to a pattern X by $MMBR_i(X)$, or by y_i, where $i = 1, \ldots, p$ and p is the number of committee members. Let $Y_X = (y_1, y_2, \ldots, y_p, 1)$ be the input to the vote-taking TLU and $COM(X)$ denote the response of the committee to X.

2.5.1 Local adjustment to a majority committee. A local adjustment to a majority committee due to X is defined as follows. If $COM(X) = class(X)$, none of the weights in the network are changed. If $COM(X) \neq class(X)$, let $MMBR_{i_1}, MMBR_{i_2}, \ldots, MMBR_{i_q}$ be the committee members whose response to X is the same as that of the committee, and assume WLOG that $|X \cdot W_{i_j}| \leq |X \cdot W_{i_{j+1}}|$, $j = 1, \ldots, q-1$. Let k be the smallest integer such that changing the responses of $MMBR_{i_1}, MMBR_{i_2}, \ldots, MMBR_{i_k}$ changes the response of the committee. For $j = 1, \ldots, k$ let

$$W_{i_j} \leftarrow W_{i_j} - c\, MMBR_{i_j}(X)X, \tag{2.5.3}$$

where $c > 0$ is a constant. The weights of the remaining

committee members are unchanged.

The local adjustment algorithm defined by Definition 2.5.1 is minimal. If $COM(X) \neq class(X)$, then a local adjustment to a majority committee due to X either changes the sign of $MMBR_{i_j}(X)$ or decreases $|X \cdot W_{i_j}|$ for $j = 1, \ldots, k$. Neither the response of a majority committee nor its certainty, defined as $|Y_X \cdot V|$, is necessarily changed by a single adjustment due to X; however, after a finite number of adjustments, the responses of individual committee members change, $|Y_X \cdot V|$ decreases, and eventually the response of the committee to X changes.

2.5.2 Local adjustment to a veto committee. A local adjustment to a veto committee due to X is defined as follows. If $COM(X) = class(X)$, none of the weights in the network are changed. If $COM(X) \neq class(X)$, let $MMBR_{i_1}, MMBR_{i_2}, \ldots, MMBR_{i_q}$ be as described in Definition 2.5.1. If $COM(X) = +1$, let

$$W_{i_1} \leftarrow W_{i_1} - c\, MMBR_{i_1}(X)X, \tag{2.5.4}$$

with $c > 0$, and let the weights of the remaining committee members be unchanged. If $COM(X) = -1$, for $j = 1, \ldots, q$ let

$$W_{i_j} \leftarrow W_{i_j} - c\, MMBR_{i_j}(X)X, \tag{2.5.5}$$

and let the weights of the remaining members be unchanged. On lines (2.5.4) and (2.5.5), c is a positive constant.

The local adjustment algorithm defined by Definition 2.5.2 is minimal. If $COM(X) \neq class(X)$ and $COM(X) = +1$, successive adjustments due to X are to $MMBR_{i_1}$, and after a finite number of adjustments, the response of $MMBR_{i_1}$ changes from $+1$ to -1 and so does the response of the committee. If $COM(X) \neq class(X)$ and $COM(X) = -1$, after a finite number of successive adjustments due to X, $MMBR_{i_j}(X) = +1$, $j = 1, \ldots, q$, and $COM(X) = +1$.

Note that in both algorithms the number of committee members is determined before training begins, and there is no provision for changing the number of committee members during training. If the committee has too few members, the committee is not able to recognize all the patterns. Even if the committee has the minimum number of members needed to solve the problem, there are instances in which the adjustment algorithms above do not yield a solution [24]. Experience also shows that the training is more likely to converge to a solution if the number of members is larger than the minimum number needed [24]. For a given problem, however, there is no way of knowing in advance how many members to place on the committee.

2.6 Modified Veto Committee

In this section we describe a new logic for a committee called modified veto logic. The choice of the name and the meaning will become clear as we proceed.

2.6.1 Modified veto logic. For an arbitrary partition of the set $\{v_i : v_i$ a component of V and $i = 1, \ldots, p\}$ into two subsets, where p is the number of committee members, we say that the component v_1 and all components in the same subset as v_1 are of the first type and the remaining components are of the second type. For $i = 1, \ldots, p$ let

$$R_i = \{v_j : j \leq i \text{ and } v_j \text{ is of the first type}\},$$
$$T_i = \{v_j : j \leq i \text{ and } v_j \text{ is of the second type}\},$$

and let

$$v_i = \begin{cases} 1 & \text{if } i = 1 \\ 1 + \sum_{v_j \in T_i} v_j & \text{if } v_i \text{ is of the first type} \\ \sum_{v_j \in R_i} v_j & \text{if } v_i \text{ is of the second type,} \end{cases} \qquad (2.6.1)$$

i.e., for $i > 1$, if v_i is of the first type, its value is one larger than the sum of all v_j, $j < i$, of the second type; and if v_i is of

the second type, its value is equal to the sum of all v_j, $j < i$, of the first type. The last component of V is

$$v_{p+1} = \sum_{v_j \in T_p} v_j - \sum_{v_j \in R_p} v_j .$$

For example, if there are five committee members with every second one, starting with the first, of the first type, then $V = (1, 1, 2, 3, 5, -4)$.

2.6.2 Ignored. We say that the response of a committee member is ignored if the response is $+1$ and the member is of the first type or if the response is -1 and the member is of the second type, where the type of $MMBR_i$ is defined to be the type of v_i.

2.6.3 Theorem. Given a modified veto committee, let

$$i' = \max\{i : i = 0 \text{ or } MMBR_i(X) \text{ is not ignored}\}.$$

Then

$$COM(X) = \begin{cases} +1 & \text{if } i' = 0 \\ MMBR_{i'}(X) & \text{if } i' \neq 0 . \end{cases}$$

Proof. For a pattern X let

$$k = \max\{i : i = 0, \text{ or } MMBR_i(X) \text{ is of the first type and not ignored}\},$$

and let

$$\ell = \max\{i : i = 0, \text{ or } MMBR_i(X) \text{ is of the second type and not ignored}\}.$$

The value of $COM(X)$ depends on the value of

$$Y_X \cdot V = \sum_{i=1}^{p} y_i v_i + \sum_{v_j \in T_p} v_j - \sum_{v_j \in R_p} v_j$$

$$= \sum_{v_j \in T_p} (1+y_j)v_j + \sum_{v_j \in R_p} (y_j - 1)v_j . \qquad (2.6.2)$$

Note that on line (2.6.2) each term of the first sum is $2v_j$ or zero and of the second sum zero or $-2v_j$ as y_j is $+1$ or -1 respectively. Also note that $v_j \geq 0$ for $j = 1, \ldots, p$.

If $k > \ell$, then by line (2.6.2)

$$Y_X \cdot V \leq \sum_{v_j \in T_k} (1+y_j)v_j + \sum_{v_j \in R_p} (y_j - 1)v_j$$

as $j > k$ and v_j of the second type imply $j > \ell$ and $y_j = -1$

$$\leq \sum_{v_j \in T_k} (1+y_j)v_j - 2v_k$$

as each term of the second sum is less

than or equal to zero, as $y_k = -1$ and

as $v_k \in R_p$

$$\leq \sum_{v_j \in T_k} 2v_j - 2v_k$$

as $1+y_j \leq 2$

$$\leq \sum_{v_j \in T_k} 2v_j - 2(1+ \sum_{v_j \in T_k} v_j)$$

by line (2.6.1)

$$\leq -2 .$$

Hence, $COM(X) = -1$. As $i' = k$, $MMBR_{i'}(X) = -1$ also.

If $k < \ell$, then by line (2.6.2)

$$Y_X \cdot V \geq \sum_{v_j \in T_p} (1+y_j)v_j + \sum_{v_j \in R_\ell} (y_j-1)v_j$$

as $j > \ell$ and v_j of the first type

imply $j > k$ and $y_j = +1$.

$$\geq 2u_\ell - \sum_{v_j \in R_\ell} 2v_j$$

as each term of the first sum is greater

than or equal to zero, as $y_\ell = +1$ and

as $v_\ell \in T_p$

$$\geq 2 \sum_{v_j \in R_\ell} v_j - \sum_{v_j \in R_\ell} 2v_j$$

by line (2.6.1)

$$\geq 0.$$

Hence, $COM(X) = +1$. As $i' = \ell$, $MMBR_{i'}(X) = +1$ also.

If $k = \ell = 0$, then for all v_j of the first type $y_j = +1$, for all v_j of the second type $y_j = -1$, and by line (2.6.2) $Y_X \cdot V = 0$. Hence $COM(X) = +1$. ●

2.6.4 Member making the decision. If $i' \neq 0$ in Theorem 2.6.3, then $MMBR_{i'}$ is said to be the member making the decision. If $i' = 0$, the decision is said to be made by default, and we make the convention that the decision is made by a non-existent member, $MMBR_0$, of the second type whose response to all patterns is $+1$.

Stated informally, Theorem 2.6.3 says that responses of $+1$ by committee members of the first type and -1 by committee members of the second type are ignored. The response of the committee is the last response not ignored. If all responses are ignored, the response of the committee is $+1$. Thought of in another way, some members vote YES or abstain and others vote NO or abstain. If all abstain, the committee decides YES, otherwise the vote of the highest ranking, i.e., highest index, member prevails.

2.7 Capacity of Modified Veto Committee

In this section we show that any 2-class problem containing a finite number of patterns whose components are binary can be solved by a modified veto committee. First we prove the result for a veto committee. For a veto committee, the result is so obvious that undoubtedly many people have proved it before, and the proof is included here only because it does not seem to be in the literature. The proof is constructive and the number of committee members equals the number of samples in the first class, even if many fewer members would be sufficient.

2.7.1 Lemma. Let S be a set of patterns whose components are ± 1, let $A \cap B = \phi$ and $A \cup B = S$, and let p equal the number of elements in A. Then there exists a veto committee of p members such that

$$\text{COM}(X) = \begin{cases} -1 & \text{if } X \text{ is in } A \\ +1 & \text{if } X \text{ is in } B. \end{cases}$$

Proof. Let $X_i = (x_{i1}, x_{i2}, \ldots, x_{in})$, $i = 1, \ldots, p$, be the distinct patterns in A, and let $W_i = (w_{i1}, w_{i2}, \ldots, w_{in})$ be the weight vector of MMBR_i, where

$$w_{ij} = \begin{cases} -x_{ij} & \text{if} \quad j = 1, \ldots, n-1 \\ n-2 & \text{if} \quad j = n \, . \end{cases}$$

If $X = X_i$, then

$$X \cdot W_i = n-2 + \sum_{j=1}^{n-1} -x_{ij}^2 < 0.$$

If $X \neq X_i$, there exists k, $1 \leq k \leq p$, such that

$$x_k \neq x_{ik}, \quad \text{i.e.,} \quad x_k = -x_{ik} \, ,$$

and

$$X \cdot W_i = n-2 + \sum_{j=1}^{n-1} -x_j x_{ij}$$

$$\geq n-2 + \left(\sum_{j=1}^{k-1} -x_{ij}^2 \right) + x_k^2 + \left(\sum_{j=k+1}^{n-1} -x_{ij}^2 \right)$$

$$\geq 1 \, .$$

Hence for all $i = 1, \ldots, p$,

$$\text{MMBR}_i(X) = \begin{cases} -1 & \text{if} \quad X = X_i \\ +1 & \text{if} \quad X \neq X_i \, , \end{cases}$$

and

$$\text{COM}(X) = \begin{cases} -1 & \text{if} \quad X \in A \\ +1 & \text{if} \quad X \in B \, . \end{cases} \quad \bullet$$

The name modified veto logic is suggested by the following consideration. If a modified veto committee has q members of the first type and if the members of the second type respond -1 to each pattern, then the committee is equivalent to a committee containing just the q members of the first type and voting by veto logic. If, on the other hand, members of the first type respond $+1$ to all patterns, the committee responds $+1$ regardless of the responses of the members of the second type.

2.7.2 Lemma. If $T_p = \phi$, then modified veto logic and veto logic are the same.

Proof. As $T_p = \phi$, then by line (2.6.1)

$$v_i = \begin{cases} 1 & \text{if } i = 1, \ldots, p \\ -p & \text{if } i = p+1 , \end{cases}$$

which is the definition of veto logic. ●

2.7.3 Theorem. Let S be a finite set of patterns whose components are ± 1, let $A \cap B = \phi$ and $A \cup B = S$, and let p equal the number of elements in A. Then there exists a modified veto committee with p members such that

$$\mathrm{COM}(X) = \begin{cases} -1 & \text{if } X \text{ is in } A \\ +1 & \text{if } X \text{ is in } B . \end{cases}$$

Proof. The theorem is an immediate consequence of Lemmas 2.7.1 and 2.7.2. ●

2.8 An Informal Description of a Local Adjustment Algorithm for Training a Modified Veto Committee

In this section we describe an intuitively appealing local adjustment algorithm for training a modified veto committee. In broad outline, the algorithm is as follows: Before training begins, a training sequence is defined, and the committee starts with a single member of the first type, whose weight vector is

$W_1 = (w_{11}, w_{12}, \ldots, w_{1n})$, where

$$w_{1j} = \begin{cases} 0 & \text{if } j = 1, \ldots, n-1 \\ 1 & \text{if } j = n. \end{cases}$$

For any pattern X, $X \cdot W_1 = 1$, $MMBR_1(X) = +1$, the response is ignored as $MMBR_1$ is of the first type, the decision is made by default, and $COM(X) = +1$. The committee classifies all the patterns in class B correctly and misclassifies all the patterns in class A. After many local adjustments to $MMBR_1$ of the type described on line (2.3.1), $MMBR_1$ and hence the committee recognizes all the patterns in the problem, or the recognition rate of $MMBR_1$ stops improving. In the latter instance further adjustment to $MMBR_1$ is pointless, and a new member is added to the committee.

Assuming that we have reached a point where there are $i-1$ committee members, that patterns are still misclassified, and that further adjustments to the committee do not improve the recognition rate, we add a new member, $MMBR_i$, to specialize in responding to those patterns misclassified while ignoring those already classified correctly. If $MMBR_1$ is of the first type, the components of W_i are initially

$$w_{ij} = \begin{cases} 0 & \text{if } j = 1, \ldots, n-1 \\ 1 & \text{if } j = n ; \end{cases}$$

$MMBR_i(X) = +1$ for every pattern X, the response is ignored, and the response of the committee to X is unchanged. Local adjustments to $MMBR_i$ of the type described on line (2.3.1) due to patterns in A still misclassified by the committee increase the recognition rate for patterns in A while perhaps causing some patterns in B previously classified correctly to be misclassified. Any attempt to adjust $MMBR_i$ so that the newly misclassified patterns in B are again recognized may decrease the recognition rate on A. Hopefully, W_i attains a value that makes for the greatest possible improvement in the recognition rate of the committee, at which time further improvement can be obtained by adding yet another member to the committee. If $MMBR_i$ is of the second type, the components of W_i are initially

$$w_{ij} = \begin{cases} 0 & \text{if } j = 1, \ldots, n-1 \\ -1 & \text{if } j = n ; \end{cases}$$

$MMBR_i(X) = -1$ for every pattern X, the response is ignored, and the response of the committee to X is unchanged. $MMBR_i$ is adjusted to respond $+1$ to those patterns in B still misclassified without causing too many additional patterns in A to be misclassified. Additional members continue to be added until the recognition rate is 100%, there has been a certain number of adjustments, or the committee has reached a certain size.

The local adjustment algorithm is slightly more complex than that just outlined, and if a pattern in the training sequence is misclassified, several members already on the committee and a new member to be added are considered for adjustment. The problem of deciding which member to adjust, the new member or an old member, the changing of whose response would change the response of the committee, is solved by assigning an age to each member. When a member is first added, it is given some initial age which is incremented by one every time the member is adjusted. Among the members who can change the response of the committee, the one for which the product of certainty and age is least is adjusted. In this way members recently added to the committee tend to be most active, their age is small though their certainty may be large; older members are

adjusted only due to patterns very close to the hyperplane defined by their weight vector. The value of the initial age assigned to new members determines how readily new members are added. If the initial age is small, after a few adjustments to a member, its age is many times that of a new member, and a new member is likely to be added.

A difficulty with local adjustment algorithms is that, when the algorithm does not converge to a solution, successive adjustments can cause wild oscillations in the recognition rate of the classifier, see Figures 2.3.2 and 2.3.4, and it becomes important to stop the training at the right instant. One way to avoid the difficulty is to stop the training periodically, compute the recognition rate, and save the parameters defining the current configuration of the classifier if the recognition rate is the largest to date. Such a solution entails a lot of additional computation. An alternative is to use an algorithm that is not sensitive to excessive adjustments, see Figures 2.3.3 and 2.3.4. Figure 2.3.4 illustrates that, if the size of successive adjustments decreases, the recognition rate of a single TLU oscillates little, even when the TLU is adjusted in response to the conflicting demands of patterns that are not linearly separable. The same technique is beneficial in the algorithm for training a modified veto committee, and the age of a member not only helps determine which member to adjust but is also used to determine the size of an adjustment. The size of an

adjustment to a member is inversely proportional to the member's age. The algorithm should be insensitive to excessive adjustments for a second reason. Hopefully, the members first added to the committee make the decision on most samples, while later members are added and trained to respond to fewer and fewer patterns still misclassified. If training continues too long, members continue to be added, specializing in a few troublesome patterns, and there is little effect on the overall performance of the committee. When training is completed, the members last added can be taken off the committee. In contrast, on a veto committee an adjustment due to a pattern which is incorrectly vetoed, i.e., some member responds -1 to a pattern in B, will be an adjustment to every member responding -1, even if the performance of these members is on the whole satisfactory and further adjustment to them has a bad overall effect.

A veto committee can also have additional members added as needed during training. When first added a new member would respond $+1$ to all patterns, leaving the response of the committee unchanged, and would be trained to veto some patterns. On the other hand, adding a member to a majority committee changes the response of the committee to many patterns, so one must fix the number of members before training begins.

A second difficulty with local adjustment algorithms and with complex classifiers is that after training, the recognition rate is much

higher on the training set than it is on a test set, a phenomenon Bongard [4] calls prejudice. Prejudice increases as the learning scheme becomes more complex and as the size of the training set decreases. Larson [18] performed a relevant experiment. Five groups of patterns were generated in R^{32}, each group contained 33 patterns, each component of each pattern was a random number and randomness was confirmed using the F-test. Each group was divided into a training set of 22 patterns and a test set of 11 patterns. Discriminant analysis was applied to the samples in the training set to find the four mutually orthogonal directions in the pattern space in which the ratio of between class variance to within class variance was greatest. The mean of each group of training patterns in this four dimensional space was computed, and a 5-class classifier, called the distance-to-mean classifier, was defined by assigning a pattern to the class whose mean was closest. The experiment was repeated 15 times. The overall recognition rate in the training set was 85% and in the test set 19%. As there were five classes, assigning a sample to a class at random would have given an expected recognition rate of 20%. We see that the results in the training set were completely misleading. If the distance-to-mean classifier were to be used as above but with the means computed in the original system of 32 variables, because of the randomness of each variable, we would expect the recognition rate to be 20% in both the training and the test set. In the above example

the prejudice is not due to inadequacies in the distance-to-mean classifier but to using a complex method to choose a new measurement space based on the information contained in very few training patterns. By the central limit theorem, one can expect that in general the class means determined from a training set are relatively unprejudiced estimates of the true class means.

To see how a local adjustment algorithm can cause prejudice, we compare the solution of a simple 2-class problem obtained by using the distance-to-mean classifier with the solution obtained by using the local adjustment algorithm of line (2.3.1), see Figure 2.8.1. Class A consists of the -'s and B of the +'s. The training patterns in each class are circled. The distance-to-mean classifier, as determined from the training set, is defined by the hyperplane H_1, misclassifies one training pattern, and is an excellent classifier for the problem as a whole. Because the training patterns are linearly separable, the local adjustment algorithm yields a classifier, defined by hyperplane H_2, classifying all the training samples correctly but performing poorly for the problem as a whole. The local adjustment algorithm suffers from the defect that it will tolerate no errors in the training set without taking corrective action.

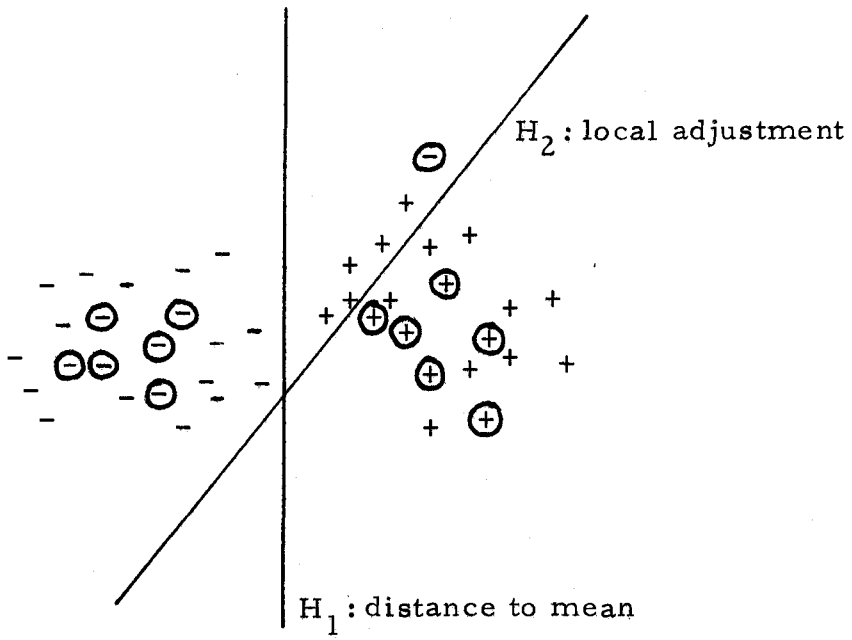


Figure 2.8.1. Local adjustment algorithm and prejudice.

The algorithm for training a modified veto committee is complex and is a local adjustment algorithm, so we expect prejudice; however, two things have been done to lessen the effects of prejudice. First, successive adjustments to a member are smaller and smaller, so that a position reached in response to many patterns is not easily changed if the member is adjusted in response to a small number of atypical patterns. Second, as a committee member ages, it is less likely to be adjusted, and the position reached in response to many patterns is again protected from the effects of adjustments due to atypical patterns. Atypical patterns cause adjustments to the committee, but to new members which ignore all but the few patterns that have called

them into existence. So it is hoped that only the later members added to the committee will learn prejudice and that improvements to the recognition rate in the training set will not come at the expense of a reduction in the recognition rate on the problem as a whole, as happens in Figure 2.8.1. Computer simulations described in Section 5.3 seem to encourage this hope.

2.9 A Local Adjustment Algorithm for Training a Modified Veto Committee

We now give a precise definition of the local adjustment algorithm described in the previous section.

2.9.1 Adjustment to a member. Let W be the weight vector of a committee member MMBR, and let $\|W\| = 1$. An adjustment to MMBR due to X is a change in W defined by

$$\alpha \leftarrow \alpha + 1 \tag{2.9.1}$$

$$W \leftarrow W - \frac{\beta}{\alpha} \mathrm{MMBR}(X) \frac{X}{\|X\|} \tag{2.9.2}$$

$$W \leftarrow W/\|W\| \ , \tag{2.9.3}$$

where α, called the age, has some initial value $\alpha_0 > 0$ before the first adjustment and where β, $\alpha_0 > \beta > 0$, is a constant.

2.9.2 Adjustable member. A committee member is adjustable in response to a pattern X if changing the response of

the member to X changes the response of the committee to X.

Three kinds of committee members are adjustable in response to a pattern X. First, a new member of the appropriate type is adjustable. If COM(X) = +1, the new member should be of the first type with all components except the last zero and the last one. Initially, the response of the new member to X is +1, which is ignored, but if the response of the new member changes to -1, so does the response of the committee. Similarly, if COM(X) = -1, the new member should be of the second type. Second, members of larger index and a different type than the member making the decision are adjustable. Currently the responses of these members are being ignored; changing the response of any one of them will cause that one to make the decision, and, as it is of a different type than the member currently making the decision, the response of the committee will change. Third, the member making the decision is sometimes adjustable. Suppose $MMBR_k$, $k > 0$, makes the decision; then changing its response causes its response to be ignored, and it is adjustable if and only if the member of next highest index not ignored is of a different type. Members of lower index than the member making the decision are not adjustable as their responses do not effect the response of the committee.

2.9.3 Resistance. The resistance of a member with weight

vector W and age α at a pattern X is

$$\text{rest}(X) = |X \cdot W|^{(\alpha+\gamma)}, \tag{2.9.4}$$

where $\gamma \geq -\alpha_0$ is constant.

2.9.4 Local adjustment to a modified veto committee. A local adjustment to a modified veto committee due to a pattern X is defined as follows. If $\text{COM}(X) = \text{class}(X)$, none of the weights in the network are changed. If $\text{COM}(X) \neq \text{class}(X)$, then the least resistant adjustable member is adjusted.

2.9.5 The local adjustment algorithm for training a modified veto committee. Let T_X be a training sequence; let $\alpha_0 > 0$, $\beta > 0$, and $\gamma > -\alpha_0$ be constants. Form a modified veto committee having a single member of the first type with weight vector $W_1 = (0, 0, \ldots, 0, 1)$. Taking the samples of the training sequence in turn, for each pattern make a local adjustment to the committee. If at any point a new member is added to the committee, let the initial value of the weight vector be $(0, 0, \ldots, 0, 1)$ if the member is of the first type and $(0, 0, \ldots, 0, -1)$ if the member is of the second type. Adjustments continue until the recognition rate is 100%, or a certain number of adjustments have been made, or the committee has reached a certain size.

If line (2.9.3) were omitted, the adjustment described in

Definition 2.9.1 would, like all the adjustments to a TLU described so far, be of the form

$$W \leftarrow W - c\,TLU(X)X\,. \tag{2.9.5}$$

The effect of line (2.9.3) is to normalize the length of the weight vector to one after each adjustment. If such a normalization has been used before, it does not seem to be mentioned in the literature. We now give the heuristic reasons for the normalization.

First, Nilsson [24] reports that, during computer simulations of the local adjustment algorithm for training a majority committee, sometimes a member stops being adjusted because its weight vector becomes much longer than the weight vector of any other member, making its certainty at all the training patterns larger than the certainty of other members. Should this happen when the weight vector has assumed a value that prevents a solution of the problem, further training becomes futile. Fortunately, on a committee using modified veto logic a single member cannot block all further improvement in the recognition rate of the committee, but it can cause an excessive number of members to be added. Normalizing the weight vectors after adjustments eliminates this problem.

Second, there is the companion difficulty, not mentioned by Nilsson, of the weight vector of a member becoming so short that the member receives an inordinate number of adjustments. Figure 2.9.1

illustrates a circumstance in which the weight vector of a TLU becomes shorter with successive adjustments. The -'s and +'s denote pattern vectors in classes A and B respectively. All the patterns are on the line $y = 1$ as the last component of each pattern, in this instance the second component, is one. A TLU trained by the local adjustment algorithm of line (2.3.1) might be expected to reach quickly the position illustrated, with the weight vector W lying on the x-axis and the hyperplane through the origin defined by W being the line $x = 0$. The two circled patterns $\ominus$ and $\oplus$ are classified incorrectly, and an adjustment due to $\ominus$ has the form

$$W \leftarrow W - c\ominus$$

and due to $\oplus$ the form

$$W \leftarrow W + c\oplus .$$

Suppose that successive adjustments become smaller, as is the case if adjustments are defined by lines (2.9.1) and (2.9.2); then the sum of alternate adjustments to W due to $\ominus$ and $\oplus$ lies approximately along the x-axis, but in the opposite direction as W. W gets shorter and shorter with little change in direction until it points in the opposite direction, at which time $\ominus$ and $\oplus$ are classified correctly and the remaining patterns are misclassified. Further adjustments cause W to point again in its original direction but also make W still shorter.

So there would seem to be the danger that adjustments to a committee member due to the conflicting demands of patterns that are not linearly separable may cause the weight vector to become shorter and shorter. Normalizing the length of a weight vector after each adjustment eliminates this problem.

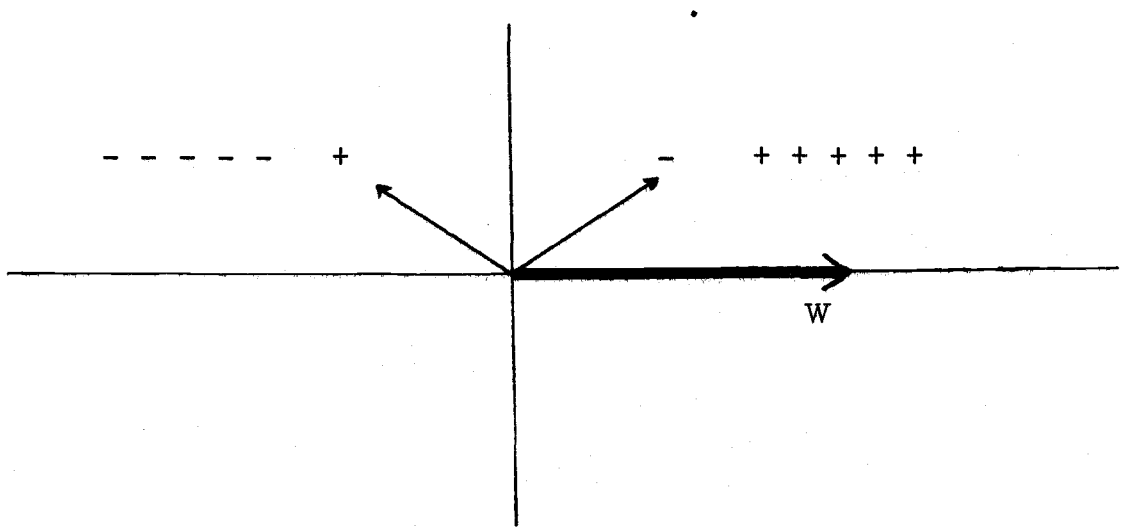


Figure 2.9.1. An inseparable 2-class problem.

Third, the effect of an adjustment to a member depends on the relative length of the adjustment vector and the weight vector. If the length of the weight vector is constant, then adjustments as defined on lines (2.9.1), (2.9.2), and (2.9.3) not only become successively smaller, but also the ratio of the length of the adjustment vector to the length of the weight vector becomes smaller, and successive adjustments have a diminishing effect on the weight vector. If on the other hand the weight vector is allowed to become arbitrarily short,

adjustments can cause dramatic and drastic changes in the response of the member, see Section 5.2 for an example.

Fourth, if $\|W\| = 1$, then the certainty at X, $|X \cdot W|$, equals the distance from X to the hyperplane through the origin defined by W, and the resistance at X is the product of age and distance. (We explain later the significance of the constant γ on line (2.9.4).)

Fifth, when the length of a weight vector W is normalized after each adjustment, the effect on W of an adjustment due to X depends inversely on $|X \cdot W|$. In Section 2.3 we cited Mueller's work [23] showing the advantage of making the size of an adjustment depend inversely on $|X \cdot W|$, see Figures 2.3.2 and 2.3.3 and the accompanying discussion. If the length of W before and after an adjustment is one, the only effect of an adjustment is to change the angle between X and W in a direction that decreases the certainty. The amount the angle changes and the amount the certainty decreases depend inversely on $|X \cdot W|$. For instance, when X and W are close to orthogonal, i.e., $|X \cdot W|$ is close to zero, the change is greatest; and when X and W are close to being scalar multiples of each other, i.e., $|X \cdot W|$ is close to $\|X\| \cdot \|W\|$, the change is smallest.

Normalizing the weight vector after each adjustment has two disadvantages. First, the amount of computation increases. Second, when training a single TLU, if the pattern vector is a scalar multiple

of the weight vector, successive adjustments due to the pattern do not change the response, so even if the patterns in a problem are linearly separable, the local adjustment algorithm may not converge to a solution.

We now examine the effect of varying the constants a_0, β, and γ. The length of the vector added to W decreases with each adjustment, and it is possible to choose a_0 and β so that the lengths of the first and m-th vector added to W are p/q and r/s respectively, where $p/q > r/s > 0$ and p/q, r/s are rational numbers. Let

$$k = rm/(sp-rq) ;$$

then

$$kp/(kq+m) = r/s .$$

If $\beta = kp$ and $a_0 = kq$, then the length of the first vector added is

$$\beta/a_0 = p/q ,$$

and the length of the m-th vector added is

$$\beta/(a_0+m) = kp/(kq+m) = r/s .$$

Before the first adjustment the resistance of a member is the product of the certainty and $(a_0+\gamma)$, and after the m-th adjustment the resistance is the product of the certainty and $(a_0+m+\gamma)$. Then for

any $a > 1$,

$$(a_0+m+\gamma)/(a_0+\gamma) = a$$

if

$$\gamma = (a_0(1-a)+m)/(a-1),$$

and it is possible to control how quickly the resistance of a member increases with age. The requirement that γ be greater than $-a_0$ means that

$$(a_0(1-a)+m)/(a-1) > -a_0 ,$$

i.e., that

$$(a_0(1-a)+m) > a_0(1-a),$$

which is always true.

2.10 Successive Adjustments Due to X

In order to show that the algorithm for training a modified veto committee is minimal, we now examine the effect on a single member and then on the committee as a whole of successive adjustments due to a fixed pattern X. If line (2.9.3) of Definition 2.9.1 is omitted, it is easy to show that a finite number of adjustments due to X changes the response of a member to X; but with line (2.9.3) included, this becomes more difficult and is established in Lemma 2.10.3.

If the criterion for choosing which member to adjust depended on certainty rather than resistance, we would be assured that, after a finite number of adjustments to the committee due to X, the response of the committee would be changed. The first adjustment would be to the least certain adjustable member, an adjustment to this member would either change its response or reduce its certainty, and all successive adjustments to the committee would also be to this member until, after a finite number of adjustments, the response of the member and the committee changed. As the criterion for choosing which member to adjust depends on resistance and as the resistance of a member at X may be larger after an adjustment than before, each adjustment to the committee may effect a different member or require the addition of a new member. Hence, if the first adjustment to a member is not large enough to change the member's response, the response of the committee may never change, no matter how many adjustments are made and how many members are added.

In Lemma 2.10.4 we show that, if γ is sufficiently large, an adjustment decreases the resistance of a member, and in Theorem 2.10.5 we show that, if γ is sufficiently large, the response of the committee changes after a finite number of adjustments. In Theorem 2.10.6 we express the γ of Lemma 2.10.4 and Theorem 2.10.5 in terms of a_0, β, and the dimension of the pattern space.

Let ΔABC denote a triangle with vertices A, B, and C; let AB denote the side connecting A and B; and let $\angle ABC$ denote the radian measure of the angle with vertex at B and with sides AB and BC.

2.10.1 Lemma. Let $\Delta A_1B_1C_1$ and $\Delta A_2B_2C_2$ be plane triangles; let $A_1B_1 = A_2B_2$, let $B_1C_1 = B_2C_2$, let $\angle A_2C_2B_2$ be obtuse, and let $\angle A_2B_2C_2 > \angle A_1B_1C_1$. Then $\angle B_2A_2C_2 > \angle B_1A_1C_1$.

Proof. We refer to Figure 2.10.1. As $\angle A_2B_2C_2 > \angle A_1B_1C_1$, there is a point D on A_2C_2 such that $\angle A_2B_2D = \angle A_1B_1C_1$. As $\angle A_2C_2B_2$ is obtuse, $B_2D > B_2C_2$, and there is a point E on B_2D such that $B_2E = B_2C_2$. As ΔA_2B_2E is congruent to $\Delta A_1B_1C_1$ and as $\angle B_2A_2C_2 > \angle B_2A_2E$, then $\angle B_2A_2C_2 > \angle B_1A_1C_1$. ●

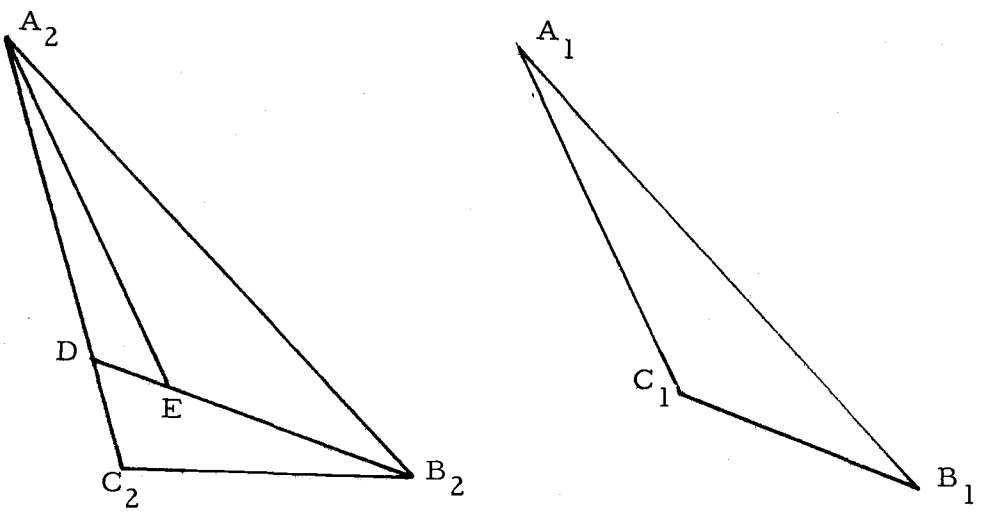


Figure 2.10.1. Two triangles.

2.10.2 Lemma. Let $\{\Delta A_i B_i C_i\}_{i=1}^{\infty}$ be a sequence of plane triangles; for all i let $\angle A_i C_i B_i$ be obtuse, let $A_i B_i = \xi_1$, let $\angle A_i B_i C_i = \xi_2$, where ξ_1 and ξ_2 are constants, and let $\sum_{i=1}^{\infty} B_i C_i = \infty$. Then $\sum_{i=1}^{\infty} \angle B_i A_i C_i = \infty$.

Proof. By the law of sines

$$\frac{B_i C_i}{\sin \angle B_i A_i C_i} = \frac{A_i C_i}{\sin \angle A_i B_i C_i},$$

so that

$$B_i A_i C_i = \sin^{-1} \frac{B_i C_i \sin \xi_2}{A_i C_i}$$

$$\geq \sin^{-1} \frac{B_i C_i \sin \xi_2}{A_i B_i}$$

$$\geq \frac{B_i C_i \sin \xi_2}{\xi_1}$$

The lemma follows immediately. ●

2.10.3 Lemma. After a finite number of adjustments to a member due to a pattern X, the response of the member to X changes, provided that X is not a scalar multiple of the member's weight vector.

Proof. Let W denote the weight vector. If W is not a

scalar multiple of X, then adding a multiple of X to W changes the angle between X and W in a direction that decreases $|X \cdot W|$ or changes the sign of $X \cdot W$. We show that, after a finite number of adjustments, the angle changes sufficiently to change the response to X. If X and W are orthogonal, then a single adjustment changes the response.

We refer to Figure 2.10.2. Assume that X and W are not orthogonal and the response is not changed by a finite number of adjustments. Let W_0 and W_i denote the initial value of the weight vector and the value after the i-th adjustment respectively. Let α_i' denote the age after the i-th adjustment, and let

$$\lambda_i = \frac{\beta \text{MMBR}(X)}{\alpha_i' \| X \|} .$$

(Note that W_0 does not necessarily equal the value of the weight vector of a new member as the member may have been adjusted prior to the moment we are speaking of. Similarly, α_0' does not necessarily equal α_0.) Let $\mu = \text{sgn}(X \cdot W_0)$. As X, W_0 and W_i lie in the same plane and as $\text{sgn}(X \cdot W_i) = \text{sgn}(X \cdot W_0)$, then for all i

$$\angle (W_0)(X)(W_i) < \pi/2, \tag{2.10.1}$$

where Z is the zero vector and (V) denotes the head of a vector V. As

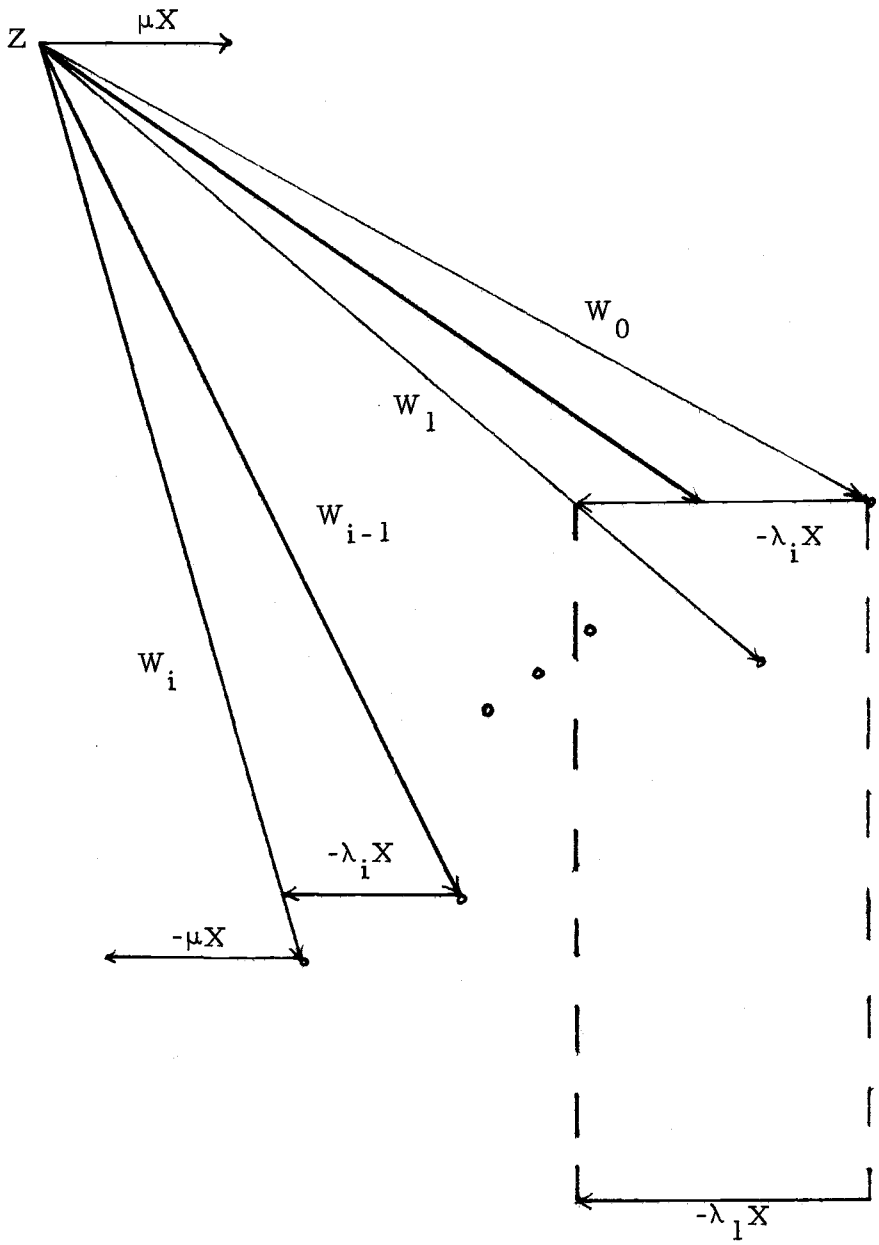


Figure 2.10.2. Adjustments to W due to X.

$$\angle(W_0)(Z)(W_i) = \sum_{j=1}^{i} \angle(W_{j-1})(Z)(W_j) ,$$

line (2.10.1) implies

$$\sum_{i=1}^{\infty} \angle(W_{i-1})(Z)(W_i) \leq \pi/2. \qquad (2.10.2)$$

With the aid of Lemma 2.10.1 we now show that

$$\angle(W_{i-1})(Z)(W_i) > \angle(W_0)(Z)(W_0 - \lambda_i X) \qquad (2.10.3)$$

and with the aid of Lemma 2.10.2 that

$$\sum_{i=1}^{\infty} \angle(W_0)(Z)(W_0 - \lambda_i X) = \infty. \qquad (2.10.4)$$

Hence, the inequality in line (2.10.2) and the assumption that the response is not changed by a finite number of adjustments will be shown to be false.

We now establish the inequality on line (2.10.3). As we are assuming that the response is not changed by any number of adjustments and as the effect of each adjustment further decreases the certainty, given that X is not a scalar multiple of W_0, then

$$\pi/2 > \angle(X)(Z)(W_i) > \angle(X)(Z)(W_{i-1}) > 0 \quad \text{if} \quad X \cdot W_0 > 0,$$

and

$$\pi/2 < \angle(X)(Z)(W_i) < \angle(X)(Z)(W_{i-1}) < \pi \quad \text{if} \quad X \cdot X_0 < 0,$$

which implies

$$\pi/2 > \angle(\mu X)(Z)(W_i) > \angle(\mu X)(Z)(W_{i-1}) > 0.$$

As

$$\angle(\mu X)(Z)(W_i) = \angle(Z)(W_i)(W_i - \mu X),$$

then

$$\pi/2 > \angle(Z)(W_i)(W_i - \mu X) > \angle(Z)(W_{i-1})(W_{i-1} - \mu X) > 0, \qquad (2.10.5)$$

which implies

$$\angle(Z)(W_{i-1})(W_{i-1} - \mu X) > \angle(Z)(W_0)(W_0 - \mu X). \qquad (2.10.6)$$

Now

$$\angle(Z)(W_i)(W_i - \mu X) = \angle(Z)(W_{i-1} - \lambda_i X)(W_{i-1} - \lambda_i X - \mu X)$$

$$= \angle(Z)(W_{i-1} - \lambda_i X)(W_{i-1} - \lambda_i X - \lambda_i X),$$

and the complement of

$$\angle(Z)(W_{i-1} - \lambda_i X)(W_{i-1} - \lambda_i X - \lambda_i X)$$

is

$$\angle(Z)(W_{i-1} - \lambda_i X)(W_{i-1} - \lambda_i X + \lambda_i X) = \angle(Z)(W_{i-1} - \lambda_i X)(W_{i-1}),$$

so by line (2.10.5)

$$\pi > \angle(Z)(W_{i-1} - \lambda_i X)(W_{i-1}) > \pi/2. \qquad (2.10.7)$$

Referring to Lemma 2.10.1, we let

$$A_1 = A_2 = (Z),$$

$$B_1 = (W_0), \quad B_2 = (W_{i-1}),$$

$$C_1 = (W_0 - \lambda_i X), \quad C_2 = (W_{i-1} - \lambda_i X).$$

By lines (2.10.6) and (2.10.7) and as $\|W_0\| = \|W_{i-1}\|$, the hypotheses of Lemma 2.10.1 are satisfied and the inequality on line (2.10.3) follows.

Now we establish the equality on line (2.10.4). By line (2.10.7) and as $\lambda_i < \lambda_1$,

$$\pi > \angle(Z)(W_0 - \lambda_i X)(W_0) > \pi/2. \qquad (2.10.8)$$

Referring to Lemma 2.10.2, we let

$$A_i = (Z),$$

$$B_i = (W_0),$$

$$C_i = (W_0 - \lambda_i X).$$

By line (2.10.8) and as $\sum_{i=1}^{\infty} \lambda_i = \infty$, the hypotheses of Lemma 2.10.2 are satisfied, and the equality on line (2.10.4) follows. ●

2.10.4 Lemma. If X is not a scalar multiple of the weight

vector of a member and if γ is sufficiently large, see line (2.9.4), then, until the response changes, each adjustment to the member due to X decreases the resistance at X.

Proof. By Lemma 2.10.3 the response to X changes after a finite number of adjustments due to X. Let k be the index of the adjustment which changes the response. Each adjustment preceding the k-th decreases the certainty. Let δ denote the smallest of these decreases, a'_0 and W_0 the age and the weight vector respectively prior to the first adjustment due to X, and a'_i and W_i the age and the weight vector respectively after the i-th adjustment due to X. If $i < k$, then the difference in the resistance before and after the i-th adjustment is

$$
\begin{aligned}
&|X \cdot W_{i-1}|(a'_{i-1}+\gamma) - |X \cdot W_i|(a'_i+\gamma) \\
&= |X \cdot W_{i-1}|(a'_{i-1}+\gamma) - |X \cdot W_i|(a'_{i-1}+1+\gamma) \\
&= (a'_{i-1}+\gamma)(|X \cdot W_{i-1}| - |X \cdot W_i|) - |X \cdot W_i| \\
&> (a'_0+\gamma)\delta - |X \cdot W_0| .
\end{aligned}
$$

Hence, if

$$
\gamma > \frac{|X \cdot W_0|}{\delta} - a'_0 ,
$$

the resistance decreases. ●

Note that in Lemma 2.10.4, γ depends on $|X \cdot W_0|$, δ and a_0', while δ depends on the quantities a_0', β and the angle between X and W_0. Of the three only β is independent of the previous history of adjustments to the member. If the member is new at the time of the first adjustment due to X, then γ depends on only a_0, β and the angle between X and the vector $(0, 0, \ldots, 0, 1)$. If in addition each component of X equals ± 1, we can express γ in terms of a_0, β and the dimension of the pattern space, see Theorem 2.10.6.

2.10.5 Theorem. The response of a modified veto committee to X is changed after a finite number of adjustments due to X with the addition of at most one new member to the committee, provided that X is not a scalar multiple of $(0, 0, \ldots, 0, 1)$ and that γ is sufficiently large.

Proof. Let γ be large enough to insure that Lemma 2.10.4 applies to a new member adjustable in response to X. Each of the successive adjustments due to X is made to the least resistant adjustable member. Suppose that at some point a new member is chosen for adjustment. Each adjustment to the new member reduces its resistance at X, Lemma 2.10.4, and it will continue to be the least resistant adjustable member until, after a finite number of adjustments, its response and consequently the response of the

committee change, Lemma 2.10.3.

If a new member is never added, after a finite number of adjustments the response of one of the adjustable members already on the committee changes, Lemma 2.10.3, and so does the response of the committee. ●

2.10.6 Theorem. Let MMBR be a new member and let X be a pattern of n components, each equal to ±1. If

$$\beta > \frac{1}{\sqrt{n}} \quad \text{and} \quad a_0 > \frac{-\sqrt{n}\,\beta\gamma}{\sqrt{n}\,\beta - 1},$$

then, until the response changes, each adjustment to MMBR due to X decreases the resistance at X.

Proof. Let a_0 and W_0 be the initial values of the age and the weight vector respectively; and let a_i, W_i, and $rest_i(X)$ be the values of the age, the weight vector, and the resistance respectively after the i-th adjustment. For notational simplicity and WLOG we assume that every component of X and the last component of W_0 is +1. Assume that the response is changed by the k-th adjustment and that $i < k$. Then for some $a > 0$,

$$W_{i-1} = (-a, -a, \ldots, -a, 1-a),$$

$$X \cdot W_{i-1} = 1 - na > 0,$$

$$rest_{i-1}(X) = (1-na)(\alpha_{i-1}+\gamma),$$

and for some $b > 0$,

$$W_i = (-b, -b, \ldots, -b, 1-b),$$

$$X \cdot W_i = 1 - nb > 0, \qquad (2.10.9)$$

$$rest_i(X) = (1-nb)(\alpha_i+\gamma) .$$

The difference in the resistance before and after the i-th adjustment is

$$rest_{i-1}(X) - rest_i(X) = nb(\alpha_i+\gamma) - na(\alpha_{i-1}+\gamma) - 1$$

and is positive if

$$b > a + \frac{1}{n(\alpha_i+\gamma)} . \qquad (2.10.10)$$

We now determine the conditions needed to establish the inequality on line (2.10.10). As the i-th adjustment consists of subtracting a scalar multiple of X from W_{i-1}, for some scalar $c > 0$, followed by a normalization of the resulting vector, then for $c = \beta/(\alpha_i \| X \|)$

$$b = (a+c)/\{(n-1)(a+c)^2+(1-a-c)^2\}^{1/2} .$$

Now

$$\{(n-1)(a+c)^2+(1-a-c)^2\}^{1/2} > 1$$

implies

$$n(a+c) > 2$$

implies

$$X \cdot (-a-c, \ldots, -a-c, 1-a-c) = 1 - n(a+c) < 0$$

implies

$$X \cdot W_i < 0$$

which contradicts line (2.10.9). Hence,

$$b \geq a + c. \tag{2.10.11}$$

If

$$\beta > \frac{1}{\sqrt{n}} \quad \text{and} \quad \alpha_0 > \frac{-\sqrt{n}\,\beta\gamma}{\sqrt{n}\,\beta - 1},$$

then

$$\frac{\beta}{\alpha_i} > \frac{1}{\sqrt{n}\,(\alpha_i+\gamma)},$$

and

$$c = \frac{\beta}{\alpha_i \|X\|} = \frac{\beta}{\alpha_i\sqrt{n}} > \frac{1}{n(\alpha_i+\gamma)} \quad . \tag{2.10.12}$$

The inequalities on lines (2.10.11) and (2.10.12) imply that the inequality on line (2.10.10) is true and that the resistance is decreased by the i-th adjustment. ●

III. ESTIMATION

3.1 The Problem of Estimation

A more general problem than multiclass classification is that of function estimation. Let R be the set of real numbers, let f be a function from a subset of R^n to R, and let $TR = \{(T_j, y_j)\}_{j=1}^{N'}$ be a set of observations, called the training set, where T_j is in the domain of f and y_j is in R. The relationship between y_j and $f(T_j)$ is

$$y_j = f(T_j) + e_j, \quad j = 1, \ldots, N' ,$$

where the e_j's are some unknown errors introduced during the process of making observations and hence can be expected to have mean zero and have some unknown variance uncorrelated to the y_j's. The problem of estimation is to infer from the training set a function g which estimates f well, i.e., for which $\|g-f\|$ is small for some function norm. As the function f is not known, otherwise there would be no problem, the criterion $\|g-f\|$, though ideal, cannot be used, and instead $\|(y_j)_{j=1}^{N'} - (g(T_j))_{j=1}^{N'}\|$ is used for some vector norm. Because of its convenience in statistical analysis, the norm is usually chose to be the Euclidean norm. It is important that the training set be representative of the function f and that the

process of inference not produce a complex function which is valid on the training set but not elsewhere. A measure of the generality of g is given by $\| (z_j)_{j=1}^{M} - (g(S_j))_{j=1}^{M} \|$, where $TE = \{S_j, z_j\}_{j=1}^{M}$ is some test set drawn from the same population as the training set.

If the form of f is known, i.e., only the values of some parameters need to be determined, if $y_j = f(T_j)$ for all j, and if N' is sufficiently large, it may be possible to find $g \equiv f$, within the limits of computational accuracy. For example we can find f directly if f is a polynomial, using Lagrange's interpolating polynomial; or if f is a linear transformation, we can find f indirectly using a gradient technique. Of course as the y_j's deviate from the $f(T_j)$'s, g deviates from f. If the points in the training set do not fit any function of the proposed form, there may be statistical techniques for finding the function g of the proposed form which best fits the training data in some sense (e.g., linear and non-linear regression).

Suppose the form of f is not known. If $\{\psi_i\}_{i=1}^{\infty}$ is a set of orthonormal functions spanning a space of functions which can reasonably be expected to include f, then g may be set equal to the linear combination of ψ_i's which best fits the training data. Alternatively, for any point X in the domain of f, $g(X)$ can be set equal to a weighted average of the y_j's (e.g., linear interpolation). A very simple scheme for choosing weights is to let the weight

of the y_j for which T_j is closest to X equal one and all other weights equal zero. For this scheme if f is continuous, if the domain of f is closed and bounded and if $y_j = f(T_j)$, then for any L_p-norm $\|g-f\|_p$ goes to zero as $\{T_j\}_{j=1}^{N'}$ becomes dense in the domain of f. A technique which computes $g(X)$ as a weighted average of y_j's becomes impractical as N' increases unless there is a method for summarizing the information contained in the training data.

In this chapter we show how 2-class classifiers can be used to summarize the information in the training set and how this summarized information can be used to construct an estimate g of f that is a weighted average of points in the range of f. The estimate g will be statistical in an indirect sense inasmuch as the 2-class classifiers are statistical, but not directly as are linear and nonlinear regression, both of which find the estimate of some pre-defined form that fits the training data best in the least-squares sense. Because 2-class classifiers can be trained without a priori information about the form of f, g is independent of the form of f. Theorems will be proven describing conditions under which g is continuous and converges to f. Also the effect on g of small changes in the 2-class classifiers will be examined.

3.2 Kernel Functions

Here and throughout the chapter all integrals will be Lebesgue integrals, the norm of a vector in R^n will be the Euclidean norm, h will denote a positive real number, and for $a > 0$ and X in R^n let $A_a(X) = \{z : \|Z-X\| \geq a\}$ and $B_a(X) = R^n \setminus A_a(X)$. A continuous and bounded function k from R^n to the nonnegative real numbers R^+ for which $\int_{R^n} k(Z)dZ = 1$ will be called a kernel function. When $n = 1$ and we wish to draw attention to the fact, we will use the letter ℓ rather than the letter k to denote a kernel function. Several properties of kernel functions that will be useful later are:

(i) $$\int_{R^n} (1/h^n)k((Z-X)/h)dZ = \int_{R^n} k(Z)dZ ;$$

(ii) $$\int_{A_a(X)} (1/h^n)k((Z-X)/h)dZ = \int_{A_a(0)} (1/h^n)k(Z/h)dZ$$

$$= \int_{A_{a/h}(X)} k(Z-X)dZ = \int_{A_{a/h}(0)} k(Z)dz ;$$

(iii) $$\int_{B_a(X)} (1/h^n)k((Z-X)/h)dZ = \int_{B_a(0)} (1/h^n)k(Z/h)dZ$$

$$= \int_{B_{a/h}(X)} k(Z-X)dZ = \int_{B_{a/h}(0)} k(Z)dZ ;$$

(iv) $\lim_{h \to 0} \int_{A_{a/h}(0)} k(Z)dZ = \lim_{a \to \infty} \int_{A_{a/h}(0)} k(Z)dZ = 0$ and

$\lim_{h \to 0} \int_{B_{a/h}(0)} k(Z)dZ = \lim_{a \to \infty} \int_{B_{a/h}(0)} k(Z)dZ = 1$;

(v) $k_h(Z) = (1/h^n)k(Z/h)$ and $k_{X,h}(Z) = (1/h^n)k((Z-X)/h)$

define kernel functions;

(vi) if f is a continuous and bounded function from R^n to R, then the convolution of f and k_h converges pointwise to f as h converges to zero; if f is uniformly continuous, the convergence is uniform;

(vii) $\int_{-\infty}^{0} (1/h)\ell((z-x)/h)dz = \int_{-\infty}^{0} (1/h)\ell(z/h)dz$;

(viii) when $a > 0$,

$\lim_{h \to 0} \int_{-\infty}^{-a} (1/h)\ell(z/h)dz = \lim_{h \to 0} \int_{a}^{\infty} (1/h)\ell(z/h)dz = 0$ and

$\lim_{h \to 0} \int_{-a}^{a} (1/h)\ell(z/h)dz = 1$;

(ix) if $a > 0$ and if $\gamma > 0$, where

$\gamma = \min\{\int_{-\infty}^{0} \ell(z)dz, \int_{0}^{\infty} \ell(z)dz\}$, then

$$\lim_{h \to 0} \int_{-a}^{0} (1/h)\ell(z/h)dz > \gamma/2 , \quad \text{and}$$

$$\lim_{h \to 0} \int_{0}^{a} (1/h)\ell(z/h)dz > \gamma/2 .$$

Because the techniques involved will be useful later, we now indicate the usual proof of (vi), see [10] for instance. Let $\epsilon > 0$ and X be given and let $\delta > 0$ be sufficiently small to insure that $|f(X)-f(Z)| < \epsilon/2$ whenever $\|X-Z\| < \delta$. If f is uniformly continuous, the choise of δ does not depend on X. If $b = \sup\{|f(X)| : X \text{ in } R^n\}$, let h be sufficiently small to insure that $\int_{A_\delta(0)} (1/h^n)k(Z/h)dZ < \epsilon/(4b)$. Then

$$|f(X) - \int_{R^n} f(Z)(1/h^n)k((Z-X)/h)dZ|$$

$$\leq \int_{A_\delta(X)} |f(X)-f(Z)|(1/h^n)k((Z-X)/h)dZ$$

$$+ \int_{B_\delta(X)} |f(Z)-f(Z)|(1/h^n)k(Z-X)/h)dZ$$

$$\leq 2b\int_{A_{\delta/h}(0)} k(Z)dZ + (\epsilon/2)\int_{B_{\delta/h}(0)} k(Z)dz$$

$$< \epsilon .$$

Henceforth, when confusion seems unlikely, we will write $\int_{S_1} k(X,h)$ and $\int_{S_2} \ell(x,h)$ for $\int_{S_1} (1/h^n)k((Z-X)/h)dz$ and $\int_{S_2} (1/h)\ell((z-x)/h)dz$ respectively, where S_1 is a Lebesgue measurable subset of R^n and S_2 of R.

It will strengthen our intuitive understanding of what follows to think of the kernel function $k(Z)$ as a nonincreasing function of $\|Z\|$, maximum at the origin and decreasing to zero as $\|Z\|$ diverges to infinity. Under these circumstances $k_{X,h}(Z)$ is a non-increasing function of $\|Z-X\|$, maximum at X and decreasing to zero as $\|Z-X\|$ diverges to infinity. If we think of $\int_S k(X,h)$ as a measure of the "attraction" between a point X and a measurable subset S of R^n, then (iv) above implies that, as h converges to zero, the attraction of X to S converges to one whenever X is an interior point of S and converges to zero whenever X is an interior point of the complement of S.

3.3 The First Method of Estimating f

By statement (vi) of Section 3.2 the convolution of f and k_h converges pointwise to f as h converges to zero whenever f is a continuous, real valued, bounded function defined on R^n. (From the proof of (vi) it can be seen that the result also holds when the domain of f is a proper subset of R^n at points X interior to the

domain.) The convolution can be approximated by

$$g_1(X) = \sum_{i=1}^{N} b_i u_i(X) , \tag{3.3.1}$$

where

$$u_i(X) = \int_{R_i} k(X,h) , \tag{3.3.2}$$

b_i is in the range of f, R_i is a subset of the domain of f on which $f(Z)$ is close to b_i, and the R_i's are pairwise disjoint. For example let $\{I_i\}_{i=1}^{N}$ be a set of pairwise disjoint intervals whose union includes the range of f, where for later convenience we adopt the convention that interval I_i is to the left of interval I_{i+1}, $i = 1,\ldots,N-1$; let each

$$R_i = f^{-1}(I_i) ; \tag{3.3.3}$$

and let each

$$b_i = \begin{cases} (r_i+s_i)/2 & \text{if} \quad -\infty < r_i, s_i < \infty \\ s_1 - c & \text{if} \quad r_1 = -\infty \\ r_N + c & \text{if} \quad s_N = \infty , \end{cases} \tag{3.3.4}$$

where r_i, s_i are the left and right end point respectively of I_i and where c is a positive constant. g_1 is an estimate of f of the

form suggested at the end of Section 3.1. If $\bigcup_{i=1}^{N} R_i$ is a proper subset of R^n, then

$$\sum_{i=1}^{N} u_i(X) = \int_{\bigcup_{i=1}^{N} R_i} k(X,h) \qquad (3.3.5)$$

may be less than one, and $g_1(X)$ may not be a weighted average of the b_i's. However, if X is interior to $\bigcup_{i=1}^{N} R_i$, $\lim_{h \to 0} \sum_{i=1}^{N} u_i(X) = 1$, and $g_1(X)$ approaches a weighted average of the b_i's.

When we wish to estimate f, we are unlikely to have the information needed to determine the $f^{-1}(I_i)$'s, while the information we are likely to have is the value of f at some points in the domain of f. Suppose then that we are given an N-class classifier and a set of pairs $\{(T_j, y_j)\}_{j=1}^{N'}$ satisfying $y_j \simeq f(T_j)$ and $y_j \leq y_{j+1}$, $j = 1, \ldots, N'-1$. Let

$$\{T_j\}_{j=\ell_{i-1}+1}^{\ell_i} \qquad (3.3.6)$$

be the training samples for the i-th class, where

$$\ell_0 = 0, \quad \ell_N = N' \quad \text{and} \quad \ell_i < \ell_{i+1}, \quad i = 1, \ldots, N-1, \qquad (3.3.7)$$

and let each

$$b_i = \psi_i(y_j : j = 1, \ldots, N'), \tag{3.3.8}$$

where ψ_i is some function of its arguments, for instance the median of the numbers y_j, $j = \ell_{i-1}+1, \ldots, \ell_i$. After training, the classifier defines a partition of R^n, and if the partition is Lebesgue measurable, let

$$R_i = \{Z : \text{class}(Z) = i\}. \tag{3.3.9}$$

As $\bigcup_{i=1}^{N} R_i = R^n$, $g_1(X)$ is a weighted average of the b_i's, see line (3.3.5). For instance, the nearest-neighbor-classifier [10] defines regions

$$R_i = \bigcup_{j=\ell_{i-1}+1}^{\ell_i} \text{nn}(T_j), \tag{3.3.10}$$

where for $j = 1, \ldots, N'$

$$\begin{aligned} \text{nn}(T_j) = \{Z : \|Z-T_j\| < \|Z-T_k\| \text{ for all } k \neq j\} \\ \cup \{Z : \|Z-T_j\| = \|Z-T_k\| \text{ for all } k < j\}. \end{aligned} \tag{3.3.11}$$

Given $N-1$ 2-class classifiers instead of one N-class classifier, let the b_i's be as defined on line (3.3.8), and let the training samples of the i-th 2-class problem be

$$\{T_j\}_{j=1}^{\ell_i} \quad \text{and} \quad \{T_j\}_{j=\ell_i+1}^{N'} \tag{3.3.12}$$

for the first and second class respectively. After training, the classifiers define sets

$$A_i = \{Z : \text{class}_i(Z) = -1\}, \quad i = 1, \ldots, N-1, \tag{3.3.13}$$

where $\text{class}_i(Z)$ is the decision of the i-th 2-class classifier. In addition let

$$A_0 = \phi \quad \text{and} \quad A_N = R^n . \tag{3.3.14}$$

If the A_i's are measurable, let

$$u_i(X) = \int_{A_i} k(X, h) - \int_{A_{i-1}} k(X, h). \tag{3.3.15}$$

$\sum_{i=1}^{N} u_i(X) = 1$ for all X, and whenever $A_i \subseteq A_{i+1}$, $i = 1, \ldots, N-1$, each $u_i(X)$ is nonnegative for all X. From the sets A_i, sets A_i' can be defined for which $A_i' \subseteq A_{i+1}'$, for instance by letting

$$A_i' = \bigcup_{j=0}^{i} A_i , \tag{3.3.16}$$

insuring that

$$u_i(X) = \int_{A_i'} k(X, h) - \int_{A_{i-1}'} k(X, h) \tag{3.3.17}$$

is nonnegative for all X.

To compute $g_1(X)$ it is necessary to evaluate integrals over subregions of R^n, and this can be done by evaluating the equivalent multiple integrals. A simpler computational scheme is possible if each 2-class classifier is defined by a discriminant function d_i, $i = 1, \ldots, N-1$. For $i = 1, \ldots, N-1$, let

$$A_i = \{Z : d_i(Z) < 0\}. \tag{3.3.18}$$

Then d_i maps A_i into $(-\infty, 0)$ and X to $d_i(X)$, which suggests letting

$$\begin{aligned} u_i(X) &= \int_{-\infty}^{0} \ell(d_i(X), h) - \int_{-\infty}^{0} \ell(d_{i-1}(X), h) \\ &= \int_{-\infty}^{-d_i(X)} \ell(0, h) - \int_{-\infty}^{-d_{i-1}(X)} \ell(0, h) \\ &= \int_{-d_{i-1}(X)}^{-d_i(X)} \ell(0, h) , \end{aligned} \tag{3.3.19}$$

where $i = 1, \ldots, N$, $d_0(X) = \infty$ and $d_N(X) = -\infty$. $\sum_{i=1}^{N} u_i(X) = 1$, and if $d_{i-1}(X) \geq d_i(X)$, then $u_i(X)$ is nonnegative. In the definition of $u_i(X)$ we have replaced a term expressing the attraction of X to A_i by a term expressing the attraction of $d_i(X)$ to $(-\infty, 0)$. Even if the kernel functions k and ℓ are the same nonincreasing

function of the Euclidean norm of their argument, there is no reason to suppose that $\int_{A_i} k(X, h)$ equals $\int_{-\infty}^{0} \ell(d_i(X), h)$, but it is reasonable to hope that the value of the two integrals is highly correlated. From the functions d_i, functions d_i' can be defined for which $d_{i-1}'(X) \geq d_i'(X)$, for instance by letting

$$d_i'(X) = \min\{d_j(X) : j = 1, \ldots, i\}, \tag{3.3.20}$$

insuring that

$$u_i(X) = \int_{-d_{i-1}'(X)}^{-d_i'(X)} \ell(0, h) \tag{3.3.21}$$

is positive for all X.

3.4 The Second Method of Estimating f

Computer simulations testing the effectiveness of the estimate g_1 indicate that, for those values of h which minimize the difference between f and g simultaneously at a number of different points in the domain of f, where f is concave upward g tends to be greater than f and where f is concave downward less. We now examine a simple example that reveals the probable reason for this phenomenon.

Let I_k, I_{k+1}, I_{k+2} be three contiguous intervals of equal length in the range of f, and suppose that $f : R \to R$ is continuous,

increasing and concave upward on $f^{-1}(I_k \cup I_{k+1} \cup I_{k+2})$. Then $f^{-1}(I_k)$, $f^{-1}(I_{k+1})$, $f^{-1}(I_{k+2})$ are three contiguous intervals in the domain of f, and the length of $f^{-1}(I_k)$ is greater than the length of $f^{-1}(I_{k+1})$ which in turn is greater than the length of $f^{-1}(I_{k+2})$. Let the kernel function be

$$\ell(z) = \begin{cases} 1 - |z| & \text{if } |z| \leq 1 \\ 0 & \text{otherwise,} \end{cases}$$

and suppose we wish to estimate f at the point x which separates $f^{-1}(I_k)$ from $f^{-1}(I_{k+1})$ using Equations (3.3.1), (3.3.2), (3.3.3) and (3.3.4). Figure 3.4.1 summarizes these assumptions and also indicates the graph of $(1/h)\ell((z-x)/h)$ and the attraction of x to the intervals $f^{-1}(I_k)$ and $f^{-1}(I_{k+1})$. Figure 3.4.2 is the same as Figure 3.4.1 except that h is larger and the attraction of x to $f^{-1}(I_{k+2})$ is nonzero. In Figure 3.4.1 the attractions of x to $f^{-1}(I_k)$ and $f^{-1}(I_{k+1})$ are both $1/2$, and

$$g_1(x) = \frac{1}{2} b_k + \frac{1}{2} b_{k+1}$$

$$= f(x),$$

as the intervals I_k, I_{k+1}, I_{k+2} are of equal length and b_k, b_{k+1}, b_{k+2} are their respective midpoints. In Figure 3.4.2 the attractions of x to $f^{-1}(I_k)$, $f^{-1}(I_{k+1})$ and $f^{-1}(I_{k+2})$ are $1/2$,

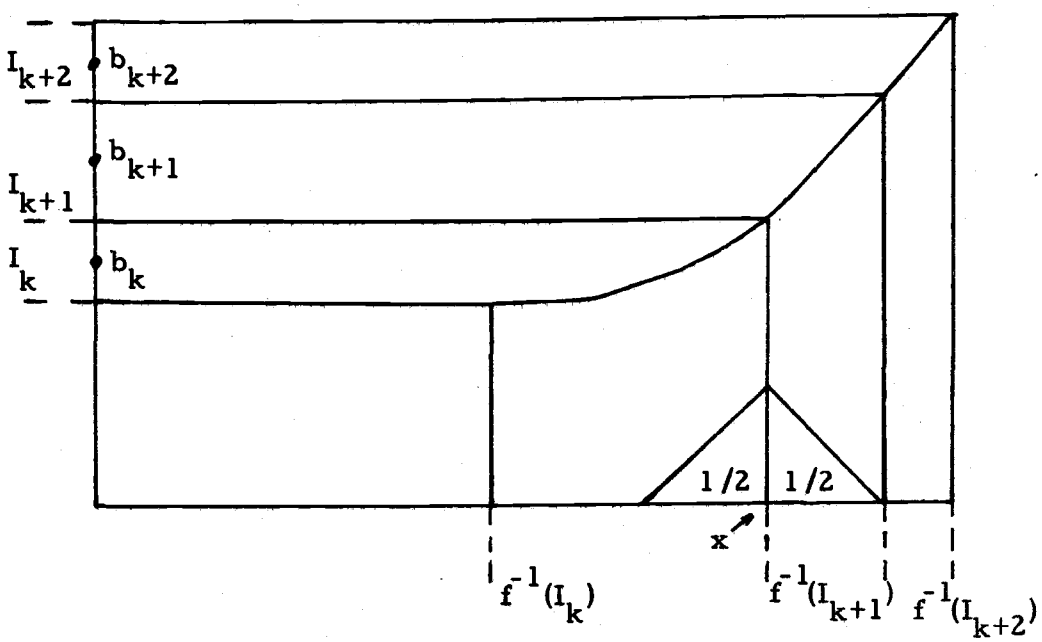


Figure 3.4.1. Attraction when h is small.

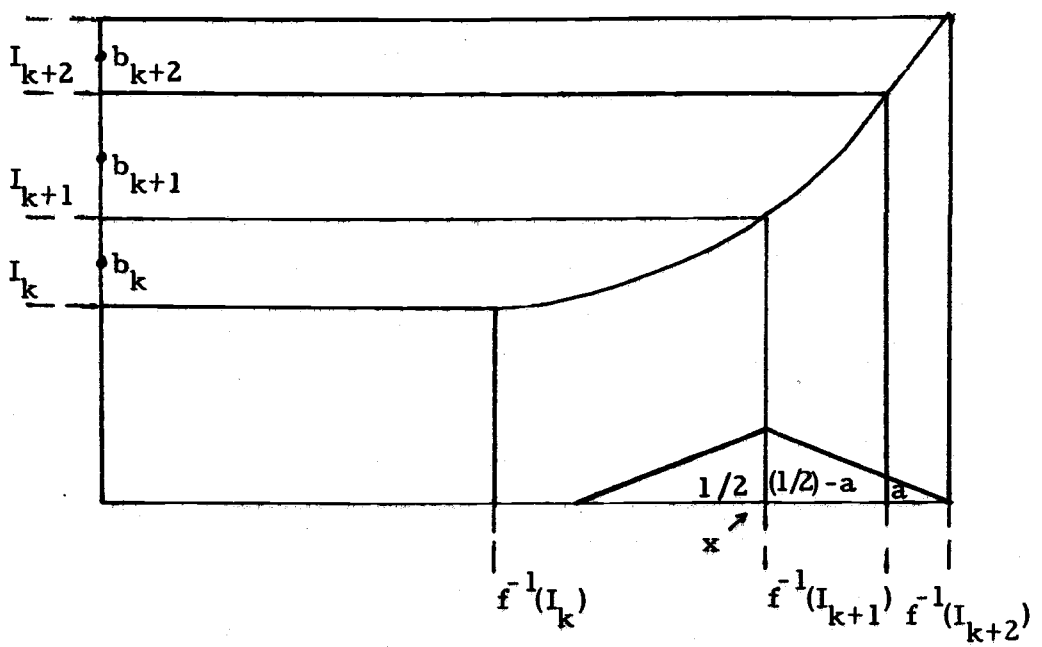


Figure 3.4.2. Attraction when h is large.

$(1/2) - a$ and a respectively, and

$$g_1(x) = \frac{1}{2} b_k + (\frac{1}{2} - a)b_{k+1} + ab_{k+2}$$

$$= \frac{1}{2} b_k + \frac{1}{2} b_{k+1} + a(b_{k+2} - b_{k+1})$$

$$= f(x) + a\Delta b, \qquad (3.4.1)$$

where $\Delta b = b_{k+2} - b_{k+1} = b_{k+1} - b_k$. At x, g_1 overestimates f by the amount $a\Delta b$. Similar examples can be constructed if f is decreasing and concave upward, increasing and concave downward, or decreasing and concave downward.

We now propose an estimate g_2 of f that has less the tendency to over and underestimate f when f is concave upward and downward respectively. In the rest of the chapter we will assume that $b_i < b_{i+1}$, $i = 1, \ldots, N-1$. Let X be in R^n; for $i = 1, \ldots, N$ let u_i be as defined in any of the several ways mentioned in Section 3.3; let

$$u_0 \equiv 0 ; \qquad (3.4.2)$$

let

$$v_i(X) = \sum_{j=0}^{i} u_j(X), \quad i = 0, \ldots, N; \qquad (3.4.3)$$

let

$$w_i(X) = \prod_{j=0}^{i-1} (1 - v_j(X)) \prod_{j=i}^{N} v_j(X) , \quad i = 1, \ldots, N; \qquad (3.4.4)$$

and let

$$g_2(X) = \sum_{i=1}^{N} b_i w_i(X) \Big/ \sum_{i=1}^{N} w_i(X) \qquad (3.4.5)$$

whenever $\sum_{i=1}^{N} w_i(X) \neq 0$. For each of the definitions of the u_i's given in Section 3.3, $0 \leq v_i(X) \leq 1$, implying that each $w_i(X) \geq 0$ and that $g_2(X)$, when defined, is a weighted average of the b_i's.

We now examine the circumstances under which $\sum_{i=1}^{N} w_i(X) = 0$.

3.4.1 Theorem. $\sum_{i=1}^{N} w_i(X) = 0$ if and only if

(i) there are indices j_1 and j_2 with $j_1 < j_2$ such that

$v_{j_1}(X) = 1$ and $v_{j_2}(X) = 0$ or

(ii) $v_N(X) = 0$.

Proof. Suppose (i) is true. For $i \leq j_2$, $w_i(X)$ is a multiple of $v_{j_2}(X)$, and for $i > j_1$, $w_i(X)$ is a multiple of $1 - v_{j_1}(X)$. Hence, each $w_i(X) = 0$.

Suppose (ii) is true. Each $w_i(X)$ is a multiple of $v_N(X)$ and equals zero.

Conversely, suppose that (i) and (ii) are false. If any $v_i(X) = 1$, let $\ell = \min\{i : v_i(X) = 1\}$. As (i) is false, $w_\ell(X) \neq 0$. If no $v_i(X) = 1$, let $\ell = \max\{i : v_i(X) = 0\}$. As (ii) is false, $\ell < N$ and $w_{\ell+1}(X) \neq 0$. ●

When each $u_i(X)$ is nonnegative, then $v_i(X) \leq v_{i+1}(X)$ for all X, and $\sum_{i=1}^{N} w_i(X) = 0$ if and only if $v_N(X) = 0$. When the u_i's are defined by Equation (3.3.2), $v_N(X) = \int k(X, h)$. If $\bigcup_{i=1}^{N} R = R^n$, then $v_N(X) = 1$; and if $\bigcup_{i=1}^{N} R_i \subsetneq R^n$, then for X interior to $\bigcup_{i=1}^{N} R_i$, $v_N(X) = 1$ for all sufficiently small h. When the u_i's are defined by Equation (3.3.15), (3.3.17), (3.3.19) or (3.3.21), then $v_N(X) = 1$.

The definition of g_2 is suggested by the following consideration. Suppose that $E_1, E_2, \ldots, E_N$ are independent events, that F_i is the complement of E_i and that p_i is the probability of event E_i. The probability of event $\bigcap_{i=0}^{N} G_i$ is $\prod_{i=0}^{N} r_i$, where

$$r_i = \begin{cases} p_i & \text{if } G_i = E_i \\ 1 - p_i & \text{if } G_i = F_i \end{cases}$$

If b_i, $i = 1, \ldots, N$, is the payoff associated with event $\bigcap_{k=0}^{i-1} F_k \bigcap_{k=i}^{N} E_k$, then the expected value of the b_i's is

$$E\{b\} = \frac{\sum_{i=1}^{N} b_i \left\{ \prod_{k=0}^{i-1} (1-p_i) \prod_{k=i}^{N} p_i \right\}}{\sum_{i=1}^{N} \left\{ \prod_{k=0}^{i-1} (1-p_i) \prod_{k=i}^{N} p_i \right\}} . \qquad (3.4.6)$$

If we draw a parallel between E_i and A_i' of line (3.3.16) and between p_i and $v_i(X) = \int_{A_i'} k(X, h)$, then Equations (3.4.5) and (3.4.6) are similar. We ignore the fact that the E_i's are independent events while the A_i''s are not independent "events". Corresponding to the event $\bigcap_{k=0}^{i-1} F_k \bigcap_{k=i}^{N} E_k$ is the set

$$\bigcap_{k=0}^{i-1} (R^n \setminus A_k') \bigcap_{k=i}^{N} A_k' = R_i, \quad \text{where} \quad R_i = A_i' \setminus A_{i-1}',$$

and we can interpret $g_2(X)$ as being a weighted average of the b_i's with the weight of each b_i being the "probability" that X is in R_i.

We return now to the example discussed at the beginning of the section and illustrated in Figures 3.4.1 and 3.4.2. For the situation illustrated in Figure 3.4.1,

$$v_i(x) = \begin{cases} 0 & \text{if} \quad i = 0, \ldots, k-1 \\ 1/2 & \text{if} \quad i = k \\ 1 & \text{if} \quad i = k+1, \ldots, N, \end{cases}$$

$$w_i(x) = \begin{cases} 0 & \text{if} \quad i = 0, \ldots, k-1 \\ 1/2 & \text{if} \quad i = k, k+1 \\ 0 & \text{if} \quad i = k+2, \ldots, N, \end{cases}$$

and

$$g_2(x) = \frac{1}{2} b_k + \frac{1}{2} b_{k+1} = f(x).$$

For the situation illustrated in Figure 3.4.2,

$$v_i(x) = \begin{cases} 0 & \text{if } i = 0, \ldots, k-1 \\ 1/2 & \text{if } i = k \\ 1-a & \text{if } i = k+1 \\ 1 & \text{if } i = k+2, \ldots, N, \end{cases}$$

$$w_i(x) = \begin{cases} 0 & \text{if } i = 0, \ldots, k-1 \\ \frac{1}{2}(1-a) & \text{if } i = k, k+1 \\ \frac{1}{2}a & \text{if } i = k+2 \\ 0 & \text{if } i = k+3, \ldots, N, \end{cases}$$

$$\sum_{i=1}^{N} w_i(x) = 1 - \frac{a}{2},$$

and

$$g_2(x) = \left(\tfrac{1}{2}(1-a)b_k + \tfrac{1}{2}(1-a)b_{k+1} + \tfrac{1}{2}ab_{k+2}\right) / \left(1 - \tfrac{a}{2}\right)$$

$$= \tfrac{1}{2}b_k + \tfrac{1}{2}b_{k-1} + \frac{3a\Delta b}{2(2-a)}$$

$$= f(x) + \frac{3a\Delta b}{2(2-a)} \qquad (3.4.7)$$

At x, g_2 overestimates f by the amount $\frac{3a\Delta b}{2(2-a)}$, compared to $a\Delta b$ for g_1, line (3.4.1). The length of the interval $f^{-1}(I_{k+2})$ is less than the length of the interval $f^{-1}(I_{k+1})$, so the value of a is less than $1/8$. Therefore, the amount by which g_2 overestimates

f at x on line (3.4.7) is between 20% and 25% less than on line (3.4.1). The improvement is not great and comes at the cost of additional computation. Section 5.5 summarizes the results of several computer simulations that compare the effectiveness of the g_1 and g_2. The results in Section 5.5 indicate that g_2, in addition to providing a more accurate estimate of f then g_1, is less sensitive at the best values of h to changes in the value of h.

3.5 Properties of the Estimates g_1 and g_2

In the remainder of this chapter, we will examine some of the properties of the estimates g_1 and g_2. g_1 is defined at all points of R^n, but there may be points at which g_2 is not defined. It will be an implicit assumption throughout the rest of the chapter that whenever we describe the behavior of g_2 we are restricting our attention to that subset of R^n on which g_2 is defined.

First, we show that g_1 and g_2 are continuous functions not only of X, but also of other parameters. When each $u_i(X) = \int_{R_i} k(X, h)$, these other parameters are the R_i's, k, h and, of course, the b_i's. In Section 3.6 we define a metric for each parameter and show that g_1 and g_2 are continuous from the product space, called $\mathcal{T}_1$, of these metric spaces and of the space of X to the real numbers. When each $R_i = f^{-1}(I_i)$ and b_i is in

I_i, then it is appropriate to consider g_1 and g_2 as functions of the I_i's rather than of the R_i's and b_i's. In Section 3.7 we define a metric for the I_i's, form a product space $\mathcal{J}_2$ similar to $\mathcal{J}_1$, except that the metric space of the I_i's replaces the metric spaces of the R_i's and b_i's, and prove that g_1 and g_2 are continuous from $\mathcal{J}_2$ to R whenever (i) f is continuous, (ii) the inverse image under f of every singleton set has measure zero, and (iii) each b_i is a continuous function of I_i. The scheme of Section 3.7 is repeated in Section 3.8, with each R_i determined by an N-class classifier trained on the training set $\{(T_j, y_j)\}_{j=1}^{N'}$ and each b_i a function of some subset of the y_j's. Defining a metric on the T_j's, we get a product space $\mathcal{J}_3$. The continuity of g_1 and g_2 on the space $\mathcal{J}_3$ depends on the degree to which the classifier is effected by small changes in the training set. In Section 3.8 we assume that (i) the classifier is the nearest-neighbor-classifier, (ii) $y_j = f(T_j)$, (iii) f is continuous and (iv) b_i is a continuous function of a fixed subset of the y_j's. Finally in Section 3.9, we let each $u_i(X) = \int_{-d_{i-1}(X)}^{-d_i(X)} \ell(0, h)$, where the d_i's are continuous discriminant functions. Defining a metric for the d_i's and replacing the metric space of the R_i's in $\mathcal{J}_1$ by the metric space of the d_i's, we get a space $\mathcal{J}_4$ on which g_1 and g_2 are continuous.

Second, we will describe the circumstances in which g_1 and g_2 converge to f, from which we can deduce the behavior of the least-square error between g_1 and f and between g_2 and f. Let E be a measurable subset of R^n, $m(E) > 0$, and let ρ be a nonnegative measurable density function, i.e., $\int_{R^n} \rho = 1$. If f and g are measurable, if g is an estimate of f, if the domain of each includes E and if $\int_E \rho \neq 0$, then the least-square error between f and g conditioned by the density function ρ is

$$LSE(f, g, E, \rho) = \left(\int_E (f-g)^2 \rho \right)^{1/2} \Big/ \int_E \rho . \qquad (3.5.1)$$

If α and β are points in a metric space, if g_α is a measurable estimate of f whose domain includes E and if $\lim_{\alpha \to \beta} g_\alpha = g$ in the uniform norm, then $\lim_{\alpha \to \beta} LSE(f, g_\alpha, E, \rho) = LSE(f, g, E, \rho)$, which equals zero if $g \equiv f$.

We now summarize the results on the convergence of g_1 and g_2 to f. Let

(i) $\{R_i\}_{i=1}^N$ be pairwise disjoint measurable subsets of R^n,

(ii) R_i^0 be the interior of R_i,

(iii) $R_i^\alpha = \{Z : B_\alpha(Z) \subset R_i\}$, where $\alpha > 0$

(iv) $u_i(X) = \int_{R_i} k(X, h)$,

(v) D be the domain of f,

(vi) $\Delta f_i = \sup\{|f(Z)-b_i| : Z \in D \cap R_i\}$, $i = 1, \ldots, N$,

(vii) $\Delta f = \max\{\Delta f_i : i = 1, \ldots, N\}$, and

(viii) $q = \sum_{i=1}^{N} b_i \xi_{R_i}$,

where ξ_{R_i} is the characteristic function of the set R_i. In Section 3.10 we show that, when h converges to zero, g_1 and g_2 converge to q pointwise on $\bigcup_{i=1}^{N} R_i^0$ and to q uniformly on $\bigcup_{i=1}^{N} R_i^a$. If in addition Δf converges to zero, then g_1 and g_2 converge to f pointwise and uniformly on $D \cap \bigcup_{i=1}^{N} R_i^0$ and $D \cap \bigcup_{i=1}^{N} R_i^a$ respectively. At some points of $(\bigcup_{i=1}^{N} R_i)^0$, a set which includes $\bigcup_{i=1}^{N} R_i^0$, the limit as h converges to zero of g_1 and g_2 may not exist; however, in Section 3.12 we show that, under the additional hypothesis of f continuous and bounded, g_1 and g_2 converge to f pointwise on $D \cap (\bigcup_{i=1}^{N} R_i)^0$ and to f uniformly on every closed and bounded subset of $D \cap (\bigcup_{i=1}^{N} R_i)^a$ as h and Δf converge to zero. Sections 3.11 and 3.13 are similar to Sections 3.10 and 3.12 respectively, except that $u_i(X) = \int_{-d_{i-1}(X)}^{-d_i(X)} \ell(0, h)$.

Third, in Section 3.14 we will examine the effect of changes in the discriminant functions $d_0, d_1, \ldots, d_N$ on the values of g_1 and g_2 when $u_i(X) = \int_{-d_{i-1}(X)}^{-d_i(X)} \ell(0,h)$. The discriminant functions are determined from the training set $\{(T_j, y_j)\}_{j=1}^{N'}$ by some training procedure, and a change in the training set means that discriminant functions $d_0', d_1', \ldots, d_N'$ will result. Of course, $d_0 \equiv d_0' \equiv \infty$ and $d_N \equiv d_N' \equiv -\infty$, but otherwise we can expect the d_i's to be different than the d_i''s. Let $u_i'(X) = \int_{-d_{i-1}'(X)}^{-d_i'(X)} \ell(0,h)$, and let g_1' and g_2' be the estimates determined by the u_i''s. As

$$|u_i'(X) - u_i(X)| = \left| \int_{-d_{i-1}'(X)}^{-d_i'(X)} \ell(0,h) - \int_{-d_{i-1}(X)}^{-d_i(X)} \ell(0,h) \right|$$

$$\leq \left| \int_{-d_i(X)}^{-d_i'(X)} \ell(0,h) - \int_{-d_{i-1}(X)}^{-d_{i-1}'(X)} \ell(0,h) \right| , \qquad (3.5.2)$$

it will be convenient to analyze the difference between g_1 and g_1' and between g_2 and g_2' in terms of

$$e_i(X) = \int_{-d_i(X)}^{-d_i'(X)} \ell(0,h), \qquad (3.5.3)$$

$i = 0, \ldots, N$. Note that $e_i(X) \leq (1/h) \sup(\ell) |d_i(X) - d_i'(X)|$. As already mentioned, g_1 and g_2 are continuous functions of the

d_i's, a result which will be established in Lemma 3.9.2. From Lemma 3.9.2 it can be seen that, as the $e_i(X)$'s simultaneously converge to zero, $g_1(X)$ and $g_2(X)$ converge to $g_1'(X)$ and $g_2'(X)$ respectively.

The measure we use for the difference between g_1 and g_1' and between g_2 and g_2' at X is

$$\text{Error}(g_1(X)) = |g_1(X)-g_1'(X)| \ / \ (b_N-b_1)$$

and

$$\text{Error}(g_2(X)) = |g_2(X)-g_2'(X)| \ / \ (b_N-b_1) \ .$$

By finding bounds for $\text{Error}(g_1(X))$ and $\text{Error}(g_2(X))$ expressed in terms of the $e_i(X)$'s, we get a measure of the stability of g_1 and g_2 with respect to the d_i's. We will prove that if $d_{i-1}(X) \leq d_i(X)$ and $d_{i-1}'(X) \leq d_i'(X)$ for all i and X, then $\text{Error}(g_1(X)) \leq e(X)$, where $e(X) = \max\{|e_i(X)| : i = 0, \ldots, N\}$. $\text{Error}(g_2(X))$ does not seem to have any easily expressed yet reasonable bound, not surprisingly considering the complexity of g_2 in terms of the u_i's. But when $e_i(X) = u_i(X) = 0$, $i = 0, \ldots, k-2$ and $i = k+1, \ldots, N$ for some index k and when the kernel function is symmetric about the origin, then

$$\text{Error}(g_1(X)) \leq 2e(X) \ / \ (N-1)$$

and

$$\text{Error}(g_2(X)) \leq 2e(X) / \{(N-1)(.75-e(X))\}, \quad \text{provided} \quad .75 > e(X).$$

3.6 Continuity of g_1 and g_2 on $\mathcal{T}_1$

Let $\mathcal{R}$ be the metric space of N-tuples $(R_i)_{i=1}^N$ of measurable subsets of R^n, and let the metric be

$$d_1((R_i)_{i=1}^N, (R_i')_{i=1}^N) = \max\{d_2(R_i, R_i') : i = 1, \ldots, N\}, \tag{3.6.1}$$

where

$$d_2(R_i, R_i') = \min\{c, m(R_i \Delta R_i')\} . \tag{3.6.2}$$

$R_i \Delta R_i'$ is the symmetric difference of R_i and R_i', $m(S)$ is the Lebesgue measure of a set S, c is a positive constant, and R_i and R_i' are considered equal if $m(R_i \Delta R_i') = 0$. For notational simplicity, we shall sometimes omit writing the bounds of indices when the omission does not seem likely to cause confuction, for instance writing (R_i) insteady of $(R_i)_{i=1}^N$. Also there will be a number of different distance functions being used, and we will denote them all by the letter d.

Let R^N, R^n and $R^+ = \{z : z > 0\}$ be the metric spaces of $B = (b_i)$, X and h respectively with the Euclidean metric. Let $\mathcal{K}$ be the metric space of the kernel function k, and let the metric be

$$d(k, k') = \int_{R^n} |k(Z) - k'(Z)| dZ. \tag{3.6.3}$$

Let

$$\mathcal{J}_1 = \mathcal{R} * R^N * R^n * R^+ * \mathcal{K}, \tag{3.6.4}$$

where "*" denotes the Cartesian product. Let each

$u_i(X) = \int_{R_i} k(X, h)$.

g_1 is defined on all of $\mathcal{J}_1$ and g_2 at all points of $\mathcal{J}_1$ for which $\sum_{i=1}^{N} w_i(X) \neq 0$. We now show that g_1 and g_2 are continuous at all points $((R_i), B, X, h, k)$ of their domains. The scheme is to prove a number of lemmas establishing the continuity of u_j, $j = 1, \ldots, N$, in each variable when all but that variable is held fixed. A function of several variables continuous in each separately is not necessarily continuous; however, for u_j the lemmas are established in such a way that the continuity of u_j follows. The continuity of g_1 and g_2 is an immediate consequence of the continuity of the u_j's. The same scheme will be followed in proving the continuity of g_1 and g_2 on $\mathcal{J}_2$, $\mathcal{J}_3$ and $\mathcal{J}_4$.

3.6.1 Lemma. For every $\epsilon > 0$, there exists $\delta > 0$ such that $|u_j((R_i), B, X, h, k) - u_j((R_i'), B, X, h, k)| < \epsilon$ whenever $d((R_i), (R_i')) < \delta$. δ is independent of j, B, X and dependent on h, k.

Proof.

$$|u_j((R_i), B, X, h, k) - u_j((R_i'), B, X, h, k)|$$

$$\leq \left| \int_{R_j} k(X,h) - \int_{R_j'} k(X,h) \right|$$

$$\leq \int_{R_j \Delta R_j'} k(X,h)$$

$$\leq (\sup(k)/h^n) m(R_j \Delta R_j')$$

$$\leq (\sup(k)/h^n) d((R_i), (R_i'))$$

if we assume WLOG that $m(R_j \Delta R_j') < c$, where $c > 0$ is the constant previously defined in conjunction with the distance function

$$< \epsilon$$

whenever $d((R_i), (R_i')) < \delta$ and $\delta = (h^n/\sup(k))\epsilon$. ●

3.6.2 Lemma. For every $\epsilon > 0$ there exists $\delta > 0$ such that

$$|u_j((R_i), B, X, h, k) - u_j((R_i), B, X', h, k)| < \epsilon$$

whenever

$$\|X - X'\| < \delta .$$

δ is independent of j, (R_i), B and dependent on h, k.

Proof.

$$|u_j((R_i), B, X, h, k) - u_j((R_i), B, X', h, k)|$$

$$\leq \left| \int_{R_j} (1/h^n)[k(\frac{Z-X}{h}) - k(\frac{Z-X'}{h})]dZ \right|$$

$$\leq \int_{R^n} (1/h^n)\left| k(\frac{Z-X}{h}) - k(\frac{Z-X}{h} - \overline{X}) \right| dZ$$

where $\overline{X} = (X'-X)/h$

$$\leq \int_{R^n} |k(Z)-k(Z-\overline{X})|\,dZ$$

$$\leq \int_{A_a(0)} |k(Z)-k(Z-\overline{X})|\,dZ \qquad (3.6.5)$$

$$+ \int_{B_a(0)} |k(Z)-k(Z-\overline{X})|\,dZ \qquad (3.6.6)$$

for any $a > 0$.

We will now show that for a sufficiently large a, whose value depends on h and k, the integral on line (3.6.5) is less than $\epsilon/2$. For this a there exists $\delta > 0$ such that the integral on line (3.6.6) is less than $\epsilon/2$ whenever $\|X-\overline{X}'\| < \delta$. From this, the lemma will follow.

The integral on line (3.6.5) is less than or equal to

$$\int_{A_\alpha(0)} k(Z)dZ + \int_{A_\alpha(0)} k(Z-\overline{X})dZ$$

$$\leq \int_{A_\alpha(0)} k(Z)dZ + \int_{A_\gamma(0)} k(Z)dZ$$

provided that $\alpha > \|(X'-X)/h\|$ and where $\gamma = \alpha - \|(X'-X)/h\|$, as whenever Z is in $A_\alpha(0)$ then $Z-\overline{X}$ is in $A_\gamma(0)$

$$\leq 2\int_{A_\gamma(0)} k(Z)dZ$$

as $A_\gamma(0) \supset A_\alpha(0)$

$$< \epsilon/2$$

whenever γ, and hence α, is sufficiently large.

As k is continuous, it is uniformly continuous on the closure of $B_{2\alpha}(0)$. So there exists $\delta_1 > 0$ such that $\|(X'-X)/h\| < \delta_1$ implies $|k(Z)-k(Z-\overline{X})| < \epsilon \Big/ \int_{B_\alpha(0)} 4dZ$ for all Z and $Z-\overline{X}$ in $B_{2\alpha}(0)$ and consequently for all Z in $B_\alpha(0)$ (note that $\alpha > \|(X'-X)/h\|$ from above). Therefore, the integral on line (3.6.6) is less than or equal to

$$\int_{B_\alpha(0)} \epsilon \Big/ \int_{B_\alpha(0)} 4dZ \; dZ < \epsilon/2$$

whenever $\|X-X'\| < \delta$ and $\delta = h\delta_1$. ●

3.6.3 Lemma. For every pair $h' > 0$ and $\epsilon > 0$ there exists $\delta > 0$ such that $|u_j((R_i), B, X, h, k) - u_j((R_i), B, X, h', k)| < \epsilon$ whenever $|h-h'| < \delta$. δ is independent of j, (R_i), B, X and dependent on k.

Proof.

$$|u_j((R_i), B, X, h, k) - u_j((R_i), B, X, h', k)|$$

$$\leq \int_{R^n} |k(X, h)-k(X, h')|$$

$$\leq \int_{A_a(X)} |k(X, h)-k(X, h')| \qquad (3.6.7)$$

$$+ \int_{B_a(X)} |k(X, h)-k(X, h')| \qquad (3.6.8)$$

for every $a > 0$.

We now show that for a sufficiently large a, whose value depends on h' and k, the integral on line (3.6.7) is less than $\epsilon/2$. For this a there exists $\delta > 0$ such that the integral on line (3.6.8) is less than $\epsilon/2$ whenever $|h-h'| < \delta$. From this the lemma will follow.

The integral on line (3.6.7) is less than or equal to

$$\int_{A_{a/h}(0)} k(0,1) + \int_{A_{a/h}(0)} k(0,1)$$

$$\leq \int_{A_{\gamma}(0)} k(0,1)$$

where WLOG we assume that δ will be sufficiently small to insure that $|h-h'| < \delta$ implies $h > h'/2$ and where $\gamma = 2a/h$

$$\leq \epsilon/2$$

whenever γ, and hence a, is sufficiently large.

As $k(Z)$ is uniformly continuous on the closure of $B_a(0)$, there exists $\delta > 0$ such that $|h-h'| < \delta$ implies $|(1/h^n)k(Z/h) - (1/h'^n)k(Z/h')| < \epsilon \Big/ \int_{B_a(0)} 4dZ$ whenever Z is in $B_a(0)$. Therefore, the integral on line (3.6.8) is less than or equal to

$$\int_{B_a(0)} |(1/h^n)k(Z/h) - (1/h'^n)k(Z/h')| dZ$$

on making a change of variables

$$< \epsilon/2$$

whenever $|h-h'| < \delta$. ●

3.6.4 Lemma. For $\epsilon > 0$ there exists $\delta > 0$ such that $|u_j((R_i), B, X, h, k) - u_j((R_i), B, X, h, k')| < \epsilon$ whenever $d(k, k') < \delta$. δ is independent of j, (R_i), B, X and h.

Proof.

$$|u_j((R_i), B, X, h, k) - u_j((R_i), B, X, h, k')|$$

$$\leq \int_{R^n} |k(X, h) - k'(X, h)|$$

$$\leq \int_{R^n} |k(0, 1) - k'(0, 1)|$$

$$< \epsilon \qquad \text{whenever} \quad d(k, k') < \delta \quad \text{and} \quad \delta = \epsilon. \quad \bullet$$

3.6.5 Lemma. For $j = 1, \ldots, N$, u_j is a continuous function from $\mathcal{J}_1$ to R.

Proof. Let $((R_i'), B', X', h', k')$ in $\mathcal{J}_1$ and $\epsilon > 0$ be given, and let $((R_i), B, X, h, k)$ be any other point in $\mathcal{J}_1$. Then

$$|u_j((R_i'), B', X', h', k') - u_j((R_i), B, X, h, k)| \qquad (3.6.9)$$

$$\leq |u_j((R_i'), B', X', h', k') - u_j((R_i), B', X', h', k')| \qquad (3.6.10)$$

$$+ |u_j((R_i), B', X', h', k') - u_j((R_i), B, X', h', k')| \qquad (3.6.11)$$

$$+ |u_j((R_i), B, X', h', k') - u_j((R_i), B, X, h', k')| \qquad (3.6.12)$$

$$+ |u_j((R_i), B, X, h', k') - u_j((R_i), B, X, h, k')| \qquad (3.6.13)$$

$$+ |u_j((R_i), B, X, h, k') - u_j((R_i), B, X, h, k)| \; . \qquad (3.6.14)$$

By Lemma 3.6.1 there exists $\delta_1 > 0$ such that (R_i) in $B_{\delta_1}((R_i'))$ implies the expression on line (3.6.10) is less than $\epsilon/5$.

As the value of u_j is independent of B and B', the expression on line (3.6.11) is zero.

By Lemma 3.6.2 there exists $\delta_2 > 0$ whose value is independent (R_i) and B such that X in $B_{\delta_2}(X')$ implies the expression on line (3.6.12) is less than $\epsilon/5$.

By Lemma 3.6.3 there exists $\delta_3 > 0$ whose value is independent of (R_i), B and X such that h in $B_{\delta_3}(h')$ implies the expression on line (3.6.13) is less than $\epsilon/5$.

By Lemma 3.6.4 there exists $\delta_4 > 0$ whose value is independent of (R_i), B, X and h such that k in $B_{\delta_4}(k')$ implies the expression on line (3.6.14) is less than $\epsilon/5$.

For each of the Lemmas 3.6.1, 3.6.2, 3.6.3, and 3.6.4 the value of delta depends on some of the arguments of u_j and does not depend on the others. In breaking the expression on line (3.6.9) into the sum of five terms, care has been taken to do it in such an order that the lemmas apply. Hence for every point of

$$B_{\delta_1}((R_i')) * B_1(B') * B_{\delta_2}(X') * B_{\delta_3}(h') * B_{\delta_4}(k')$$

the expression on line (3.6.9) is less than ϵ. ●

3.6.6 Theorem. g_1 and g_2 are continuous functions from $\mathcal{T}_1$ to R.

Proof. The theorem is an immediate consequence of Lemma 3.6.5. ●

3.7 Continuity of g_1 and g_2 on $\mathcal{J}_2$

Let I_f^N be the metric space of all N-tuples $(I)_{i=1}^N$ of disjoint intervals whose union includes the range of f and for which I_i is to the left of I_{i+1}. Let the metric be

$$d_1((I_i), (I_i')) = \max\{d_2(I_i, I_i') : i = 1, \ldots, N\}, \tag{3.7.1}$$

where

$$d_2(I_i, I_i') = \min\{c, m(I_i \Delta I_i')\} \tag{3.7.2}$$

for some constant $c > 0$. Let

$$\mathcal{J}_2 = I_f^N * R^n * R^+ * \mathcal{K}. \tag{3.7.3}$$

Let

$$d_a((R_i), (R_i')) = \max_i \{m(R_i \Delta R_i' \cap B_a(0)\}. \tag{3.7.4}$$

Let each $u_i(X) = \int_{R_i} k(X, h)$, $R_i = f^{-1}(I_i)$, and b_i belong to I_i.

3.7.1 Lemma. Let

(i) A be a closed and bounded subset of R^n,

(ii) f be continuous,

(iii) the domain of f be closed, and

(iv) $m(f^{-1}(y)) = 0$ for every y in the range of f.

Then for every y in the range of f

$$\lim_{\delta \to 0} m(f^{-1}(B_\delta(y)) \cap A) = 0$$

Proof. Let y in the range of f and $\epsilon > 0$ be given. As $m(f^{-1}(y)) = 0$, there exists an open set U containing $f^{-1}(y)$ whose measure is less than ϵ. If we can show that there exists $\delta > 0$ such that $f^{-1}(B_\delta(y)) \cap A \subset U$, we are through.

On the contrary, assume that no such $\delta > 0$ exists. Then for $k = 1, 2, 3, \ldots$, there exists a point Z_k in $f^{-1}(B_{1/k}(y)) \cap A$ that is not in U. As A is closed and bounded, the sequence $\{Z_k\}_{k=1}^{\infty}$ has a convergent subsequence $\{Z_k'\}_{k=1}^{\infty}$ whose limit point Z is contained in A. As f is continuous, $f(Z) = \lim_{k \to \infty} f(Z_k') = y$, so that Z is in U, and U contains infinitely many points of $\{Z_k'\}_{k=1}^{\infty}$. But this contradicts the way in which the sequence $\{Z_k\}_{k=1}^{\infty}$ is constructed. Hence there exists $\delta > 0$ such that $f^{-1}(B_\delta(y)) \cap A \subset U$. ●

3.7.2 Lemma. Let

(i) f satisfy the constraints of Lemma 3.7.1 and

(ii) $\alpha > 0$ be an arbitrary constant.

Then

$$\lim_{(I_i)\to(I_i')} d_\alpha((R_i),(R_i')) = 0,$$

where R_i and R_i' equal $f^{-1}(I_i)$ and $f^{-1}(I_i')$ respectively.

Proof. For $i = 1,\ldots,N$ let r_i and s_i denote the left and right end point respectively of the interval I_i, and r_i' and s_i' those of I_i'. With the possible exceptions of r_1, r_1', s_N and s_N', all the end points are finite. In order for (I_i) to converge to (I_i') it is necessary that r_1 and r_1' both be finite or both be infinite. Hence WLOG we assume that both are finite or both are infinite. For the same reason we assume that both s_N and s_N' are finite or both are infinite. For $i = 1,\ldots,N$ let

$$\delta_i = \begin{cases} |s_1 - s_1'| & \text{if } r_1, r_1' \text{ are infinite} \\ |r_N - r_N'| & \text{if } s_N, s_N' \text{ are infinite} \\ \max\{|r_i - r_i'|, |s_i - s_i'|\} & \text{otherwise,} \end{cases}$$

and let $\delta = 2\max_i\{\delta_i\}$. For notational convenience we make the convention that $B_\delta(\infty) = B_\delta(-\infty) = \phi$.

As for $i = 1,\ldots,N$,

$$R_i \Delta R_i' = f^{-1}(I_i \Delta I_i') \subseteq f^{-1}(B_\delta(r_i')) \cup f^{-1}(B_\delta(s_i')),$$

then

$$\lim_{(I_i)\to(I_i')} d_a((R_i),(R_i'))$$

$$\leq \lim_{(I_i)\to(I_i')} \sum_{i=1}^{N} \{m(R_i \Delta R_i') \cap B_a(0)\}$$

where $B_a(0)$ is a subset of R^n

$$\leq \lim_{(I_i)\to(I_i')} \sum_{i=1}^{N} m[f^{-1}(B_\delta(r_i')) \cap B_a(0)] + m[f^{-1}(B_\delta(s_i')) \cap B_a(0)]$$

$$\leq \lim_{(I_i)\to(I_i')} 2N \max_i \{m[f^{-1}(B_\delta(r_i') \cap A],\ m[f^{-1}(B_\delta(s_i')) \cap A]\}$$

where A is the closure of $B_a(0)$

$= 0$ by Lemma 3.7.1 as

$$\lim_{(I_i)\to(I_i')} \delta = 0. \quad \bullet$$

3.7.3 Lemma. Let the constraints of Lemmas 3.7.1 be satisfied. Then for every (I_i') and $\epsilon > 0$ there exists $\delta > 0$ such that $|u_j((I_i), X, h, k) - u_j((I_i'), X, h, k)| < \epsilon$ whenever $d((I_i),(I_i')) < \delta$. δ is independent of j and dependent on X, h, k.

Proof.

$$|u_j((I_i), X, h, k) - u_j((I_i'), X, h, k)|$$

$$\leq \left| \int_{R_j} k(X,h) - \int_{R_j'} k(X,h) \right|$$

$$\leq \int_{\overline{R}_j} k(X,h) + \int_{\overline{R}'_j} k(X,h) \qquad (3.7.5)$$

$$+ \left| \int_{\underline{R}_j} k(X,h) - \int_{\underline{R}'_j} k(X,h) \right| \qquad (3.7.6)$$

where for a set S, $\overline{S} = S \cap A_\alpha(X)$ and $\underline{S} = S \cap B_\alpha(X)$, $\alpha > 0$.

The expression on line (3.7.5) is less than or equal to $2\int_{A_\alpha(X)} k(X,h) < \epsilon/2$ for some sufficiently large α dependent on k, h.

For this α the expression on line (3.7.6) is less than or equal to

$$\int_{\underline{R}_j \Delta \underline{R}'_j} k(X,h)$$

$$\leq (\sup(k)/h^n) m(R_j \Delta R'_j \cap B_\alpha(X))$$

$$\leq (\sup(k)/h^n) d_{\alpha+\beta}((R_i),(R'_i)) \qquad (3.7.7)$$

as every point of $B_\alpha(X)$ is in $B_{\alpha+\beta}(0)$ where $\beta = \|X\|$.

Lemma 3.7.2 applies and there exists $\delta > 0$ dependent on X, h, k such that the term on line (3.7.7) is less than $\epsilon/2$ whenever $d((I_i),(I'_i)) < \delta$. ●

3.7.4 Lemma. Let the constraints of Lemma 3.7.1 be satisfied. Then for $j = 1, \ldots, N$, u_j is a continuous function from $\mathcal{J}_2$ to R.

Proof. Let $((I_i'), X', h', k')$ in $\mathcal{J}_2$ and $\epsilon > 0$ be given, and let $((I_i), X, h, k)$ be any other point of $\mathcal{J}_2$. Then

$$|u_j((I_i'), X', h', k') - u_j((I_i), X, h, k)| \tag{3.7.8}$$

$$\leq |u_j((I_i'), X', h', k') - u_j((I_i), X', h', k')| \tag{3.7.9}$$

$$+ |u_j((I_i), X', h', k') - u_j((I_i), X, h', k')| \tag{3.7.10}$$

$$+ |u_j((I_i), X, h', k') - u_j((I_i), X, h, k')| \tag{3.7.11}$$

$$+ |u_j((I_i), X, h, k') - u_j((I_i), X, h, k)| . \tag{3.7.12}$$

By Lemma 3.7.3 there exists $\delta_1 > 0$ such that (I_i) in $B_{\delta_1}((I_i'))$ implies the expression on line (3.7.9) is less than $\epsilon/4$.

By Lemma 3.6.2 there exists $\delta_2 > 0$ whose value is independent of (R_i) and B and consequently of (I_i) such that X in $B_{\delta_2}(X')$ implies the expression on line (3.7.10) is less than $\epsilon/4$.

By Lemma 3.6.3 there exists $\delta_3 > 0$ whose value is independent of (R_i), B and X and consequently of (I_i) and X such that h in $B_{\delta_3}(h')$ implies the expression on line (3.7.11) is less than $\epsilon/4$.

By Lemma 3.6.4 there exists $\delta_4 > 0$ whose value is independent of (R_i), B, X and h and consequently of (I_i), X and h

such that k in $B_{\delta_4}(k')$ implies the expression on line (3.7.12) is less than $\epsilon/4$.

Hence, the lemma is true. ●

3.7.5 Theorem. Let

(i) f be continuous,

(ii) the domain of f be closed,

(iii) $m(f^{-1}(y)) = 0$ for every y in the range of f,

(iv) $\lim_{I_i \to I_i'} b_i = b_i'$, $i = 1, \ldots, N$.

Then g_1 and g_2 are continuous functions from $\mathcal{T}_2$ to R.

Proof. The theorem is an immediate consequence of Lemma 3.7.4 and of condition (iv) above. ●

3.8 Continuity of g_1 and g_2 on $\mathcal{T}_3$

Let $T_f^{N'}$ be the metric space of all N'-tuples $(T_i)_{i=1}^{N'}$ of points in the domain of f and let the metric be

$$d((T_i), (T_i')) = \max\{\| T_i - T_i' \| : i = 1, \ldots, N'\} \tag{3.8.1}$$

Let

$$\mathcal{T}_3 = T_f^{N'} * R^n * R^+ * \mathcal{K} \tag{3.8.2}$$

For measurable subsets A, B and C of R^n let

$$d_C(A, B) = m((A \Delta B) \cap C). \tag{3.8.3}$$

Let each $u_i(X) = \int_{R_i} k(X, h)$, let $R_i = \bigcup_{j=\ell_{i-1}+1}^{\ell_i} nn(T_j)$, see line (3.3.10), and let b_i be a continuous function of the y_j's, $j = \ell_{i-1}+1, \ldots, \ell_i$.

3.8.1 Lemma. Let

(i) X, X', Y, Y' be points in R^n,

(ii) A be a measurable subset of R^n of finite diameter containing X, X', Y, Y', and

(iii) $C = \{Z : \|Z-X\| \sim \|Z-Y\|\}$,

$C' = \{Z : \|Z-X'\| \sim \|Z-Y'\|\}$, where "$\sim$" can mean "$\leq$" or "$<$".

Then $\lim_{(X, Y) \to (X', Y')} d_A(C, C') = 0$.

Proof. WLOG

(a) let the components of X' and Y' be

$$x_i' = \begin{cases} 0 & \text{if } i < n \\ a & \text{if } i = n \end{cases} \quad \text{and} \quad y_i' = \begin{cases} 0 & \text{if } i < n \\ -a & \text{if } i = n, \end{cases}$$

where $a > 0$,

(b) let $A = B_\alpha(0)$, where $\alpha > 0$, and

(c) let the n-th components of X and Y be greater than zero and less than zero respectively.

Condition (c) is allowed because (X, Y) is assumed to converge to (X', Y'), and it insures that $x_n > y_n$.

Let H' be the hyperplane of points equidistant from X' and Y'. If $t = \sup_{P \in (C \Delta C') \cap A} \{d(P, H')\}$, where $d(p, H')$ is the perpendicular distance from P to H', then t is finite and

$$d_A(C, C') = m((C \Delta C') \cap A$$
$$\leq 2ta.$$

We now show that t converges to zero as (X, Y) converges to (X', Y'). For any point P of $C \Delta C'$, the distance from P to H' is $|p_n|$. If P is in C and not in C', then $\|P-X\| \leq \|P-Y\|$ and $\|P-X'\| \geq \|P-Y'\|$, which implies

$$\left\{(X-Y)\cdot(X+Y) - 2\sum_{i=1}^{n-1} p_i(x_i-y_i)\right\} \Big/ 2(x_n-y_n) \leq p_n \leq 0.$$

Similarly if P is not in C and is in C', then

$$\left\{(X-Y)\cdot(X+Y) - 2\sum_{i=1}^{n-1} p_i(x_i-y_i)\right\} \Big/ 2(x_n-y_n) \geq p_n \geq 0.$$

Hence

$$t \leq \sup_{P \in (C \Delta C') \cap A} \left\{\left|(X-Y)\cdot(X+Y)\right| + 2\sum_{i=1}^{n-1} |p_i(x_i-y_i)|\right\} \Big/ 2(x_n-y_n)$$
$$\leq \left\{\left|(X-Y)\cdot(X+Y)\right| + 2a\sum_{i=1}^{n-1} |x_i-y_i|\right\} \Big/ 2(x_n-y_n),$$

from which we see that t converges to zero as (X, Y) converges to (X', Y'). ●

3.8.2 Lemma. Let $(T_j)_{j=1}^{N'}$ and $(T_j')_{j=1}^{N'}$ be N'-tuples of points taken from $B_a(0) \subset R^n$, $a > 0$. Then

$$\lim_{(T_j) \to (T_j')} d_a((R_i), (R_i')) = 0 .$$

Proof. As for $i = 1, \ldots, N$,

$$R_i \Delta R_i' \subseteq \bigcup_{j=\ell_{i-1}+1}^{\ell_i} nn(T_j) \Delta nn(T_j') ,$$

the lemma will follow if we establish that for $j = 1, \ldots, N'$

$$\lim_{(T_h)_{h=1}^{N'} \to (T_h')_{h=1}^{N'}} d_a(nn(T_j), nn(T_j')) = 0.$$

For $j, k = 1, \ldots, N'$ and for $j \neq k$ let

$$C_{jk} = \{Z : \| Z - T_j \| \sim \| Z - T_k \|\}$$

$$C_{jk}' = \{Z : \| Z - T_j' \| \sim \| Z - T_k' \|\}$$

where "$\sim$" means "$\leq$" if $j < k$ and "$<$" if $j > k$.

Then for $j = 1, \ldots, N'$

$$nn(T_j) = \bigcap_{\substack{k=1 \\ k\neq j}}^{N'} C_{jk} \quad \text{and} \quad nn(T'_j) = \bigcap_{\substack{k=1 \\ k\neq j}}^{N'} C'_{jk} .$$

For $j = 1, \ldots, N'$

$$Z \in nn(T_j) \Delta nn(T'_j) \Rightarrow (Z \in nn(T_j) \wedge Z \notin nn(T'_j)) \vee (Z \notin nn(T_j) \wedge Z \in nn(T'_j)).$$

$$Z \in nn(T_j) \wedge Z \notin nn(T'_j) \Rightarrow (Z \in C_{jk} \forall k \neq j) \wedge (\exists k \neq j \ni Z \notin C'_{jk})$$

$$\Rightarrow \exists k \neq j \ni Z \in C_{jk} \Delta C'_{jk} .$$

$$Z \notin nn(T_j) \wedge Z \in nn(T'_j) \Rightarrow \exists k \neq j \ni Z \in C_{jk} \Delta C'_{jk} .$$

Hence

$$nn(T_j) \Delta nn(T'_j) \subseteq \bigcup_{\substack{k=1 \\ k\neq j}}^{N'} C_{jk} \Delta C'_{jk} .$$

Hence for $j = 1, \ldots, N'$

$$\lim_{(T_h) \to (T'_h)} d_\alpha(nn(T_j), nn(T'_j))$$

$$\leq \lim_{(T_h) \to (T'_h)} \sum_{\substack{k=1 \\ k\neq j}}^{N'} d_\alpha(C_{jk}, C'_{jk})$$

$$= 0$$

by Lemma 3.8.1. ●

3.8.3 Lemma. For every set of points $(T_i')_{i=1}^{N'}$ and for every $\epsilon > 0$ there exists $\delta > 0$ such that $|u_j((T_i), X, h, k) - u_j((T_i'), X, h, k)| < \epsilon$ whenever $d((T_i),(T_i')) < \delta$. δ is independent of j, $j = 1, \ldots, N$, and dependent on X, h, k.

Proof. The proof is identical to the proof of Lemma 3.7.3 except that (i) on line (3.7.7) $\beta = \|X\|$ is replaced by $\beta = \max\{\max_i\{\|T_i'\| + 1\}, \|X\|\}$, and (ii) Lemma 3.7.2 is replaced by Lemma 3.8.2. ●

3.8.4 Lemma. For $j = 1, \ldots, N$, u_j is a continuous function from $\mathcal{T}_3$ to R.

Proof. The proof is so similar to that of Lemma 3.7.4 that it is omitted. ●

3.8.5 Theorem. Let

(i) $y_i = f(T_j)$, $j = 1, \ldots, N'$,

(ii) f be continuous, and

(iii) b_i be a continuous function of the y_j's, $j = \ell_{i-1}+1, \ldots, \ell_i$, see line (3.3.7), for $i = 1, \ldots, N$.

Then g_1 and g_2 are continuous functions from $\mathcal{T}_3$ to R.

Proof. The theorem is an immediate consequence of Lemma 3.8.4 and of conditions (i), (ii), and (iii) above. ●

3.9 Continuity of g_1 and g_2 on $\mathcal{T}_4$

Let $\mathcal{D}$ be the metric space of (N+1)-tuples $(d_i)_{i=0}^{N}$ of continuous discriminant functions, where $d_0 \equiv \infty$ and $d_N \equiv -\infty$. Let the metric be

$$d_1((d_i), (d_i')) = \max\{d_2(d_i, d_i') : i = 0, \ldots, N\}, \tag{3.9.1}$$

where

$$d_2(d_i, d_i') = \min\{c, \sup\{|d_i(Z) - d_i'(Z)| : Z \in R^n\}\} \tag{3.9.2}$$

for some positive constant c. We make the convention that

$$d_2(d_0, d_0) = d_2(d_N, d_N) = 0$$

and that

$$\int_{-\infty}^{-d_0(X)} \ell(0, h) = 0.$$

Let $\mathcal{K}$, as defined in Section 3.6 with $n = 1$ and $R^n = R$, be the metric space of the kernel function ℓ. Let

$$\mathcal{T}_4 = \mathcal{D} * R^N * R^n * R^+ * \mathcal{K}. \tag{3.9.3}$$

Let each

$$u_i(X) = \int_{-d_{i-1}(X)}^{-d_i(X)} \ell(0, h).$$

3.9.1 Lemma. For every pair X' in R^n and $\epsilon > 0$ there exists $\delta > 0$ such that $|u_j((d_i), B, X, h, \ell) - u_j((d_i), B, X', h, \ell)| < \epsilon$ whenever $\|X-X'\| < \delta$. δ is independent of B and dependent on j, (d_i), h, ℓ.

Proof.

$$|u_j((d_i), B, X, h, \ell) - u_j((d_i), B, X', h, \ell)|$$

$$\leq \left| \int_{-d_{j-1}(X)}^{-d_j(X)} \ell(0, h) - \int_{-d_{j-1}(X')}^{-d_j(X')} \ell(0, h) \right|$$

$$\leq (1/h)\sup(\ell)\{|d_i(X)-d_j(X')| + |d_{j-1}(X)-d_{j-1}(X')|\}$$

$$< \epsilon$$

whenever $\|X-X'\| < \delta$ and δ is small enough to insure that $|d_j(X) - d_j(X')|$ and $|d_{j-1}(X) - d_{j-1}(X')|$ are less than $h\epsilon/(2 \sup(\ell))$. Such a δ exists as d_{j-1} and d_j are continuous. ●

3.9.2 Lemma. For every $\epsilon > 0$ there exists $\delta > 0$ such that $|u_j((d_i), B, X, h, \ell) - u_j((d_i'), B, X, h, \ell)| < \epsilon$ whenever $d((d_i), (d_i')) < \delta$. δ is independent of j, B, X and dependent on h, ℓ.

Proof.

$$|u_j((d_i), B, X, h, \ell) - u_j((d_i'), B, X, h, \ell)|$$

$$\leq \left| \int_{-d_{j-1}(X)}^{-d_j(X)} \ell(0,h) - \int_{-d_{j-1}'(X)}^{-d_j'(X)} \ell(0,h) \right|$$

$$\leq \left| \int_{-d_j'(X)}^{-d_j(X)} \ell(0,h) \right| + \left| \int_{-d_{j-1}'(X)}^{-d_{j-1}(X)} \ell(0,h) \right|$$

$$\leq (1/h)\ \sup(\ell)\{|d_j(X)-d_j'(X)| + |d_{j-1}(X)-d_{j-1}'(X)|\}$$

$$< \epsilon$$

whenever $d((d_i), (d_i')) < \delta$ and

$\delta = h\epsilon/(2\ \sup(\ell))$. ●

3.9.3 Lemma. For every pair $h' > 0$ and $\epsilon > 0$ there exists $\delta > 0$ such that $|u_j((d_i), B, X, h, \ell) - u_j((d_i), B, X, h', \ell)| < \epsilon$ whenever $|h-h'| < \delta$. δ is independent of j, (d_i), B, X and dependent on ℓ.

Proof.

$$|u_j((d_i), B, X, h, \ell) - u_j((d_i), B, X, h', \ell)|$$

$$\leq \left| \int_{-d_{j-1}(X)}^{-d_j(X)} \ell(0,h) - \ell(0,h') \right|$$

$$\leq \int_{A_a(0)} |\ell(0,h)-\ell(0,h')| + \int_{B_a(0)} |\ell(0,h)-\ell(0,h')|$$

for every $a > 0$.

The proof now proceeds as in Lemma 3.6.3. ●

3.9.4 Lemma. For every $\epsilon > 0$ there exists $\delta > 0$ such that $|u_j((d_i), B, X, h, \ell) - u_j((d_i), B, X, h, \ell')| < \epsilon$ whenever $d(\ell, \ell') < \delta$. δ is independent of j, (d_i), B, X, h.

Proof.

$$|u_j((d_i), B, X, h, \ell) - u_j((d_i), B, X, h, \ell')|$$

$$\leq \left| \int_{-d_{j-1}(X)}^{-d_j(X)} \ell(0, h) - \ell'(0, h) \right|$$

$$\leq \int_{-\infty}^{\infty} |\ell(0, 1) - \ell'(0, 1)|$$

$$< \epsilon$$

whenever $d(\ell, \ell') < \delta$ and $\delta = \epsilon$. ●

3.9.5 Lemma. For $j = 1, \ldots, N$, u_j is a continuous function from $\mathcal{T}_4$ to R.

Proof. The proof is so similar to the proof of Lemma 3.6.5 that we omit it. ●

3.9.6 Theorem. g_1 and g_2 are continuous functions from $\mathcal{T}_4$ to R.

Proof. The theorem is an immediate consequence of Lemma 3.9.5. ●

3.10 Convergence of g_1 and g_2 to f and q when $u_i(X) = \int_{R_i} k(X,h)$, First Proof

3.10.1 Theorem. Let the u_i's be as defined on line (3.3.2) and let $q = \sum_{i=1}^{N} b_i \xi_{R_i}$. Then as h converges to zero,

(a) g_1 and g_2 converge pointwise to q on $\bigcup_{i=1}^{N} R_i^0$, and

(b) g_1 and g_2 converge uniformly to q on $\bigcup_{i=1}^{N} R_i^a$.

Let D be the domain of f. As h and Δf, converge to zero,

(c) g_1 and g_2 converge pointwise to f on $D \cap \bigcup_{i=1}^{N} R_i^0$, and

(d) g_1 and g_2 converge uniformly to f on $D \cap \bigcup_{i=1}^{N} R_i^a$.

Proof. (a) Let X in $\bigcup_{i=1}^{N} R_i^0$ be given, and WLOG assume X is R_j^0 for some index j between 1 and N. There exists $\delta > 0$, whose value may depend on X, such that $B_\delta(X) \subseteq R_j^0$. As

$$u_j(X) = \int_{R_j} k(X,h)$$

$$\geq \int_{B_\delta(X)} k(X,h)$$

$$\geq \int_{B_{\delta/h}(0)} k(0,1)$$

and as for all $i \neq j$

$$u_i(X) \leq \int_{A_\delta(X)} k(X, h)$$

$$\leq \int_{A_{\delta/h}(0)} k(0, 1),$$

then

$$\lim_{h \to 0} u_i(X) = \begin{cases} 0 & \text{if } i \neq j \\ 1 & \text{if } i = j . \end{cases}$$

From the definitions of g_1 and g_2 it now follows that

$$\lim_{h \to 0} g_1(X) = \lim_{h \to 0} g_2(X) = q(X).$$

(b) Note that the value of h needed to insure that $u_i(X)$ is within some $\epsilon > 0$ of its limit depends on δ, only indirectly on X, and not on i. Hence, the value of h needed to insure that $g_1(X)$ and $g_2(X)$ are within some $\epsilon > 0$ of $q(X)$ depends on δ and only indirectly on X. On $\bigcup_{i=1}^{N} R_i^a$, δ can be chosen independently of X, and g_1 and g_2 converge uniformly to q.

(c, d) On $D \cap \bigcup_{i=1}^{N} R_i$, $|f(X)-q(X)| \leq \Delta f$, so that parts (c) and (d) of the theorem follow from parts (a) and (b). ●

Theorem 3.10.1 describes the behavior of g_1 and g_2, as h and Δf converge to zero, at all points of $\bigcup_{i=1}^{N} R_i^0$. In Theorem

3.12.1 we will extend the results of Theorem 3.10.1 to $(\bigcup_{i=1}^{N} R_i)^0$. There will be some difficulty establishing the extension, because as we shall now see, g_1 and g_2 can behave badly at points outside $\bigcup_{i=1}^{N} R_i^0$.

Let (i)

$$k(x) = \begin{cases} 1/2 & \text{if } |x| \leq 1 \\ 0 & \text{otherwise} \end{cases}$$

be a kernel function defined on R,

(ii) $N = 2$,

(iii) $b_1 = 1$, $b_2 = 0$, and

(iv) $R_1 = \bigcup_{n=0}^{\infty} [-2^{-2n}, -2^{-2n-1}] \cup [2^{-2n-1}, 2^{-2n}]$ and

$R_2 = [-1, 1] \setminus R_1$.

Then for x in $[-1, 1]$

$$g_1(x) = g_2(x) = b_1 \int_{R_1} k(x, h) + b_2 \int_{R_2} k(x, h)$$

$$= \int_{R_1} k(x, h)$$

$$= (1/(2h))m(R_1 \cap [-h+x, h+x]) ,$$

as

$$(1/h)k((y-x)/h) = \begin{cases} 1/(2h) & \text{if} \quad |(y-x)/h| \leq 1 \\ 0 & \text{otherwise .} \end{cases}$$

Hence

$$g_1(0) = g_2(0) = \begin{cases} \frac{2^n}{2} \; \frac{1}{2^n} \; \frac{4}{3} & \text{if} \quad h = \frac{1}{2^n} \quad \text{and} \quad n = 0, 2, 4, \ldots \\ \frac{2^n}{2} \; \frac{1}{2^{n+1}} \; \frac{4}{3} & \text{if} \quad h = \frac{1}{2^n} \quad \text{and} \quad n = 1, 3, 5, \ldots \end{cases}$$

$$\text{as} \quad m(R_1 \cap [-1/2^n, 1/2^n]) = \begin{cases} \frac{1}{2^n} \; \frac{4}{3} & \text{if} \quad n = 0, 2, 4, \ldots \\ \frac{1}{2^{n+1}} \; \frac{4}{3} & \text{if} \quad n = 1, 3, 5, \ldots \end{cases}$$

$$= \begin{cases} \frac{2}{3} & \text{if} \quad h = \frac{1}{2^n} \quad \text{and} \quad n = 0, 2, 4, \ldots \\ \frac{1}{3} & \text{if} \quad h = \frac{1}{2^n} \quad \text{and} \quad n = 1, 3, 5, \ldots, \end{cases}$$

and $\lim_{h \to 0} g_1(0)$ and $\lim_{h \to 0} g_2(0)$ do not exist. Notice that zero is not in R_1^0 or R_2^0 but is in $(R_1 \cup R_2)^0$.

Now k is not continuous; however,

$$k'(x) = \begin{cases} 1/2 & \text{if} \quad |x| < c \\ (1/2)|(d-x)/(d-c)| & \text{if} \quad c < |x| \leq d \\ 0 & \text{otherwise,} \end{cases}$$

where $c = 11/12$ and $d = 13/12$, is continuous, and

$$\left| \int_{R_1} k(0,h) - k'(0,h) \right| \leq (d-c)/2 \leq 1/12$$

for all $h > 0$. Hence, if g_1 and g_2 are defined in terms of k' instead of k,

$$g_1(0) = g_2(0) \begin{cases} \geq 7/12 & \text{if } h = 1/2^n \text{ and } n = 0, 2, 4, \ldots \\ \geq 5/12 & \text{if } h = 1/2^n \text{ and } n = 1, 3, 5, \ldots, \end{cases}$$

and $\lim_{h \to 0} g_1(0)$ and $\lim_{h \to 0} g_2(0)$ do not exist.

3.11 Convergence of g_1 and g_2 to f and q when $u_i(X) = \int_{-d_{i-1}(X)}^{-d_i(X)} \ell(0,h)$, First Proof

Let $d_0, d_1, \ldots, d_N$ be discriminant functions with $d_0 \equiv \infty$, $d_N \equiv -\infty$ and $d_{i-1}(Z) \geq d_i(Z)$ for all Z. Let

$$A_i = \{Z \mid d_i(Z) < 0\}, \qquad i = 0, \ldots, N,$$

$$R_i = A_i \setminus A_{i-1}, \qquad i = 1, \ldots, N,$$

$${}^0R_i = \{Z \mid d_{i-1}(Z) > 0 \text{ and } d_i(Z) < 0\}, \qquad i = 1, \ldots, N, \text{ and}$$

$${}^\alpha R_i = \{Z \mid d_{i-1}(Z) > \alpha \text{ and } d_i(Z) < \alpha\}, \qquad i = 1, \ldots, N,$$

where $\alpha > 0$ is some constant. As $d_{i-1}(Z) \geq d_i(Z)$ for all Z, then $A_{i-1} \subseteq A_i$, the R_i's are pairwise disjoint, and

$\bigcup_{i=1}^{N} R_i = R^n$. Each R_i can also be characterized by

$$R_i = \{Z \mid d_{i-1}(Z) \geq 0 \text{ and } d_i(Z) < 0\} .$$

There is no particular relationship between ${}^0R_i$ and R_i^0 and between ${}^aR_i$ and R_i^a, but if the discriminant functions are continuous, then ${}^0R_i \subseteq R_i^0$.

3.11.1 Theorem. Let

(i) $d_0, d_1, \ldots, d_N$ be discriminant functions with $d_0 \equiv \infty$, $d_N \equiv -\infty$ and $d_{i-1}(Z) \geq d_i(Z)$ for all Z,

(ii) the u_i's be as defined on line (3.3.19), and

(iii) $q = \sum_{i=1}^{N} b_i \xi_{R_i}$.

Then as h converges to zero,

(a) g_1 and g_2 converge pointwise to q on $\bigcup_{i=1}^{N} {}^0R_i$, and

(b) g_1 and g_2 converge uniformly to q on $\bigcup_{i=1}^{N} {}^aR_i$.

Let D be the domain of f. As h and Δf converge to zero,

(c) g_1 and g_2 converge pointwise to f on $D \cap \bigcup_{i=1}^{N} {}^0R_i$, and

(d) g_1 and g_2 converge uniformly to f on $D \cap \bigcup_{i=1}^{N} {}^aR_i$.

Proof. (a) Let X in $\bigcup_{i=1}^{N} {}^{0}R_i$ be given, and WLOG assume X is in ${}^{0}R_j$ for some index j between 1 and N. For $i \neq j$ the interval $(-d_{i-1}(X), -d_i(X))$ does not contain the origin, while for $i = j$ the interval does. Hence

$$\lim_{h \to 0} u_i(X) = \lim_{h \to 0} \int_{-d_{i-1}(X)}^{-d_i(X)} \ell(0,h) = \begin{cases} 0 & \text{if } i \neq j \\ 1 & \text{if } i = j . \end{cases}$$

From the definitions of g_1 and g_2, it now follows that

$$\lim_{h \to 0} g_1(X) = \lim_{h \to 0} g_2(X) = q(X).$$

(b) If X is in ${}^{a}R_j$, then

$$u_i(X) \begin{cases} \geq \int_{-a}^{a} \ell(0,h) & \text{if } i = j \\ \geq \int_{-\infty}^{-a} \ell(0,h) + \int_{a}^{\infty} \ell(0,h) & \text{if } i \neq j . \end{cases}$$

The limit as h converges to zero of $u_i(X)$ is one if $i = j$ and zero otherwise. The value of h needed to insure that $u_i(X)$ is within some fixed distance of its limit depends on a and not at all on X. Hence, there exists $h' > 0$, whose value is independent of X, such that $g_1(X)$ and $g_2(X)$ are within some $\epsilon > 0$ of $q(X)$ whenever $h < h'$. This proves part (b).

(c, d) On $D \cap \bigcup_{i=1}^{N} R_i$, $|f(X)-q(X)| \leq \Delta f$, so parts (c) and (d) follow from parts (a) and (b). ●

3.12 Confergence of g_1 and g_2 to f when $u_i(X) = \int_{R_i} k(X,h)$, Second Proof

3.12.1 Theorem. Let f be continuous and bounded on its domain D, and let the u_i's be as defined on line (3.3.2). Then as h and Δf converge to zero,

(a) g_1 and g_2 converge pointwise to f on $D \cap (\bigcup_{i=1}^{N} R_i)^0$, and

(b) g_1 and g_2 converge uniformly to f on every closed and bounded subset of $D \cap (\bigcup_{i=1}^{N} R_i)^a$.

Proof. In part (A) below we prove the theorem for g_1. The proof does not seem to extend to g_2. In part (B) we introduce a second approach to proving the theorem, first proving the theorem for g_1 in part (C) and then for g_2 in part (D).

(A.a) Let X in $D \cap (\bigcup_{i=1}^{N} R_i)^0$ and $\epsilon > 0$ be given. Suppose that

$$\Delta f < \epsilon/4 . \tag{3.12.1}$$

There exists $\delta > 0$ such that

$$|f(X)-f(Y)| < \epsilon/4 \quad \text{whenever} \quad \|X-Y\| < \delta \tag{3.12.2}$$

and

$$B_\delta(X) \subseteq \bigcup_{i=1}^{N} R_i \,. \tag{3.12.3}$$

For $i = 1, \ldots, N$ let

$$\overline{R}_i = R_i \cap A_\delta(X) \quad \text{and} \quad \underline{R}_i = R_i \cap B_\delta(X). \tag{3.12.4}$$

For $i = 1, \ldots, N$ if $\underline{R}_i \neq \phi$, then

$$|f(X)-b_i| \leq |f(X)-f(Y)| + |f(Y)-b_i|$$

where Y is in $\underline{R}_i$

$$\leq \epsilon/4 + \Delta f_i$$

$$< \epsilon/2$$

by lines (3.12.1) and (3.12.2)

Now

$$|f(X)-g(X)| \leq \Big|f(X) - \sum_{i=1}^{N} b_i \int_{R_i} k(X,h)\Big|$$

$$\leq \Big|f(X) \int_{A_\delta(X)} k(X,h) + \int_{B_\delta(X)} k(X,h)$$

$$- \sum_{i=1}^{N} b_i \int_{\overline{R}_i} k(X,h) + \int_{\underline{R}_i} k(X,h) \Big|$$

$$\leq b \int_{A_\delta(X)} k(X,h) + b \int_{\bigcup_{i=1}^{N} \overline{R}_i} k(X,h) \qquad (3.12.6)$$

$$+ \left| f(X) \int_{B_\delta(X)} k(X,h) - \sum_{i=1}^{N} b_i \int_{\underline{R}_i} k(X,h) \right| , \qquad (3.12.7)$$

where $b = \max\{ \sup_{X \in D} \{|f(X)|\}, \max_i \{|b_i|\}\}$.

The expression on line (3.12.6) is less than or equal to

$$2b \int_{A_\delta(X)} k(X,h)$$

as $\bigcup_{i=1}^{N} \overline{R}_i \subseteq A_\delta(X)$ by line (3.12.3)

$$\leq 2b \int_{A_{\delta/h}(X)} k(0,1)$$

$$< \epsilon/2 ,$$

whenever $0 < h < h'$ and h' is sufficiently small. The value of h' depends on b and δ and only indirectly on X.

By line (3.12.3) $B_\delta(X) = \bigcup_{i=1}^{N} \underline{R}_i$, so that the expression on line (3.12.7) is less than or equal to

$$\left| \sum_{i=1}^{N} (f(X) - b_i) \int_{\underline{R}_i} k(X,h) \right| < \epsilon/2$$

by line (3.12.5).

(A.b) For any X in $D \cap (\bigcup_{i=1}^{N} R_i)^0$ the proof of part (A.a) depends on X only inasmuch as the choise of a $\delta > 0$ small enough to satisfy the conditions on lines (3.12.2) and (3.12.3) may depend on X. If X is in a closed and bounded subset E of $D \cap (\bigcup_{i=1}^{N} R_i)^a$, then, as f is uniformly continuous on E, a $\delta > 0$ may be chosen, independent of X, such that the condition on line (3.12.2) is satisfied. If in addition $\delta < a$, then the condition on line (3.12.3) is also satisfied.

(B.a) Let X in $D \cap (\bigcup_{i=1}^{N} R_i)^0$ and $\epsilon > 0$ be given. Suppose that

$$\Delta f < \epsilon/8. \tag{3.12.8}$$

There exists $\delta > 0$ such that

$$|f(X)-f(Y)| < \epsilon/8 \quad \text{whenever} \quad \|X-Y\| < \delta \tag{3.12.9}$$

and

$$B_\delta(X) \subseteq \bigcup_{i=1}^{N} R_i . \tag{3.12.10}$$

Let $i' = \min_{1 \le i \le N} \{i | \underline{R}_i \neq \phi\}$ and $i'' = \max_{1 \le i \le N} \{i | \underline{R}_i \neq \phi\}$; let $b' = b_{i'}$ and $b'' = b_{i''}$. Then

$$\bigcup_{i=i'}^{i''} \underline{R}_i = B_\delta(X) \quad \text{and} \quad b' \leq b_i \leq b'' \tag{3.12.11}$$

for $i = i', \ldots, i''$ (remember that $b_i \leq b_{i+1}$, $i = 1, \ldots, N-1$).

Let Y' and Y'' be in $\underline{R}_{i'}$ and $\underline{R}_{i''}$ respectively. Then

$$|b'-f(X)| \leq |b'-f(Y')| + |f(Y')-f(X)|$$

$$\leq \epsilon/4 \qquad \text{by lines (3.12.8) and (3.12.9),}$$

and

$$|b''-f(X)| \leq |b''-f(Y'')| + |f(Y'')-f(X)|$$

$$\leq \epsilon/4 \qquad \text{by lines (3.12.8) and (3.12.9).}$$

Hence

$$b''-b' \leq \epsilon/2, \tag{3.12.12}$$

and

$$-\epsilon/4+b' \leq f(X) \leq b''+\epsilon/4. \tag{3.12.13}$$

In part (C) we will show that, for a sufficiently small $h' > 0$,

$$-\epsilon/4+b' \leq g_1(X) \leq b''+\epsilon/4, \tag{3.12.14}$$

whenever $0 < h < h'$. Combining lines (3.12.13) and (3.12.14) yields

$$-\epsilon/2-(b''-b') \leq f(X) - g_1(X) \leq (b''-b')+\epsilon/2,$$

and hence by line (3.12.12)

$$|f(X)-g_1(X)| < \epsilon.$$

In part (D) we will show that, for a sufficiently small $h' > 0$,

$$-\epsilon/4+b' \leq g_2(X) \leq b''+\epsilon/4 \qquad (3.12.15)$$

whenever $0 < h < h'$ and consequently that

$$|f(X)-g_2(X)| < \epsilon.$$

(B.b) In parts (C) and (D) it will be seen that the choice of a $h' > 0$ needed to satisfy the inequalities on lines (3.12.14) and (3.12.15) depends on $b = \max_i\{|b_i|+1\}$ and on δ and only indirectly on X. Therefore g_1 and g_2 converge uniformly to f on every closed and bounded subset $D \cap (\bigcup_{i=1}^{N} R_i)^a$.

(C) Let $h' > 0$ be sufficiently small to insure that for all $0 < h < h'$

$$\int_{A_{\delta/h}(0)} k(0,1) \leq \epsilon/(8Nb) \quad \text{and} \quad \int_{B_{\delta/h}(0)} k(0,1) \leq 1-\epsilon/(8b), \qquad (3.12.16)$$

where WLOG we assume that $\epsilon/(8b) < 1$ and where $b = \max_i\{|b_i|+1\}$. Note that the value of h' depends on b and δ and only indirectly on X.

For all $i \neq i', \ldots, i''$, $R_i = \phi$, so that

$$u_i(X) = \int_{\underline{R}_i} k(X,h)$$

$$\leq \int_{A_\delta(X)} k(X,h)$$

$$\leq \int_{A_{\delta/h}(0)} k(0,1)$$

$$\leq \epsilon/(8Nb) \qquad (3.12.17)$$

whenever $0 < h < h'$ by line (3.12.16).

Also, for all $0 < h < h'$

$$\sum_{i=i'}^{i''} u_i(X) \leq 1, \qquad (3.12.18)$$

and

$$\sum_{i=i'}^{i''} u_i(X) = \sum_{i=i'}^{i''} \left\{ \int_{\underline{R}_i} k(X,h) + \int_{\overline{R}_i} k(X,h) \right\}$$

$$\geq \int_{B_\delta(X)} k(X,h)$$

by line (3.12.11)

$$\geq \int_{B_{\delta/h}(0)} k(0,1)$$

$$\geq 1 - \epsilon/(8b) \qquad \text{by line (3.12.16).} \qquad (3.12.19)$$

From lines (3.12.18) and (3.12.19) it follows that

$$b'(1-\epsilon/(8b)) \leq b' \sum_{i=i'}^{i''} u_i(X) \quad \text{if} \quad b' \geq 0,$$

$$b' \leq b' \sum_{i=i'}^{i''} u_i(X) \quad \text{if} \quad b' < 0,$$

and

$$b'-\epsilon/8 \leq b' \sum_{i=i'}^{i''} u_i(X). \tag{3.12.20}$$

Similarly,

$$b'' \sum_{i=i'}^{i''} u_i(X) \leq b''+\epsilon/8. \tag{3.12.21}$$

As

$$g_1(X) = \sum_{i=1}^{N} b_i u_i(X),$$

then for all $0 < h < h'$

$$-(N-k)b\frac{\epsilon}{8Nb} + \sum_{i=i'}^{i''} b_i u_i(X) \leq g_1(X) \leq \sum_{i=i'}^{i''} b_i u_i(X) + (N-k)b\frac{\epsilon}{8Nb},$$

by line (3.12.17), where $k = i''-i'+1$

$$-\epsilon/8 + b' \sum_{i=i'}^{i''} u_i(X) \leq g_1(X) \leq b'' \sum_{i=i'}^{i''} u_i(X) + \epsilon/8,$$

by line (3.12.11)

and

$$-\epsilon/4 + b' \leq g_1(X) \leq b'' + \epsilon/4$$

by lines (3.12.20) and (3.12.21).

(D) For all $i = 1, \ldots, (i'-1)$, $R_i = \phi$, so that $u_i(X) \leq \int_{A_{\delta/h}(0)} k(0, 1)$, see line (3.12.17), and $\lim_{h \to 0} u_i(X) = 0$. As

$$v_i(X) = \sum_{\ell=0}^{i} u_\ell(X),$$

remember $u_0 \equiv 0$, then $\lim_{h \to 0} v_i(X) = 0$ whenever $i = 1, \ldots, (i'-1)$. For all $i = i'', \ldots, N$, $\bigcup_{\ell=1}^{i} R_\ell \supset B_\delta(X)$, see line (3.12.11),

$$v_i(X) = \int_{\bigcup_{\ell=1}^{i} R_\ell} k(X, h)$$

$$\geq \int_{B_\delta(X)} k(X, h)$$

$$\geq \int_{B_{\delta/h}(0)} k(0, 1),$$

and $\lim_{h \to 0} v_i(X) = 1$. Hence, for all $i \neq i', \ldots, i''$

$$\lim_{h \to 0} w_i(X) = 0, \tag{3.12.22}$$

and there exists $h_1 > 0$ such that

$$w_i(X) \leq \epsilon/(8Nb(2N)^N), \tag{3.12.23}$$

where $b = \max_i \{|b_i| + 1\}$, whenever $0 < h < h_1$.

Let $h_2 > 0$ be sufficiently small to insure that $\int_{B_{\delta/h}(0)} k(0, 1) \geq 1/2$ whenever $0 < h < h_2$. Then for $0 < h < h_2$

$$\max_{i' \leq i \leq i''} \{u_i(X)\} \geq 1/(2N), \quad \text{as}$$

$$\sum_{i=i'}^{i''} u_i(X) = \int_{\bigcup_{i=i'}^{i''} R_i} k(X, h)$$

$$\geq \int_{B_\delta(X)} k(X, h)$$

by line (3.12.11)

$$\geq \int_{B_{\delta/h}(0)} k(0, 1)$$

$$\geq 1/2.$$

Let $\ell, i' \leq \ell \leq i''$, be any index at which $u_\ell(X) \geq 1/(2N)$. Then for

$i = 1, \ldots, (\ell-1)$, $(1-v_i(X)) \geq 1/(2N)$, and for $i = \ell, \ldots, N$, $v_i(X) \geq 1/(2N)$. Hence, $w_\ell(X) \geq (1/(2N))^N$ as $1-v_0(X) = 1$; and

$$\sum_{i=1}^{N} w_i(X) \geq (1/(2N))^N \tag{3.12.24}$$

whenever $0 < h < h_2$.

Let $h' = \min\{h_1, h_2\}$. Then for all $0 < h < h'$

$$\sum_{i=i'}^{i''} w_i(X) / \sum_{i=1}^{N} w_i(X) \geq 1 - \sum_{i \neq i', \ldots, i''} w_i(X) / \sum_{i=1}^{N} w_i(X)$$

$$\geq 1 - \frac{(N-k)\epsilon/(8Nb(2N)^N)}{(1/2N)^N}$$

by lines (3.12.23) and (2.12.24)

where $k = i''-i'+1$

$$\geq 1 - \epsilon/(8b) . \tag{3.12.25}$$

Clearly,

$$\sum_{i=i'}^{i''} w_i(X) / \sum_{i=1}^{N} w_i(X) \leq 1. \tag{3.12.26}$$

From lines (3.12.25) and (3.12.26) it follows that

$$b' - \epsilon/8 \leq b' \sum_{i=i'}^{i''} w_i(X) / \sum_{i=1}^{N} w_i(X), \tag{3.12.27}$$

$$b'' \sum_{i=i'}^{i''} w_i(X) \Big/ \sum_{i=1}^{N} w_i(X) \leq b'' + \epsilon/8. \qquad (3.12.28)$$

As $g_2(X) = \sum_{i=1}^{N} b_i w_i(X) \Big/ \sum_{i=1}^{N} w_i(X)$, then for all $0 < h < h'$

$$\frac{-(N-k)b\epsilon/(8Nb(2N)^N)}{\sum_{i=1}^{N} w_i(X)} + \frac{\sum_{i=i'}^{i''} b_i w_i(X)}{\sum_{i=1}^{N} w_i(X)} \leq g_2(X)$$

$$\leq \frac{\sum_{i=i'}^{i''} b_i w_i(X)}{\sum_{i=1}^{N} w_i(X)} + \frac{(N-k)b\epsilon/(8Nb(2N)^N)}{\sum_{i=1}^{N} w_i(X)},$$

by line (3.12.23)

$$-\epsilon/8 + b' \sum_{i=i'}^{i''} w_i(X) \Big/ \sum_{i=1}^{N} w_i(X) \leq g_2(X)$$

$$\leq b'' \sum_{i=i'}^{i''} w_i(X) \Big/ \sum_{i=1}^{N} w_i(X) + \epsilon/8,$$

by line (3.12.24)

and

$$-\epsilon/4 + b' \leq g_2(X) \leq b'' + \epsilon/4,$$

by lines (3.12.27) and (3.12.28). ●

3.13 Convergence of g_1 and g_2 to f when $u_i(X) = \int_{-d_{i-1}(X)}^{-d_i(X)} \ell(0,h)$, Second Proof.

3.13.1 Corollary. Let

(i) f be continuous and bounded on its domain D,

(ii) $d_0, d_1, \ldots, d_N$ be discriminant functions with $d_0 \equiv \infty$ and $d_N \equiv -\infty$,

(iii) $A_{i-1} \subseteq A_i$, $R_i = A_i \setminus A_{i-1}$, $i = 1, \ldots, N$, see line (3.3.18),

(iv) the u_i's be as defined on line (3.3.19),

(v) $d_i(X) \leq -q(\delta)$ if $B_\delta(X) \subseteq A_i$, and

$d_i(X) > q(\delta)$ if $B_\delta(X) \subseteq B_i$,

where $\delta > 0$, $i = 1, \ldots, N-1$, q is a function from the positive real numbers to the positive real numbers and $B_i = R^n \setminus A_i$.

(vi) $\gamma = \frac{1}{2} \min\left\{\int_{-\infty}^{0} \ell(0,1), \int_{0}^{\infty} \ell(0,1)\right\} > 0.$

Then as h and Δf converge to zero,

(a) g_1 and g_2 converge pointwise to f on D, and

(b) g_1 and g_2 converge to f on every closed and bounded subset of D.

Proof. The proof is very similar to the proof given in parts (B), (C) and (D) of Theorem 3.12.1. We now describe the modifications to

these parts needed in the present situation. All of part (B.a) of Theorem 3.12.1 applies, but the set relationships on lines (3.12.10) and (3.12.11) are not needed here. The remarks of part (B.b) remain valid--except that the choice of a $h' > 0$ now depends on b and $a = q(\delta)$, rather than on b and δ, and still only indirectly on X. Part (C) followed from the relationships on lines (3.12.17), (3.12.18) and (3.12.19). For the present situation we will establish equivalent relationships on lines (3.13.3), (3.13.4), (3.13.5) and (3.13.6). Part (D) followed from the fact that

$$\lim_{h \to 0} v_i(X) = \begin{cases} 0 & \text{if} \quad i = 1, \ldots, (i'-1) \\ 1 & \text{if} \quad i = i'', \ldots, N, \end{cases}$$

where the value of $h > 0$ needed to insure that $v_i(X)$ was within some $\epsilon > 0$ of the limit depended on δ and only indirectly on X, and from the fact that there existed ℓ, $1 \leq \ell \leq N$, and $h' > 0$ dependent on δ such that

$$1 - v_i(X) \geq 1/(2N) \quad \text{if} \quad i = 1, \ldots, (\ell - 1)$$

and

$$v_i(X) \geq 1/(2N) \quad \text{if} \quad i = \ell, \ldots, N$$

whenever $0 < h < h'$. We will establish equivalent relationships on lines (3.13.7), (3.13.8), (3.13.9) and (3.13.10).

Let $h' > 0$ be sufficiently small to insure that for all $0 < h < h'$

$$\int_{-\infty}^{-a} \ell(0,h) + \int_{a}^{\infty} \ell(0,h) \leq \epsilon/(8Nb) \tag{3.13.1}$$

and

$$\int_{-a}^{a} \ell(0,h) \geq 1 - \epsilon/(8b), \tag{3.13.2}$$

where WLOG we assume that $\epsilon/(8b) < 1$ and where $a = q(\delta)$. Note that the value of h' depends on $b = \max_i \{|b_i| + 1\}$ and on a and only indirectly on X.

For all $i = 1, \ldots, (i'-1)$, $B_\delta(X) \subseteq B_i$ and $d_i(X) > a$. Hence, for all $0 < h < h'$

$$\begin{aligned} |u_i(X)| &\leq \left| \int_{-d_{i-1}(X)}^{-d_i(X)} \ell(0,h) \right| \\ &\leq \int_{-\infty}^{-a} \ell(0,h) \\ &\leq \epsilon/(8Nb) . \end{aligned} \tag{3.13.3}$$

For all $i = i'', \ldots, N$, $B_\delta(X) \subseteq A_i$ and $d_i(X) < -a$. Hence for all $i = (i''+1), \ldots, N$ and for all $0 < h < h'$,

$$\begin{aligned} |u_i(X)| &\leq \int_{a}^{\infty} \ell(0,h) \\ &\leq \epsilon/(8Nb). \end{aligned} \tag{3.13.4}$$

As $B_\delta(X) \subseteq B_{i'-1}$ and as $B_\delta(X) \subseteq A_{i''}$, then $d_{i'-1}(X) > a$ and $d_{i''}(X) < -a$. Hence for all $0 < h < h'$

$$\sum_{i=i'}^{i''} u_i(X) = v_{i''}(X) - v_{i'-1}(X)$$

$$= \int_{-d_{i'-1}(X)}^{-d_{i''}(X)} \ell(0,h)$$

$$\geq \int_{-a}^{a} \ell(0,h)$$

$$\geq 1 - \epsilon/(8b) \qquad \text{by line (3.13.2).} \qquad (3.13.5)$$

Also, for all $0 < h < h'$,

$$\sum_{i=i'}^{i''} u_i(X) \leq 1. \qquad (3.13.6)$$

For $i = 1, \ldots, (i'-1)$,

$$B_\delta(X) \subseteq B_i, \quad d_i(X) > a,$$

$$v_i(X) = \int_{-\infty}^{-d_i(X)} \ell(0,h) \leq \int_{-\infty}^{-a} \ell(0,h),$$

and hence

$$\lim_{h \to 0} v_i(X) = 0. \qquad (3.13.7)$$

For $i = i'', \ldots, N$, $B_\delta(X) \subseteq A_i$, $d_i(X) < -a$,

$$v_i(X) \leq \int_{-\infty}^{a} \ell(0, h),$$

and hence

$$\lim_{h \to 0} v_i(X) = 1. \tag{3.13.8}$$

Let $\gamma > 0$ be defined by

$$\gamma = (1/2)\min\left\{\int_{-\infty}^{0} \ell(0, 1), \int_{0}^{\infty} \ell(0, 1)\right\}.$$

Let $\ell = \min\{i \mid X \text{ in } A_i \text{ and } 1 \leq i \leq N\}$. For $i = 1, \ldots, (\ell - 1)$, X is not in A_i, $d_i(X) \geq 0$, and

$$\begin{aligned} 1 - v_i(X) &= \int_{-\infty}^{\infty} \ell(0, h) - \int_{-\infty}^{-d_i(X)} \ell(0, h) \\ &= \int_{-d_i(X)}^{\infty} \ell(0, h) \\ &\geq \int_{0}^{\infty} \ell(0, h) \\ &\geq \gamma \qquad \text{whenever} \quad 0 < h < h'. \end{aligned} \tag{3.13.9}$$

For $i = \ell, \ldots, N$, X is in A_i (as $A_i \subseteq A_{i+1}$, $i = 1, \ldots, N-1$), $d_i(X) < 0$, and

$$v_i(X) = \int_{-\infty}^{-d_i(X)} \ell(0,h)$$

$$\geq \int_{-\infty}^{0} \ell(0,h)$$

$$\geq \gamma \qquad \text{whenever} \quad 0 < h < h'. \bullet \quad (3.13.10)$$

3.14 Stability of g_1 and g_2 with Respect to the Discriminant Functions

Let

(i) d_i and d_i' be discriminant functions for $i = 0, \ldots, N$,

(ii) $d_0 \equiv d_0' \equiv \infty$ and $d_N \equiv d_N' \equiv -\infty$.

(iii) $d_{i-1}(X) \geq d_i(X)$ and $d_{i-1}'(X) \geq d_i'(X)$ for all X and for $i = 1, \ldots, N$,

(iv) $$u_i(X) = \int_{-d_{i-1}(X)}^{-d_i(X)} \ell(0,h),$$

$$u_i'(X) = \int_{-d_{i-1}'(X)}^{-d_i'(X)} \ell(0,h),$$

(v) g_1, g_2 and g_1', g_2' be the estimates determined by the u_i's and u_i''s respectively,

(vi) $$\text{Error}(g_1(X)) = |g_1(X) - g_1'(X)| / (b_N - b_1),$$

$$\text{Error}(g_2(X)) = |g_2(X) - g_2'(X)| / (b_N - b_1),$$

(vii) $e_i(X) = \int_{-d_i(X)}^{-d_i'(X)} \ell(0,h)$, and

$$e(X) = \max\{|e_i(X)| : i = 0, \ldots, N\}.$$

The condition $d_{i-1}(X) \geq d_i(X)$ and $d_{i-1}'(X) \geq d_i'(X)$ for all i and X insures that g_2 and g_2' are defined on all of R^n, see Theorem 3.4.1 and the discussion following the theorem.

3.14.1 Theorem. $\text{Error}(g_1(X)) \leq e(X)$.

Proof.

$$|g_1(X) - g_1'(X)| = \left| \sum_{i=1}^{N} b_i(u_i(X) - u_i'(X)) \right|$$

$$= \left| \sum_{i=1}^{N} b_i(e_i(X) - e_{i-1}(X)) \right|$$

by line (3.5.2)

$$= \left| -e_0(X)b_1 + \sum_{i=1}^{N-1} e_i(X)(b_i - b_{i+1}) + e_N(X)b_N \right|$$

$$= \left| \sum_{i=1}^{N-1} e_i(X)(b_i - b_{i+1}) \right| \qquad (3.14.1)$$

as $e_0(X) = e_N(X) = 0$

$$\leq e(X)|b_1 - b_N| .$$

Hence $\text{Error}(g_1(X)) \leq e(X)$. ●

For $i = 1, \ldots, N$, let $\epsilon_i(X) = w_i'(X) - w_i(X)$, where w_i' and w_i are derived from u_i' and u_i in the manner defined on lines (3.4.3) and (3.4.4). Let $\epsilon(X) = \max\{|\epsilon_i(X)| : i = 1, \ldots, N\}$. Let $\mathcal{E}(X) = (\epsilon_i(X))_{i=1}^N$, $W(X) = (w_i(X))_{i=1}^N$, $B = (b_i)_{i=1}^N$, $B_1 = (b_1)_{i=1}^N$ and $B' = (B - B_1)/(b_N - b_1)$. And for any m-tuple $Q = (q_1, \ldots, q_m)$ of real numbers let $[Q]$ denote $\sum_{i=1}^{m} q_i$. Note that the components of B' are between zero and one.

3.14.2 Theorem.

$$\text{Error}(g_2(X)) \leq \frac{|\mathcal{E}(X) \cdot B' - [\mathcal{E}(X)]((g_2(X) - b_1)/(b_N - b_1))|}{|[W(X) + \mathcal{E}(X)]|} \qquad (3.14.2)$$

Proof. For convenience we write W, W', $\mathcal{E}$ for $W(X)$, $W'(X)$, $\mathcal{E}(X)$ respectively.

$$\begin{aligned} |g_2'(X) - g_2(X)| &= \left| \frac{B \cdot W'}{[W']} - \frac{B \cdot W}{[W]} \right| \\ &= \left| \frac{B \cdot (W + \mathcal{E})}{[W + \mathcal{E}]} - \frac{B \cdot W}{[W]} \right| \\ &= \left| \frac{([W]\mathcal{E} - [\mathcal{E}]W) \cdot B}{[W + \mathcal{E}][W]} \right| \\ &= \left| \frac{([W]\mathcal{E} - [\mathcal{E}]W) \cdot ((b_N - b_1)B' + B_1)}{[W + \mathcal{E}][W]} \right| . \end{aligned}$$

Hence,

$$\text{Error}(g_2(X)) \le \left| \frac{([W]\mathcal{E}-[\mathcal{E}]W)\cdot B'}{[W+\mathcal{E}][W]} \right| + \left| \frac{([W]\mathcal{E}-[\mathcal{E}]W)\cdot B_1}{[W+\mathcal{E}][W]} \right| / (b_N-b_1)$$

$$\le \frac{([W]\mathcal{E}-[\mathcal{E}]W)\cdot B'}{[W+\mathcal{E}][W]}$$

as $[W]\mathcal{E}\cdot B_1 = [W][\mathcal{E}]b_1 = [\mathcal{E}]W\cdot B_1$

$$\le \frac{\cdot B'-[\mathcal{E}]\frac{W}{[W]}\cdot\frac{(B-B_1)}{b_N-b_1}}{[W+\mathcal{E}]}$$

$$\le \frac{\mathcal{E}\cdot B'-[\epsilon]((g_2(X)-b_1)/(b_N-b_1))}{[W+\mathcal{E}]} \cdot \bullet$$

The bound for $\text{Error}(g_2(X))$ on line (3.14.2) is expressed in terms of $\mathcal{E}(X)$ rather than $E(X) = (e_i(X))_{i=0}^{N}$. We now express each $\epsilon_i(X)$ in terms of the $e_i(X)$'s. As $d_{i-1}(X) \ge d_i(X)$ and $d'_{i-1}(X) \ge d'_i(X)$, then

$$v_i(X) = \sum_{j=0}^{i} u_j(X) = \int_{-\infty}^{-d_i(X)} \ell(0,h) ,$$

$$v'_i(X) = \int_{-\infty}^{-d'_i(X)} \ell(0,h) ,$$

and $v'_i(X)-v_i(X) = e_i(X)$. To maintain readability we write v_i, v'_i, e_i, w_i, w'_i, ϵ_i instead of $v_i(X)$, $v'_i(X)$, $e_i(X)$, $w_i(X)$, $w'_i(X)$, $\epsilon_i(X)$ respectively.

$$\epsilon_i = w_i' - w_i$$

$$= \left(\prod_{k=0}^{i-1} (1-v_k') \prod_{k=i}^{N} v_i' \right) - w_i$$

$$= \left(\prod_{k=0}^{i-1} (1-v_k-e_k) \prod_{k=i}^{N} (v_k-e_k) \right) - w_i$$

$$= w_i + \sum_{j=0}^{N} \operatorname{sgn}(j-i) \left(\prod_{k=0}^{j-1} (1-v_k) \right) e_j \prod_{k=j+1}^{N} v_k + \mathcal{O}(e^2) - w_i$$

$$= \sum_{j=0}^{N} \operatorname{sgn}(j-i) \left(\prod_{k=0}^{j-1} (1-v_k) \right) e_j \prod_{k=j+1}^{N} v_k + \mathcal{O}(e^2) . \qquad (3.14.3)$$

As $0 \leq v_k$, $(1-v_k) \leq 1$, $k = 0, \ldots, N$, then

$$|\epsilon_i(X)| \leq (N+1)e(X) + \mathcal{O}(e^2). \qquad (3.14.4)$$

The expressions on lines (3.14.2) and (3.14.3) are too complex to be easily interpreted, and the bound for $|\epsilon_i(X)|$ on line (3.14.4) is unaesthetically large. We now make some simplifying assumptions in an attempt to find an easily interpreted bound for $\text{Error}(g_2(X))$.

3.14.3 Theorem. In addition to the assumptions made at the beginning of the section let

(i) the b_i's be equally spaced and $\Delta b = b_i - b_{i-1}$,

(ii) $d_{k-1}(X) \geq 0 \geq d_k(X)$ for some index k, and

(iii) $e_i(X) = 0, \quad i = 0, \ldots, k-2 \quad \text{or} \quad i = k+1, \ldots, N.$ Then

$$\text{Error}(g_1(X)) \leq 2e(X)/(N-1). \tag{3.14.5}$$

If in addition

(iv) $u_i(X) = 0, \quad i = 0, \ldots, k-2 \quad \text{or} \quad i = k+1, \ldots, N,$ and

(v) the kernel function is symmetric about the origin, then

$$\text{Error}(g_2(X)) \leq 2e(X)/\{(N-1)(.75-e(X))\} \tag{3.14.6}$$

provided $.75 > e(X)$.

Proof. By (iii) above and line (3.14.1),

$$\begin{aligned}\text{Error}(g_1(X)) &\leq \{e_{k-1}(X)(b_{k-1}-b_k)+e_k(X)(b_k-b_{k+1})\}/(b_N-b_1)\\ &\leq 2e(X)\Delta b/(b_N-b_1)\\ &\leq 2e(X)/(N-1),\end{aligned}$$

which establishes the inequality on line (3.14.5).

It is more difficult to prove the inequality on line (3.14.6). By (iv) above and the condition $d_{i-1}(X) \geq d_i(X), \quad i = 0, \ldots, N,$

$$v_i(X) = \begin{cases} 0 & \text{if} \quad i = 1, \ldots, k-2 \\ 1 & \text{if} \quad i=k+1, \ldots, N, \end{cases} \tag{3.14.7}$$

and

$$w_i(X) = \begin{cases} 0 & \text{if } i = 1,\ldots,k-2 \text{ or } i = k+2,\ldots,N \\ v_{k-1}v_k & \text{if } i = k-1 \\ (1-v_k)v_k & \text{if } i = k \\ (1-v_{k-1})(1-v_k) & \text{if } i = k+1, \end{cases} \tag{3.14.8}$$

where the argument X is implicitly assumed when writing v_{k-1} and v_k. By lines (3.14.3) and (3.14.7)

$$\epsilon_i(X) = \begin{cases} 0 & \text{if } i = 1,\ldots,k-2 \text{ or } i = k+2,\ldots,N \\ e_{k-1}v_k + v_{k-1}e_k & \text{if } i = k-1 \\ -e_{k-1}v_k + (1-v_{k-1})e_k & \text{if } i = k \\ -e_{k-1}(1-v_k) - (1-v_{k-1})e_k & \text{if } i = k+1 \end{cases} \tag{3.14.9}$$

if we ignore the term $\mathcal{O}(e^2)$. The absolute value of the numerator of the fraction on the right of inequality (3.14.2) equals

$$\left|\epsilon_{k-1}(b'_{k-1}-c)+\epsilon_k(b'_k-c)+\epsilon_{k+1}(b'_{k+1}-c)\right|$$

where b'_{k-1}, b'_k, b'_{k+1} are components of B' and where $c = (g_2(X)-b_1)/(b_N-b_1)$

$$\leq \left|(e_{k-1}v_k+v_{k-1}e_k)(b'_{k-1}-c) + (-e_{k-1}v_k+(1-v_{k-1})e_k)(b'_k-c)\right.$$
$$\left. + (-e_{k-1}(1-v_k)-(1-v_{k-1})e_k)(b'_{k+1}-c)\right|$$

by line (3.14.9)

$$\leq \left| (e_{k-1}v_k+(1-v_{k-1})e_k)\Delta b'+e_{k-1}(1-v_k)(b'_{k+1}-c) \right.$$

$$\left. + v_{k-1}e_k(c-b'_{k-1}) \right| \qquad (3.14.10)$$

obtained by rearranging terms, multiplying by -1, and letting $\Delta b'$ be the distance between any two contiguous b'_i's.

$$\leq e\{v_k\Delta b'+(1-v_{k-1})\Delta b'+(1-v_k)(b'_{k+1}-c)+v_{k-1}(c-b'_{k-1})\}$$

$$\leq e\{v_k\Delta b'+(1-v_{k-1})\Delta b'+(1-v_k)(2\Delta b'-a)+v_k a\}$$

where $a = c - b'_{k-1}$

$$\leq e\{3\Delta b'-a-\Delta b'(v_k+v_{k-1})+a(v_k+v_{k-1})\}$$

$$\leq e\{3\Delta b'-a-(\Delta b'-a)(v_k+v_{k-1})\}$$

$$\leq e\{2\Delta b'+(\Delta b'-a)-(\Delta b'-a)(v_k+v_{k-1})\}$$

$$\leq e\{2\Delta b'-(\Delta b'-a)(v_k+v_{k-1}-1)\}. \qquad (3.14.11)$$

Now

$$c = (g_2(X)-b_1)/(b_N-b_1)$$

$$= (b'_{k-1}w_{k-1}+b'_k w_k+b'_{k+1}w_{k+1})/(w_{k-1}+w_k+w_{k+1}). \qquad (3.14.12)$$

On line (3.14.11), $\Delta b'-a > 0$ implies

$$c-b'_{k-1} < \Delta b'$$

implies $b'_{k-1} \leq c \leq b'_k$

implies $w_{k-1} > w_{k+1}$ by line (3.14.12)

implies $v_{k-1}v_k > (1-v_{k-1})$ by line (3.14.8)

implies $0 > 1-v_{k-1}-v_k$

implies $(\Delta b'-a)(v_k+v_{k-1}-1) > 0.$

Similarly, $\Delta b'-a < 0$ implies $(\Delta b'-a)(v_k+v_{k-1}-1) > 0$. If $\Delta b'-a = 0$, then $(\Delta b'-a)(v_k+v_{k-1}-1) = 0$. Hence, the expression on line (3.14.11) is less than or equal to

$$2e\Delta b' \leq 2e/(N-1) \qquad (3.14.13)$$

which is a bound for the numerator of the fraction on the right side of inequality (3.14.2).

The absolute value of the denominator of the fraction on the right side of inequality (3.14.2) is

$$|[W+\mathfrak{E}]| = |v_{k-1}v_k+(1-v_{k-1})v_k+(1-v_{k-1})(1-v_k)+[\mathfrak{E}]|$$

by line (3.14.8)

$$= |1-v_{k-1}(1-v_k)+[\mathfrak{E}]|$$

as $v_{k-1}v_k+(1-v_{k-1})v_k+(1-v_{k-1})(1-v_k)$ $+v_{k-1}(1-v_k) = 1$

$$= \left| 1-v_{k-1}(1-v_k)+v_{k-1}e_k-e_{k-1}(1-v_k) \right| \qquad (3.14.14)$$

by line (3.14.9)

$$\geq \left| 1-v_{k-1}(1-v_k) \right| -e(v_{k-1}+(1-v_k))$$

$$\geq .75 - e \qquad (3.14.15)$$

as conditions (ii), (iv), (v) assure that

$$0 \leq v_{k-1} \quad \text{and} \quad (1-v_k) \leq 1/2.$$

Combining lines (3.14.13) and (3.14.15) establishes the inequality on line (3.14.6). ●

Table 3.14.1 shows the values of

$$\left| g_1(X)-g_1'(X) \right| = \text{Error}(g_1(X)) \cdot (N-1)$$

and

$$\left| g_2(X)-g_2'(X) \right| = \text{Error}(g_2(X)) \cdot (N-1)$$

computed from Equations (3.14.1) and (3.14.2) respectively for a situation in which all the constraints of Theorem 3.14.3 are satisfied. The function $f(x) = x$ is estimated at the points

$$x_1 = (b_{k-1}+b_k)/2$$

$$x_2 = (b_{k-1}+3b_k)/4$$

$$x_3 = b_k$$

$$x_4 = (3b_k+b_{k+1})/4 \quad \text{and} \quad x_5 = (b_k+b_{k+1})/2.$$

The kernel function is

$$\ell(z) = \begin{cases} 1-|z| & \text{if } |z| \leq 1 \\ 0 & \text{otherwise .} \end{cases}$$

The meaning of each column of Table 3.14.1 is as follows:

x - the value at which the estimates g_1 and g_2 are being made;

ℓe_{k-1} - the size of $e_{k-1}(x)$ for the various values of x when $d'_{k-1}(x) = d_{k-1}(x) - .1$;

re_{k-1} - the size of $e_{k-1}(x)$ for the various values of x when $d'_{k-1}(x) = d_{k-1}(x) + .1$;

ℓe_k - similar to the column labeled ℓe_{k-1};

re_k - similar to the column labeled re_{k-1};

$\ell\ell g_1$ - the value of $|g_1(x)-g'_1(x)|$ when $e_{k-1}(x)$ and $e_k(x)$ are determined from columns ℓe_{k-1} and ℓe_k respectively.

$\ell r g_1$ - similar to $\ell\ell g_1$ except that $e_{k-1}(x)$ and $e_k(x)$ are determined from columns ℓe_{k-1} and re_k respectively.

$r\ell g_1$ and $rr g_1$ - obvious

$\ell\ell g_2$, $\ell r g_2$, $r\ell g_2$ and $rr g_2$ - similar to preceding column except $|g_2(x)-g'_2(x)|$ replaces $|g_1(x)-g'_1(x)|$.

As can be seen from Table 3.14.1, $|g_1(x)-g_1'(x)|$ is approximately equal to $|g_2(x)-g_2'(x)|$. In the worst case the error is as large as the bound given on line (3.14.5) and never as large as the bound given on line (3.14.6). Notice that the table considers the four possible cases that occur as each error alternates between being positive and negative.

Table 3.14.1. Computation of $|g_1(x)-g_1'(x)|$ and $|g_2(x)-g_2'(x)|$.

x	ℓe_{k-1}	re_{k-1}	ℓe_k	re_k	$\ell\ell$		ℓr		$r\ell$		rr	
					g_1	g_2	g_1	g_2	g_1	g_2	g_1	g_2
x_1	-.1	.1	0	0	-.100	-.100	-.100	-.100	.100	.100	.100	.100
x_2	-.07	.08	-.03	.02	-.100	-.101	-.050	-.053	.050	.055	.100	.101
x_3	-.045	.055	-.055	.045	-.100	-.102	.000	.000	.000	.000	.100	.102
x_4	-.02	.03	-.8	.07	-.100	-.101	.050	.053	-.050	-.055	.100	.101
x_5	0	0	-.1	.01	-.100	-.100	.100	.100	-.100	-.100	.100	.100

IV. BINARY VARIABLES

4.1 Advantages of Binary Variables

Whether by choice or necessity, the inputs to recognition networks are often binary [10, p. 111]. We give examples from the work of other investigators showing that in some instances replacing multi-leveled variables by binary variables is advantageous, while in other instances at least it is not harmful.

Binary variables can simplify the implementation of a given classifier. For instance, Bayes's criterion for classifying the patterns in a 2-class problem can be implemented by a TLU when the loss function is symmetric and the variables are binary and independent [24]. As another example, a matrix inversion is usually required to find the linear function which best approximates a given function in the mean-square sense; however, if the variables are binary, matrix inversion is unnecessary [10].

Shoenfelt *et al.* [30] state that binary variables can increase the recognition rate of a classifier while simultaneously reducing storage requirements, classification time, and the cost of making measurements of a problem's environment. They also note that distortions arising from differences in the scale of the variables disappear. They base their remarks on experiments made on a speech recognition problem. To put the discussion of the next two sections in

proper perspective, it is useful to describe the experiments in some detail [30]:

> These data were obtained by IBM for preliminary study concerned with their Expo 70 word-recognition demonstration. The isolated Japanese word data set employed in this study consisted of 4000 utterances-40 words, 5 speakers, and 20 repetitions per word per speaker. Several confusable word groups are represented. For example, Shichi, Shi, Nichi, Ichi, and Hachi are very similar. Forty-nine feature measurements corresponding to the outputs of a set of bandpass filters were employed. Each feature used eight binary digits of storage. It therefore took a total of 392 binary digits to represent the 49 features. The 4000 utterances were recorded in five different sessions; four repetitions per word per speaker were recorded in each. The utterances were divided into a training set and a test set. Each set consisted of 10 repetitions of each of the 40 words by the 5 speakers.

Each of the multi-leveled variables (the variables had 256 levels corresponding to the values $0, 1, \ldots, 255$) was replaced in the usual way by a number of binary variables. Each binary variable corresponded to a threshold in the range of a multi-leveled variable and was $+1$ for a pattern if the value of the multi-leveled variable at the pattern was above the threshold, otherwise it was -1. The number of binary variables used to represent each multi-leveled variable was the smallest integer not less than the minimum of the rate-distortion bound and of the average mutual information between the classes and the multi-leveled variable. The number was one for each multi-leveled variable. The rate distortion bound for a variable is

$$R = \frac{1}{2} \log_2 \frac{S}{N} ,$$

where S/N is the variance of the class means divided by the average within-class variance, and [30]:

> The average mutual information between all classes and some feature X_k is defined as
>
> $$I(X_k, C) = \sum_{j=1}^{L} \sum_{i=1}^{M} P(X_k = j | C_i) P(C_i) \log_2 \frac{P(X_k = j | C_i)}{P(X_k = j)} , \quad (4.1.1)$$
>
> where M is the number of classes and L is the number of feature levels.

The placement of each threshold was determined by a modified mutual information measure as follows [30]: the expression within braces on line (4.1.1)

> is the (discrete) contribution of a discrete feature value to the average mutual information measure and can be written as $I(X_k = j, C)$. The average mutual information distribution function $D(\ell)$ can then be defined as
>
> $$D(\ell) = \sum_{j=1}^{\ell} I(X_k = j, C),$$
>
> where $I \leq \ell \leq L$. The modified mutual information procedure used to select the thresholds is to place a threshold at θ_A for each $A = 1, 2, \ldots, 2^B - 1$ where
>
> $$D(\theta_A) = A(\frac{1}{2})^B I(X_k, C)$$
>
> and
>
> $$B = \min\lceil R, I(X_k, C) \rceil .$$
>
> $\lceil \ \rceil$ indicates rounding to next higher integer.

The distance-to-mean classifier as determined from the training set had recognition rates of 82%, 88%, and 90% for 30, 40, and 49 binary variables respectively; but when the experiment was repeated with the multi-leveled variables, the recognition rates were 75%, 85%, and 87% for 30, 40, and 49 multi-leveled variables respectively.

Duda and Fossum [7] found that replacing multi-leveled variables by binary variables did not affect the recognition rate of a classifier composed of linear discriminant functions. The data was obtained by adding Gaussian random noise to 80 ten-dimensional prototype vectors and forming 32 classes, each class containing one or more prototypes and the patterns obtained by adding noise to these prototypes. Binary variables were obtained by introducing ten equally spaced thresholds into the range of each multi-leveled variable.

4.2 Additional Advantages of Binary Variables

In this section we describe additional advantages of using binary variables. By Theorem 2.7.3 whenever the components of patterns are binary, every 2-class problem in which the classes are disjoint can be solved by a modified veto committee. The proof of Theorem 2.7.3 calls for as many committee members as there are patterns in the smaller of the two classes; however, simulations to be described in Section 5.3 show that when the local adjustment algorithm of Definition 2.9.5 is used, a committee of many fewer members is obtained.

When non-binary variables are replaced by binary variables in the manner described in the last section, the improvement in the recognition rate of a classifier can be much larger than the improvement Shoenfelt [30] observed. The two computer simulations whose results are summarized in Figure 5.4.1 and Table 5.4.1 show that when binary variables replace multi-leveled variables, a TLU is capable of a high recognition rate on a problem for which is otherwise would be totally unsuited.

The implementation of the estimates g_1 and g_2 can be easier when binary variables replace multi-leveled variables. It was pointed out in Section 3.3 that the simplest computational scheme for computing g_1 and g_2 is obtained when

$$u_i(X) = \int_{-d_{i-1}(X)}^{-d_i(X)} \ell(0, h), \quad i = 1, \ldots, N.$$

Given 2-class classifiers which are TLUs, discriminant functions can be determined easily. Binary variables increase the number of instances in which TLUs can be trained to have a high recognition rate. A discriminant function can be defined by a TLU as follows. Let W be the TLU's weight vector, let X be a pattern, and let $\overline{W}$ and $\overline{X}$ be the vectors obtained by dropping the last component of W and X respectively. Let

$$d(X) = \frac{X \cdot W}{\| \bar{W} \|} \quad . \qquad (4.2.1)$$

We can interpret $d(X)$ as the signed distance between $\bar{X}$ and the hyperplane in $\bar{X}$-space defined by $\bar{W} \cdot Y - w_n = 0$, where w_n is the last component of W.

4.3 Selection of Binary Variables

While the method outlined in Section 4.1 and used by Schoenfelt [30] to replace a multi-leveled variable by a number of binary variables chooses the number of binary variables and the thresholds well, it has the disadvantage of treating each multi-leveled variable separately, perhaps creating a set of binary variables containing redundant information, especially if the original variables contain redundant information. It would be better to create a set of binary variables which among the sets of that size contains the most information possible.

Mucciardi and Gose [22] describe a method for choosing a "good" subset of variables from a set of variables. They note that "the only guaranteed technique for choosing the best subset of N properties from a set of M is to try all $\binom{M}{N}$ possible combinations" but that "this is impractical for sets of even moderate size, so heuristic techniques are required." They might also have added that the subset which is best for one classifier is not necessarily the best for another.

We now give a brief description of their technique, called the weighted sum or WS technique.

The variables are ranked, and those with the highest rank are retained. The variable with the highest rank is that for which the expected probability of error, POE, is smallest. Suppose the k-1 highest ranking variables have been chosen. For some variable with index i that has not been chosen let

$$POE_i' = \frac{POE_i - POE_{min}}{POE_{max} - POE_{min}},$$

where POE_{min} and POE_{max} are the smallest and largest values of the probability of error when the probability is computed for each variable and where POE_i is the probability of error for variable i. Let ACC_i equal the average of the absolute values of the correlation coefficients between variable i and the ones already chosen. The k-th ranked feature is the one for which the weighted sum of POE_i' and ACC_i is smallest for some predetermined set of weights.

Using a set of data derived from EKG graphs, Mucciardi and Gose found that for suitably chosen weights the WS technique compares favorably with the Karhunen-Loeve expansion and the technique of choosing at each step that feature which in conjunction with those already chosen does most to improve the performance of the classifier. The WS technique has the advantage that is is much easier to

implement and is much faster than the other two.

For a 2-class problem, the WS technique is useful in choosing a set of binary variables to represent a set of multi-leveled variables. Given n multi-leveled variables, we introduce m equally spaced thresholds into the range of each variable, use the method described in Section 4.1 to associate a binary variable with a threshold, and use the WS technique to choose a "good" subset of these binary variables. In Section 5.4, a simulation using data on cranial capacities confirms the effectiveness of this scheme for choosing binary variables.

V. RESULTS OF COMPUTER SIMULATIONS

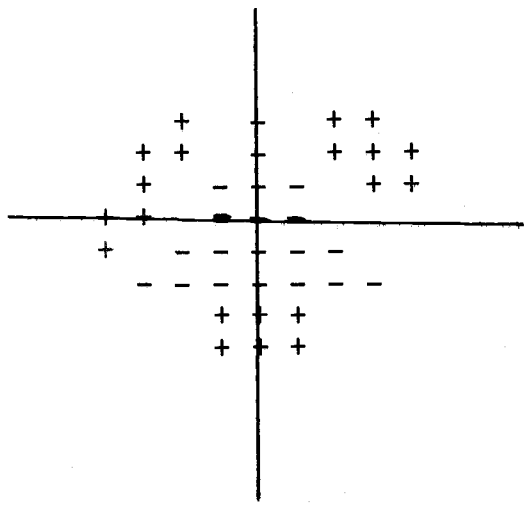
5.1 Training a Network

A local adjustment algorithm for training a network was described in Definition 2.4.5. We now describe the results obtained when this algorithm is used to train a committee of three members to recognize the 40 patterns shown in Figure 5.1.1 part (a), where "-" and "+" denote a pattern of class A and B respectively. Each pattern has a third component, constantly one, not shown. A variable a, called the age, is associated with each TLU in the network. Before training begins, the age of each TLU is 100, and immediately prior to an adjustment to a TLU the TLU's age is incremented by one. The value of the constant c on line (2.4.1) is

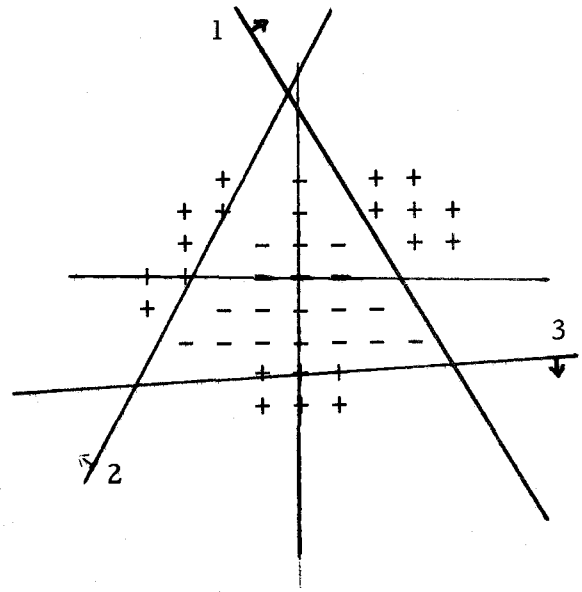
$$c = \frac{10}{a\|Y_X\|} .$$

Also, before training begins, the weight vector of each committee member is $(0, 0, -1)$. The training sequence is divided into groups of 40 patterns, and each group contains all the patterns in some random order. Only by chance are the patterns in two different groups in the same order.

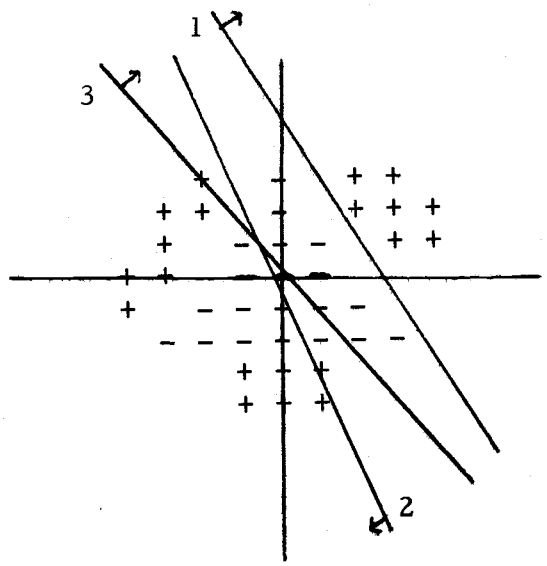
Three simulations are performed, with the weight vector of the vote-taking TLU having a different initial value in each simulation.



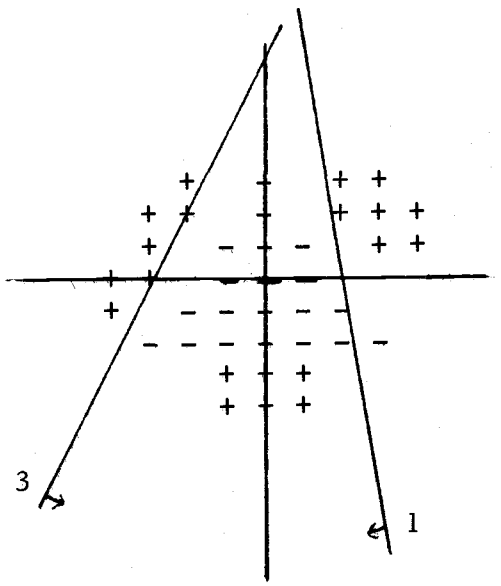
(a) The problem.



(b) Simulation 1.



(c) Simulation 2.



(d) Simulation 3.

Figure 5.1.1. A 2-class problem for a network.

The results are summarized in Table 5.1.1 with each column of the table describing the results of a different simulation. The first row shows the initial value of the vote-taking TLU; the second row shows the amount the ages of the members and the vote-taking TLU, indicated by VTT, increase during training; and the third row shows the recognition rate when the training stops. As we can see in Table 5.1.1, the age of the vote-taking TLU does not change during training and consequently neither does the committee's logic. In the first simulation it is acceptable that the logic does not change, as after a few adjustments the committee recognizes all the patterns. In the second and third simulation, the algorithm, to be successful, should change the logic of the committee. Meuller [23] proved that the logic used in the second simulation, which, incidently is majority logic, is not sufficient for solving the problem, and it is not hard to show that neither is the logic used in the third simulation.

Table 5.1.1. Results of training a network.

	Simulation 1	Simulation 2	Simulation 3
Initial value of the logic vector	(1, 1, 1, 2.5)	(1, 1, 1, 0)	(-1, 1, -1, 1)
Increase in age of TLUs			
$MMBR_1$	4	4	1256
$MMBR_2$	16	70	0
$MMBR_3$	4	1958	66
VTT	0	0	0
Recognition rate	100%	70%	83%

Referring to Figure 5.1.1 parts (c) and (d), we now explain why the logic does not change. Each line defines the decision boundary of a committee member, and the number beside the line indicates which member. The response of a committee member to a pattern is +1 if the pattern is on the same side of the line as the arrow attached to the line, otherwise the response is -1. In part (d), as $MMBR_2$ receives no adjustments and consequently responds -1 to every pattern, the line defining the decision boundary of $MMBR_2$ is not shown. In part (c) we see that if training continues, the response of the committee to a misclassified pattern can be changed by changing the response of $MMBR_3$, and this is the case regardless of the position of $MMBR_3$. As in addition the length of $MMBR_3$'s weight vector is much less than the length of $MMBR_1$'s and $MMBR_2$'s weight vector, $MMBR_3$ is chosen for adjustment even if $MMBR_1$ or $MMBR_2$ are also in the set P. After many additional adjustments, $MMBR_3$'s weight vector can be expected to remain by far the shortest, as it will be adjusted due to the conflicting demands of patterns that are not linearly separable. Hence, we can expect that no matter how long training continues the logic will not change. A similar explanation accounts for the outcome of the third simulation.

5.2 The Need to Normalize the Weight Vectors

In Section 2.9, when describing the algorithm for training a modified veto committee, we gave heuristic reasons for normalizing the length of a member's weight vector after each adjustment to the member. We now discuss four computer simulations of the algorithm which confirm the need for the normalization. In particular we shall see that, if the normalization is omitted, an adjustment can cause a drastic change in the response of a member, and the length of a member's weight vector may become so short that the member receives a disproportionate number of adjustments.

The simulations are illustrated in the four parts of Exhibit 1 (following Chapter VI). The patterns are vectors in R^3, and the last component of each pattern is one. When the third component is omitted, the patterns are as indicated in Exhibit 1 part (a) frame (1), where "·" and "*" indicate a pattern of class A and B respectively.

In the first simulation, the training sequence consists of successive groups, each containing all the patterns; the order of the patterns within each group is the same and is illustrated in part (a) frame (1). (The coordinate axis is shown only in frame (1).) The lines in the frames indicate the decision boundaries of committee members, the number beside a line indicates the index of the corresponding member, and the arrowpoints in the direction of those

patterns which a member does not ignore. A solid line represents a member of the first type and a broken line a member of the second type. When the position of a line makes it impossible to draw the line within a frame, the inclination of the line is shown, and an ordered pair indicates the x and y intercepts respectively of the line (see for instance part (a) frame (6)). If the decision boundary cannot be indicated by a line in R^2, then the weight vector of the member is shown preceded by a pair of humbers separated by a hyphen, where the first number is the index of the member and the second the member's type (see part (a) frame (3)). The successive frames of part (a) show the successive patterns in the training sequence that are misclassified. The pattern labeled "1" in frame (1) is the first misclassified, the committee is adjusted, and $MMBR_1$ assumes the position shown in frame (2). The n-th pattern misclassified is the pattern circled in frame (n), and the position of the committee members after the adjustment is shown in frame (n+1).

In a similar way the frames in parts (b), (c) and (d) describe the second, third and fourth simulations respectively. In parts (b) and (c) the first frames are not shown. In Part (d), although, $MMBR_2$ is added to the committee in frame (3), its decision boundary can never be indicated in R^2, and it is only explicitly mentioned after receiving an adjustment, i.e., frames (3), (10), (14), (18), (22), (26).

The results of the simulations are summarized in Table 5.2.1. For each simulation, the table indicates: whether or not the weight vectors are normalized after each adjustment; the initial values of the parameters α_0, β, and γ; the recognition rate; and the members on the committee with their type, increase in age, and final length.

Referring to Exhibit 1 part (a), we see that without normalization, adjustments can cause drastic changes in the response of a member (see for instance the transition between frames (5) and (6), (7) and (8), (14) and (15)). Even if the size of each adjustment is smaller, there continue to be drastic changes in the response of a member (see for instance part (b) frames (67) and (68), (69) and (70)). With normalization, adjustments do not cause drastic changes (see part (d)).

Referring to the third row of Table 5.2.1, we see that without normalization a member's weight vector may become so short that the member receives a disproportionate number of adjustments, in this case all the adjustments. In the fourth row we see that $MMBR_1$ receives a disproportionate number of adjustments; however, as the weight vector is normalized after each adjustment, eventually new members are added.

Table 5.2.1. Results obtained by the algorithm for modified veto logic without and with normalization.

Simulation	Normalized	a_0	β	γ	Recognition Rate	Member	Type	Increase in Age	Final Length
1	no	6	1	0	100%	1	1	4	.82
						2	1	11	.21
						3	2	5	.94
2	no	20	1	0	100%	1	1	25	.79
						2	1	49	.04
						3	2	7	.91
3	no	20	1	0	63%	1	1	100	.02
4	yes	6	1	0	100%	1	1	26	1
						2	1	6	1
						3	2	7	1

5.3 The Performance of Modified Veto Committees on Three Sets of Patterns

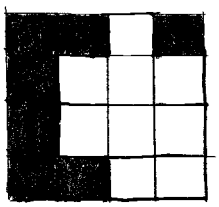
Mueller [23] describes three 2-class problems which he used to compare Ridgway's local adjustment algorithm for training a majority committee [26] with his own improved version of the algorithm. We now describe the problems and simulations performed on the problems using the algorithm for training a modified veto committee.

The first problem consists of 294 hand printed letters collected by Munson of Stanford University; the letters in one class being A's and in the other class being R's. The letters were written on a 24 x 24 grid and were of approximately uniform size and position. Thus each letter can be expressed as a vector of 576 zeros and ones, with a component being one if and only if the letter passes through the corresponding square in the grid. Mueller used just 19 of these components to represent each letter, and we use the same 19, augmenting as usual each pattern by the constant one. Mueller divided the patterns into a training set of 240 patterns and a test set of 54 patterns. We will call this division of the patterns AR1. Adding a duplicate of three of the A's and three of the R's chosen at random, shuffling the A's and shuffling the R's, we formed a training set of 240 patterns and a test set of 60 patterns. We call this division of the patterns AR2. Repeating the procedure a second time, we formed

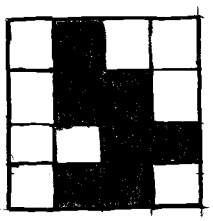
another division of the patterns which we call AR3. In the training set of AR1 the two classes intersect, and every classifier must have an error rate of at least 2% to 3% when judging the patterns, even if every pattern that is not in the intersection is judged correctly. In the test set the two classes do not intersect. For AR2, the minimum error rate is 3% to 5% in the training set and 3% in the test set, while for AR3 the minimum error rate is 2% to 3% in the training set and 2% in the test set.

Mueller's description of the second problem follows.

> Given a 4 x 4 array of squares with each array containing seven black squares. An array will be called disconnected if the seven black squares are neither face-wise nor corner-wise connected. A disconnected array will be represented by a 17-dimensional vector with binary components, a 1 standing for a black square, with the 17th component equal to 1 for all samples. Thus a pattern vector representing a disconnected array would be the following.

X =  = (1, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1)

> An array will be called connected if the seven black squares are all face-wise connected. Thus a pattern vector representing a connected array would be the following.

X =  = (0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1)

> This data was randomly generated. . . . In the experiment we have used 240 samples for the training set and an equal number for a test set.

We call this set of patterns CON1. Generating 360 more patterns in each class, making a training set of 240 samples and a test set of 960, we get a set of patterns we call CON2. Reversing the roles of the training and test sets yields CON3.

In the third problem the patterns have 11 binary components, the last component being one, and a pattern is in the first class if among the first 10 components the string 1101 appears, otherwise it is in the second class. We call the set of patterns used by Mueller, with 240 patterns in the training set and 240 in the test set, CODE 1. Generating 360 more patterns in each class, placing 240 patterns in the training set and 960 in the test set, we get CODE 2, while reversing the roles of the training and test sets we get CODE 3.

In all the simulations of the algorithm for training a modified veto committee, the training sequence consists of groups of patterns, each group containing all the training patterns; within each group every second pattern is in the same class, and only by chance are the patterns in different groups in the same order. Adjusting the committee in response to the patterns in a group is called an iteration.

The results of several of the most successful simulations are shown in the column labeled "Modified Veto Committee" of Table 5.3.1. The number preceding each hyphen is the recognition rate of

a modified veto committee on a training set, and the other number is the recognition rate on the corresponding test set. (Hyphenated numbers are used in all the tables of this section in a similar way, with the first and second number being the recognition rate of some classifier on a training and test set respectively.) In the column to the right of each pair of hyphenated numbers is a number indicating the number of members on the committee. The committees of various sizes shown for a given set of patterns are all subcommittees of a single committee, where a subcommittee is formed by dropping members added most recently. The last three columns give the values of the parameters α_0, β and γ. It is not significant that the parameters have values like 99, 401, -203 rather than 100, 400, -200 respectively as the algorithm is not sensitive to such small changes in the values of the parameters. The numbers separated by colons in the first column are the size of the training and test sets. The recognition rates of the distance-to-mean classifier is shown in the second column. The third and fourth columns show the results obtained by Mueller when he used Ridgway's and his own algorithm to train a majority committee with five members.

From Table 5.3.1 we see that in many instances the distance-to-mean classifier, while not performing as well as the committees on the training sets, performs just about as well on the test sets, indicating that the committees are learning prejudice. As a pattern can

be classified much more quickly by the distance-to-mean classifier than by a committee, it is to be preferred when speed is important. The distance-to-mean classifier can also be trained much more quickly than a committee, but as training needs to be done only once, the time it takes to train a classifier is not as important as the time it takes for the classifier to classify a pattern. For the data sets CON3, CODE 1, CODE 2 and CODE 3, the committees perform better than the distance-to-mean classifier on the training and test sets. When the training set is large, a modified veto committee performs excellently on both the training and test sets. The performance on the set CON3 might have improved if training had continued; however, training stopped when the committee wished to add more members than the computer implementation had allowed for.

Kaylor's conjecture [16], that for a problem in which the patterns have binary components there is a majority committee among the committees of minimum size, seems to be without practical significance, as in all instances a modified veto committee with five members performs better than a majority committee with five members. Further, for the data sets CON and CODE the modified veto committees with five members are subcommittees of larger committees and perhaps do not perform as well as modified veto committees restricted during training to a maximum of five members. The fact that subcommittees perform so well indicates that members added

most recently to the committee specialize in recognizing a few troublesome patterns without creating a prejudice that adversely affects the performance of the committee on the problem as a whole.

Table 5.3.2 shows the behavior on the set CON2 of the algorithm for training a modified veto committee. The first column of each row shows the number of an iteration. The remainder of the row shows under the heading "Number of Adjustments...", the number of adjustments made to each member of the committee from the beginning of training until the end of the current iteration and under the heading "Number of Decisions...", the number of decisions made by each member when, at the end of the current iteration, all the training patterns are classified by the committee. Under the heading "Number of Adjustments...", rather than label each column according to the type of member represented in the column, it is more convenient to let odd and even numbered columns represent members of the second and first type respectively. The first column represents $MMBR_0$ which, as $MMBR_0$ is never adjusted, naturally contains zeros. Other columns containing zeros indicate the absence of a member from the committee. The i-th column under both headings refers to the same member. Although $MMBR_0$ is never adjusted, it makes many decisions, i.e., every time the committee makes decisions by default. The last column shows the recognition rate on the training set. Tables 5.3.3 and 5.3.4 give the same information

for CON3 as Table 5.3.2 gives for CON2.

Several things can be seen in Tables 5.3.2, 5.3.3 and 5.3.4. Because a modified veto committee initially responds +1 to all patterns, the first members added to a committee are of the first type. Until a member of the second type is added, a modified veto committee is a veto committee. Although the recognition rate does not improve with each iteration (especially in the test set the recognition rate fluctuates), it does tend to improve as training continues. The number of decisions made by a particular member fluctuates from iteration to iteration, and old members receive just as many adjustments as new members. For instance, in Table 5.3.3 between iterations six and eleven the first three members on the committee receive more adjustments than the last five. If the first three members are being adjusted due to the conflicting demands of patterns that are not linearly separable--which seems likely or otherwise they would already have obtained suitable positions--it would be better for the committee to add new members. Decreasing the value of γ is one way to encourage the committee to stop adjusting old members and to add new members, but if γ is too small, new members are added too often. Alternatively, the algorithm can include a rule that prevents adjustments to an old member once a certain number of newer members have been added. All simulations which incorporated this rule gave poor results. A less drastic alternative, namely to

let the resistance equal the certainty times $(\alpha+\gamma)^2$, had little effect on the performance of the algorithm although it did change the best values of α_0, β and γ.

Tables 5.3.5 and 5.3.6 show the performance of subcommittees of the committees described in Tables 5.3.2, 5.3.3 and 5.3.4. Each time a member is dropped there is a small reduction in the recognition rate of the committee.

Tables 5.3.7 and 5.3.8 show the effect of varying the parameters α_0, β and γ and of increasing the number of constant components in a pattern. The number of constant terms is the number appearing in the column headed "N.V." minus 16. Each line of the tables shows the number of adjustments, etc., at the end of five iterations for different values of the parameters. In Table 5.3.8 some of the committees have so many members that the information on the number of adjustments and decisions continues on a second line. Comparing lines 2-3-4 and 6-7 of Table 5.3.7 and lines 2-3 of Table 5.3.8, we see that, if γ is decreased while the other parameters are held constant, new members are added more frequently. Members are added less frequently when adjustments are smaller, (see lines 4-5 of Table 5.3.7 and 3-4 of Table 5.3.8). The difference between lines four and six of Table 5.3.7 is that the number of constant terms in the representation of each pattern is different. When more constant terms are added, fewer members are added to the

Table 5.3.1. Recognition rate of several classifiers.

Data	Distance-to-Mean Classifier	Majority Committee 5-members (Ridgway)	Majority Committee 5-members (Mueller)	Modified Veto Committee	Number of Members	a_0	β	γ
AR1	92-85	97-85	94-85	95-89	3	99	10	0
240:54			97-	98-87	3	"	"	"
				98-85	4	"	"	"
AR2	91-87			95-93	4	99	10	0
240:60				96-88	5	"	"	"
AR3	91-88			92-92	3	99	10	0
240:60				97-92	5	"	"	"
				98-90	5	"	"	"
CON1		90-71	90-80					
240:240								
CON2	85-81			95-81	3	99[1]	10	299
240:960				98-81	5	"	"	"
				100-82	6	"	"	"
CON3	84-86			86-84	4	401	41	-300
960:240				89-88	5	"	"	"
				95-94	12	"	"	"
CODE 1		91-60	92-71					
240:240								
CODE 2	75-61			93 -78	5	301	29	-203
240:960				99.6-80	7	"	"	"
				100 -80	8	"	"	"
CODE 3	68-75			83 -85	4	499	10	-300
960:240				91 -93	5	"	"	"
				97 -98	6	"	"	"
				99.7-99.6	7	"	"	"

[1] $MMBR_1$ initially equals the hyperplane separating the means, and there are 20 variables, the last four being the constant one.

Table 5.3.2.[1] Training record showing number of decisions and adjustments: CON2; $\alpha_0 = 99$; $\beta = 10$; $\gamma = 299$.

It..	Number of Adjustments to Each Member										Number of Decisions Made by Each Member on the Training Set										R.R.
5	0	66	0	37	0	38	20	0	7		108	77	0	16	0	32	4	0	3		91 -
10	0	80	0	70	0	83	49	0	7	6	113	65	0	15	0	30	5	0	2	10	97 -
15	0	80	0	87	0	93	50	0	7	17	110	61	0	14	0	34	9	0	2	10	99.5-
18	0	80	0	93	0	93	50	0	7	17	109	61	0	15	0	34	9	0	2	10	100 -

[1] $MMBR_1$ initially equals the hyperplane separating the means of the two classes, and there are 20 variables, the last four being the constant one.

Table 5.3.3. Training record showing number of adjustments: CON3; $\alpha_0 = 401$; $\beta = 41$; $\gamma = -300$.

It.	Number of Adjustments to Each Member																				R.R.
1	0	38	0	48	0	40	0	37	1	0	3										84-85
2	0	65	0	75	0	74	0	63	4	0	25	1									87-84
3	0	77	0	98	0	89	0	93	20	0	38	20	2								90-87
4	0	92	0	111	0	99	0	114	30	0	48	45	12	1							90-85
5	0	107	0	118	0	116	0	133	39	0	55	62	22	3	1						94-87
6	0	124	0	125	0	126	0	146	47	0	61	72	32	14	3	0	1				90-84
7	0	139	0	137	0	131	0	158	50	0	62	76	34	25	5	0	2	1			95-89
8	0	157	0	145	0	135	0	172	50	0	70	84	45	28	5	0	6	2			94-90
9	0	171	0	146	0	140	0	180	55	0	70	86	55	38	6	0	19	7			95-94
10	0	182	0	146	0	142	0	186	57	0	71	87	62	50	7	0	25	17			94-91
11	0	182	0	146	0	142	0	186	57	0	71	87	62	50	7	0	25	18	1		95-91

Table 5.3.4. Training record showing number of decisions: CON3; $\alpha_0 = 401$; $\beta = 41$; $\gamma = -300$.

It.	Number of Decisions Made by Each Member or the Training and Test Sets																		R. R.
1	601	17	0	234	0	193	0	155	0	0	0								84-85
2	537	118	0	127	0	141	0	199	1	0	41	0							87-84
3	466	85	0	64	0	135	0	112	45	0	66	227	0						90-87
4	469	123	0	75	0	55	0	161	37	0	41	228	11	0					90-84
5	375	120	0	105	0	109	0	225	79	0	39	103	43	2	0				94-87
6	308	75	0	92	0	78	0	115	146	0	65	123	68	130	0	0	0		90-84
7	316	102	0	77	0	73	0	141	135	0	87	126	78	63	2	0	0	0	95-89
8	315	93	0	57	0	86	0	185	131	0	64	141	64	53	2	0	9	0	94-90
9	258	97	0	54	0	56	0	211	146	0	62	88	106	85	4	0	26	7	95-94
10	229	67	0	42	0	76	0	133	147	0	65	110	93	62	3	0	62	111	94-91
11	229	66	0	43	0	76	0	143	148	0	65	110	93	62	3	0	72	90	0	95-91

Table 5.3.5.[1] Recognition rate of subcommittees: CON2; $\alpha_0 = 99$; $\beta = 10$; $\gamma = 299$.

No.	Number of Decisions Made by Each Member on the Training and Test Sets										R. R.
1	672	528									86-81
2	621	494	0	85							90-80
3	527	356	0	85	0	232					95-81
4	488	347	0	85	0	209	71				97-82
5	486	346	0	80	0	208	71	0	9		98-81
6	476	333	0	73	0	197	67	0	7	47	100-82

[1] $MMBR_1$ initially equals the hyperplane separating the means of the two classes, and there are 20 variables, the last four being the constant one.

Table 5.3.6. Recognition rate of subcommittees: CON3; $\alpha_0 = 401$; $\beta = 41$; $\gamma = -300$.

No.	Number of Decisions Made by Each Member on the Training and Test Sets																		R. R.
1	1009	191																	62-64
2	752	132	0	316															79-77
3	676	129	0	244	0	151													83-80
4	533	121	0	113	0	133	0	300											86-84
5	415	111	0	102	0	121	0	271	180										89-88
6	349	111	0	97	0	117	0	242	170	0	114								89-87
7	331	111	0	56	0	102	0	218	161	0	90	131							91-90
8	284	109	0	56	0	88	0	201	150	0	79	121	112						92-90
9	263	92	0	48	0	76	0	184	150	0	78	115	106	88					94-91
10	263	92	0	48	0	76	0	184	150	0	78	115	105	86	3				94-91
11	231	78	0	46	0	76	0	159	149	0	66	113	103	86	3	0	90		93-90
12	299	66	0	43	0	76	0	143	148	0	65	110	93	62	3	0	72	90	0 95-91

Table 5.3.7. Effect of varying parameters: CON2; five iterations.

Line	a_0	β	γ	N.V.	Number of adjustments to Each Member	Number of Decisions Made by Each Member on the Training Set	R. R.
1	599	59	400	17	0 196 0 48	101 118 0 21	83-
2	99	10	900	17	0 183 0 51	107 107 0 26	35-
3	99	10	201	17	0 152 0 29 0 3 4	113 86 0 36 0 3 2	90-
4	99	10	-1	17	0 132 0 21 0 9 0 9 7 0 1	120 78 0 22 0 7 0 11 2 0 0	94-
5	99	5	-1	17	0 154 0 30 0 10 3	116 97 0 24 0 3 0	89-
6	99	10	-1	20	0 203 41 15	75 117 22 26	87-
7	99[1]	10	299	20	0 66 0 37 0 38 20 0 7	108 77 0 16 0 32 4 0 3	91-

[1]Initially $MMBR_1$ is the hyperplane separating the means of the two classes.

Table 5.3.8. Effect of varying parameters: CON3; five iterations.

Line	a_0	β	γ	N.V.	Number of Adjustments to Each Member	Number of Decisions Made by Each Member on the Training and Test Sets	R. R.
1	401	41	-300	17	0 107 0 118 0 116 0 113 39 0 55 62 22 3 1	375 120 0 105 0 109 0 225 79 0 39 103 43 2 0	94-87
2	99	10	-1	20	0 316 0 132 0 114 109 26 0 38 32 0 3	376 68 0 111 0 104 131 121 0 284 5 0 0	87-84
3	99	10	299	20	0 631 0 105 83 7 1	359 553 0 112 116 10 0	85-85
4	99	5	-1	20	0 796 0 123 104	637 389 0 174 0	83-80

committee, because most of the adjustments to a member are of the opposite sign to the initial value of the last component of the weight vector and consequently the certainty at all patterns is reduced.

Tables for the other sets of patterns are so similar that there seems little point in including them.

5.4 Replacing Multi-Leveled Variables by Binary Variables

In this section we describe three computer simulations which test the effect of replacing multi-leveled variables by binary variables. The first and second simulations use artificially generated data; and the third, data on cranial capacities.

The first simulation, involving a seemingly difficult 2-class problem in R^2, is illustrated in Figure 5.4.1 part (a), where "-" and "+" denote patterns in class A and B respectively. Assume that the patterns in Figure 5.4.1 part (a) are located at the grid points of a coordinate system, with the patterns in the lower left hand corner and upper right hand corner having coordinates $(0, 0)$ and $(14, 14)$ respectively. The pattern at position (I, J) is mapped into a pattern vector X with 30 binary valued components by

$$X_{IJ} = (x_1, x_2, \ldots, x_{29}, 1),$$

where

$$x_i = \begin{cases} 1 & \text{if } i \leq I \text{ or } 15 < i \leq 15+J \\ -1 & \text{otherwise .} \end{cases}$$

Parts (b), (c), and (d) of Figure 5.4.1 illustrate the performance of the three indicated classifiers when they are trained on the binary patterns obtained from part (a). In parts (b) and (c) the response of the classifiers to a pattern is -1 if and only if the pattern is in the hatched region of the figure, and the classifiers have recognition rates of 80% and 90% respectively. In pard (d) the classifier, a modified veto committee, recognizes all but the circled pattern. The hatched regions in the four parts of Figure 5.4.2 include just those patterns not ignored by the four members on the committee referred to in Figure 5.4.1 part (d).

The second simulation involves five similar 2-class problems. For

$$S = \{X_{IJ} : I, J = 0, \ldots, 15\} \tag{5.4.1}$$

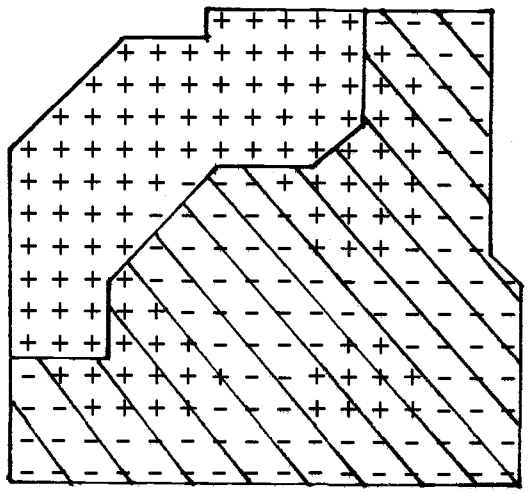
$$X_{IJ} = (x_1, x_2, \ldots, x_{30}, 1) \tag{5.4.2}$$

$$x_j = \begin{cases} 1 & \text{if } j \leq I \text{ or } 15 < j \leq 15+J \\ -1 & \text{otherwise} \end{cases} \tag{5.4.3}$$

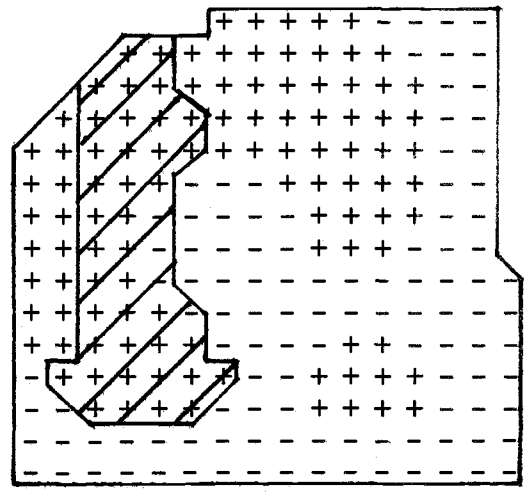
$$j = 1, \ldots, 30$$

$$f(x, y) = (x-7.5)^2 + (y-7.5)^2 \tag{5.4.4}$$

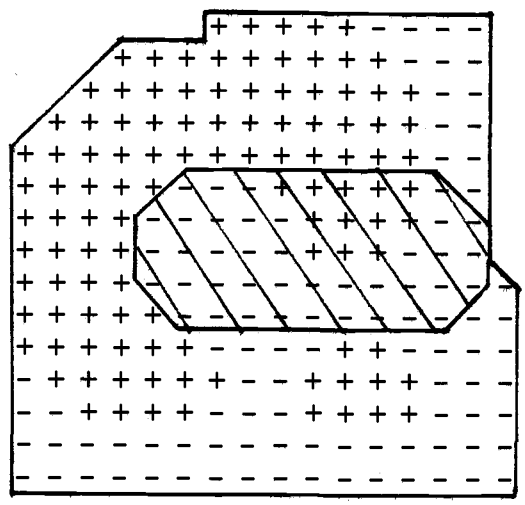
$$(t_1, t_2, \ldots, t_5) = (11.7, 34.1, 56.5, 78.9, 101.3), \tag{5.4.5}$$



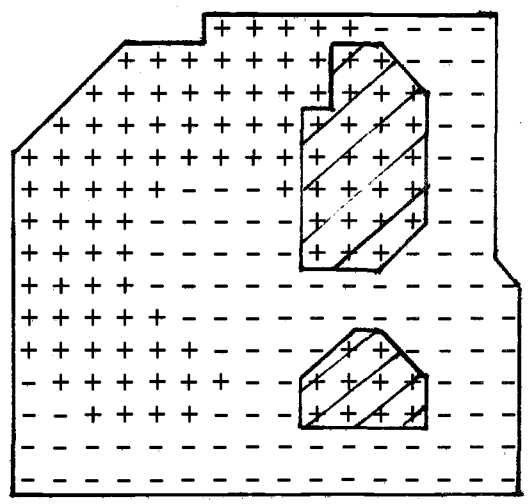
(a) $MMBR_1$-first type



(b) $MMBR_2$-second type



(c) $MMBR_3$-first type



(d) $MMBR_4$-second type

Figure 5.4.2. Decisions of each committee member.

let the i-th 2-class problem, $i = 1, \ldots, 5$, be

$$A_i = \{X_{IJ}: f(I, J) < t_i\} \tag{5.4.6}$$

$$B_i = S \setminus A_i \,. \tag{5.4.7}$$

Table 5.4.1 shows, for each 2-class problem, the recognition rate of the distance-to-mean classifier and the number of iterations needed of the local adjustment algorithm for training a TLU before the TLU recognizes all the patterns. For each 2-class problem, an iteration consists in adjusting a TLU in response to each of the patterns in S, and the initial weight of a TLU is that defined by the distance-to-mean classifier.

Table 5.4.1. The recognition rate in five similar 2-class problems.

2-Class Problem i	Number of Patterns in A_i and B_i	Recognition Rate of the Distance-to-mean Classifier	Number of Iterations Needed to Attain a Recognition Rate of 100%
1	32/224	89%	7
2	112/144	100%	0
3	172/84	94%	4
4	232/24	89%	1
5	252/4	92%	5

The third simulation also involves five similar 2-class problems. The data come from a table of cranial capacities with associated distances and arcs collected by Hooke [13]. Hooke's table contains

measurements for 245 adult skulls; but because for many skulls some measurements are missing, we select a subset of 110 skulls for which the cranial capacity and ten other measurements are present. Table 5.4.2 lists the 110 skulls and the ten measurements selected. The indices for the skulls and names of the measurements are those in Hooke's table. The largest of the 110 skulls has a cranial capacity of 1632.5cc; and the smallest, a capacity of 1121.0cc. Five 2-class problems are formed from the selected skulls by letting

$$t_i = 1121.0\text{cc} + (2i-1)(1632.5\text{cc}-1121.0\text{cc})/10 \qquad (5.4.8)$$

$$A_i = \{\text{skulls with a cranial capacity less than } t_i\} \qquad (5.4.9)$$

$$B_i = \{\text{skulls not in } A_i\}\ , \qquad (5.4.10)$$

for $i = 1, \ldots, 5$. The components of the patterns in A_i and B_i are the ten measurements named in Table 5.4.2. The cranial capacity is not a component, being used instead to define the 2-class problems. A training and test set for each 2-class problem is defined by ordering the data for the 110 skulls by their cranial capacities and using the data of every second skull as a training pattern and the remainder as test patterns.

Table 5.4.2. List of skulls and variables selected from Hooke's data.

Skulls Selected
2, 3, 4, 5, 6, 7, 8, 10, 11, 13, 14, 16, 17, 19, 20, 21, 22, 23, 24, 27, 28, 30, 31, 32, 33, 34, 35, 36, 37, 38, 40, 41, 42, 43, 44, 47, 48, 49, 52, 53, 58, 63, 64, 67, 69, 70, 71, 74, 83, 84, 85, 87, 88, 91, 93, 94, 95, 158, 161, 162, 163, 165, 166, 167, 176, 177, 184, 186, 188, 189, 190, 191, 192, 194, 196, 198, 199, 200, 201, 203, 204, 205, 206, 208, 209, 211, 213, 215, 216, 217, 218, 220, 221, 223, 224, 225, 227, 228, 229, 230, 231, 232, 236, 238, 239, 241, 242, 243, 244, 245
Variables Selected
L, B, B', H', OH, Q, S, S_3, S'_1, U

The patterns are mapped to patterns with binary components using the method outlined in Section 4.3. Twenty equally spaced thresholds are introduced into the range of each of the ten measurements to give 200 binary variables, and the WS technique is used to select a subset of 20 from these 200. Only the training patterns are referred to during the selection process, and each of the five 2-class problems is used in the selection of four of the 20 binary variables. The weights for POE and ACC are three and one respectively. Table 5.4.3 shows for each 2-class problem the measurements corresponding to the four binary variables chosen by the WS technique. The order in which the measurements are listed is the order in which the corresponding binary variables are selected. Table 5.4.4, compiled from Table 5.4.3, shows how often each measurement corresponds to a binary variable chosen by the WS technique and how often to a binary

variable ranked first. Rao [25] points out that "three important measurements from which cranial capacity (C) may be predicted are the glabella - occipital length (L), the maximum parietal breadth (B), and the basio - bregmatic height (H')." From Tables 5.4.3 and 5.4.5 we see that L, B, and H' are also the measurements the WS technique identifies as being most important.

Table 5.4.3. Measurements selected by the WS technique.

2-Class Problem Number	Name of Measurements Corresponding to the Binary Variables Chosen by the WS Technique			
1	H'	U	B	OH
2	H'	H'	U	U
3	H'	B	Q	U
4	B	H'	U	S
5	L	L	S_3	U

Table 5.4.4. Frequency with which each measurement is selected by the WS technique.

Measurement	L	B	B'	H'	OH	Q	S	S_3	S'_1	U
Number of times selected by the WS technique	2	3	0	5	1	1	1	1	0	6
Number of times ranked first by the WS technique	1	1	0	3	0	0	0	0	0	0
Selected by Rao	1	1	0	1	0	0	0	0	0	0

Table 5.4.5. Recognition rate for skulls of the distance-to-mean classifier.

2-Class Problem Number i	Number of Samples in the Training Set of Classes A_i and B_i	Recognition Rate on the Training Set		
		20 Binary Variables	4 Binary Variables	4 Real Variables L, B, H', ?
1	4/51	87%	--	--
2	14/41	82%	95%	--
3	29/26	96%	96%	93%
4	40/15	84%	91%	--
5	50/5	91%	--	--

The third column of Table 5.4.5 shows the recognition rate on the training set of the distance-to-mean classifier for the five 2-class problems when all 20 binary variables are used to represent a pattern. The fourth column shows the recognition rate on the training set of a problem when the patterns for that problem are represented using only the four binary variables chosen in conjunction with that problem--again the distance-to-mean classifier is used.

5.5 Testing the Estimates g_1 and g_2 Under Ideal Circumstances

As Δf and h converge to zero, g_1 and g_2 converge to the function f being estimated, but g_1 and g_2 cannot be expected to equal f for any particular values of their parameters. To get a notion of how large the differences between g_1 and f and between g_2 and f are likely to be in the best of

circumstances, we perform a number of simulations in which

$$u_i(X) = \int_{R_i} k(X, h)$$

can be evaluated accurately. Throughout the section f has domain and range $[0, 10]$, and

$$u_i(X) = \int_{R_i} \ell(x, h),$$

where

$$R_i = f^{-1}(I_i)$$

$$I_i = \begin{cases} [0, \frac{1}{2}] & \text{if } i = 1 \\ [i-\frac{3}{2}, i-\frac{1}{2}] & \text{if } i = 2, \ldots, 10 \\ [\frac{19}{2}, 10] & \text{if } i = 11. \end{cases}$$

ℓ is one of the functions

$$\ell_1(z) = \frac{1}{2\pi} e^{-z^2/2} \qquad (5.5.1)$$

$$\ell_2(z) = \begin{cases} \frac{3}{2}(-|z|+1)^2 & \text{if } -1 \leq z \leq 1 \\ 0 & \text{otherwise} \end{cases}$$

$$\ell_3(z) = \begin{cases} \frac{3}{4}(-|z|+1)^{1/2} & \text{if } -1 \leq z \leq 1 \\ 0 & \text{otherwise} \end{cases}$$

$$\ell_4(z) = \begin{cases} \frac{1}{2} & \text{if } -1 \le z \le 1 \\ 0 & \text{otherwise.} \end{cases}$$

The b_i's are

$$b_i = i-1, \quad i = 1, \ldots, 11.$$

The accuracy with which g_j, $j = 1, 2$, estimates f is measured by

$$ERR(f, g_j) = \left(\frac{\sum_{i=1}^{m} (f(x_i)-g_j(x_i))^2}{m} \right)^{1/2} \frac{100}{\Delta f}, \qquad (5.5.2)$$

where the x_i's are equally spaced points in $[0, 10]$. For all $x \in \bigcup_{i=1}^{N} R_i^0$, it follows from Theorem 3.10.1 that $\lim_{h \to 0} |f(x)-g_j(x)|$ is never greater than Δf and is close to zero when $f(x)$ is close to some b_i. Hence, as $h \to 0$, $ERR(f, g_j)$ is usually close to 50; and for some values of h we can hope that $ERR(f, g_j)$ is less than 50. If the factor $\frac{100}{\Delta f}$ is dropped from line (5.5.2), $ERR(f, g_j)$ is an approximation of $LSE(f, g_j, E, \rho)$.

Tables 5.5.1, 5.5.2, and 5.5.3 summarize the results of computer simulations which compute the value of $ERR(f, g_j)$. When h is close to zero, we see from Table 5.5.1 that $ERR(f, g_2)$ is close to 50; and for other values of h, $ERR(f, g_2)$ is considerably less than 50. The best value of h depends on the form of f.

Table 5.5.2 confirms the remark made in Section 3.4 that g_2 is a better estimate of f than g_1. For instance, when $f(x) = x^2/10$, the smallest value of $ERR(f, g_2)$ is 19% less than the smallest value of $ERR(f, g_1)$; and when $f(x) = \sqrt{10x}$, 30% less. Table 5.5.2 also confirms that at the best values of h, g_2 is less sensitive than g_1 to changes in the value of h. Table 5.5.3 shows that the choice of the kernel function is not critical, although the best value of h varies with the choice of the kernel function. Table 5.5.4 shows that the estimates are less accurate near the boundary of f's domain.

Table 5.5.1. $ERR(f, g_2)$ given ℓ_1 and test set $x = .2$ to 9.6 step $.32$.

	$ERR(f, g_2)$ when $f(x) =$			
h	x	$x^2/10$	$(x-5)^2/2.5$	$5 \sin(\frac{x\pi}{5}) + 5$
.01	54	52	56	50
.1	--	34	24	22
.3	--	18	16	12
.5	6	10	22	26
.7	--	8	30	44
1.0	16	12	48	42

Table 5.5.2. Comparison of $ERR(f, g_1)$ and $ERR(f, g_2)$ given ℓ_1 and test set x = .5 to 9.49 step .31.

h	$f(x) = x^2/10$		$f(x) = \sqrt{10x}$	
	$ERR(f, g_1)$	$ERR(f, g_2)$	$ERR(f, g_1)$	$ERR(f, g_2)$
.3	18.2	18.2	20.0	19.6
.4	14.0	13.8	14.6	13.0
.5	11.4	11.0	13.2	9.6
.6	10.4	9.2	14.8	9.2
.7	11.2	8.4	17.0	10.2
.8	13.2	8.4	19.4	11.4
.9	16.4	9.2	22.0	12.6
1.0	20.0	10.8	24.6	13.6

Table 5.5.3. Smallest values of $ERR(f, g_2)$ for different kernel functions given $f(x) = x^2/10$ and test set x = .5 to 9.49 step .31.

	ℓ_1	ℓ_2	ℓ_3	ℓ_4
$ERR(f, g_2)$	8.4	8.4	8.6	10.0
Corresponding value of h	.7	2.3	1.4	1.0

Table 5.5.4. Comparison of $f - g_1$ and $f - g_2$ near the boundary of the domain of f given ℓ_1, h = .7, $f(x) = x^2/10$.

x	f(x)	$f(x) - g_1(x)$	$f(x) - g_2(x)$
9.5	9.025	.144	.070
9.6	9.216	.197	.116
9.7	9.409	.261	.176
9.8	9.604	.336	.252
9.9	9.801	.423	.342
10.0	10.000	.522	.448

5.6 Two More Tests of the Estimate g_2

The simulations described in the previous section were performed under ideal conditions. Here we describe two simulations in which the function f is multi-variate; g_2 is the estimate; and

$$u_i(X) = \int_{-d_{i-1}(X)}^{-d_i(X)} \ell_1(0, h),$$

where ℓ_1 is defined on line (5.5.1) and the d_i's are derived from 2-class classifiers in the manner shown on line (4.2.1). In the first simulation

$$f_1(x, y) = (x-7.5)^2 + (y-7.5)^2 ,$$

and in the second f_2 is the function whose value is the cranial capacity of a skull and whose arguments are the 20 binary variables chosen by the WS technique (see the simulation using skulls described in Section 5.4).

In both simulations, five 2-class classifiers are used. In the first simulation the five 2-class problems are defined by lines (5.4.1) to (5.4.7), and the classifiers are the five TLU's referred to in the last column of Table 5.4.1--each TLU has a recognition rate of 100%. The training set for each classifier includes all the patterns in S (see line (5.4.1)), and $ERR(f, g_2)$ is computed on the set $\{X_{II}\}_{I=0}^{15}$.

The b_i's are

$$b_i = (i-1)(22.5), \quad i = 1, \ldots, 6;$$

so for the t_i's of line (5.4.5)

$$b_i = \begin{cases} t_i - \Delta f/2 & \text{if} \quad i = 1 \\ (t_{i-1}+t_i)/2 & \text{if} \quad i = 2, \ldots, 5 \\ t_5 + \Delta f/2 & \text{if} \quad i = 6. \end{cases} \tag{5.6.1}$$

In the second simulation, which uses the data on skulls, the five 2-class problems are defined by lines (5.4.8) to (5.4.10), and the classifiers are the distance-to-mean classifiers whose recognition rates are given in the third column of Table 5.4.5. As we explained a few lines after line (5.4.10), the training set for each 2-class problem contains data on the same 55 skulls. $ERR(f_2, g_2)$ is computed on data for the remaining 55 skulls. The b_i's are

$$b_i = 1121.0cc + 2(i-1)(1632.5cc-1121.0cc)/10, \quad i = 1, \ldots, 6;$$

so for the t_i's of line (5.4.8), the b_i's satisfy the equation on line (5.6.1).

Table 5.5.1 shows $ERR(f_1, g_2)$ and $ERR(f_2, g_2)$. It is interesting to compare the accuracy of g_2 in predicting cranial capacity with the accuracy of linear regression in predicting cranial capacity. Rao [25] notes that it is reasonable to expect that cranial

capacity C is a function of the measurements L, B, and H' and that for suitably chosen parameters α, β_1, β_2, and β_3

$$C = \alpha L^{\beta_1} B^{\beta_2} H'^{\beta_3} . \qquad (5.6.2)$$

If

$$y = \log_{10} C$$

$$x_1 = \log_{10} L$$

$$x_2 = \log_{10} B$$

$$x_3 = \log_{10} H' ,$$

then

$$y = \alpha + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 ,$$

and linear regression can be used to find values for α, β_1, β_2, and β_3. Using data from 86 of Hooke's [13] skulls, Rao obtained

$$\alpha = .00241$$

$$\beta_1 = .878$$

$$\beta_2 = 1.041$$

$$\beta_3 = .733 .$$

These values are used in the formula on line (5.6.2). The last column of Table 5.6.1 shows the root-mean-square error obtained when this formula is used to predict the capacity of 76 test skulls. (Twenty-one of the test skulls are also in the training set of 86 skulls.)

For the sake of comparison with $ERR(f_2, g_2)$, the root-mean-square error is multiplied by $\frac{100}{\Delta f_2}$.

Table 5.6.1. $ERR(f_1, g_2)$, $ERR(f_2, g_2)$, error for linear regression on skulls.

	$ERR(f_1, g_2)$	$ERR(f_2, g_2)$	Linear Regression on Skulls, (Root-Mean-Square Error) $* \frac{100}{\Delta f_2}$
Error	22	114	106
h	.5	5	-

VI. CONCLUSIONS

6.1 Networks

Every 2-class pattern recognition problem can be solved by a network of TLUs. If the patterns are linearly separable a single TLU will suffice; otherwise, a majority committee, containing perhaps as many members as there are patterns, can be found [16]. Networks of TLUs more complex than a committee offer the possibility of solving complex pattern recognition problems while using fewer TLUs than a committee.

The local adjustment algorithm for training a single TLU (see line (2.3.1)), which is convergent when the patterns are linearly separable, has been extended to algorithms for training committees of TLUs. Unfortunately, the existence of a solution does not insure convergence, but none the less the algorithms have proven to be useful in solving a variety of problems (see [21] and Section 5.3). The attempts described in Sections 2.4 and 5.1 to extend the algorithm for training a single TLU to an algorithm for training a network were unsuccessful. Even when the structure of the network was restricted to that of a committee whose logic was allowed to change, the algorithm failed.

Other investigators have had some success in training networks but they have been forced to introduce severe restrictions. For

instance, Ridgway [26] fixes the number of members and the logic of a committee before training begins. Francalangia [8] adjusts a network at a single level. Holderman [11] allows the number of TLUs in a network to change during training and adjusts all of the TLUs in the network, but he fixes the order in which TLUs can be adjusted and uses a global criterion to determine the adjustments. Networks may indeed be "monsters of vacuous generality" [18] for which excellent training algorithms may never be found.

6.2 Modified Veto Committees

The modified veto committee with its local adjustment algorithm for training is a new and useful addition to the theory of committees of TLUs. A modified veto committee has the capacity to solve any 2-class problem in which the classes are disjoint and the variables are binary, and the algorithm for training a modified veto committee seems to converge to a solution in the cases tested. It is necessary to say "seems to converge" as in two of the simulations summarized in Table 5.3.1 training was stopped before a solution was reached. In contrast, a majority committee is capable of solving any 2-class problem in which the classes are disjoint, regardless of whether or not the variables are binary, provided there are only a finite number of patterns.

When the variables are binary, Ridgway's [26] and Mueller's [23] local adjustment algorithms for training a majority committee do not seem to perform as well as the local adjustment algorithm for training a modified veto committee (Table 5.3.1). It could be that for the problems used a five member modified veto committee has more power than a five member majority committee, or it could be that the algorithm for adjusting a modified veto committee is more powerful than the algorithms of adjusting a majority committee. Because of the nature of modified veto logic, the algorithm for adjusting a modified veto committee can add new members as needed during training; and when training ends, members most recently added can be dropped from the committee at the cost of a reasonable reduction in the recognition rate. Adding a member to or dropping a member from a majority committee has a major effect on the logic of the committee, and there is unlikely to be an algorithm for training a majority committee that allows the number of members to change during training. If at the end of training it is decided a majority committee would perform better with a different number of members, then training must start again from scratch.

The algorithm for training a modified veto committee requires the normalization of weight vectors after each adjustment. The heuristic reasons given for this normalization in Section 2.9 are supported by the results of the simulation in Section 5.3 (see the

discussion at the end of Section 5.2).

As we discussed in Section 2.8 and illustrated in Figure 2.8.1, prejudice is to be expected when a local adjustment algorithm is used, especially when the classifier is complex. The hope expressed in Section 2.8 that only the last members added to a modified veto committee learn prejudice is confirmed by the simulations on CON2 and CODE2; for dropping the last members of the committees decreased the recognition rate on the training set much more than on the test set (Tables 5.3.1 and 5.3.5). When the training set is large enough to be representative of the test set, dropping members affects the recognition on the training and test set about equally (see CON3 and CODE3 in Tables 5.3.1 and 5.3.6).

The simulations summarized in Tables 5.3.7 and 5.3.8 show that the algorithm for adjusting a modified veto committee is sensitive to changes in the parameters a_0, β, and γ, and to the changes in the number of constant terms in the representation of a pattern. This sensitivity is unfortunate in that one has to search for a suitable set of parameters and useful in that it enables the algorithm to obtain a high recognition rate for different problems. Fortunately, we had no difficulty finding suitable values for the parameters, first, because it is possible to predict the effect of changing a parameter, and second, because the algorithm performs well for a wide range of values of the parameters.

6.3 Estimation

The extension of the use of 2-class classifiers to problems of function estimation appears to be a new contribution to the theory of pattern recognition. The simulations described in Sections 5.5 and 5.6 show that g_1 and g_2 are reasonably accurate estimates for a variety of functions. The estimates g_1 and g_2 can be used regardless of the number of independent variables and the form of the function being estimated. The greatest difficulty encountered in using g_1 and g_2 is in computing the discriminant functions d_i. When the 2-class classifiers are TLUs the discriminant functions can be defined by line (4.2.1); and the number of problems that can be handled satisfactorily by a TLU is extended by using binary variables. Future research could investigate ways of defining discriminant functions in terms of other classifiers.

We saw in Chapter III that the estimates g_1 and g_2 have several nice properties. They are continuous on the product space formed by their parameters and by X. Under reasonable conditions, they converge to the function being estimated. When $u_i(x)$ is given by

$$u_i(X) = \int_{-d_{i-1}(X)}^{-d_i(X)} \ell(0, h),$$

then g_1 and g_2 are stable with respect to changes in the d_i's.

The simulations described in Section 5.5. indicate limits on the accuracy of g_1 and g_2. The simulations show that: g_2 is more accurate than g_1; g_2 is less sensitive than g_1 at the best values of h to changes in the value of h; the choice of the kernel function is not critical; and near the boundary of the domain of the function being estimated, g_2 and g_1 are less accurate than elsewhere.

The simulations in Section 5.6 are a more interesting test of the accuracy of the estimate g_2 than are the simulations in Section 5.5, because in Section 5.6 the u_i's are defined in terms of the d_i's, and the functions are multivariate. The estimation performed on the function

$$f_1(x, y) = (x-7.5)^2 + (y-7.5)^2$$

shows the performance of g_2 when the function being estimated is "smooth" and the training data is accurate. The estimation performed to predict cranial capacities shows the performance of g_2 when the function being estimated may not be "smooth" in the measurements given. The effect, if any, of inaccuracies in the data is undetermined.

In predicting cranial capacity, g_2 is nearly as accurate as linear regression. As we can see from line (5.6.2), to use linear regression we must be able to make a good guess about the form of the function we are estimating and then be sufficiently ingenious to translate this form to a linear form. We wonder: Would the accuarcy of

linear regression have been improved if the equation on line (5.6.2) were changed? Would the accuracy of g_2 have been improved if more 2-class classifiers were used?

In retrospect we see from Table 5.4.5 that g_2 might have estimated cranial capacities more accurately if each 2-class problem had been solved using only the four variables chosen for that problem by the WS technique. If this had been done, each pattern would still have been described by 20 binary variables, but each discriminant function would have been evaluated in terms of just four binary variables. As a variable may contain useful information about patterns that map to one part of a function's range and misleading information about patterns that map to another part, it seems worthwhile to use different variables for the different 2-class problems. Using different variables for the different 2-class problems has the additional advantage of shortening the computation of the d_i's.

Further simulations testing the effectiveness of g_2 and the idea of using different variables for each 2-class problem are planned.

6.4 Binary Variables

In the first two simulations in Section 5.4, replacing multi-leveled variables by binary variables permitted a distance-to-mean classifier and a TLU to have a high recognition rate on problems they otherwise could not handle. The third simulation in Section 5.4, on

skulls, confirms Schoenfelt's [30] remarks that using binary variables can increase the recognition rate for a problem while reducing storage requirements, (see Table 5.4.5). The third simulation also shows that the WS technique is an effective way to choose a "good" subset of binary variables from a larger set as evidenced by the recognition rates in Table 5.4.5). Moreover, it seems worthwhile to note that the variables selected most often and ranked highest by the WS technique are those considered most important by Rao [25] (see Table 5.4.4).

A modified veto committee performs well when binary variables are used; and replacing multi-leveled variables by binary variables extends the range of application of the estimates g_1 and g_2. Further research could be done to determine the beneficial effect, if any, on the performance of other classifiers when binary variables are used.

Exhibit 1. Training records of a modified veto committee
Part (a) $a_0 = 6$, $\beta = 1$, $\gamma = 0$ and weight vectors not normalized.

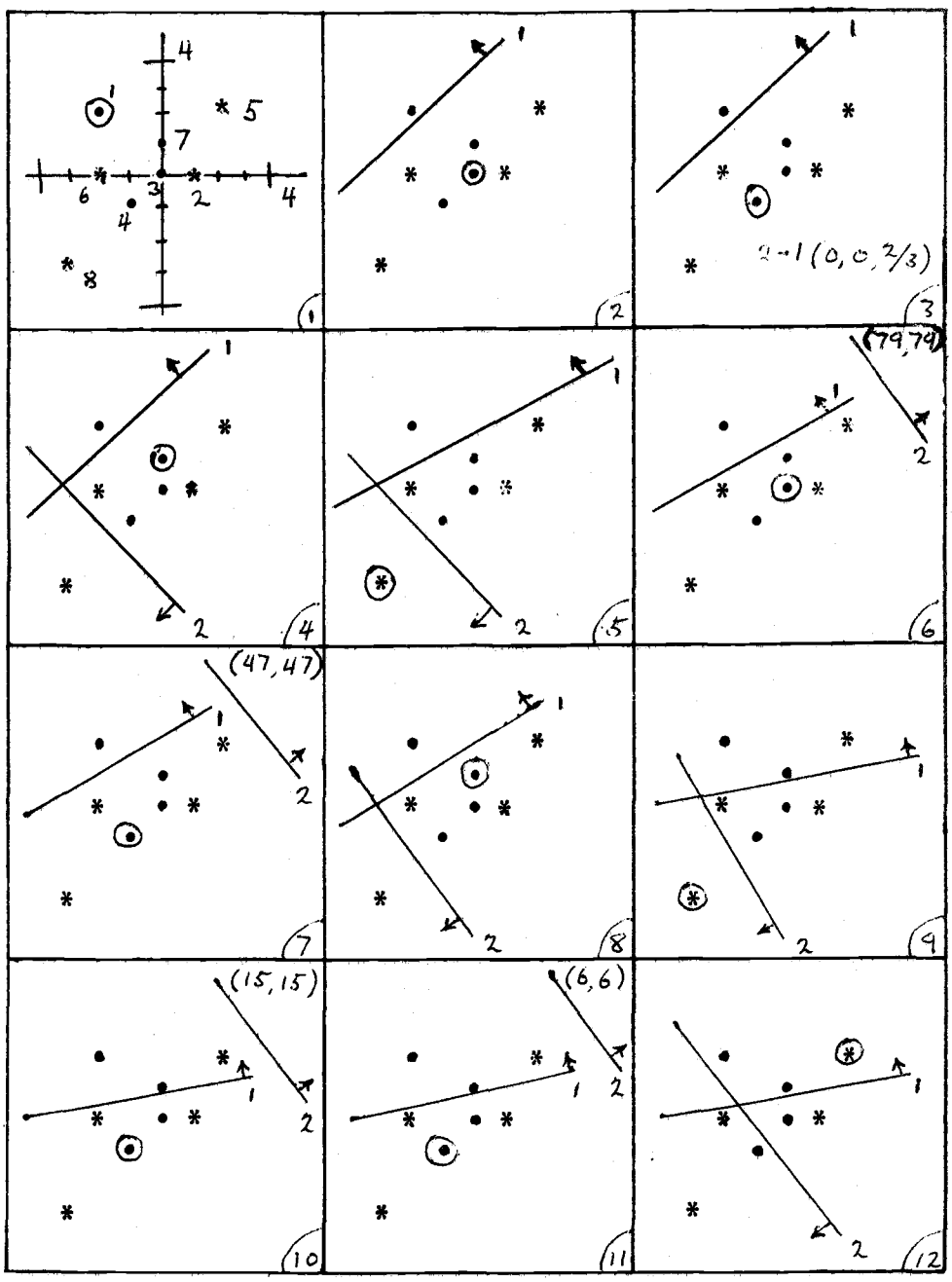


Exhibit 1. Part (a) continued.

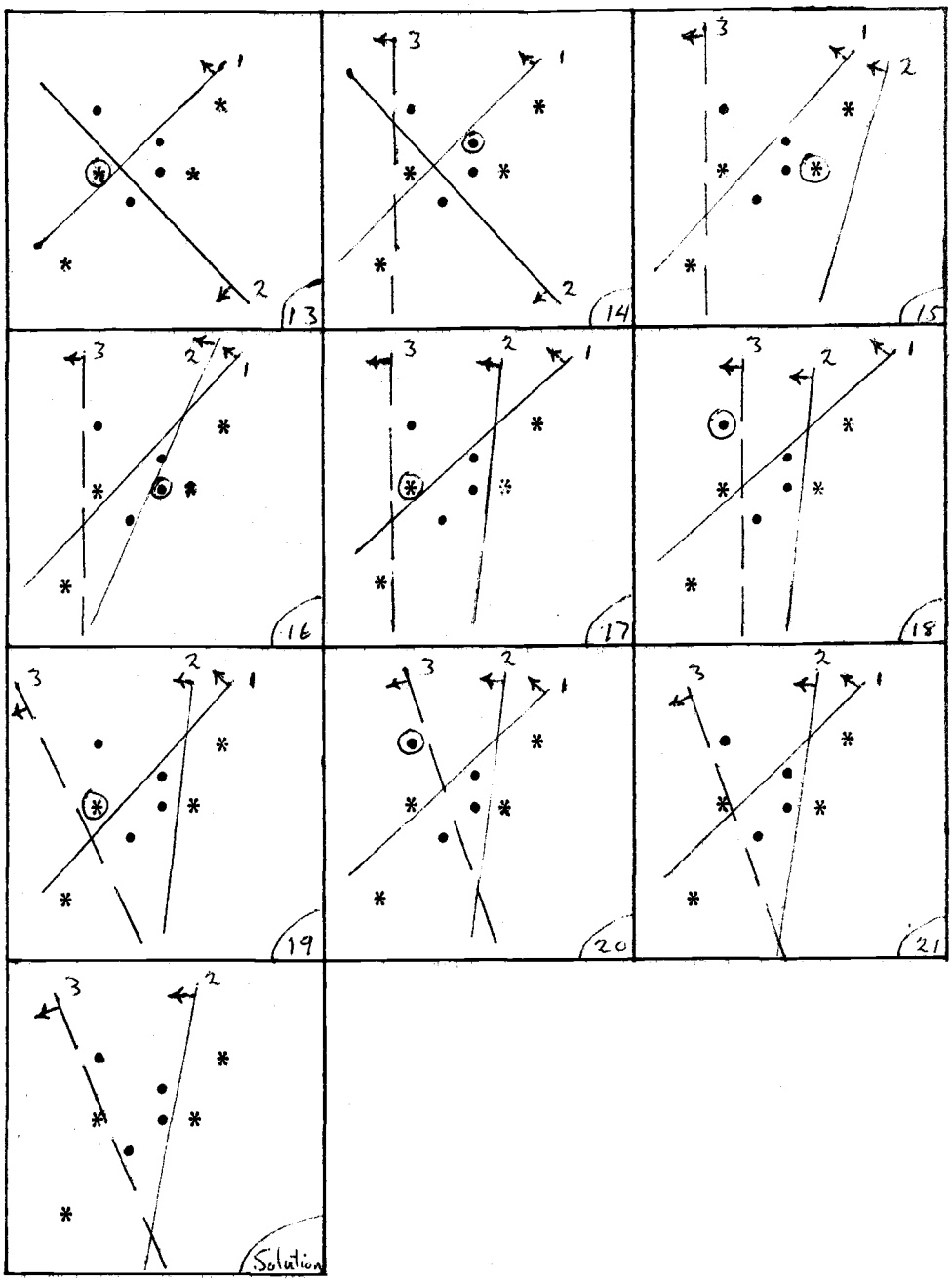


Exhibit 1. Part (b) $\alpha_0 = 20$, $\beta = 1$, $\gamma = 0$ and weight vectors not normalized.

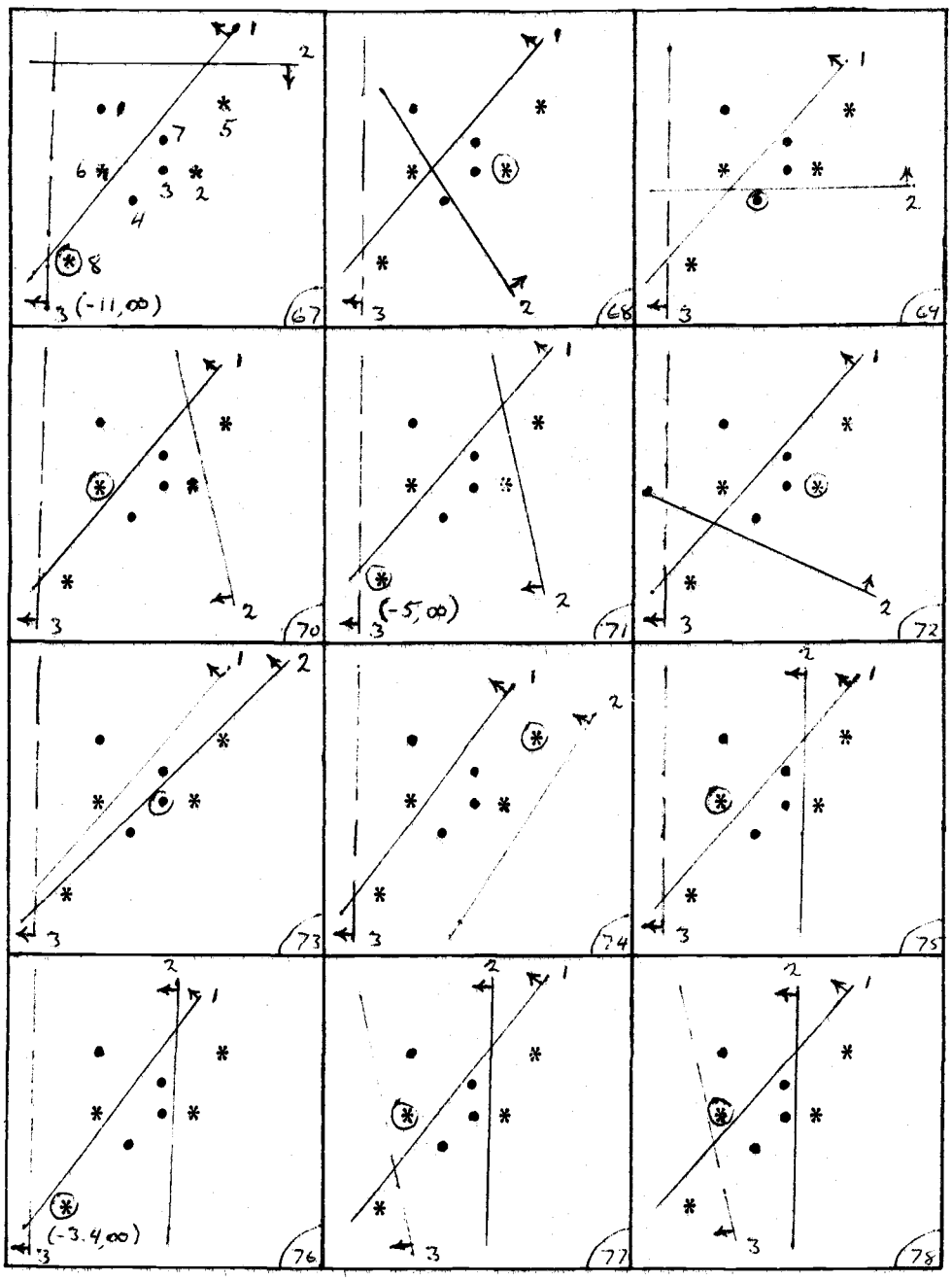


Exhibit 1. Part (b) continued.

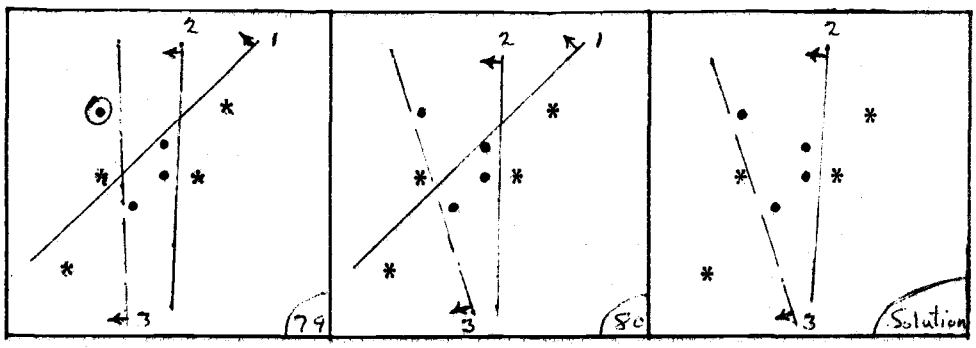


Exhibit 1. Part (c) $\alpha_0 = 20$, $\beta = 1$, $\gamma = 0$ and weight vectors not normalized.

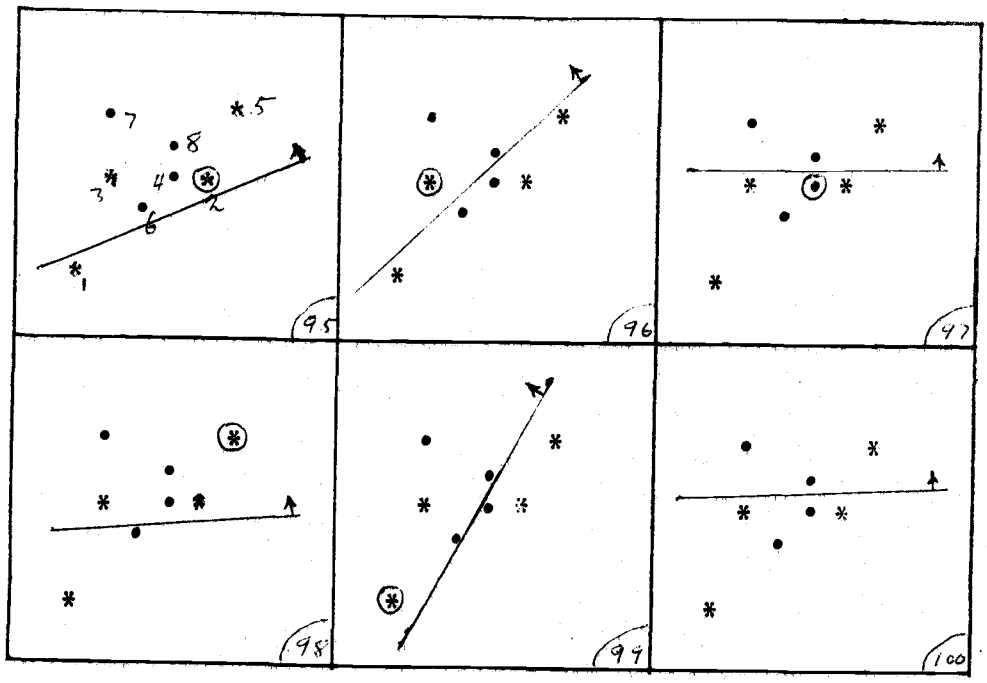


Exhibit 1. Part (d) $\alpha_0 = 6$, $\beta = 1$, $\gamma = 0$ and weight vectors normalized.

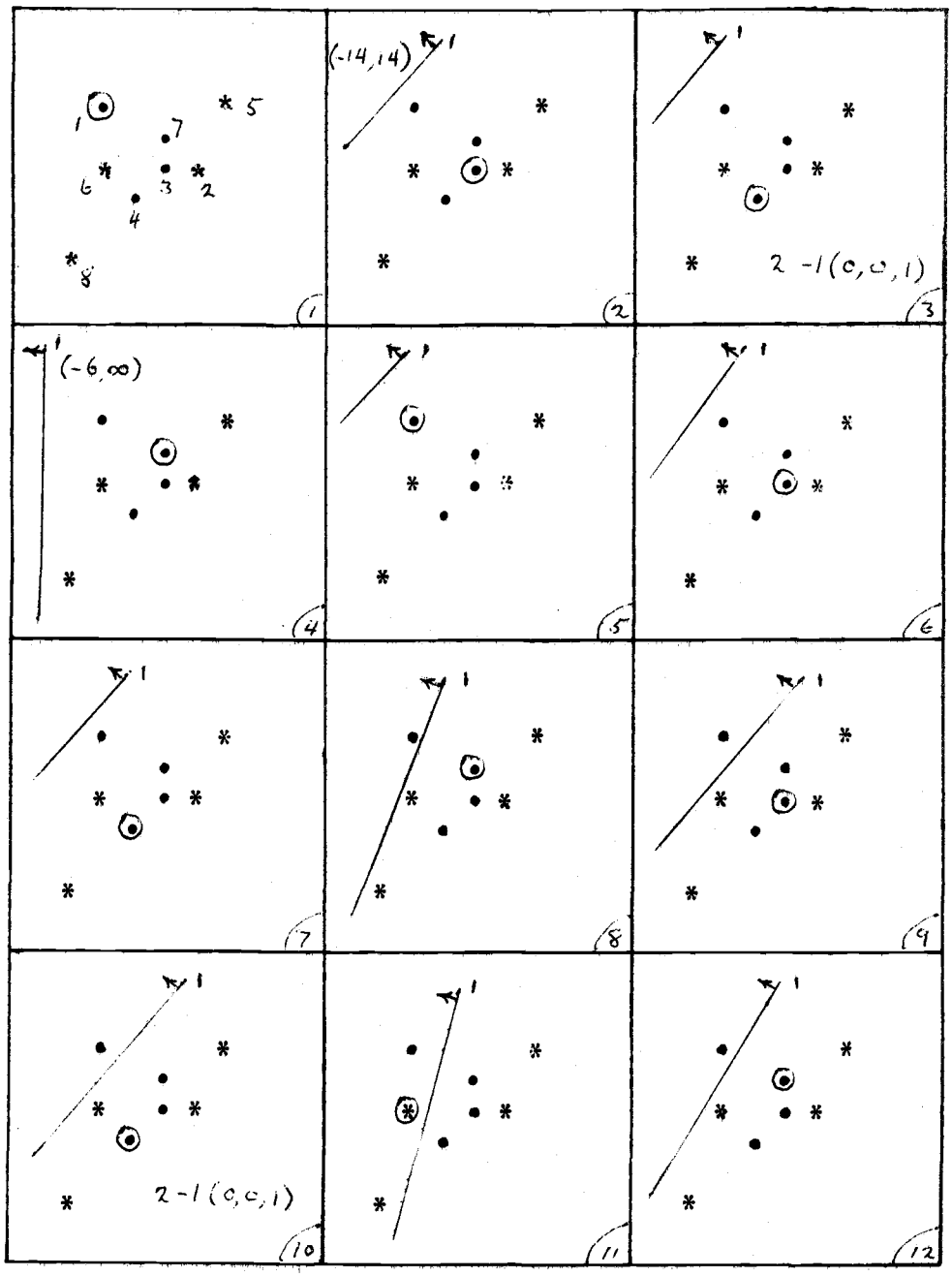


Exhibit 1. Part (d) continued.

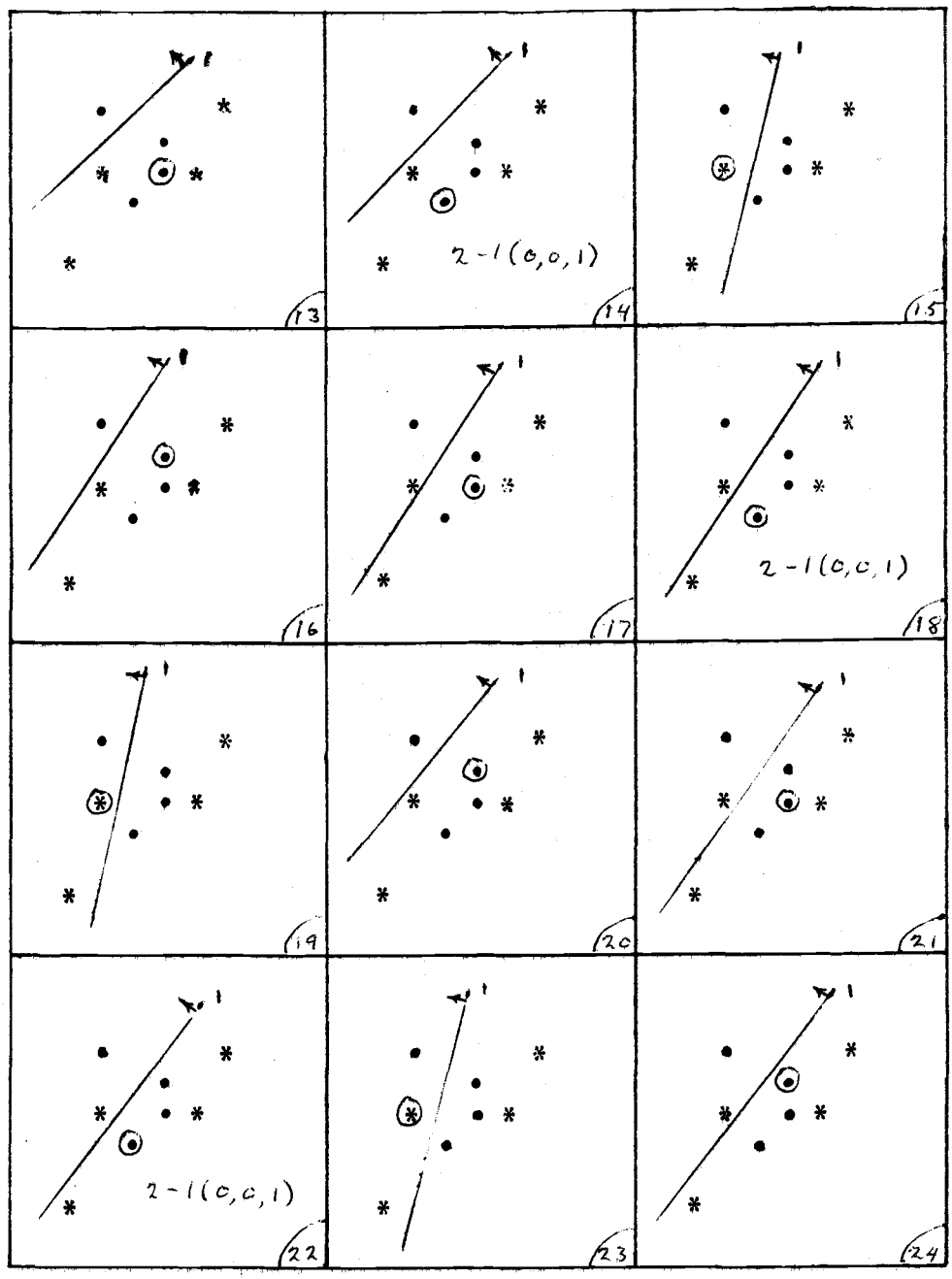


Exhibit 1. Part (d) continued.

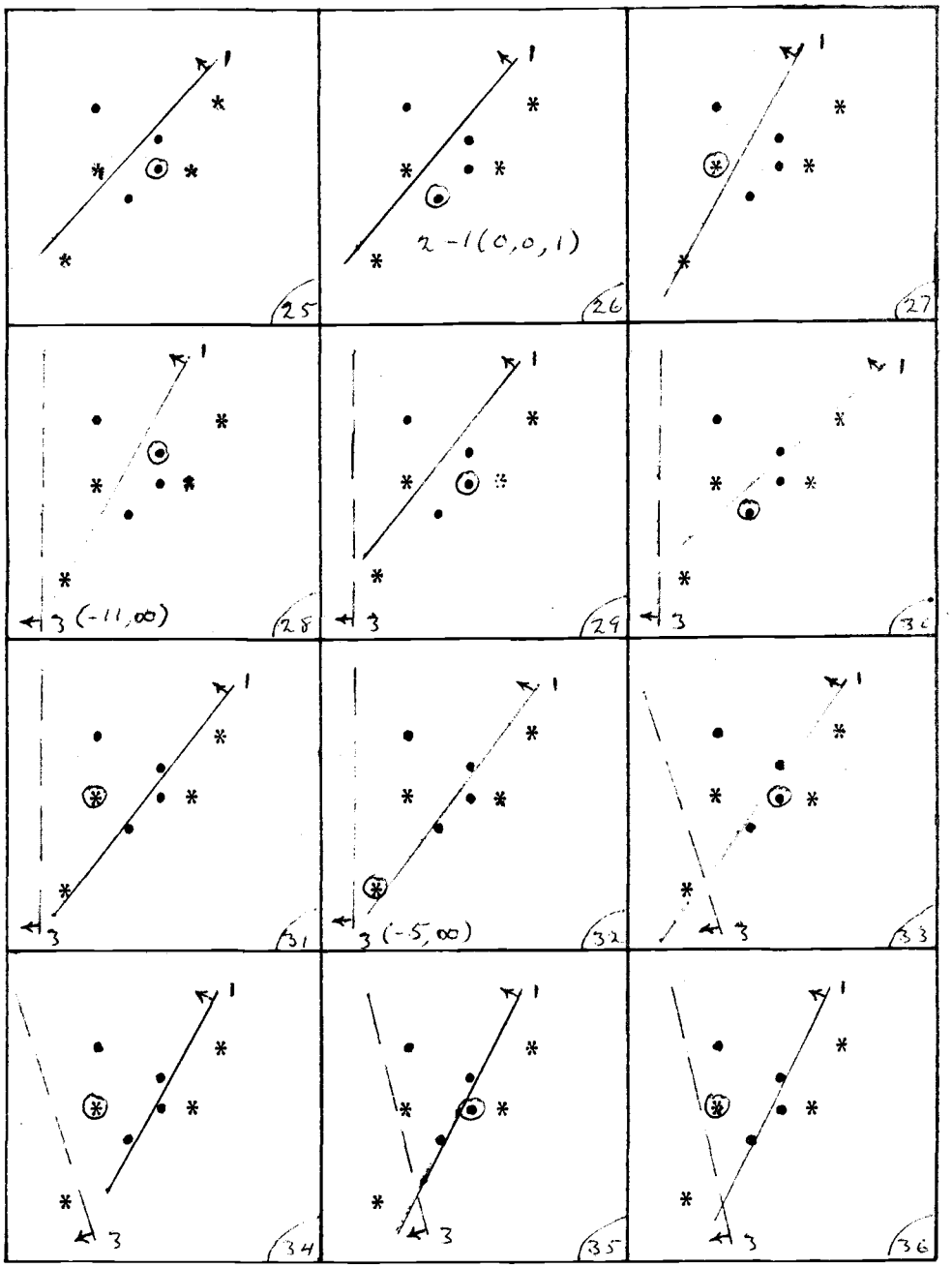
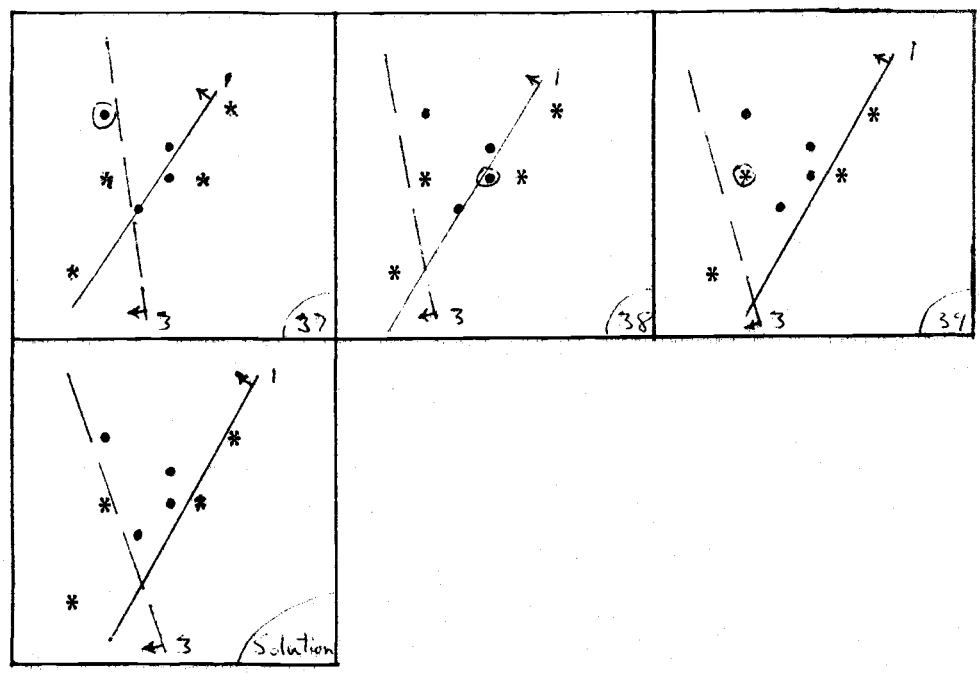


Exhibit 1. Part (d) continued.



BIBLIOGRAPHY

1. Ablow, C.M. and Kaylor, D.J. "A Committee Solution of the Pattern Recognition Problem." IEEE Trans. on Information Theory, Vol. IT-11, No. 3. July, 1965. pp. 453-455.

2. Agmon, Samuel, "The Relaxation Method for Linear Inequalities." Canadian Journal of Mathematics IV. 1954. pp. 382-392.

3. Block, H.E. "The Perceptron: A Model for Brain Functioning." Reviews of Modern Physics, 34. 1962. pp. 123-135.

4. Bongard, N. Pattern Recognition. Spartan Books, Washington, D.C. 1970.

5. Brain, A.E., et al. "A Large Self-Contained Learning Machine." 1963 WESCON Paper 6.1. August, 1963.

6. Chang, Chin-Liang. "Pattern Recognition by Piecewise Linear Discriminant Functions." IEEE Trans. Comp., Vol. C-22, No. 9. September, 1973. pp. 859-862.

7. Duda, R.O. and Fossum, H. "Pattern Classification by Iteratively Determined Linear and Piecewise Linear Discriminant Functions." IEEE Trans. Electronic Computers, Vol. EC-15, No. 2, April, 1966. pp. 220-232.

8. Duda, R.O. and Singleton, R.C. "Training a Threshold Logic Unit with Imperfectly Classified Patterns." 1964 WESCON Paper 3.1. Los Angeles, California. August, 1964.

9. Francalangia, J.L. A General Class of Layered, Trainable, Threshold Logic Networks for Pattern Classification. Ph.D. Thesis, University of Washington, 1969.

10. Fukunaga, K. Introduction to Statistical Pattern Recognition. Academic Press, New York. 1972.

11. Ho, Y.C. and Kashyap, R.L. "A Class of Iterative Procedures for Linear Inequalities." J. SIAM Contra. 4. 1966. pp. 112-115. (Chapter 4).

12. Holdermann, F. "Classification by Cascade Threshold Elements." Pattern Recognition, Vol. 3, 1971. pp. 243-251.

13. Hooke, B.G.E. "A Third Study of the English Skull with Special Reference to the Farrington Street Crania." Biometrika, 18, 1926, pp. 1-55.

14. Ibaraki, T. and Muroga, S. "Adaptive Linear Classifiers by Linear Programming." IEEE Trans. Syst. Sci. Cybern., Vol. SSC-6. January, 1970. pp. 53-62.

15. Kaminuma and Watanabe. "Fast-converging Adaptive Algorithms for Well-balanced Separating Linear Classifiers." Pattern Recognition, October, 1972. pp. 289-305.

16. Kaylor, D.J. A Mathematical Model of a Two-layered Network of Threshold Logic Elements. Rome Air Development Center Technical Documentary Report RADC-TDR-63-534. March, 1964.

17. Koford, J. and Grover, G. "The Use of Adaptive Threshold Elements to Design a Linear Optimal Classifier." IEEE Trans. on Information Theory, IT-12, 1966. pp. 42-50.

18. Larson, L.E., et al. "On the Problem of Bias in Error Rate Estimation for Discriminant Analysis." Pattern Recognition, Vol. 3, October, 1971. pp. 217-223.

19. Meisel, W. Computer-oriented Approaches to Pattern Recognition. Academic Press, New York, 1972.

20. Minsky and Papert. The Perceptron--An Introduction to Computational Geometry. MIT Press, Cambridge, Mass. 1969.

21. Motzkin, T.S. and Schoenberg, I.J. "The Relaxation Method for Linear Inequalities." Canadian Journal of Mathematics VI. 1954. pp. 393-404.

22. Mucciardi, A.N. and Gose, E.E. "A Comparison of Seven Techniques for Choosing Subsets of Pattern Recognition Properties." IEEE Trans. Comp., Vol. C-20, No. 9, September, 1971. pp. 1023-1031.

23. Mueller, T. Teemal, an Adaptive Training Procedure for a Two Layer System of Linear Threshold Elements. Ph.D. Thesis, Oregon State University, 1973.

24. Nilsson, N. Learning Machines: Foundations of Trainable Pattern - Classifying Systems. McGraw-Hill, New York. 1965.

25. Rao, C.R. Advanced Statistical Methods in Biometric Research. Hafner, New York. 1970.

26. Ridgway, W.C. An Adaptive Logic System with Generalized Properties. Stanford Electronic Laboratories Technical Report 1556-1, Stanford Univ., April, 1962.

27. Rosenblatt, F. Principles of Neurodynamics. Spartan Books, Washington, D.C. 1961.

28. Sammon, J.W. Jr. "An Optimal Discriminant Plane." IEEE Trans. Comp. Vol. C-19, Sept. 1970. pp. 826-829.

29. _____. "Interactive Pattern Analysis and Classification." IEEE Trans. Comp. Vol. C-19. July, 1970. pp. 594-616.

30. Shoenfelt, J.E., et al. "Techniques for Efficient Encoding of Features in Pattern Recognition." IEEE Trans. Comp., Vol. C-20, No. 9, September, 1971. pp. 1104-1106.

31. Warmack, R.E. and Gonzalez, R.C. "An Algorithm for Optimal Solution of Linear Inequalities and its Application to Pattern Recognition." IEEE Trans. Comp. Vol. C-22. December, 1973. pp. 1065-1075.