BOOTSTRAP LEARNING IN THRESHOLD LOGIC SYSTEMS

Bernard Widrow*

Abstract

A new adaptive process, called "bootstrap learning," has been developed for use with linear threshold logic elements. Such elements are generally trained by supplying input-pattern signals and the corresponding desired binary responses (decisions). Often, however, the desired responses to specific input patterns are not known. In this situation, bootstrap adaptation can be used to train the element by deriving desired responses from actual responses evaluated in the light of the performance quality resulting from a series of such actual responses.

One example of the application of bootstrap adaptation is found in the playing of "Blackjack" or "21." After playing several hundred computer-simulated games without being given any information other than whether it has won or lost each game, a single threshold element can adapt its parameters in such a way as to play the game with close to an optimal strategy. An analytical expression has been derived for the time of convergence, and has been checked by experiment.

Bootstrap adaptation is slower than conventional adaptation, since adaptation with respect to each individual pattern is not always effected in the correct direction. However, bootstrap adaptation can be used where conventional methods are not applicable. It is possible that its use will allow convergent adaptation of threshold logic networks of complex configurations, both parallel and multilayered. Applications to the development of nonlinear dynamic control system switching surfaces are being studied.

*Department of Electrical Engineering, Stanford University, Stanford, California

"BOOTSTRAP LEARNING" IN THRESHOLD LOGIC SYSTEMS

Bernard Widrow

Introduction

The purpose of this paper is to describe research on a new kind of adaptive process involving linear threshold logic elements. By means of this new process, called "bootstrap adaptation," an adaptive logic element learns what is required of it solely through the receipt of favorable or unfavorable reactions resulting from the application of an overall performance criterion to the outcome of a series of decisions made by the element.

Adaptive linear threshold logic elements ("Adalines") and other forms of adaptive systems have been under study at Stanford University[1,2,3,4] for the past several years. Training (weight-adjusting) algorithms have been developed for threshold logic elements and networks of such elements[2,3,5,6]. Useful results have been obtained in many areas, including the design of devices for the physical realization of adaptive circuits[4,7], and applications of adaptive pattern-recognition techniques to speech recognition, weather forecasting, diagnosis of heart conditions by electrocardiogram, and relay contactor ("bang-bang") control systems[8,9].

Until recently, threshold elements in the above applications have been used only as trainable pattern-classifying systems. When these elements are trained, for each array of input signals (input pattern), a desired response (representing the pattern class) is specified. This kind of process may be called "learning with a teacher". The present paper is concerned with the analysis of adaptive processes wherein the desired responses cannot be supplied for each input pattern. Applications for such adaptive processes, called "learning without a teacher", may arise in certain sequential-decision processes, in pattern generation processes (the inverse of pattern classification), and possibly in convergent adaptation procedures for multilayered and more generally-connected networks of adaptive threshold elements.

Conventional Adaptation Processes for Threshold Elements

In order to understand the process of learning without a teacher, it is convenient to begin by summarizing the process whereby an adaptive threshold element learns with a teacher. Figure 1 shows a functional diagram and schematic symbol for an adaptive threshold logic element. The diagram indicates the terminology used and the input-output relationships. The zeroth input signal is always +1. Thus the zeroth weight w_0 controls the threshold partitioning level. Before adaptation, an analog error ϵ (defined as the difference between the analog sum s and the desired binary response d) exists for each input pattern. The j^{th} input pattern would have an analog error of

[Fig. 1

$$\epsilon(j) \triangleq d(j) - s(j) = d(j) - [\vec{X}(j) \cdot \vec{W}] \ . \qquad (1)$$

A least-mean-square error (LMS) adaptation procedure developed by Widrow and Hoff[2,3] causes the weights to relax automatically toward a set that minimizes the function $\sum_{j=1}^{N} \epsilon^2(j)$ when training is performed on a finite group of N training patterns; or toward a set that minimizes the mean-square-error function $\overline{\epsilon^2}$ for a continuous nonrepeated flow of input training patterns derived from a large statistical population. On the i^{th} adaptation cycle, the change in the weight vector in adapting on the j^{th} pattern is made to be

$$\Delta\vec{W}_i = \left(\frac{\lambda}{n+1}\right) \epsilon(j)\ \vec{X}(j) \quad , \tag{2}$$

where $n+1$ is the total number of weights (including w_0), and λ is a coefficient which determines the fraction of the analog error that is corrected with each adaptation. Thus λ controls the rate of adaptation.

After adaptation, the new, $(i+1)^{st}$ value of the weight vector is

$$\vec{W}_{i+1} = \vec{W}_i + \Delta\vec{W}_i \quad . \tag{3}$$

It should be particularly noted that during the LMS training process, adaptation is always performed upon the presentation of a new pattern and its desired response, even when the quantized output response, the decision q, agrees with the desired response. It has been shown[1,2,3] that when binary input signals are mutually uncorrelated and are also uncorrelated with d, the weight values undergo noisy geometric (exponential-like) transients in relaxing toward optimal values. These transients are such that the "time constant" of adaptation Γ is

$$\Gamma = \frac{(n+1)}{\lambda} \text{ adaptation cycles} \quad . \tag{4}$$

A "learning-curve" plot of mean-square-error, MSE, versus the number of adaptation cycles for the uncorrelated input-signal case is also a noisy exponential. The MSE in this case can be shown to be equal to a constant plus a component which is proportional to the square of the magnitude of the difference between the weight vector and the LMS-optimal weight vector. The magnitude of this weight difference relaxes with time constant Γ. The square of this magnitude relaxes with time constant $\Gamma/2$. Therefore the time constant of the MSE learning curve is given by

$$\Gamma_{mse} = \frac{(n+1)}{2\lambda} \text{ adaptations .} \tag{5}$$

Bootstrap Adaptation

It is a straightforward process to present an adaptive threshold element with an input pattern, and to adapt it toward producing the desired response, as described in the preceding section. The question is what to do when an adaptive element is connected to an environment providing a stream of input patterns, but the desired response for each input pattern is not known and/or not supplied to the adaptive element.

One possibility would be to connect the binary output of the threshold element to the desired-output input terminal, as shown in Figure 2(a). Fig. 2
Under this plan, the adaptive element assumes when a new pattern is applied that its own binary output is the correct desired output. It adapts its weights accordingly, applying the LMS algorithm, or some other algorithm[5,6], which has been designed into its adaptation machinery, Figure 1(a). The tendency here is to maintain the binary responses that already exist (i.e.,

responses established by the initial weight settings), although some pattern responses (s-values) close to the threshold may reverse during this process. In a sense, the adaptive element has the attitude "don't bother me with the facts, my mind is made up". Let this procedure be called positive bootstrap adaptation.

An alternative means of supplying the desired response from the output signal is shown in Figure 2(b). Here the output signal goes through an inverter which forms its complement. The inverted output is then taken as the desired output. Let this form of adaptation be called negative bootstrap adaptation. Now, whenever a new input pattern is applied, adaptation takes place to change the analog output $s(j)$ a certain amount in the direction which moves it closer to a binary output (+1 or −1) which is itself opposite to the actual binary output $q(j)$. A sustained application of negative bootstrap adaptation will eventually cause all weight values to approach zero, which will neutralize the effects of initial weight conditions. A threshold element adapting this way would have the attitude, "everything I do is wrong."

Neither positive nor negative bootstrap adaptation is as useful in itself as is the combination of these methods illustrated in Figure 3. Fig. 3
In this configuration, two kinds of input information are again required to produce an adaptation: the input pattern $\vec{X}(j)$ and a bootstrap control signal $b(j)$. When $b(j)$ is positive (switch up in Figure 3), positive bootstrapping takes place; when $b(j)$ is negative (switch down) negative bootstrapping is performed. Let this process be called selective bootstrap adaptation.

The kind of information supplied as $b(j)$ in Figure 3 will be quite different in practice from that supplied as $d(j)$ in Figure 1. The $b(j)$ signals are more qualitative than specific. If some external evaluator indicates that the present decision or chain of decisions (whatever these decisions actually may be) appear to produce satisfactory performance in accordance with a pre-arranged criterion, a positive signal is applied to $b(j)$; otherwise a negative signal is applied. Selective bootstrap adaptation may be thought of as learning with a <u>critic</u>, as opposed to learning with a teacher. The critic is qualitative. The teacher is specific.

Application of Bootstrap Adaptation to Simulated Blackjack Play

In order to make the idea of selective bootstrap adaptation clearer and to stimulate ideas for its application, an example will be presented relating to the playing of the game "blackjack" or "21". It has been found that using selective bootstrap adaptation, a single threshold element is able to learn to play this game very well without knowing the rules or the objectives of the game[10]. All that is needed is the knowledge, at the end of each game, of whether the game was won or lost.

In the game of blackjack or 21, the objective of the player is to draw a series of cards from the dealer such that the values of these cards sum to less than 21, yet come closer to 21 than the sum of the cards drawn by the dealer. Whoever goes over 21 loses. When a card is offered to the player, he has the choice of drawing or not drawing ("hit" or "stick"). He must make a binary decision. The dealer has no choice. He performs a purely mechanical function, playing a fixed house strategy. Thus it was possible to simulate the dealer by means of a computer which "dealt" using a random-number generator. The computer did all score keeping, and periodically typed out the performance of the "player", an adaptive threshold element (Adaline), which it also simulated. The game of blackjack was simplified by removing all special features such as "splitting pairs," "doubling down," etc. Blackjacks were counted.

Figure 4 shows how the simulated Adaline was able to represent the player in the blackjack game. The first card drawn by the dealer is face up, so the decision made by the player is based on the dealer's card showing, and the sum of the face values of the cards in the player's hand. These data, together with an indication of how the ace is to be counted, constituted the variable inputs to the Adaline. The variables were encoded as shown in Figure 5. Notice that the different states of a variable are represented by binary code words which are algebraically linearly independent.

[Fig. 4

[Fig. 5

The adaptive "player" began making decisions with a given set of initial weights. During each game, several "hit" or "stick" decisions were automatically made. For each state of the game (input pattern), the decision made by the "player" was recorded by the computer. At the end of the game, the computer noted whether the "player" won or lost. If the "player" won, either by luck or by good strategy, the $b(j)$ control switch shown in Figure 4 was placed in the up position and the same patterns were repeated. Thus adaptation was effected in directions which assumed the actual decision responses were the desired responses. If the Adaline lost, its performance was considered unsatisfactory (poorer than average), and the $b(j)$

switch was placed in the down position, thus adapting the element in directions which assumed the actual decision responses were the opposite of the desired responses. Adaptation was performed only at the end of each game, and then the patterns from that game were discarded. Only the weight values required long-term storage.

It has been demonstrated by Fred W. Smith[10] that a single fixed-weight threshold element can realize the optimal "basic blackjack strategy" of Thorpe[11]. When the value of the dealer's face card is encoded in a linearly independent binary code[8,9], and when the sum of the cards dealt to the player is also encoded in this way, the binary patterns representing the states of the game, together with the associated binary decisions (hit or stick) corresponding to the Thorpe optimal strategy, constitute a linearly separable set. Yet they represent a nonlinear discriminant function. Thus through the encoding procedure, such a function is made perfectly realizable by a single threshold logic element. The strategy needed to play the game of blackjack is related to that required for the "bang-bang" (contactor) control of a variety of dynamic systems[8,9]. In both cases, binary decisions must be made based on the values of several analog or multilevel state variables.

The optimal strategy for the simplified game described earlier, assuming the player has no knowledge of cards previously played (or equivalently, where the dealer shuffles before each game), is presented in Figure 6. It should be noted that even when playing with the optimal strategy for this simplified game, the player will lose at a certain small average rate. The adaptive player must learn to minimize its losses. Fig. 6

The learning process described above used selective bootstrap adaptation (positive bootstrap for a win, negative bootstrap for a loss), and has several unique features. Learning was not directed by a teacher along each step of the way. The effects of individual decisions could not be uniquely evaluated. They all had a statistical effect on a final outcome based on a composite of the quality of the responses to a series of patterns (states) which were for all practical purposes selected at random. In addition, games were sometimes won when playing with poor strategies and games were sometimes lost when playing with excellent strategies. The net result is that <u>not always did adaptation on a given pattern proceed in the proper direction</u>. Consequently, the selective bootstrap learning process takes place at a slower rate than conventional learning with a teacher. The purpose of the analysis to follow is to predict the rate of learning of the bootstrap process.

<u>Analytical Example: Modeling a Noisy Unknown System</u>

In order to make an analytical study of the performance and rate of learning possible when selective bootstrap adaptation is practiced, the modeling of an unknown noisy memoryless system having many binary inputs and a single binary output by means of an adaptive threshold element will be examined.

The modeling configuration is shown in Figure 7. The input-signal pattern vectors are supplied at discrete times; all systems are considered to be sampled-data systems. It will be assumed that the input patterns are selected in random sequence. The difference between the adaptive-model output and the actual output of the noisy unknown system (i.e., the "right" output) is a form of error signal $\epsilon_1(j)$. When the cost associated with $\epsilon_1(j)$ averaged [Fig. 7

over the present error sample and past (D-1) error samples is lower than the long-term cost average, positive bootstrap adaptation is effected with regard to the present input pattern vector, using the switch $b(j)$ to control the desired-output input. When performance averaged over the D error samples is poorer than the long-term average, negative bootstrap adaptation is indicated. The cost function could be of a very general nature, but for purposes of this study, it will be assumed that all errors $\epsilon_1(j)$ are equally costly. I.e., the "best" performance is taken to be that which produces the minimum number of errors. The long-term performance average should be taken over a number of samples many times greater than D, and should be continually up-dated to track improvements in performance resulting from the adaptation process.

In order to analyze the effects of the bootstrapping operation described above, it is postulated that there exists a statistically optimal model which is also indicated in Figure 7. This model is deterministic, since a given input pattern will produce a given output. It produces an output called "optimal" in the sense that an error $\epsilon_2(j)$, defined as the difference between its output and that of the unknown noisy system to be modeled, is of minimum expected cost. This optimal model does not exist physically and is drawn with dotted lines to denote this. It is postulated for analytical purposes only. When its output agrees with the output of the adaptive element, the latter output is said to be an <u>optimal</u> output. When the output of the adaptive element agrees with the output of the unknown noisy system being modeled, the adaptive-element output is said to be a <u>right</u> output.

The distinction between right and optimal decisions can perhaps be visualized more clearly in the context of statistical prediction of stationary time series. An optimal predictor always makes optimal decisions (by definition) which are not always perfect (i.e., not always right).

Analysis of Selective Bootstrap Adaptation Applied to Modeling an Unknown Noisy System

In a group of D decisions made by the adaptive threshold-element model of Fig. 7, it is likely that some will be optimal and right (O-R), some will be optimal and wrong (O-W), while some will be antioptimal and right (A-R), and some will be antioptimal and wrong (A-W). These four are the only possibilities. Arrayed in a group, these kinds of decisions might occur as follows:

$$\overbrace{(O\text{-}R), (A\text{-}W), (A\text{-}W), (O\text{-}R), (O\text{-}W), (A\text{-}R), (O\text{-}R), \ldots \ldots, (A\text{-}W)}^{D \text{ of these}}$$

Let the probability of (O-R) be p_1, the probability of (O-W) be p_2, the probability of (A-R) be p_3, and the probability of (A-W) be p_4. A sketch of the joint probability density for a single decision as a function of the number of right and the number of optimal decisions is shown in Fig. 8a. This function is [Fig. 8

$$P_1(g,h) = p_1\delta(1-g,1-h) + p_2\delta(1-g,1+h) + p_3\delta(1+h,1-g) + p_4(1+h,1+g) \ , \qquad (6)$$

where h is the axis of right-wrong decisions and g is the axis of optimal-antioptimal decisions. Note that a unit two-dimensional delta function is defined to have a unit volume.

The joint probability density $P_D(g,h)$, a function of the number of right and the number of optimal decisions in a chain of D decisions, is sketched in Figure 8b. The value of the g parameter is the sum of the number of optimal decisions minus the number of antioptimal decisions; the value of the h parameter is the sum of the number of right decisions, minus the number of wrong decisions. Assume that the decisions in the sequence are statistically unrelated (independent).[Assumption 1]. It then follows that the joint probability-density function for a chain of D decisions is a D-fold convolution of the density function for a single decision.

$$P_D(g,h) = \underbrace{P_1(g,h) * P_1(g,h) * \ldots * P_1(g,h)}_{D \text{ of these}} , \tag{7}$$

The independence assumption follows from the assumption that input patterns occur in random sequence. In many situations, the independence assumption will either be precise or at least reasonable.

In order to derive an expression for the time constant of the bootstrap learning process to converge toward an optimal solution, it is necessary to derive an expression for the probability p of adapting in the optimal direction. The probability of adapting in the antioptimal direction is $q = (1-p)$. <u>If the bootstrap adaptation process is to be useful, it is important that the critical parameter $(p-q)$ be greater than zero</u>. To calculate p, a certain kind of moment will have to be evaluated for the discrete joint probability density $P_D(g,h)$. In order to simplify this moment calculation, it will be assumed that D is sufficiently large so that $P_D(g,h)$ could be replaced for purposes of moment calculation by a two-dimensional gaussian density function.[Assumption 2]. The justification for this is the Central Limit Theorem. The parameters of a gaussian approximation function $\hat{P}_D(g,h)$ will be chosen to have the same mean values as $P_D(g,h)$, the same variances, and the same correlation coefficient.

The first step is to find the means, the variances, and the covariance of the simple density function $P_1(g,h)$ sketched in Figure 8a. The means are

$$\bar{g} = p_1 + p_2 - p_3 - p_4 \ . \tag{8}$$

$$\bar{h} = p_1 + p_3 - p_2 - p_4 \ . \tag{9}$$

The variance along the g-axis is

$$\sigma_g^{\ 2} \triangleq \overline{g^2} - (\bar{g})^2 = p_1 + p_2 + p_3 + p_4 - (\bar{g})^2$$

$$= 4(p_1+p_2)(p_3+p_4) \ . \tag{10}$$

The variance along the h-axis is

$$\sigma_h^{\ 2} \triangleq 4(p_1+p_3)(p_2+p_4) \ . \tag{11}$$

The covariance is

$$\sigma_{gh}^{\ 2} \triangleq \overline{gh} - (\bar{g})(\bar{h})$$

$$= p_1 + p_4 - p_2 - p_3 - (\bar{g})(\bar{h})$$

$$= 4p_1p_4 - 4p_2p_3 \ . \tag{12}$$

The correlation coefficient is

$$\rho \triangleq \frac{\sigma_{gh}^{2}}{\left(\sigma_g^{2} + \sigma_h^{2}\right)^{\frac{1}{2}}} = \frac{p_1 p_4 - p_2 p_3}{\sqrt{(p_1+p_2)(p_3+p_4)(p_1+p_3)(p_2+p_4)}} \tag{13}$$

These parameters can now be easily calculated for the probability density $P_D(g,h)$. The means of this density function are

$$D\,\bar{g} \quad \text{and} \quad D\,\bar{h} \tag{14}$$

where D is the number of decisions in the chain. The variances are

$$D\sigma_g^{2} \quad \text{and} \quad D\sigma_h^{2} \quad . \tag{15}$$

The correlation coefficient is the same as in equation 13.

The estimating density $\hat{P}_D(g,h)$ will have parameters as determined by expressions 13, 14, and 15. Accordingly, this bivariate normal density can be written as

$$\hat{P}_D(g,h) = \frac{1}{2\pi D\sigma_g\sigma_h\sqrt{1-p^{2}}} \exp \frac{-1}{2(1-p^{2})}\left[\frac{(g-\bar{g}D)^{2}}{D\sigma_g^{2}} - \frac{2\rho(g-\bar{g}D)(h-\bar{h}D)}{D\sigma_g\,\sigma_h} + \frac{(h-\bar{h}D)^{2}}{D\sigma_h^{2}}\right] \tag{16}$$

A plan-view sketch of $\hat{P}_D(g,h)$ contours is shown in Figure 9. [Fig. 9

According to the previously stated rules of adaptation, positive bootstrapping will be effected when measured performance is better than average, i.e., when the number of right decisions in the chain of D

decisions exceeds the long-term average number of right decisions. It is implicitly assumed that on the average, each decision in a chain of D decisions has equal expected effect upon measured performance. [Assumption 3].

Events where positive bootstrap adaptation takes place $(h > D\bar{h})$ are therefore indicated by the shaded area in Figure 9. The unshaded area represents all other events, where negative bootstrap adaptation takes place $(h < D\bar{h})$.

Consider all chains of events where performance is better than average. Let the probability of such chains be represented by $P(h > D\bar{h})$. The probability of chains with below-average performance is $P(h < D\bar{h}) = 1-P(h > D\bar{h})$. It follows that once the joint density $P_D(g,h)$ is represented by the gaussian density $\hat{P}_D(g,h)$,

$$P(h > D\bar{h}) = \int_{-\infty}^{\infty} \int_{D\bar{h}}^{\infty} \hat{P}_D(g,h) \; dg \; dh$$

$$= P(h < D\bar{h}) = \tfrac{1}{2} \tag{17}$$

Consider only chains with above-average performance. Among these chains, all of which will experience positive bootstrap adaptation, the expected number of optimal decisions minus the expected number of antioptimal decisions is given by

$$E[g|h > D\bar{h}] = \frac{1}{P(h > D\bar{h})} \int_{-\infty}^{\infty} g \; dg \int_{D\bar{h}}^{\infty} \hat{P}_D(g,h) \; dh \tag{18}$$

For the chains with below-average performance, all of which will experience negative bootstrapping when adapted, the expected number of antioptimal decisions minus the expected number of optimal decisions is

$$E[-g\,|h < D\bar{h}] = \frac{1}{P(h < D\bar{h})} \int_{-\infty}^{\infty} -g \; dg \int_{-\infty}^{D\bar{h}} \hat{P}_D(g,h) \; dh \quad , \qquad (19)$$

With positive bootstrapping $(h > D\bar{h})$, adaptation in the optimal direction takes place for patterns producing optimal threshold-element decisions; the expected number of optimal adaptations minus the expected number of antioptimal adaptations is given by expression 18. With negative bootstrapping $(h < D\bar{h})$, adaptation in the optimal direction takes place for patterns producing antioptimal threshold-element decisions; the expected number of optimal adaptations minus the expected number of antioptimal adaptations is accordingly given by expression 19. The average (over all h) number of optimal adaptations minus the average number of antioptimal adaptations is

$$\begin{aligned}
(p-q)D &= E[g\,|h > D\bar{h}]\; P(h > D\bar{h}) + E[-g\,|h < D\bar{h}]\; P(h < D\bar{h}) \\
&= \int_{-\infty}^{\infty} g \; dg \int_{D\bar{h}}^{\infty} \hat{P}_D(g,h) \; dh - \int_{-\infty}^{\infty} g \; dg \int_{-\infty}^{D\bar{h}} \hat{P}_D(g,h) \; dh \\
&= \int_{-\infty}^{\infty} g \; dg \left\{ \int_{D\bar{h}}^{\infty} \hat{P}_D(g,h) \; dh - \int_{-\infty}^{D\bar{h}} \hat{P}_D(g,h) \; dh \right\} . \qquad (20)
\end{aligned}$$

After some algebraic manipulation, the following simple expression for (p-q) results:

$$(p-q) = \frac{2\rho\sigma_g}{\sqrt{2\pi D}} \qquad (21)$$

The expressions for σ_g and ρ, equations 10 and 13, may be substituted in equation 21 to give

$$(p-q) = \frac{4}{\sqrt{2\pi D}} \frac{(p_1p_4 - p_2p_3)}{(p_1+p_3)(p_2+p_4)} \qquad (22)$$

Equation 22 is a precise relation for $(p-q)$, based on Assumptions 1,2, and 3.

The next step is to find the probabilities p_1, p_2, p_3, and p_4. They can be related to physical processes by using the following expressions:

$$p_1 = P(O,R) = P(R|O) \quad P(O) \qquad (23)$$

$$p_2 = P(O,W) = P(W|O) \quad P(O) \qquad (24)$$

$$p_3 = P(A,R) = P(R|A) \quad P(A) \qquad (25)$$

$$p_4 = P(A,W) = P(W|A) \quad P(A) \qquad (26)$$

Let the probability of error of the optimal model (Fig. 7) be designated as P_0. Thus, the minimum achievable error probability is P_0.

The decisions of the adaptive model will in general not always agree with the optimal decisions--i.e., those that would be made by the optimal model if it existed. It will be assumed, however, that in responding to the input patterns where the two models do agree, the probability of these decisions being wrong is the same as the probability of all optimal decisions being wrong, [Assumption 4]. Accordingly, the probability of an optimal decision made by the adaptive element being wrong is

$$P(W|O) = P_0 . \tag{27}$$

The probability of an optimal decision made by the adaptive element being right is therefore

$$P(R|O) = (1-P_0) . \tag{28}$$

Assume that the probability of an antioptimal decision being a right one is the same for input patterns where the adaptive-system outputs are antioptimal, as for all possible input patterns. [Assumption 5]. Completely antioptimal decisions would result from the inversion or complementation of the output signals of the optimal system. Accordingly, the probability of a decision being right, given that the decision is antioptimal, is

$$P(R|A) = P(W|O) = P_0 . \tag{29}$$

Also,

$$P(W|A) = P(R|O) = (1-P_0) . \tag{30}$$

Let a dimensionless quantity M' be defined to be the difference between the error probability $P(W)$ of the threshold element and P_0, normalized with respect to P_0.

$$M' \triangleq \frac{P(W) - P_0}{P_0} \tag{31}$$

$$P(W) = P_0(1+M') = P(A)(1-2P_0) + P_0 \quad , \tag{32}$$

Adding equations 24 and 26 and making use of equations 27, 30, and 32:

$$P(A) = 1-P(0) = \frac{P_0 M'}{(1-2P_0)} \quad . \tag{33}$$

The probabilities p_1, p_2, p_3, and p_4 may now be found by substituting equations 27,28,29,30, and 33 into equations 23,24,25, and 26. The quantity $(p-q)$ may then be found by substituting the expressions for p_1, p_2, p_3, and p_4 into equation 22. The result is

$$(p-q) = \frac{(4M'\sqrt{P_0})[1-P_0(4+M')+P_0(4+2M')]}{\sqrt{2\pi D}\,(1-2P_0)\sqrt{1+M'-P_0(5+6M'+M'^2) + P_0^2(8+12M'+4M'^2)-P_0^3(2+2M')^2}} \quad . \tag{34}$$

Equation 34 follows precisely from equation 22, based on the additional assumptions 4 and 5 already mentioned.

Equation 34 simplifies greatly in the important practical case of small P_0 (approximately 0.1 or less) and small M' (approximately 0.5 or less):

$$(p-q) \cong \frac{4M'\sqrt{P_0}}{\sqrt{2\pi D}} \quad . \tag{35}$$

Effects of $(p-q)$ Upon Rate of Adaptation

With reference to the modeling situation of Fig. 7, the analysis just presented indicates that the adaptive threshold system will self-adapt toward forming a best LMS fit to the optimal model of the noisy unknown process as long as $(p-q) > 0$. To see why this is so, consider for example a situation wherein $(p-q) = 0.2$. On the average, in 10 adaptations on a given pattern, 6 will be in the optimal direction and 4 will be in the antioptimal direction. The net result is a preponderance of 2 adaptations out of 10 in the optimal direction. If all the magnitudes of the weight increments among the adaptation cycles were equal (they are not equal for the LMS procedure), the rate of learning in this case would be two tenths as great as when learning directly with a teacher. In general, time constants of the learning curve would be the corresponding values for learning with a teacher, multiplied by the factor $1/(p-q)$. It has been found by experiment that use of this factor allows one to make reasonably close estimates of learning-curve time constants for bootstrap learning with LMS adaptation and with other adaptation processes that do not use exactly equal weight-increment magnitudes.[5,6] The validity of the use of this factor will be assumed. [Assumption 6.]

Application to Blackjack Situation

The functions indicated in Fig. 7 might be considered analogous to the blackjack situation in the following way: The statistically optimal model could be assumed to generate decisions corresponding to the optimal strategy of Thorpe. The noisy unknown memoryless system might be viewed as a fictitious system that generates winning decisions based on perfect knowledge of past, present, and future cards to be drawn. It generates the "right" decisions, from the standpoint of winning each game as it is played; while the optimal strategy (having only probabilistic knowledge of how the cards are to be drawn) generates only <u>statistically</u> "optimal" decisions. The threshold element outputs represent the actual decisions of the adptive Adaline player, which are sometimes right, sometimes wrong, sometimes optimal, and sometimes antioptimal.

The cost evaluation for blackjack is especially simple. When the game is won, performance is obviously better than long-term average. When the game is lost, performance is poorer than long-term average. The chain of D decisions whose performance is to be averaged by the "win or lose" criterion represents the number of decisions per game. The average D is about 4 decisions per game.

The minimum error probability P_0 of the optimal system may be estimated in the following manner: The Thorpe optimal strategy for the simplified game wins 49.5 percent of the games. In the majority of games played, three right decisions are made first. The fourth and last decision is the critical one, and this decision is right roughly half the time (corresponding to the winning games). Therefore P_0 is assumed to be 1/8. [Assumption 7.]

The quantity M' defined by equation 31 and appearing in equation 35 is a normalized measure of the departure in performance of the adaptive system from that of the optimal system. This quantity can be estimated for blackjack by subtracting the minimum rate of loss of the optimal system (50.5 percent) from the rate of loss of the adaptive system and dividing this difference by the minimum rate of loss.

The quantity $(p-q)$ may be estimated by inserting $P_0 = 1/8$ and $D = 4$ into formula 35.

$$(p-q) \cong \frac{M'}{2\sqrt{\pi}} \tag{36}$$

By Assumption 6 and formula 5, the time constant of the LMS process with bootstrap adaptation is

$$\Gamma_{\substack{mse\\bootstrap}} = \frac{(n+1)}{2\lambda(p-q)} = \frac{(n+1)\sqrt{\pi}}{\lambda M'} \tag{37}$$

Since the total number of weights of the Adaline player is $(n+1) = 21$ and since there are 4 adaptations per game on the average,

$$\Gamma_{\substack{mse\\bootstrap}} \cong \frac{37.2}{\lambda M'} \text{ adaptations}$$
$$= \frac{9.3}{\lambda M'} \text{ games} \tag{38}$$

Formula 5 is based on the asumption that the individual input signals to the Adaline player are uncorrelated. Formula 38 is therefore based on the assumption in addition to Assumptions 1 through 7.

Experimental and Theoretical Results

A series of computer-simulated experiments was made to check the assumptions made in deriving equations 37 and 38. Experimental learning curves for two different rates of adaptation are shown in Figures 10 and 11 with $\lambda = 0.4$ and $\lambda = 0.04$. These plots present functions which are related to error probability rather than to mean-square error. However, it is pointed out in references 2, 3, 4 that under conditions applicable here, the error probability is approximately proportional to the mean-square error, and that the time constant of a mean-square-error learning curve is approximately equal to that of an error-probability learning curve.

[Fig. 10]

[Fig. 11]

Formula 38 therefore indicates that the learning curves of Figures 10 and 11 should be variable-time-constant exponentials. The more closely the average performance approaches the optimal, the smaller is M' and the larger is the time constant. These plots appear to have such properties. Each point in Figures 10 and 11 represents a time and ensemble average over 10,000 games. The averaging is necessary and provides a performance evaluation with a standard deviation of error in mean of approximately 1/2 percent.

Measurements of time constants can be made directly from Figure 10 corresponding to $\lambda = 0.4$. Initially, the percentage of games won is 22 percent, and therefore $M' = 0.56$. From formula 38, the theoretical time constant is 41.5 games. Taking the initial slope from Figure 10 gives an estimated time constant of approximately 50 games. As this learning curve crosses the level of 35 percent won, $M' = 0.3$ and the theoretical time constant is 77.5 games. By experiment, this turns out to be approximately 100 games. The asymptotic level of this learning curve, used in estimating the time constant, is not as high as the Thorpe optimal level. Space does not permit an examination of the value of this asymptote. It can be stated, however, that the larger λ, the faster is the adaptation process and the lower is the asymptotic performance level.

According to formula 38, the learning curve with $\lambda = 0.04$, Figure 11, should be ten times slower than the curve with $\lambda = 0.4$. By experiment, this turns out to be almost precisely so.

When one considers the number of assumptions that were made in deriving the simple formula 37, one might expect to predict only the order of magnitude of the learning-curve time constant. A number of experiments have shown, however, that this formula is surprisingly accurate.

Current and Future Research

Preliminary studies have been made with some success toward the development of adaptation algorithms for multilayered networks of adaptive threshold elements using the selective bootstrap principle. If performance observed at a set of output terminals is "better than average," every element in the net receives positive bootstrap adaptation. If output-terminal performance is poorer than average, then all elements receive negative bootstrap adaptation. It is conjectured that formula 37 will be usable in predicting the rate of adaptation in such networks. Instead of decisions being made in a chain over time, here they are made simultaneously, in a chain over space. The ultimate objective of this research is to develop efficient adaptation algorithms for adaptive threshold-element networks of arbitrary configuration, which are capable of realizing decision functions which are not linearly separable.

Acknowledgments

The author wishes to thank Professor James A. McFadden of Purdue University for assistance in the evaluation of expression 20 and for several suggestions that were helpful in the presentation of the statistical analyses. Dr. Fred W. Smith suggested the use of the game blackjack for the testing of bootstrap adaptation principles, and Dr. James S. Koford wrote the first computer simulation program. The experiments were performed by Mr. Irwin Sobel and Mrs. Betty Earlougher.

Bert

References

1. Widrow, B. Automatic and remote control. Proceedings of the First International Congress of the International Federation of Automatic Control, Moscow (1960). London; Butterworths

2. Widrow, B. and Hoff, M.E. Adaptive switching circuits. Wescon Conv. Rec., Inst. Radio Engrs, N.Y. Pt. 4 (1960)

3. Widrow, B. and Hoff, M.E. Adaptive switching circuits. Tech. Rep. 1553-1, Stanford Electronics Laboratories, Stanford, Calif., June 1960

4. Widrow, B. Generalization and information storage in networks of Adaline 'neurons.' Self-Organizing Systems 1962. Washington, D.C.; Spartan Books

5. Ridgway, W.C. An adaptive logic system with generalizing properties. Rep. SEL-62-040, Stanford Electronics Laboratories, Stanford, Calif., April 1962.

6. Mays, C.H. Adaptive threshold logic. Rep. SEL-63-027, Stanford Electronics Laboratories, Stanford, Calif.,April 1963

7. Crafts, H.S. Components that learn, and how to use them. Electronics, March 22, 1963

8. Widrow, B. and Smith, F.W. Pattern-recognizing control systems. 1963 Computer and Information Sciences (COINS) Symposium Proceedings. 1964. Washington, D.C.; Spartan Books

9. Smith, F.W. Contactor control by adaptive pattern-recognition techniques. Rep. SEL-64-042, Stanford Electronics Laboratories, Stanford, Calif., April 1964

10. Smith, F.W. Private communication to the author

11. Thorpe, E.O. Beat the Dealer: A Winning Strategy for the Game of Twenty-One. 1962. New York; Blaisdell

Bart

List of Figure Captions

Figure 1. An automatically-adapted threshold logic element (Adaline).

Figure 2. Positive and negative bootstrap adaptation.

Figure 3. Selective bootstrap adaptation.

Figure 4. Selective bootstrap adaptation applied to game of blackjack.

Figure 5. Blackjack game states encoded as patterns for input to threshold logic element "player" shown in Figure 4.

Figure 6. Optimal blackjack strategy (when dealer resuffles for each game).

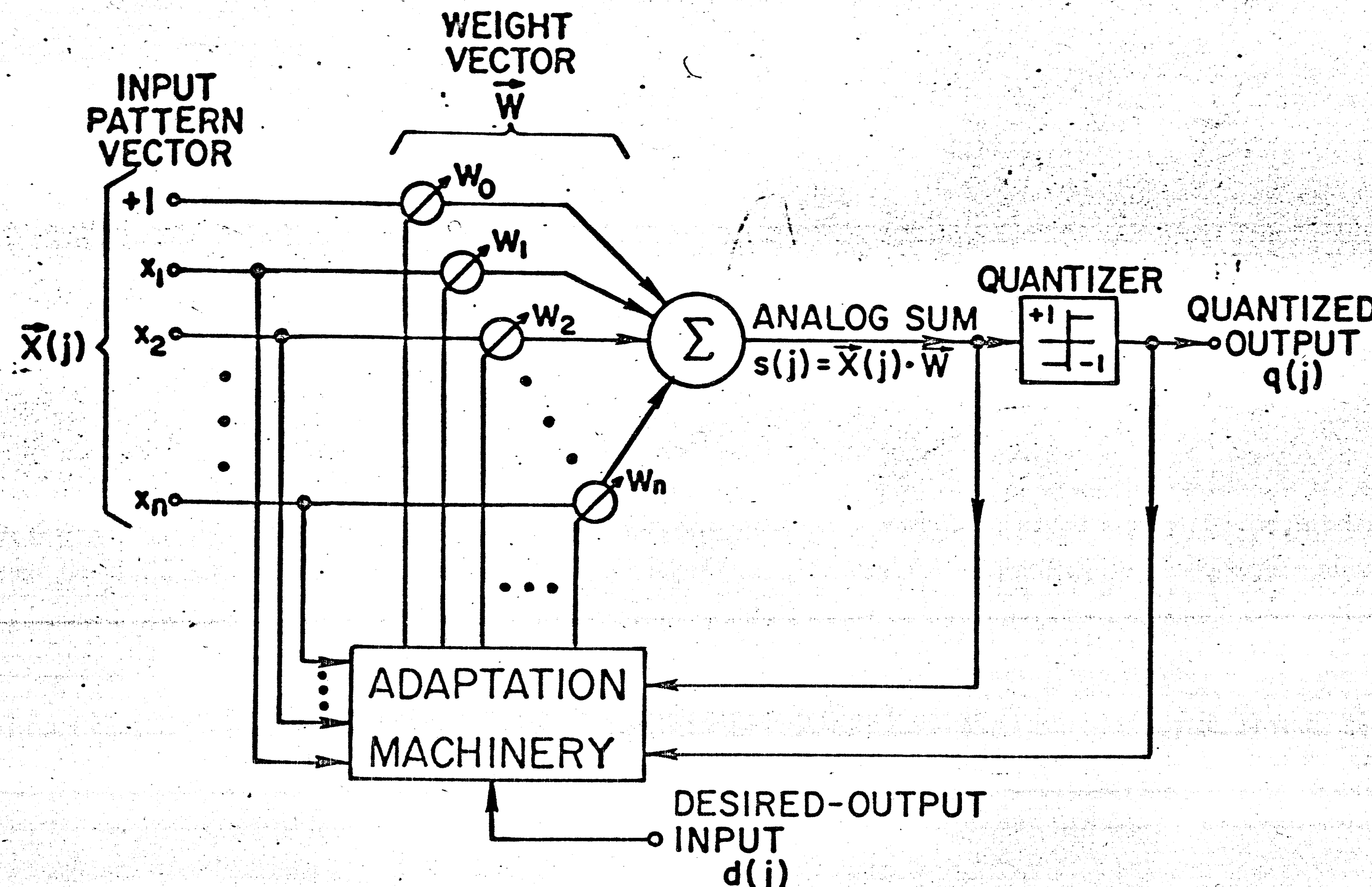
Figure 7. Modeling an unknown system by bootstrap adaptation.

Figure 8. Probability density as a function of the number of right and the number of optimal decisions.

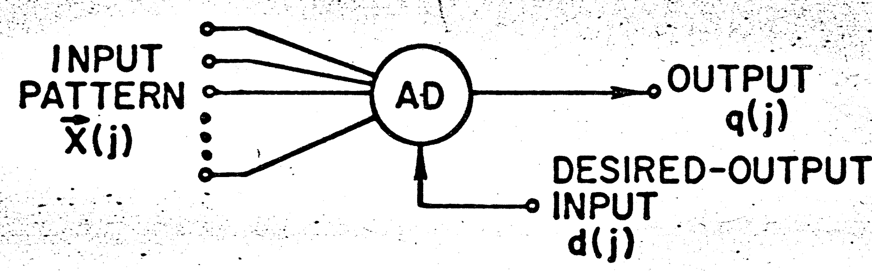
Figure 9. Regions of positive and negative bootstrapping.

Figure 10. Learning curve for Adaline playing blackjack ($\lambda = 0.4$).

Figure 11. Learning curve for Adaline playing blackjack ($\lambda = 0.04$) .

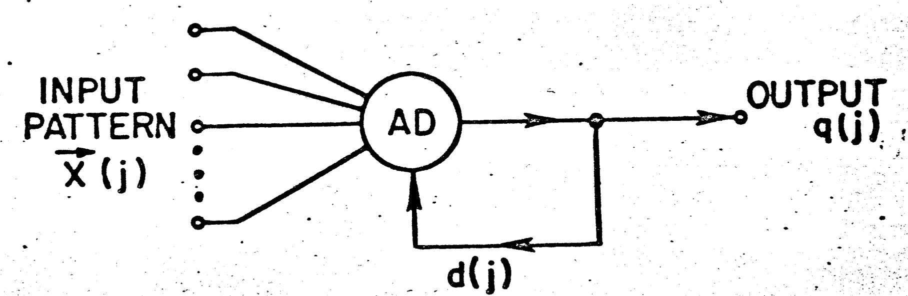


(a) FUNCTIONAL DIAGRAM

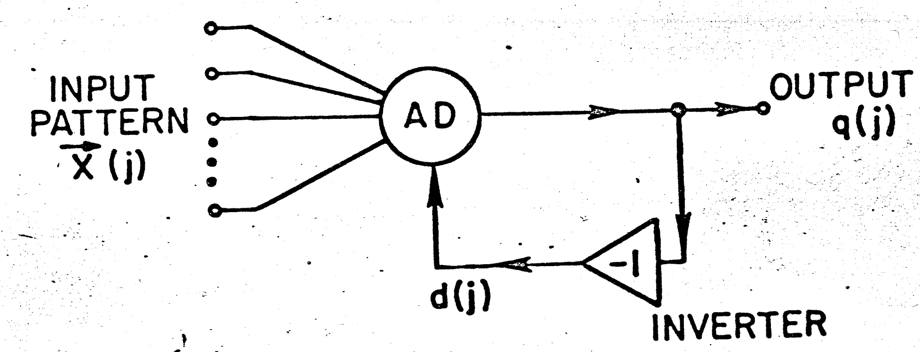


(b) SCHEMATIC SYMBOL

Fig.
1.



(a) POSITIVE BOOTSTRAP ADAPTATION



(b) NEGATIVE BOOTSTRAP ADAPTATION

Fig. 2

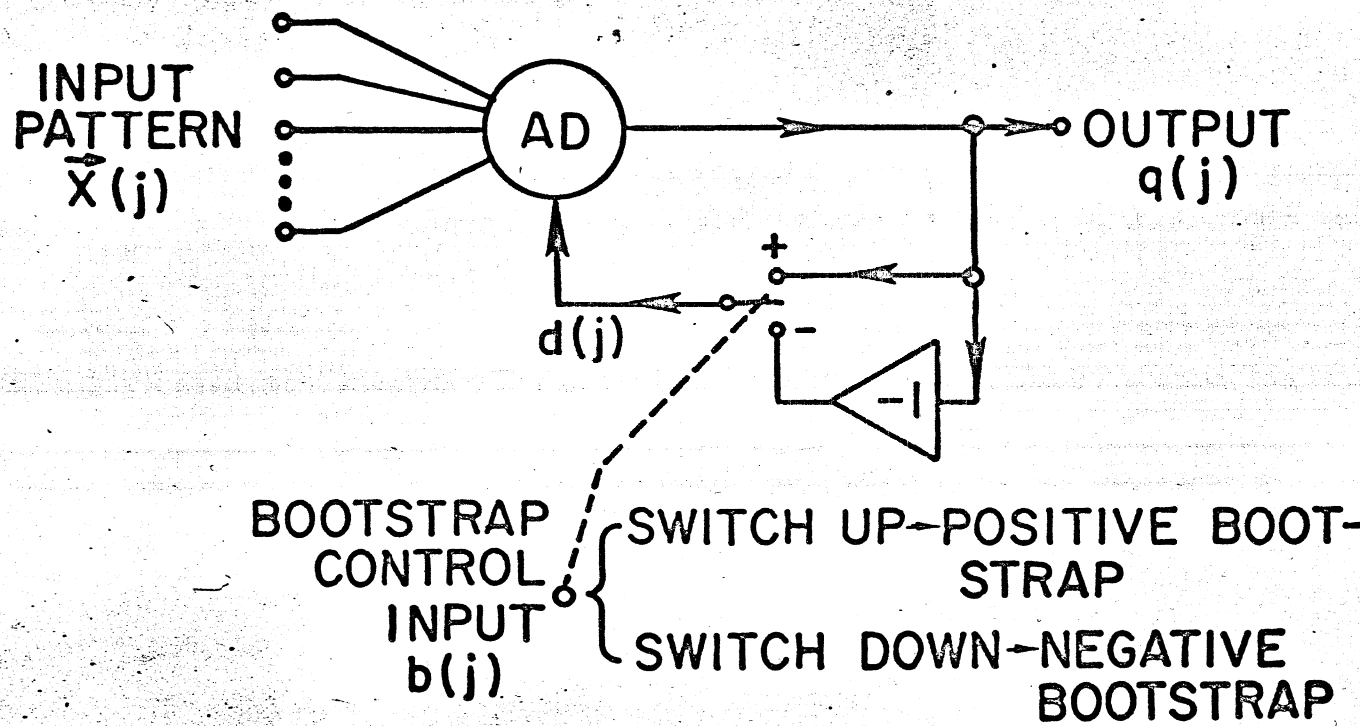

INPUT
PATTERN
$\vec{X}(j)$
AD
OUTPUT
q(j)
+
-
-1
d(j)
BOOTSTRAP
CONTROL
INPUT
b(j)
SWITCH UP→POSITIVE BOOT-
STRAP
SWITCH DOWN→NEGATIVE
BOOTSTRAP

Fig. 3

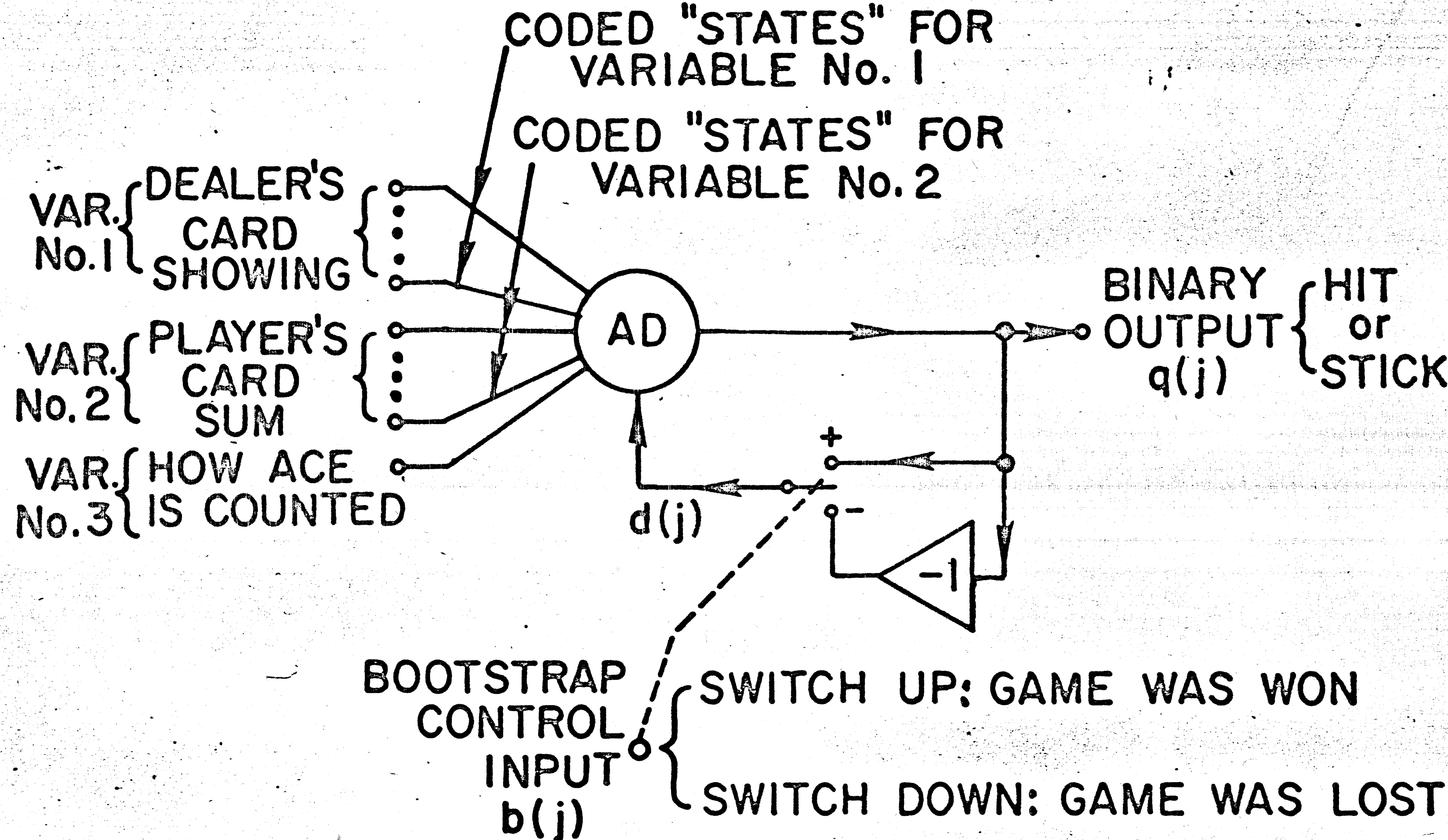

CODED "STATES" FOR VARIABLE No. 1
CODED "STATES" FOR VARIABLE No. 2
VAR. No. 1 DEALER'S CARD SHOWING
VAR. No. 2 PLAYER'S CARD SUM
VAR. No. 3 HOW ACE IS COUNTED
AD
d(j)
+
−
−1
BINARY OUTPUT q(j)
HIT or STICK
BOOTSTRAP CONTROL INPUT b(j)
SWITCH UP: GAME WAS WON
SWITCH DOWN: GAME WAS LOST

Fig. 4

VARIABLE No. 1 DEALER'S CARD SHOWING	CODE WORD
10	1 1 1 1 1 1 1 1 1
9	1 1 1 1 1 1 1 1 0
8	1 1 1 1 1 1 1 0 0
7	1 1 1 1 1 1 0 0 0
6	1 1 1 1 1 0 0 0 0
5	1 1 1 1 0 0 0 0 0
4	1 1 1 0 0 0 0 0 0
3	1 1 0 0 0 0 0 0 0
2	1 0 0 0 0 0 0 0 0
A	0 0 0 0 0 0 0 0 0

VARIABLE No. 2 ADALINE CARD SUM	CODE WORD
21	1 1 1 1 1 1 1 1 1 1
20	1 1 1 1 1 1 1 1 1 0
19	1 1 1 1 1 1 1 1 0 0
18	1 1 1 1 1 1 1 0 0 0
17	1 1 1 1 1 1 0 0 0 0
16	1 1 1 1 1 0 0 0 0 0
15	1 1 1 1 0 0 0 0 0 0
14	1 1 1 0 0 0 0 0 0 0
13	1 1 0 0 0 0 0 0 0 0
12	1 0 0 0 0 0 0 0 0 0
BELOW 12	0 0 0 0 0 0 0 0 0 0

VARIABLE No. 3

COUNTING OF ACE

1=COUNTED AS 11

0=COUNTED AS 1

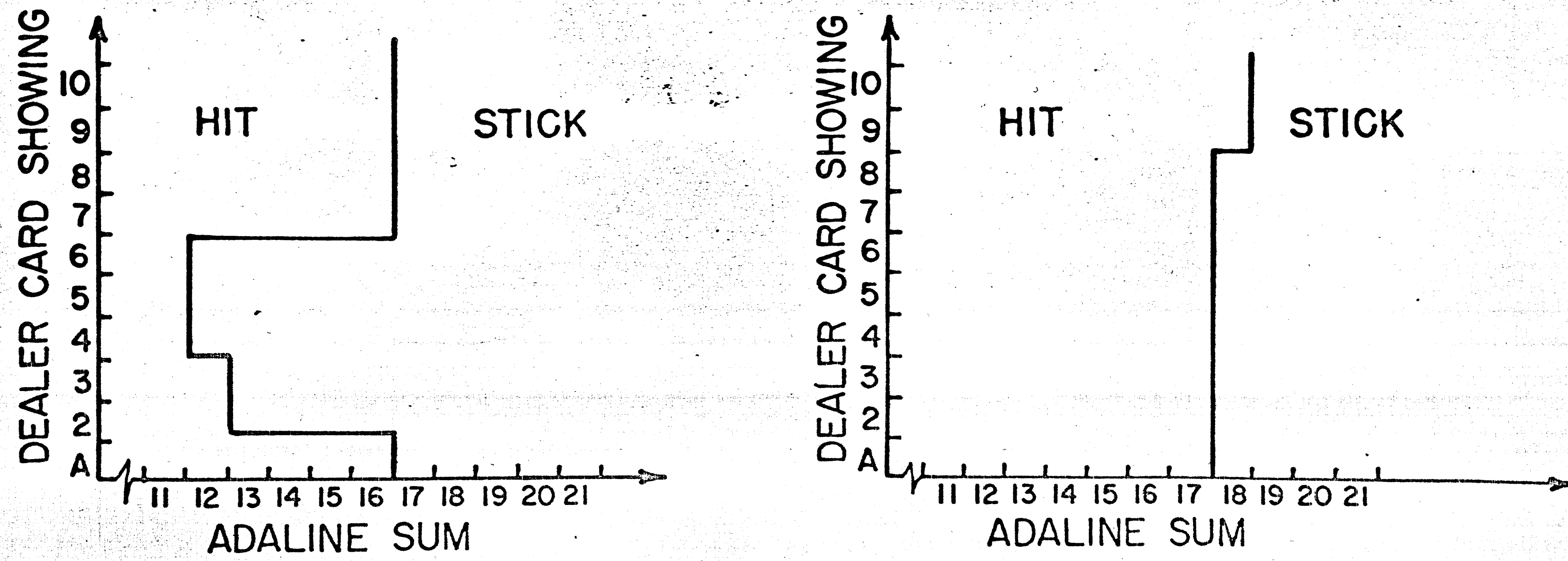
TYPICAL PATTERN 1 1 1 1 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 1

DEALER HAS 5 SHOWING

ADALINE CARD SUM IS 13

LAST BIT INDICATES ADALINE IS COUNTING AN ACE AS 11

Fig. 5



(a) ADALINE COUNTS ALL ACES AS ONES

(b) ADALINE COUNTS AN ACE AS ELEVEN

Fig. 6

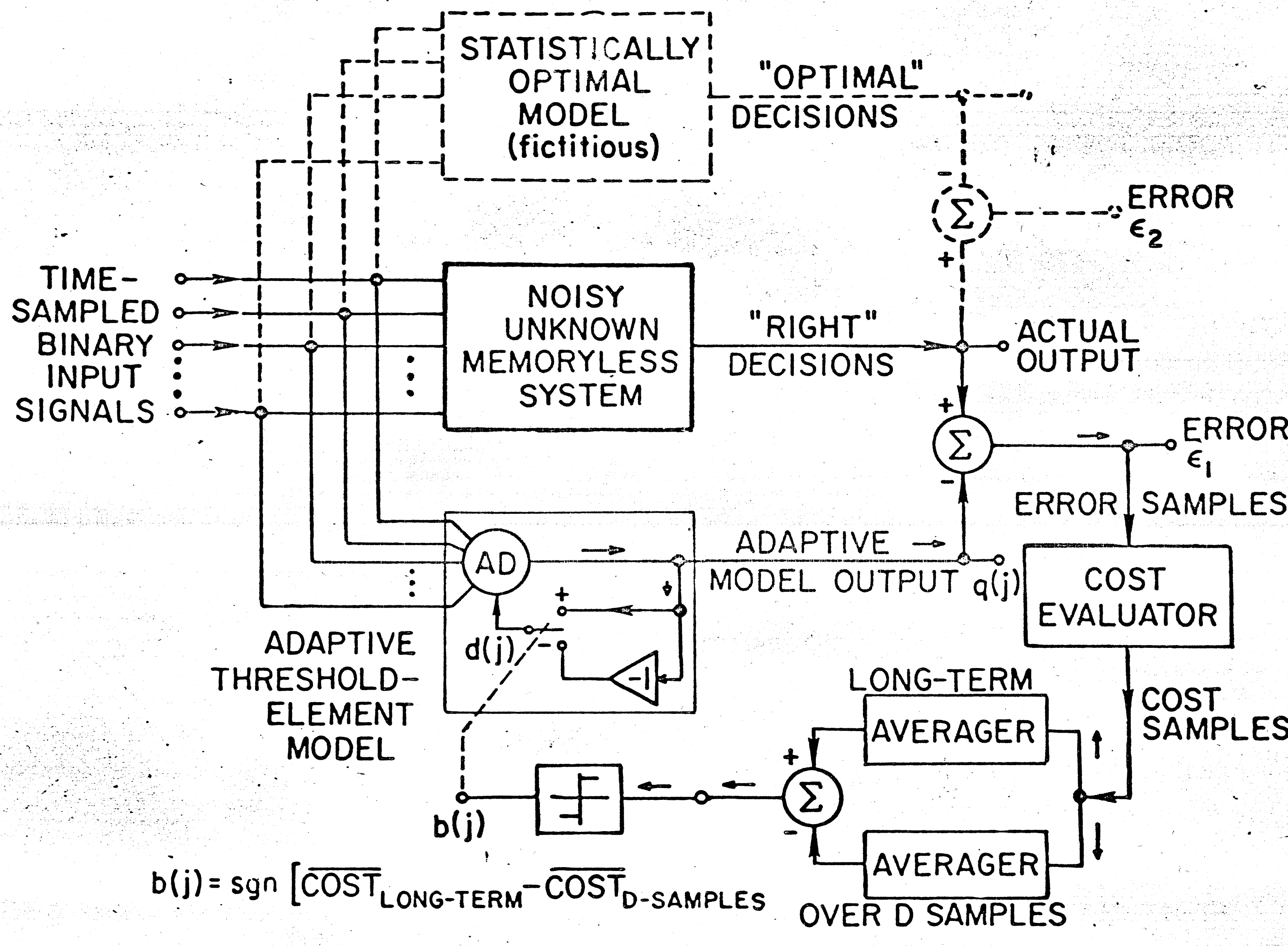
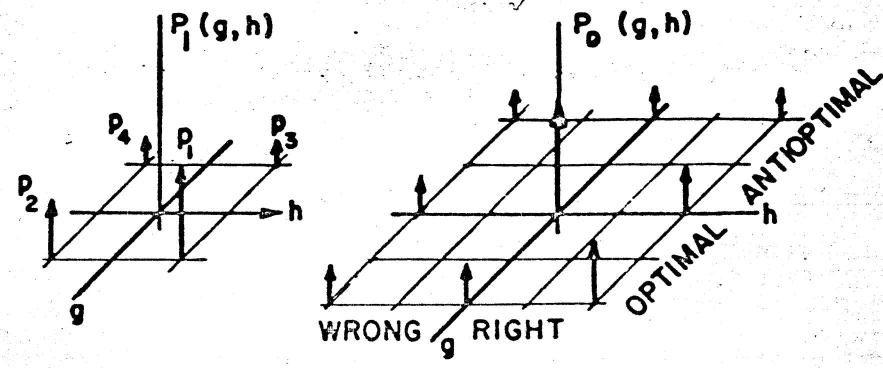

STATISTICALLY OPTIMAL MODEL (fictitious)
"OPTIMAL" DECISIONS
ERROR ϵ_2
TIME-SAMPLED BINARY INPUT SIGNALS
NOISY UNKNOWN MEMORYLESS SYSTEM
"RIGHT" DECISIONS
ACTUAL OUTPUT
ERROR ϵ_1
ERROR SAMPLES
AD
ADAPTIVE MODEL OUTPUT $q(j)$
COST EVALUATOR
$d(j)$
ADAPTIVE THRESHOLD-ELEMENT MODEL
LONG-TERM AVERAGER
COST SAMPLES
AVERAGER OVER D SAMPLES
$b(j)$
$b(j) = \mathrm{sgn}\,[\overline{COST}_{LONG\text{-}TERM} - \overline{COST}_{D\text{-}SAMPLES}$

Fig. 7



(a) ONE DECISION (b) D DECISIONS

Fi[illegible] 8

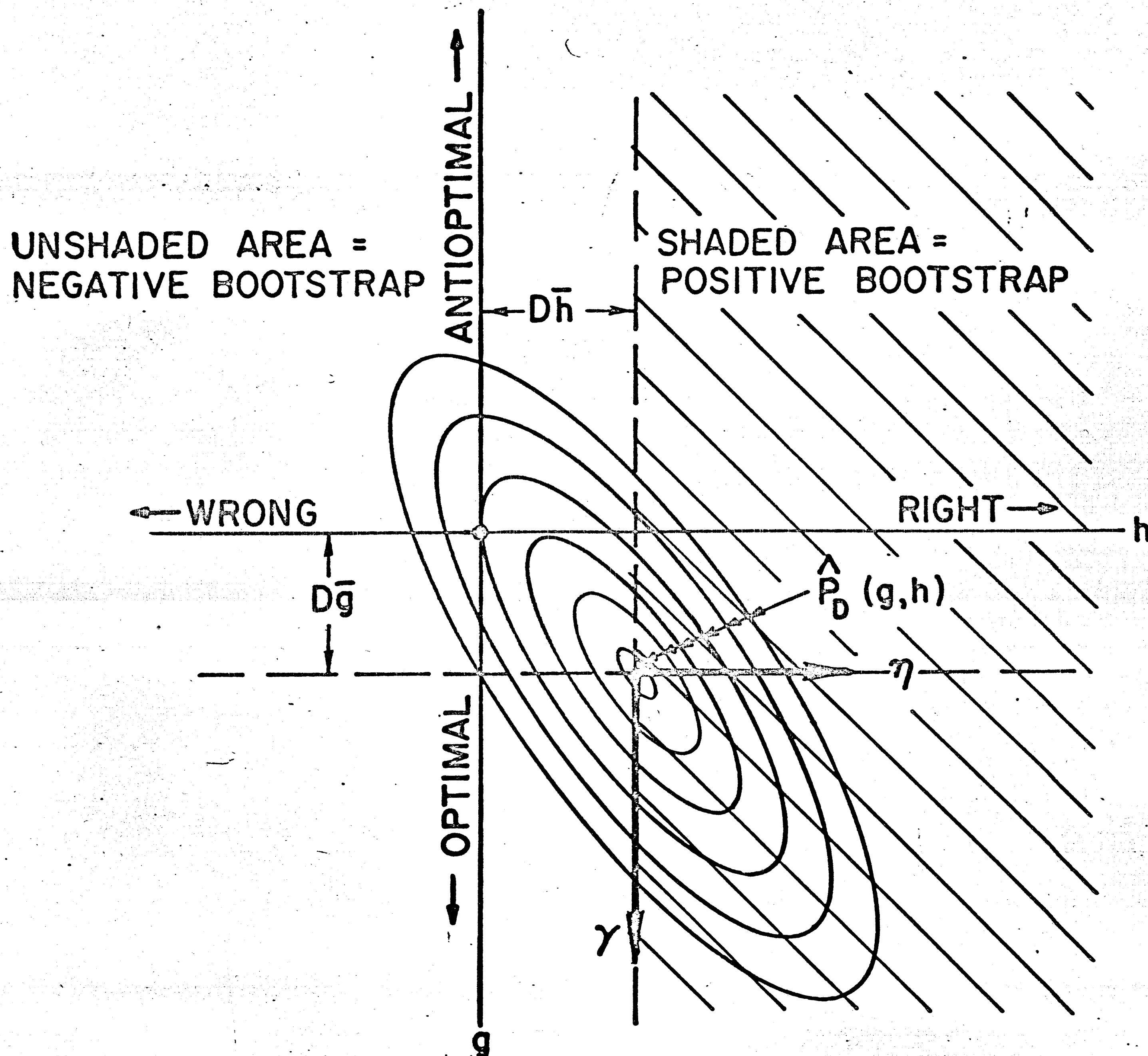
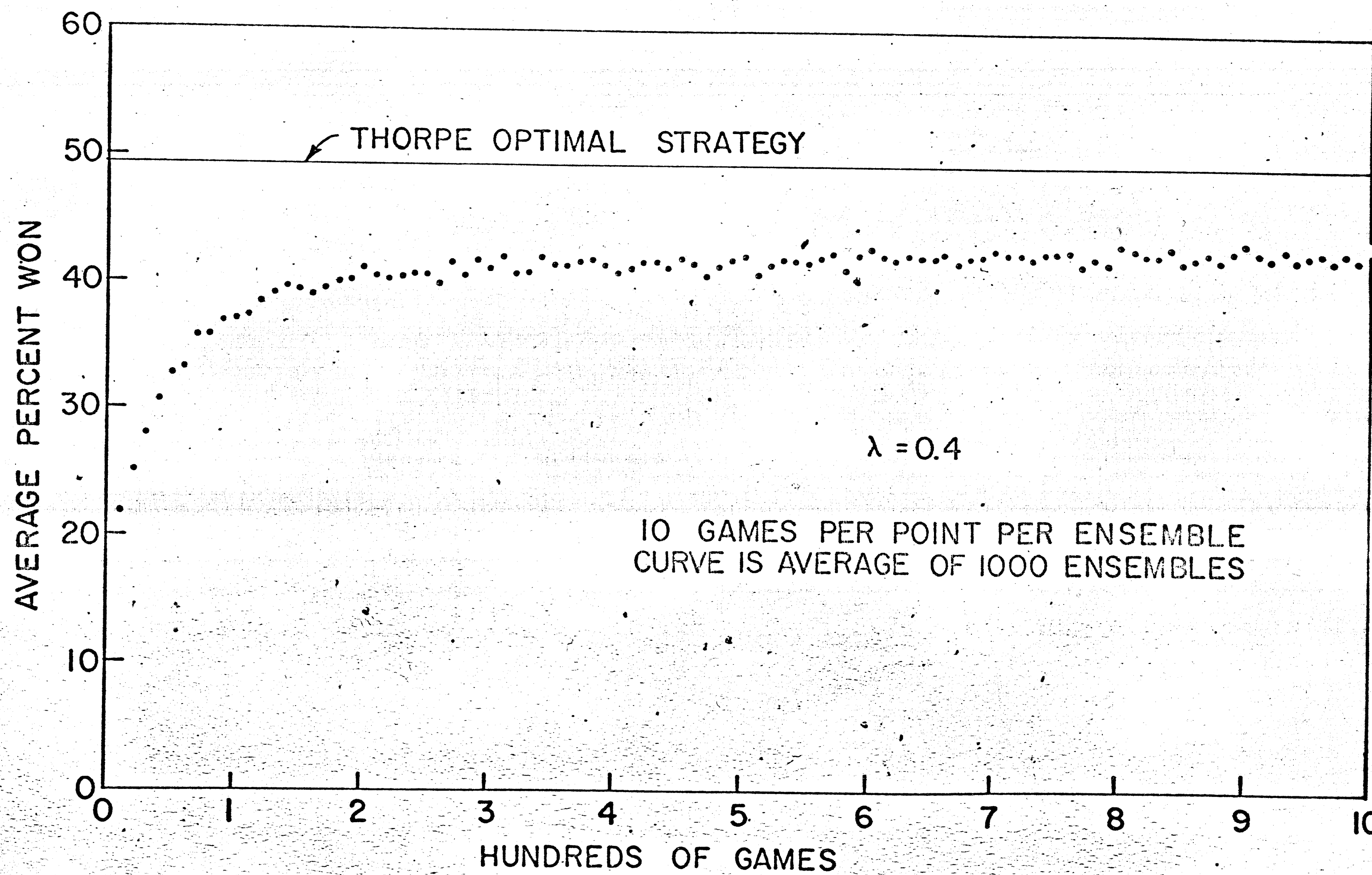

UNSHADED AREA =
NEGATIVE BOOTSTRAP
SHADED AREA =
POSITIVE BOOTSTRAP
ANTIOPTIMAL
OPTIMAL
$D\bar{h}$
$D\bar{g}$
WRONG
RIGHT
h
g
$\hat{P}_D (g,h)$
η
γ

Fig. 9


THORPE OPTIMAL STRATEGY
λ = 0.4
10 GAMES PER POINT PER ENSEMBLE
CURVE IS AVERAGE OF 1000 ENSEMBLES
AVERAGE PERCENT WON
HUNDREDS OF GAMES
0
10
20
30
40
50
60
1
2
3
4
5
6
7
8
9
10

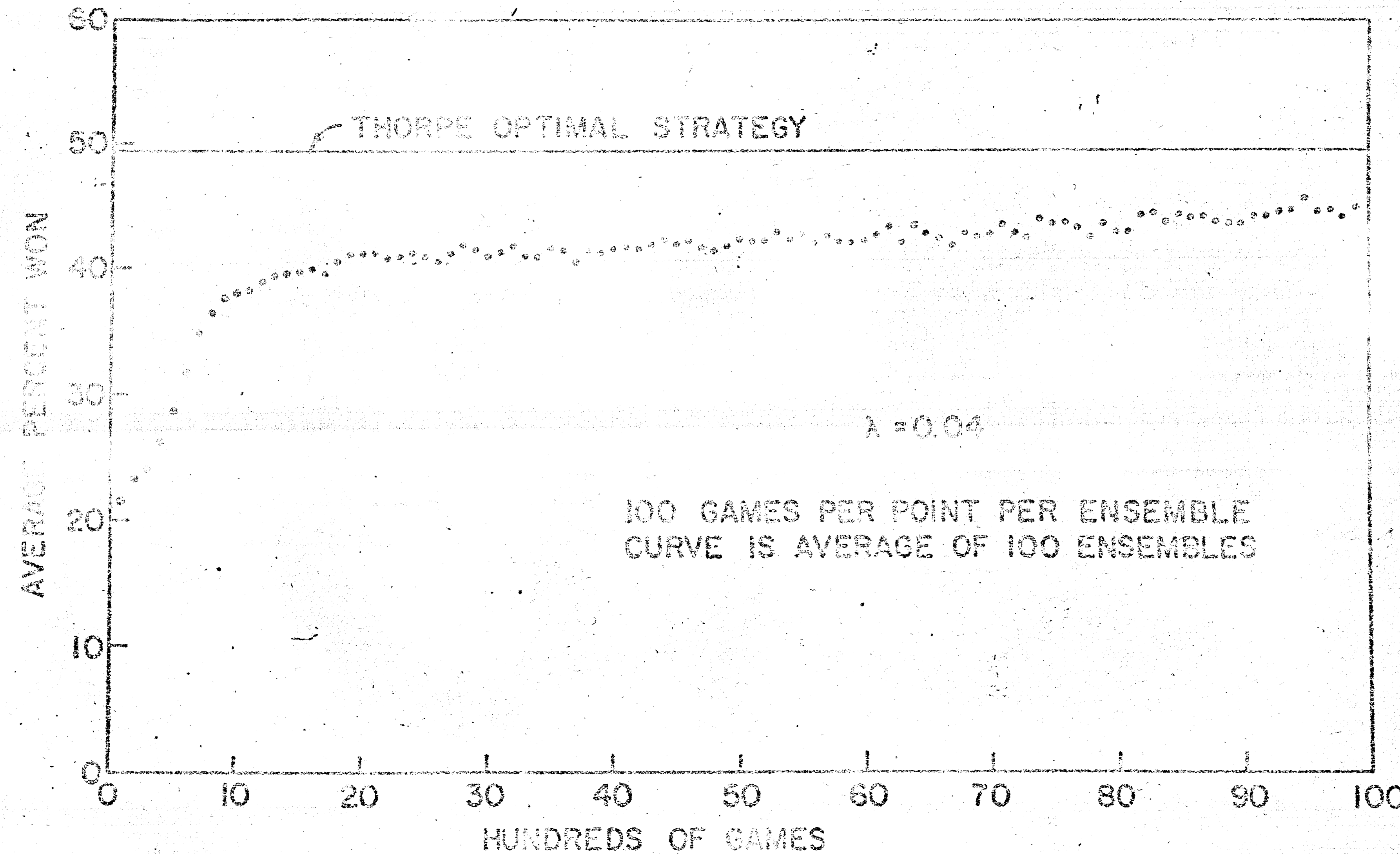

THORPE OPTIMAL STRATEGY
λ = 0.04
100 GAMES PER POINT PER ENSEMBLE
CURVE IS AVERAGE OF 100 ENSEMBLES
AVERAGE PERCENT WON
HUNDREDS OF GAMES
0
10
20
30
40
50
60
70
80
90
100

Fig.