ADAPTIVE SWITCHING CIRCUITS

Bernard Widrow
and
Marcian E. Hoff

A. Introduction

The modern science of switching theory began with work by Shannon[1] in 1938. The field has developed rapidly since then, and at present a wealth of literature exists[2] concerning the analysis and synthesis of logical networks which might range from simple interlock systems to telephone switching systems to large-scale digital computing systems.

An example illustrating the use of switching theory is that of the design of an interlock system for the control of traffic in a railroad switch yard. The first step is the preparation of a "truth table", an exhaustive listing of all input possibilities (the positions of all incoming and outgoing trains), and what the desired system output should be (what the desired control signals should be) for each input situation. The next step is the construction of a Boolean function, and the following steps are algebraic reduction and design of the logical control system.

The design of the traffic control system is an example wherein the truth table must be followed precisely and reliably. Errors would be destructive. The design of the arithmetic element of a digital computer is another example wherein the truth table must be followed precisely.

There are other situations in which some errors are inevitable, however, and here errors are usually costly but not catastrophic. These situations call for statistically optimum switching circuits. A common performance objective is the minimization of the average number of errors. An example is that of prediction of the next bit in a correlated stochastic binary number sequence. The predictor output is to be a logical combination of a finite number of previous input sequence bits. An optimum system is a sequential switching circuit that predicts with a minimum number of errors.

Suppose that a record of the binary sequence is printed on tape and cut up into pieces (with indication of the positive direction of time preserved), say 25 bits long. Place all pieces where the most recent event is ONE in one pile, and the remainder in another pile. Delete the most recent bit on each piece of tape. If the statistical scheme could be discovered by which the pieces of tape are classified, this would lead to a prediction scheme. It is apparent that prediction is a certain kind of classification.

Assuming statistical regularity, a reasonable way to proceed might be to form a truth table, and let the data from each piece of tape be an entry in the table. It might be expected that with the data of 100 pieces of tape, a fairly good predictor could be developed. The truth table would have only 100 entries however, out of a total of 2^{24}. The "best" way to fill in the remainder of the truth table depends upon the nature of the sequence statistics and the error cost criteria. Filling in the table is a difficult and a crucial part of the problem. Even if the truth table were filled in, however, the designer would have the difficult task of realizing a logical network to satisfy a truth table with 2^{24} entries.

An approach to such problems is taken in this paper which does not require an explicit use of the truth table. The design objective is the minimization of the average number of errors, rather than a minimization of the number of logical components used. The nature of the logical elements is quite unconventional. The system design procedure is adaptive, and is based upon an iterative search process. Performance feedback is used to achieve automatic system synthesis, i.e., the selection of the "best" system from a restricted but useful class of possibilities. The designer "trains" the system to give the correct responses by "showing" it examples of inputs and respective desired outputs. The more examples "seen", the better is the system performance. System competence will be directly and quantitatively related to amount of experience.

B. A Neuron Element

In Fig. 1, a combinatorial logical circuit is shown which is a typical element in the adaptive switching circuits to be considered. This element bears some resemblance to a "neuron" model introduced by von Neuman[3], whence the name.

The binary input signals on the individual lines have values of +1 or -1, rather than the usual values of 1 or 0. Within the neuron, a linear combination of the input signals is formed. The weights are the gains $a_1, a_2, \ldots$, which could have both positive and negative values. The output signal is +1 if this weighted sum is greater than a certain threshold, and -1 otherwise. The threshold level is determined by the setting of a_0, whose input is permanently connected to a +1 source. Varying a_0 varies a constant added to the linear combination of input signals.

For fixed gain settings, each of the 2^5 possible input combinations would cause either a +1 or -1 output. Thus, all possible inputs are classified into two categories. The input-output relationship is determined by choice of the gains $a_0, \ldots a_5$. In the adaptive neuron, these gains are set during the "training" procedure.

In general, there are 2^{2^5} different input-output relationships or truth functions by which the five input variables can be mapped into the single output variable. Only a subset of these, the linearly separated truth functions[4], can be realized by all possible choices of the gains of the neuron of Fig. 1. Although this subset is not all-inclusive*, it

†Department of Electrical Engineering, Stanford University.

††Doctoral student in the Department of Electrical Engineering, Stanford University.

*It becomes a vanishingly small fraction of all possible switching functions as the number of inputs gets large.

is a useful subset, and it is "searchable", i.e., the "best" function in many practical cases can be found iteratively without trying all functions within the subset.

Application of this neuron in adaptive pattern classifiers was first made by Mattson.[5,6] He has shown that complete generality in choice of switching function could be had by combining these neurons. He devised an iterative digital computer routine for finding the best set of a's for the classification of noisy geometric patterns. An iterative procedure having similar objectives has been devised by these authors and is described in the next section. The latter procedure is quite simple to implement, and can be analyzed by statistical methods that have already been developed for the analysis of adaptive sampled-data systems.

C. An Adaptive Pattern Classifier

An adaptive pattern classification machine (called "Adaline", for adaptive linear) has been constructed for the purpose of illustrating adaptive behavior and artificial learning. A photograph of this machine, which is about the size of a lunch pail, is shown in Fig. 2.

During a training phase, crude geometric patterns are fed to the machine by setting the toggle switches in the 4x4 input switch array. Setting another toggle switch (the reference switch) tells the machine whether the desired output for the particular input pattern is +1 or -1. The system learns something from each pattern and accordingly experiences a design change. The machine's total experience is stored in the values of the weights $a_0 \ldots a_{16}$. The machine can be trained on undistorted noise-free patterns by repeating them over and over until the iterative search process converges, or it can be trained on a sequence of noisy patterns on a one-pass basis such that the iterative process converges statistically. Combinations of these methods can be accommodated simultaneously. After training, the machine can be used to classify the original patterns and noisy or distorted versions of these patterns.

A block schematic of Adaline is shown in Fig. 3. In the actual machine, the quantizer is not built in as a device but is accomplished by the operator in viewing the output meter. Different quantizers (2-level, 3-level, 4-level) are realized by using the appropriate meter scales (see Fig.2). Adaline can be used to classify patterns into several categories by using multi-level quantizers and by following exactly the same adaptive procedure.

The following is a description of the iterative searching routine. A pattern is fed to the machine, and the reference switch is set to correspond to the desired output. The error (see Fig.3) is then read (by switching the reference switch; the error voltage appears on the meter, rather than the neuron output voltage). All gains including the level are to be changed by the same absolute magnitude, such that the error is brought to zero. This is accomplished by changing each gain (which could be positive or negative) in the direction which will diminish the error by an amount which reduces the error magnitude by 1/17. The 17 gains may be changed in any sequence, and after all changes are made, the error for the present input pattern is zero. Switching the reference back, the meter reads exactly the desired output. The next pattern, and its desired output, is presented and the error is read. The same adjustment routine is followed and the error is brought to zero. If the first pattern were reapplied at this point, the error would be small but not necessarily zero. More patterns are inserted in like manner. Convergence is indicated by small errors (before adaption), with small fluctuations about a stable root mean-square value. The iterative routine is purely mechanical, and requires no thought on the part of the operator. Electronic automation of this procedure will be discussed below.

The results of a typical adaption on six noiseless patterns is given in Figs. 4 and 5. The patterns were selected in a random sequence, and were classified into 3 categories. Each T was to be mapped to +60 on the meter dial, each G to 0, and each F to -60. As a measure of performance, after each adaptation, all six patterns were read in (without adaptation) and six errors were read. The sum of their squares denoted by $\Sigma\epsilon^2$ was computed and plotted. Fig. 5 shows the learning curve for the case in which all gains were initially zero.

D. Statistical Theory of Adaption for Sampled-Data Systems

This section is a summary of the portions of Widrow's statistical theory of adaption for sampled-data systems[7,8] that is useful in the analysis of adaptive switching circuits.

Consider the general linear sampled-data system formed of a tapped delay line, shown in Fig. 6. This system is intended to be a statistical predictor. The individual impulses of the impulse response may be adjusted in the following manner. Apply a mean square reading meter to $\epsilon(m)$, the difference between the present input and the delayed prediction. This meter will measure mean square error in prediction. Adjust $h_1, h_2, h_3, \ldots$, until the meter reading is minimized.

The problem of adjusting the h's is not trivial, because their effects upon performance interact. Suppose that the predictor has only two impulses in its impulse response, h_1 and h_2. The mean square error for any setting of h_1 and h_2 can be readily derived:

$$\epsilon(m) = f(m) - h_1 f(m-1) - h_2 f(m-2)$$

$$\overline{\epsilon^2}(m) = \phi_{ff}(0)h_1^2 + \phi_{ff}(0)h_1^2 - 2\phi_{ff}(1)h_1 - 2\phi_{ff}(2)h_2 + 2\phi_{ff}(1)h_1 h_2 + \phi_{ff}(0) \qquad (1)$$

The discrete autocorrelation function of the input is $\phi_{ff}(j)$.

The mean square error given by equations (1) is what the mean square meter would read if it were to average over very large sampled size. The mean square error is a parabolic function of the predictor adjustments h_1 and h_2, and, in general, can easily be shown to be a quadratic function of such adjustments, regardless of how many there are.

The optimum n-impulse predictor can be derived analytically by setting the partial derivatives of $\overline{\epsilon^2}$ of equation (1) equal to zero. This is the discrete analogue of Wiener's optimization[7] of continuous filters. Finding the optimum system experimentally is the same as finding a minimum of a paraboloid in n dimensions. This could be done manually by having a human operator read the meter and set the adjustment, or it could be done automatically

by making use of any one of several iterative gradient methods for surface-searching, as devised by numerical analysts. When either of these schemes is employed, an adaptive system results that consists essentially of a "worker" and a "boss". The worker in this case predicts, whereas the boss has the job of adjusting the worker.

Figure 7 is a block-diagram representation of such a basic adaptive unit. The boss continually seeks a better worker by trial and error experimentation with the structure of the worker. Adaption is a multidimensional performance feedback process. The "error" signal in the feedback control sense is the gradient of mean square error with respect to adjustment.

Many of the commonly used gradient methods search surfaces for stationary points by making changes in the independent variables (starting with an initial guess) in proportion to measured partial derivatives to obtain the next guess, and so forth. These methods give rise to geometric (exponential) decays in the independent variables as they approach a stationary point for second-degree or quadratic surfaces. One-dimensional surface-searching is illustrated in Fig. 8.

The surface being explored in Fig. 8 is given by Eq. (2). The first and second derivatives are given by Eqs. (3) and (4).

$$y = a(x - b)^2 + c \tag{2}$$

$$\frac{dy}{dx} = 2a(x - b) \tag{3}$$

$$\frac{d^2y}{dx^2} = 2a \tag{4}$$

A sampled-data feedback model of the iterative process is shown in Fig. 8(b). This flow-graph can be reduced, and the resulting characteristic equation is

$$-(2ak + 1)z^{-1} + 1 = 0 \tag{5}$$

The iterative process is stable when $-1/a<k<0$, and transients decay completely in one step when $k = -1/2a$.

"Noise" in the measurements of the mean square error surface due to small sample size causes noisy derivative measurements. These noises enter the adaption process, as indicated in Fig. 8(b), and cause noisy system adjustments.

Variance in x about the optimum value causes the average of y to be greater than the minimum value c. The increase in $\bar{y}$ equals the variance in x multiplied by a, as can be seen from Eq. (2). It is useful to define a dimensionless parameter M the "misadjustment", as the ratio of the mean increase in mean square error to the minimum mean square error. It is a measure of how the system performs on the average, after adapting transients have died out, compared with the fixed optimum system. With regard to the curve of Fig. 8,

$$M = \frac{\bar{y} - c}{c} \tag{6}$$

More detailed derivations of misadjustment formulas covering several different methods of surface searching and derivative measurement are presented in Refs. 7 and 8. The particular formulas which can be applied to the analysis of adaptive switching circuits are the following.

When derivatives are measured by data repeating, i.e., when the same system input data is applied for both N "forward" and N "backward" measurements of mean square error, the misadjustment is given by

$$M = \frac{1}{2(N\tau)} \tag{7}$$

τ is the time constant of the iterative process of Fig. 8, and is equal to $-1/2ak$. A unit time constant means that the adjustment error decreases by a factor 1/e per iteration cycle. Equation (7) is conservative, and appreciably so only for small values of τ, less than 1. In the limiting case of one-step adaption, $\tau = 0$ and the appropriate misadjustment formula is

$$M = \frac{1}{N} \tag{8}$$

In deriving Formulas (7) and (8), it has been assumed that the error samples are Gaussian distributed, with zero mean, and are uncorrelated. It can be shown that these results are highly insensitive to this distribution density shape, and are appreciably affected by correlation only when it exceeds 0.8. It is interesting to note that the quality of adaption depends only on the number of samples "seen" by the system in adapting.

The expressions (7) and (8) are based on the supposition that fresh data is brought in for each cycle of iteration. If the system adapts on a fixed body of N error samples the misadjustment is given by Formula (8). When there are m interacting adjustments instead of just one, Expressions (7) and (8) may be generalized by multiplication by m.

E. Statistical Theory of Adaption for the Adaptive Neuron Element

The error signal measured and used in adaption of the neuron of Fig. 1 is the difference between the desired output and the sum before quantization. This error is indicated by ϵ in Fig. 9. The actual neuron error, indicated by ϵ_n in Fig. 9, is the difference between the neuron output and the desired output.

The objective of adaption is the following. Given a collection of input patterns and the associated desired outputs, find the best set of weights $a_0, a_1, \ldots a_m$ to minimize the mean square of the neuron error, $\overline{\epsilon_n^2}$. Individual neuron errors could only have the values of +2, 0, and -2 with a two-level quantizer. Minimization of $\overline{\epsilon_n^2}$ is therefore equivalent to minimizing the average number of neuron errors.

The simple adaption procedure described in this paper minimizes $\overline{\epsilon^2}$ rather than $\overline{\epsilon_n^2}$. The measured error ϵ has zero mean (a consequence of the minimization of $\overline{\epsilon^2}$) and will be assumed to be Gaussian-distributed. By making use of certain geometric arguments or by using a statistical theory of amplitude quantization,[10] it can be shown that $\overline{\epsilon_n^2}$ is a monotonic function of $\overline{\epsilon^2}$, and that minimization of $\overline{\epsilon^2}$ is equivalent to minimization of $\overline{\epsilon_n^2}$ and to minimization of the probability of neuron error. The ratio of these mean squares has been calculated and is plotted in Fig. 9 as a function of the neuron error probability.

Given any collection of input patterns and the associated desired outputs, the measured mean square error $\overline{\epsilon^2}$ must be a precisely parabolic

function of the gain settings, $a_0, \ldots a_n$. Let the kth pattern be indicated as the vector $S(k) = s_1(k), s_2(k), \ldots s_n(k)$. The s's have values of +1 or -1, and represent the n input components numbered in a fixed manner. The kth error is

$$\epsilon(k) = d(k) - a_0 - a_1 s_1(k) - a_2 s_2(k) - \ldots - a_n s_n(k) \quad (9)$$

For simplicity, let the neuron have only two input lines and a level control. The square of the error is accordingly

$$\begin{aligned}\epsilon^2(k) = {} & d^2(k) + a_0^2 + s_1^2(k)a_1^2 + s_2^2(k)a_2^2 \\ & - 2d(k)a_0 - 2d(k)s_1(k)a_1 - 2d(k)s_2(k)a_2 \\ & + 2s_1(k)a_0a_1 + 2s_2(k)a_0a_2 + 2s_1(k)s_2(k)a_1a_2 \end{aligned} \quad (10)$$

The mean square error averaged over k is

$$\begin{aligned}\overline{\epsilon^2} = {} & a_0^2 + \phi(s_1,s_1)a_1^2 + \phi(s_2,s_2)a_2^2 - \bar{d}a_0 \\ & - 2\phi(d,s_1)a_1 - 2\phi(d,s_2)a_2 + 2\bar{s}_1a_0a_1 + 2\bar{s}_2a_0a_2 \\ & + 2\phi(s_1,s_2)a_1a_2 + \phi(d,d) \end{aligned} \quad (11)$$

The ϕ's are spatial correlations. $\phi(s_1,s_2) = \overline{s_1s_2}$, etc. Note that $\phi(s_j,s_j) = \overline{s_js_j} = 1$.

Adjusting the a's to minimize $\overline{\epsilon^2}$ is equivalent to searching a parabolic stochastic surface (having as many dimensions as there are a's) for a minimum. How well this surface can be searched will be limited by sample size, i.e., by the number of patterns seen in the searching process.

The method of searching that has proven most useful is the method of steepest descent. Vector adjustment changes are made in the direction of the gradient. Transients consist of sums of geometric sequence components (there are as many natural "frequencies" as the number of adjustments, as can be seen from generalization of the flow graph of Fig. 9--see Ref. 9). It can be shown that the method of steepest descent will be stable when the proportionality constant k between partial derivative and size of change is less than the reciprocal of the second partial derivative. It can also be shown that when k is small, transients can be approximately represented as being of the single time constant 1/2k.

The method of adaption that has been used requires an extremely small sample size per iteration cycle, namely one pattern. One-pattern-at-a-time adaption has the advantages that derivatives are very easy to measure and that no storage is required within the adaptive machinery except for the gain values (which contain the past experience of the neuron).

The square of the error for a single pattern (the mean square error for a sample size of one) is given by Eq. (10). The partial derivatives are

$$\begin{aligned}\frac{\partial\epsilon^2(k)}{\partial a_0} &= \left[-2d(k) + 2a_0 + 2s_1(k)a_1 + 2s_2(k)a_2\right] \\ \frac{\partial\epsilon^2(k)}{\partial a_1} &= s_1(k)\left[-2d(k) + 2a_0 + 2s_1(k)a_1 + 2s_2(k)a_2\right] \\ \frac{\partial\epsilon^2(k)}{\partial a_2} &= s_2(k)\left[-2d(k) + 2a_0 + 2s_1(k)a_1 + 2s_2(k)a_2\right] \end{aligned} \quad (12)$$

Comparison of the Eqs. (12) with Eq. (9) shows that the derivatives are simply related to the measured error, and suggest that the derivatives could be measured without squaring and averaging and without actual differentiation. The jth partial derivative is given by

$$\frac{\partial\epsilon^2(k)}{\partial a_j} = -2s_j(k)\ \epsilon(k) \quad (13)$$

It follows that all partial derivatives have the same magnitude, and have signs determined by the error sign and the respective input signal signs. The procedure described in Sec. C for bringing $\epsilon(k)$ to zero with each successive input pattern gives the constant k a value of $1/2(n+1)$. From the previous discussion we see that the time constant of the iterative process is therefore $\tau=(n+1)$ patterns. On the 4x4 adaline, there are n=16 input line gains plus a level control. Therefore, the time constant should be roughly 17 patterns (for verification, see the learning curve of Fig. 5). The search procedure could be readily modified to speed up or slow down the adaption process by adjusting k.

The misadjustment Formulas (7) and (8) when applied to the adaptive neuron give the per unit increase in measured mean square error as a result of adapting on a finite number of patterns. Since the ratio of probability of neuron error to the mean square error $\overline{\epsilon^2}$ is essentially constant over a wide range of error probabilities (Fig.9), the misadjustment as expressed by Formulas (7) and (8) may be interpreted in terms of the ratio of the increase in error probability to the minimum error probability.

If adaption is accomplished by injection of a fresh pattern each iteration cycle, the misadjustment, as derived from Eq. (7), is

$$M = \frac{(n+1)}{2\tau} \quad (14)$$

Following the procedure of bringing $\epsilon(k)$ to zero each iteration cycle, the misadjustment is

$$M = \frac{(n+1)}{2\tau} = \frac{(n+1)}{2(n+1)} = \frac{1}{2} \quad (15)$$

If adaption is accomplished by taking a fixed collection of N patterns and repeating them over and over for several time constants (where the time constant is long, several times N), the misadjustment, as derived from Eq. (8), is

$$M = \frac{(n+1)}{N} \quad (16)$$

Simulation tests have shown that the misadjustment formulas are highly accurate over a very wide range of pattern and noise characteristics. A description of a typical experiment and its results is given in Fig. 10.

Noisy 3x3 patterns were generated by randomly injecting errors in ten percent of the positions of the four "pure" patterns, X,T,C,J.

The best system, arrived at by slow precise adaption on the full body of 100 noisy patterns, was able to classify these patterns as desired except for twelve errors. The gains were then set to zero and ten patterns were chosen at random. The best system for these patterns was arrived at and tested on the full body of 100 patterns. Twenty-

five classification errors out of 100 were made. The misadjustment was 108 percent. The experiment was repeated three more times, and the misadjustments that resulted, in order, were 58 percent, 67 percent and 133 percent. Since N=10 patterns and n=9 input lines, the theoretical misadjustment was

$$M = \frac{n + 1}{N} = 100 \text{ percent}$$

An average taken over the four experiments gives a measured misadjustment of 91.5 percent.

The adaptive classifier can adapt after seeing remarkably few patterns. A misadjustment of 20 percent should be acceptable in most applications. To achieve this, all one has to do is supply the adaptive classifier with a number of patterns equal to five times the number of input lines, regardless of how noisy the patterns are and how difficult the "pure" patterns are to separate. Although the misadjustment formulas have been derived for the specific classifier consisting of a single adaptive neuron, it is suspected that the following "rule of thumb" will apply fairly well to all adaptive classifiers: the number of patterns required to train an adaptive classifier is equal to several times the number of bits per pattern.

F. Networks of Adaptive Neurons

Linearly separable* pure patterns and noisy versions of them are readily classified by the single neuron. Non-linearly separable pure patterns and their noisy equivalents can also be separated by a single neuron, but absolute performance can be improved and the generality of the classification scheme can be greatly increased by using more than one neuron.

Two Adalines were combined by using the following adaption procedure: if the desired output for a given input pattern applied to both machines was -1, then both machines were adapted in the usual manner to ensure this; if the desired output was +1, the machine with the smallest measured error ϵ was assigned to adapt to give a+1 output while the other machine remained unchanged. If either or both machines gave outputs of +1, the pattern was classified as +1. If both machines gave -1 outputs, the pattern was classified as -1.

This procedure assigns specific "responsibility" to the neuron that can most easily assume it. If, at the beginning of adaption, a given neuron takes responsibility for producing a +1 with a certain input pattern, it will invariably take this responsibility each time the pattern is applied during training. Notice that it is not necessary for a teacher to assign responsibility. The combination does this automatically and requires only input patterns and the associated desired outputs, like the single neuron.

Various classification problems could be solved simultaneously by multiplexing neurons or combinations of neurons. One neuron might be trained to decide whether the man in a given picture does or does not have a green tie, while another neuron or combination could be trained to decide whether or not the man has a checkered shirt, Each neuron or combination has its own output line, and each is fed the appropriate desired output signal during training. The input signals are common to all neurons. In this manner, it is possible to form adaptive classifiers that can separate with great accuracy large quantities of complicated patterns into many output categories. All that is needed is large quantities of adaptive neurons.

G. Adaptive Microelectronic Systems

The structure of the neuron described in this paper and its adaption procedure is sufficiently simple that an effort is under way to develop a physical device which is an all-electronic fully automatic Adaline. The objective is a self-contained device, like the one sketched in Fig. 11, that has a signal input line, a "desired output" input line (actuated during training only), an output line, and a power supply. The device itself should be suitable for mass production, should contain few parts, should be reliable, and probably should consist of solid-state components.

To have such an adaptive neuron, it is necessary to be able to store the gain values, which could be positive or negative, in such manner that these values could be changed electronically.

Present efforts have been based on the use of multi-aperature magnetic cores (MAD elements[11]). The special characteristics of these cores permit multilevel storage with continuous, non-destructive read out. In addition, the stored levels are easily changed by small controlled amounts, with the direction of the change being determined by logic performed by the MAD element. The results of this work have shown that macroscopic adaptive neurons made of MAD elements will soon exist, and that with the use of thin ferromagnetic films, adaptive microelectronic neurons will ultimately exist.

H. Applications for Adaptive Logical Circuit Elements

If a computer were built of adaptive neurons, details of structure could be imparted by the designer by training (showing it examples of what he would like to do) rather than by direct designing. This design concept becomes more significant as size and complexity of digital systems increase. The demands of modern technology are such that larger and more complex digital systems are continually being contemplated, and in step with this, progress in microelectronics makes such systems physically and economically possible.

The problem of reliability is greatly aggravated by increase in size and complexity.

Shannon and Moore[12] and von Neumann[3] have proposed schemes for increasing the reliability of fixed digital systems by using redundancy. The reliability of systems may be increased further by combining adaption and redundancy. Consider a multiplex consisting of three machines solving the same problem with the same input data. Let the output of each machine be a single binary number, expressed as +1 or -1. If these machines were perfectly reliable, their outputs would always agree. If not, then von Neumann proposed that the majority should rule. The neuron shown in Fig. 1 with a_0 set to zero, and the other gains set to +1 would give a majority output. Each machine has equal vote. Unequal vote (higher vote going to the more reliable machine) is possible by making the a's adjustable,

*A more complete discussion of linear separability is given in references 4 and 5.

and causing these adjustments to be made automatically to optimize performance. The adaptive vote taker, identical to the adaptive neuron, can be trained by periodically injecting a certain input when the desired output is known. The adaptive vote taker could ideally give the correct outcome with only a single correct machine by giving it a heavy vote and attenuating the votes of the unreliable machines. This is in effect an adaptive routing procedure for information flow, and allows systems in a small measure to be self-healing.

One of the most promising areas of research in computer system theory is that of problem-solving machines, theorem-proving machines, and artificially "intelligent" machines. The earliest proponents of this research were Turing and Shannon.[13] Their suggestions were put to practice with some success by Newell, Simon, and Shaw,[14] by Samuel,[15] and by others.

An automatic problem-solving computer should have a memory system from which information could be extracted according to classification rather than according to address number. The use of stored games, or "rote learning",[15] would be considerably more powerful if it were possible to extract from the memory previous situations that are similar and not necessarily identical to the current situation. Far less experience and storage would be needed to adapt to a given level of competence. The extent of classification before storing should be slight (e.g., is the pattern of checkers or of chess?), and a consistent scheme for the arrangement of the pattern bits should be established before storing. Final classification should be done within the memory itself. Each storage register might contain an Adaline or a network of Adalines.

A request from a "central control" for a certain type of information would be sent to every register in the memory simultaneously. This has the effect of setting the adjustments of all the Adalines. Only the registers whose classifiers respond properly (e.g., give +1 outputs) answer the request and transmit their information back to the "central control."

Very sophisticated learning procedures would become possible if one had such recall-by-association parallel-access memory systems. The simplicity of Adaline and the progress being made in microelectronics gives a strong indication that such memory systems will come into existence in the not too distant future.

Acknowledgment

This research was supported by the Office of Naval Research under contract with Stanford University.

* * * * * * * * *

References

1. C. E. Shannon, "A symbolic analysis of relay and switching circuits," Trans. of AIEE, Vol. 37, 1938, pp. 713-723.
2. S. H. Caldwell, Switching Circuits and Logical Design, Wiley, 1958.
3. J. von Neuman, "Probabilistic Logic and Synthesis of Reliable Organisms from Unreliable Components", in Automata Studies, Princeton University Press, 1956.
4. Robert McNaughton, "Unate truth functions", Tech. Report No. 4, Appl. Math. and Stat. Lab., Stanford University, October 21, 1957.
5. R. L. Mattson, "The design and analysis of an adaptive system for statistical classification", S. M. Thesis, MIT, May 22, 1959.
6. R. L. Mattson, "A self-organizing logical system", 1959 Eastern Joint Computer Conference Convention Record, Inst. of Radio Eng., N.Y.
7. B. Widrow, "Adaptive sampled-data systems—a statistical theory of adaptation", 1959 WESCON Convention Record, Part 4.
8. B. Widrow, "Adaptive sampled-data systems", IFAC Moscow Congress Record, Butterworth Publications, London, 1960.
9. Wiener, N., Extrapolation, Interpolation, and Smoothing of Stationary Time Series with Engineering Applications, New York, Wiley, 1949.
10. B. Widrow, "A study of rough amplitude quantization by means of Nyquist sampling theory", IRE Transactions, PGCT, Vol. CT-3, Number 4, Dec. 1956.
11. H. D. Crane, "A high-speed logic system using magnetic elements and connecting wire only", Proc. IRE, pp. 63-73, Jan. 1959.
12. E. F. Moore and C. E. Shannon, "Reliable circuits using less reliable relays", Bell Telephone System Technical Publications, Monograph 2696, January 1957.
13. C. E. Shannon, "Programming a computer for playing chess", Phil. Magn., 41, p. 256, March 1950.
14. A. Newell, J. C. Shaw, and H. A. Simon, "Chess-playing programs and the problem of complexity", IBM Journal of Res. and Dev., 2, p. 320, Oct. 1958.
15. A. L. Samuel, "Some studies in machine learning using the game of checkers", IBM Journal of Res. and Dev., Vol. 3, No. 3, p. 211-229, July 1959.

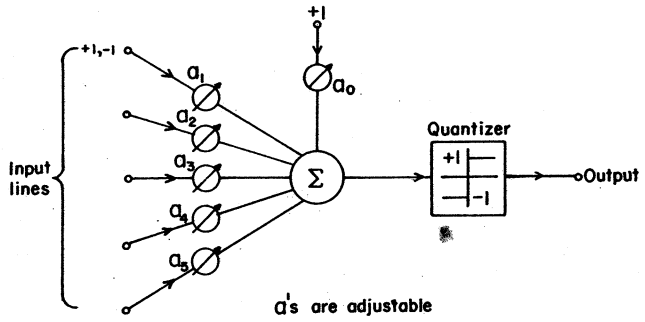


Fig. 1. An adjustable neuron.

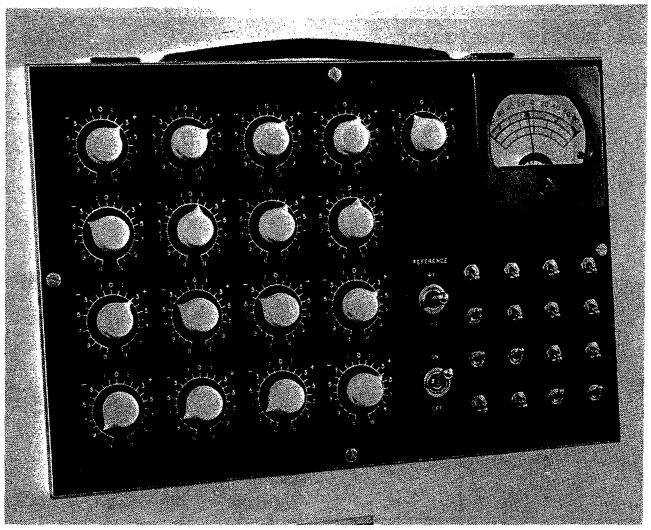


Fig. 2. Adaline.

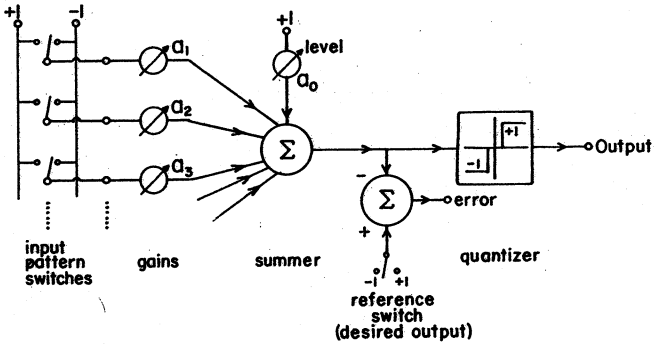


Fig. 3. Schematic of Adaline.

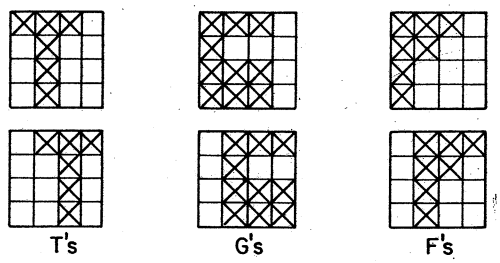


Fig. 4. Patterns for classification experiment.

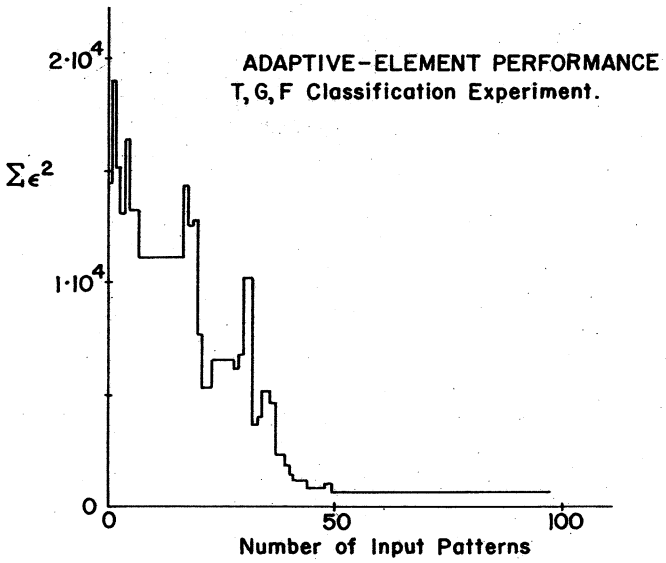


Fig. 5. Adaptive-element performance curve.

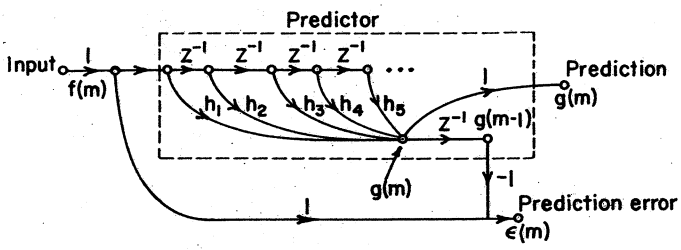


Fig. 6. An adjustable sampled-data predictor.

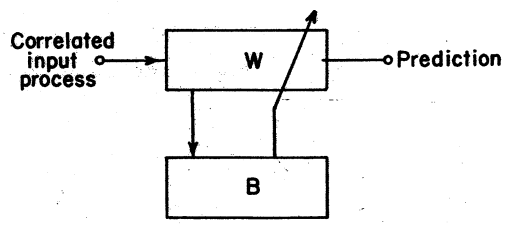


Fig. 7. An adaptive predictor.

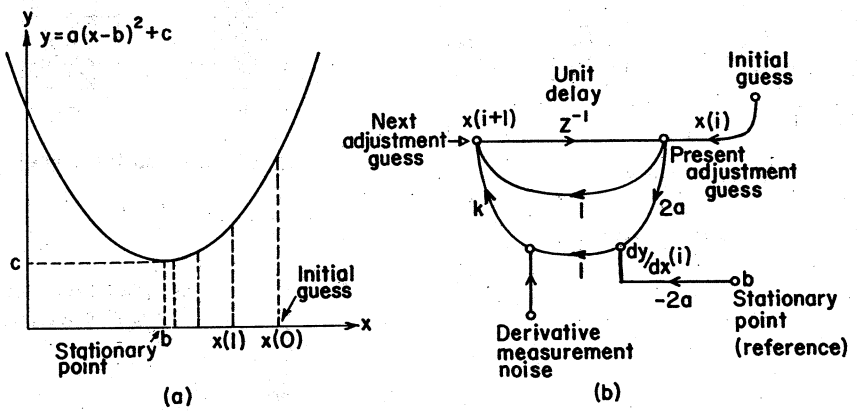


Fig. 8. One-dimensional surface searching.

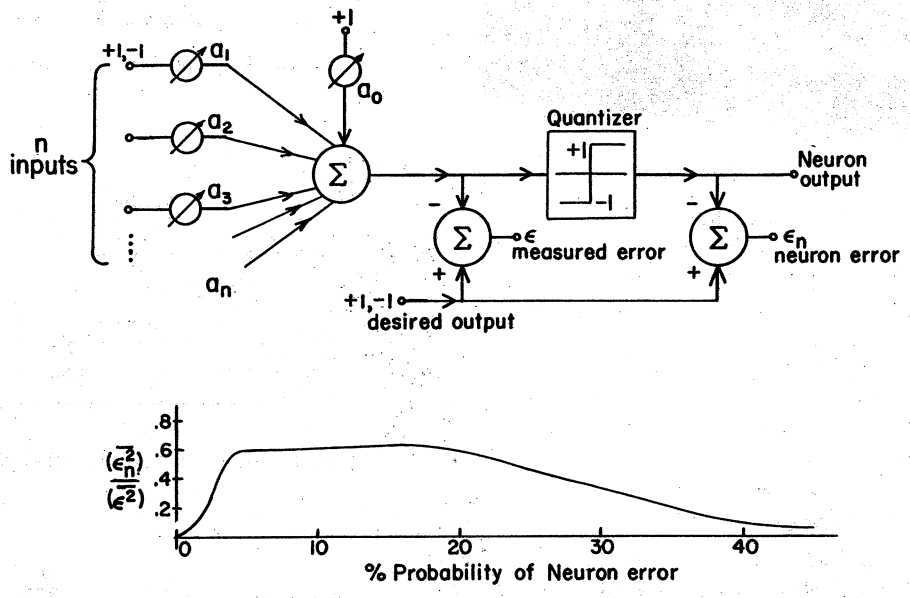
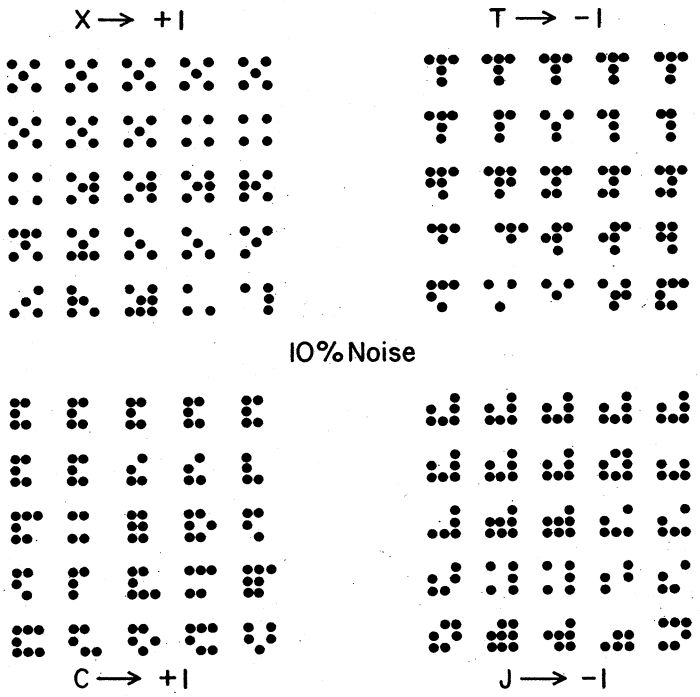


Fig. 9. Relations between neuron errors and measured errors.



Best neuron makes 12 errors out of 100

EXPERIMENT #	PATTERNS ADAPTED ON	NUMBER OF ERRORS	MISADJUSTMENT
1	95, 79, 07, 60 73, 61, 08, 02, 72, 26	25	$M = \frac{25-12}{12} = 108\%$
2	70, 69, 52, 55, 32 97, 30, 38, 87, 01	19	$M = \frac{19-12}{12} = 58\%$
3	65, 12, 84, 83, 34 38, 71, 66, 13, 80	20	$M = \frac{20-12}{12} = 67\%$
4	07, 42, 85, 88, 63 35, 37, 92, 79, 22	28	$M = \frac{28-12}{12} = 133\%$

Fig. 10. Experimental adaption on 10 noisy 3x3 patterns.

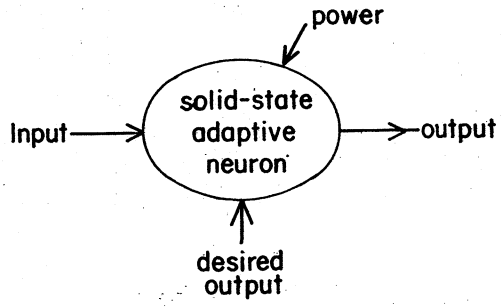


Fig. 11. Electronic automatically-adapted neuron.

Reprinted from:
1960 IRE WESCON CONVENTION RECORD
Part 4