Proceedings IFAC Symposium on "The Problems of Identification in Automatic Control Systems,"
Section 6 Special Identification Instruments, Prague June 12-19, 1967

Techniques of Identification with the Stochastic Computer

Brian R. Gaines
Standard Telecommunication Laboratories
London Rd., Harlow, Essex

Summary A major obstacle to the practical application of advanced techniques for process identification is lack of suitable hardware. By its very nature an identification computer must be able to store and adjust large numbers of variable parameters, and to use these for purposes of prediction and control. Conventional analog and digital computers both have disadvantages in this application, and there is a need for hardware specifically designed for identification purposes, economical in cost, reliable and drift-free like the digital computer, and capable of parallel operation to achieve the size-independent bandwidth of the analog computer. In particular this hardware should take full advantage of the advanced state of integrated circuit technology.

This paper outlines the principles and structure of the Stochastic Computer, and describes three identification techniques of increasing generality which take especial advantage of novel features in stochastic computing elements. The first technique is one of steepest descent to the best linear relationship between the inputs and outputs of the process to be identified—three conventional techniques, including polarity-coincidence correlation, are compared with three stochastic techniques. The second technique is one of statistical decision theory, in which Bayes Theorem is used to invert conditional probabilities and bring them to a form suitable for estimation and prediction. The final technique is one of Markov modelling of the state-class transitions of the process.

In conclusion it is suggested that advances in integrated circuit technology mean that feasible control practice will, in a few years time, have advanced beyond control theory—at least as it stands at present.

Introduction

A major obstacle to the practical application of advanced techniques for process identification is the lack of suitable hardware. By its very nature an identification computer must be able to store and adjust large numbers of variable parameters. In the analog computer this requires analog multipliers and low-leakage integrators, both liable to be costly and inaccurate. In the digital computer repeated multiplications and updating of stores during each sampling interval restrict the available bandwidth in the identification of complex systems. There is a need for hardware specifically designed for identification purposes, economical in cost, reliable and drift-free like the digital computer, and capable of parallel operation to achieve the size independent bandwidth of the analog computer. This hardware should take every possible advantage of the advanced state of integrated-circuit technology, so that sophisticated identification computers may be realized in a small size with high speed and reliability at low cost.

This paper describes techniques of identification with Stochastic Computing hardware developed as part of a program of research on the structure, realization and application of advanced automatic controllers in the form of Learning Machines [1, 2]. The Stochastic Computer [3, 4] and its derivative, the Phase Computer [5], have been described elsewhere, and only a brief introduction to the hardware is given here. Three particular identification techniques of increasing generality are then described, which take especial advantage of novel features in stochastic computing elements.

The first technique is one of steepest descent to the best linear relationship between the inputs and outputs of the process to be identified. The theory underlying this technique is already well-known, and it has been applied to many practical systems with continuous inputs and outputs. Particular attention is paid to the problem of identifying low bandwidth parameters of high bandwidth systems; to the relationship between polarity-coincidence techniques and stochastic equivalents; and to the convergence of stochastic adaptive threshold logic.

The second technique is applicable to systems with arbitrary inputs and outputs which do not necessarily lie on continua, and utilizes maximum likelihood prediction based on Bayes inversion of conditional probabilities. This has been suggested for identification in some learning controllers, but never implemented in practice because of its computational requirements. The use of stochastic computing elements enables the direct estimation of normalized likelihood ration, in a form suitable for prediction.

The third technique makes even less assumptions about the nature of the process to be identified, and is based on a Markovian model of the state-transitions of the plant. Its application has been suggested in the neighbourhood of optimal trajectories where accurate identification is especially important.

Stochastic Computing

Computations within the stochastic computer are carried out using standard logic-elements, gates, flip-flops etc., but with data represented in an unconventional form as the generating probabilities of Bernoulli sequences of logic levels. Thus a quantity in the stochastic computer is represented by a binary sequence of ON and OFF logic levels, generated by a random process such that successive levels are statistically independent, and the probability of the logic level being ON is a measure of that quantity. Because a probability cannot be measured exactly but only estimated as the relative frequency of ON logic levels in a sufficiently long sample, the information representation in the stochastic computer is generally less efficient than that in other forms of computer. For example, to represent a quantity to an accuracy of one part in N:

the analog computer requires one continuous level;

the general purpose digital computer requires $\log_2 kN$ ordered binary levels;

the pulse-counting computer (e.g. DDA) requires kN unordered binary levels;

and the stochastic computer requires kN^2 unordered binary levels;

where $k > 1$ is a constant representing the effects of round-off error or variance. The N^2 term for the stochastic computer arises in the estimation of a generating probability, where the expected error decreases as the square root of the length of sequence sampled.

Although this progression from $1 < \log_2 N < N < N^2$ shows the stochastic computer to be the least efficient in its representation of quantity, it is the only digitally-based computer to have a strictly local representation of data in which the logic levels in a sequence are independent, and through this it gains in simplicity and economy of computation. In practice also, pseudo-stochastic techniques may be used, such as those of the phase-computer [5], with little loss of simplicity and a major gain in efficiency and bandwidth.

In the next sections stochastic representations are first described, and then the range of stochastic computing hardware for multiplication, addition, integration, control of analog variables, conversion, estimation, and so on is described.

Stochastic Representations

In some applications of the Stochastic Computer, such as the Bayesian and Markovian estimators described later, one is dealing with the estimation of probabilities of external events, and the [0,1] range of generating probabilities in the computer itself has a natural interpretation. Generally, however, it is necessary to map some other variable into this range, in the same way that one maps quantities into the range of voltages of an analog computer. Many mappings are possible, but those most relevant to the present discussion are:

(1) **Asymmetric Binary** Given a quantity E in the range, $0 \leq E \leq V$, represent it by the generating probability

$$p(ON) = E/V \tag{1}$$

i.e. the magnitude of a positive bounded quantity is represented by the logic level always ON for maximum quantity, always OFF for zero quantity, and fluctuating randomly for intermediate quantities.

(2) **Symmetric Ternary** The above representation may be extended to bipolar quantities by representing the sign of the quantity as a logic level on one line, and the magnitude stochastically as above. This may be transformed to an equivalent but preferred arrangement in which positive sign and ON magnitude correspond to the UP line being ON, whilst negative sign and ON magnitude correspond to the DOWN line being ON. In this case for a quantity E, such that $V \leq E \leq V$, we have:

$$E/V = p(UP= ON) - p(DOWN= ON) \tag{2}$$

i.e. maximum positive quantity is represented by the UP line always ON, maximum negative quantity by the DOWN line always ON, and zero quantity by both lines always OFF (or possibly by equal non-zero probabilities that the UP line and DOWN line will be ON).

(3) **Symmetric Binary** Alternatively a bipolar quantity may be represented on a single line by the mapping.

$$p(ON) = E/2V + 1/2 \tag{3}$$

i.e. maximum positive quantity is represented by a logic level always ON, maximum negative quantity by it always OFF, and zero quantity by a logic level fluctuating randomly with equal probability of being ON or OFF. This last property is a disadvantage of representation (3) compared with (2) since zero quantity is represented with maximum variance in (3) but zero variance in (2)—hence the latter is to be preferred when quantities requiring accurate representation of small values, such as ‘satisfaction error’, are to be represented.

Stochastic Computing Elements

Invertors To multiply a quantity in representation (2) by -1 requires only the interchange of UP and DOWN lines. In representation (3) a simple logical invertor whose output is OFF when its input is ON performs arithmetic inversion.

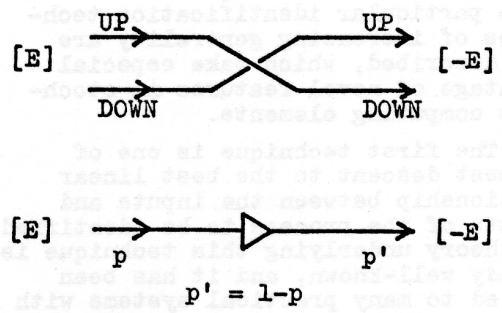


Figure 1 Inversion

Multipliers To multiply together two quantities in representation (1) a simple AND gate suffices as shown in Figure 2(i). In representation (2) it is required that the UP output be ON when both UP inputs are ON, or when both DOWN inputs are ON, and that the DOWN output should be ON when one UP input and one DOWN input is ON. This may be realized by the gating shown in Figure 2(ii). In representation (3) an inverted exclusive OR gate may be used, whose output is ON when its inputs are equal.

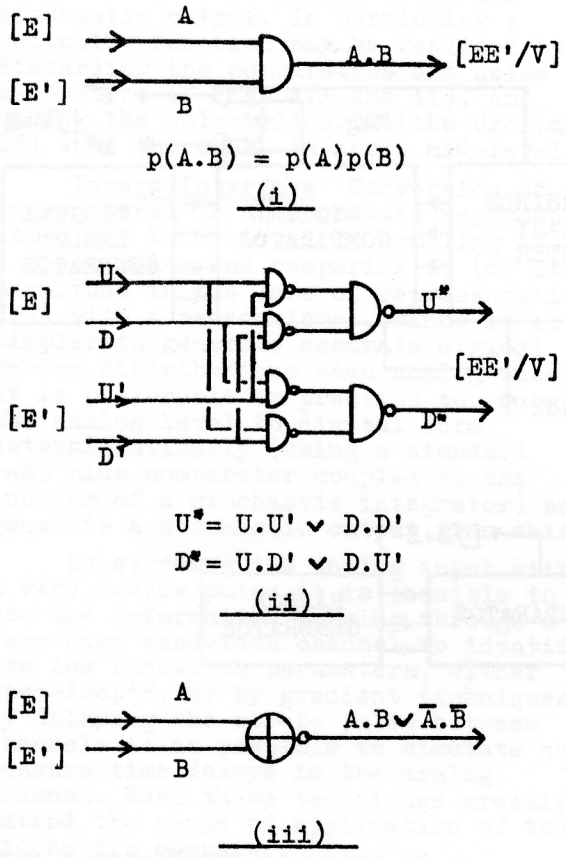


Figure 2 Multiplication

Summers Addition of the quantities represented on a number of lines in the stochastic computer may be effected by switching the output line at random to one of the input lines. For example, if there are two input lines and the output is switched to one, representing the quantity E, with probability λ, and hence to the other, representing E' with probability $1-\lambda$, then the output will

represent the quantity λE + (1-λ)E'. A two-input stochastic summer for representations (1) and (3) is shown in Figure 3(i)—Flip-Flop$_1$ is in a random state which is transferred to Flip-Flop$_2$ at a clock-pulse; dependent on the state of Flip-Flop$_2$ one or other of the input lines is reproduced at the output.

It is advantageous to use additional gating to minimize the variance of the output in representation (3), and this is shown in Figure 3(ii).

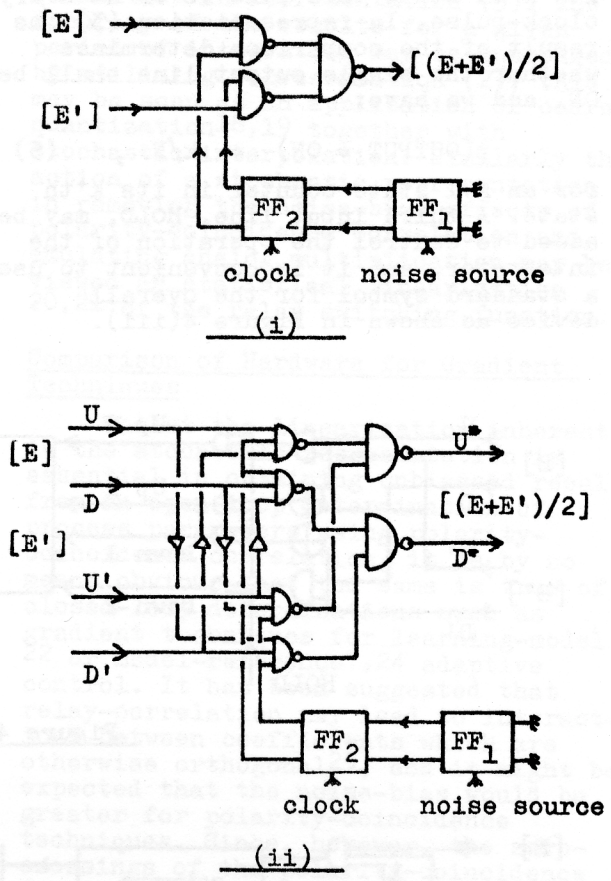


Figure 3 Summation

Integrators A reversible binary counter may be used as integrator in the stochastic computer. For quantities in representation (2) the UP and DOWN lines may be attached to the equivalent lines on the counter to obtain a single-input integrator. A two-input summing integrator may be obtained by feeding both the first and second stages of the counter to cause it to count by one or by two, using the input logic shown in Figure 4(i). For quantities in representation (3) a two-input summing integrator is naturally obtained using the logic of Figure 4(ii).

If the counter has N+1 possible states then the value of the integral when it is its k'th state is:

$$\int = V(2i / N - 1) \tag{4}$$

If this quantity is to be used in further computations it must be made available in a stochastic representation and this may be achieved by comparing it with a uniformly-distributed, digital random number (obtained from a pseudorandom shift-register or a sampled cycling counter). In

representation (2) the count is regarded as a signed number in twos-complement form, and its sign bit determines which output line is operatives whilst the result of the comparison determines whether this line is ON. Hence if the counter is above its mid-count only its UP output line may come ON, and when the counter is in its N'th state this line is ON at every clock-pulse. In representation (3) the result of the comparison determines whether the single output line shall be ON, and we have:

$$p(\text{OUTPUT} = \text{ON}) = k/N \tag{5}$$

for an N+1 state counter in its k'th state. A third input line, HOLD, may be added to control the operation of the integrator, and it is convenient to use a standard symbol for the overall device as shown in Figure 4(iii).

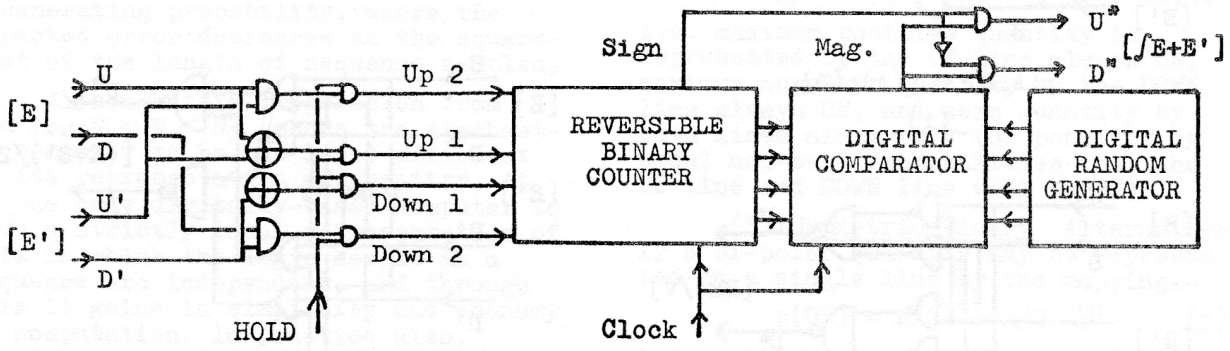


Figure 4(i)

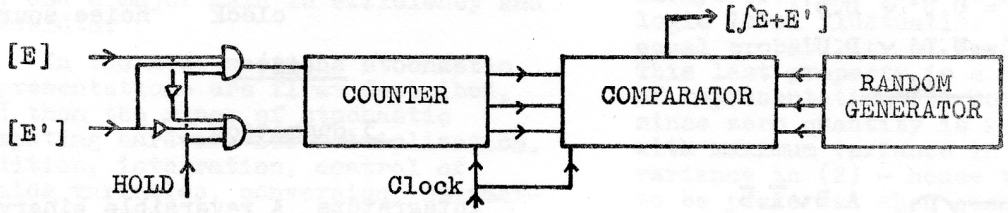
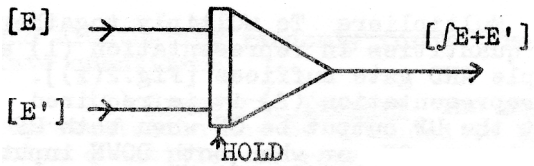
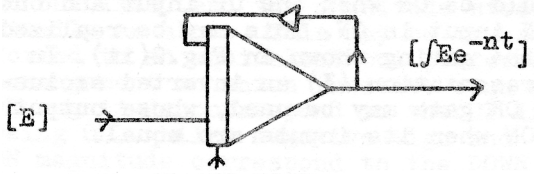


Figure 4(ii)



4(iii) Symbol



4(iv) ADDIE

Figure 4 ADDIE

Outward Interface A stochastic integrator with negative feedback to one of its inputs acts as a smoothing or estimating element (Fig.4(iv), the 'ADDIE'). In terms of the representation it may be regarded as a transfer function:

$$1/(s+1).$$

In terms of the generating probabilities it may be shown that, for the configuration of Figure 4(ii) connected as in Figure 4(iv), the fractional count in the store tends to an unbiased estimate of the probability that the input line will be ON at a clock pulse. That is, for an N+1 state store in its k'th state:

$$p(\text{INPUT=ON}) = k/N \tag{6}$$

with a time-constant of order N clock pulses, and a final variance:

$$\sigma^2(p) = p(1-p)/N \tag{7}$$

Hence any quantity represented linearly by a probability in the stochastic computer may be read out to any required accuracy by using an ADDIE with a sufficient number of states, but the more states the longer the time constant of smoothing and the lower the bandwidth of the computer. Hence variables within the computer may be regarded as degraded by Gaussian noise, whose power increases in proportion to the bandwidth required from the computer.

Since the counter has a parallel digital output, it forms the natural outward interface of the stochastic computer. In many applications this output is used to control the analog variables in a hybrid configuration. This may be done precisely by having the digital output control the switched summing resistors of an operational amplifier, but this leads to transients in the analog channel which decrease its available bandwidth. In many applications, such as model-reference adaptive control [6, 7] and adaptive line equalization, [8, 9] an exact linear relationship between the digital output and the analog multiplier is unnecessary, because the multiplier is within the adaptive loop; only a monotonic relationship without wide variation in slope is required. This may be achieved by using the digital output to control a photo-emissive source driving a photo-conductive input resistor to an amplifiers by this means low-bandwidth digital control of very high bandwidth analog channels is economically performed.

The integrator or ADDIE may be used to generate arbitrary functions by imposing suitable nonlinear relationships between the stored count and stochastic output. In particular, a switching function may be realized by discarding the comparators and noise generators of Figures 4(i and ii), and having the output line ON (the UP line ON) when the count is above mid-level.

Inward Interface Conversion of an analog level to a stochastic representation may be achieved by sampling it at a clock pulse and comparing it (or its magnitude in the case of representation (2)) with a random level. Since it is simpler to generate accurate digital random distributions than analog ones, it is preferable in practice to convert the analog level to digital form deterministically (using a standard ramp plus comparator coupled to the counter of a stochastic integrator) and generate a stochastic output from this.

By strobing the analog input with a very narrow pulse it is possible to use the information passing through a very high bandwidth channel to identify its low bandwidth parameters, either open-loop [10] or by gradient techniques. By delaying the strobe pulse between channels it is possible to simulate and measure time delays in the analog channel. Both these techniques greatly extend the range of application of the stochastic computer.

Gradient Techniques

Before considering the realization of gradient techniques in the stochastic computer, it is of interest to discuss the alternative technique of open-loop identification using cross-correlation of input and output to obtain the impulse response. There have been many attempts to simplify and reduce the cost of correlators by replacing the analog multipliers with gates or relays, using digital [11, 12] or mixed analog/digital [13, 14] computation. Under low noise conditions with Gaussian signals, these techniques may be shown to give results having a 1 to 1 correspondence with the true correlation function [15]. However, in the presence of noise or with strongly asymmetric distributions this correspondence no longer holds, and the structure of the true response is lost.

Addition of appropriate random signals to the correlator inputs has been suggested as a means of overcoming these defects, [16, 17] and leads to a structure identical to that of a stochastic multiplier and integrator for quantities in representation (3). The accuracy of the results for a given period of integration may be increased by utilizing representation (2); this may be seen as an application of coarse quantization [18, 19] together with stochastic interpolation. Similarly the action of a stochastic representation in removing the noise-bias effects of polarity-coincidence correlation and emulating analog multiplication may be viewed as statistical linearization [20, 21] of the relay switching function.

Comparison of Hardware for Gradient Techniques

Whilst the linearization inherent in the stochastic representation is essential in obtaining unbiased results from an open-loop determination of process parameters using polarity coincidence correlation, it is by no means obvious that the same is true of closed-loop determinations such as gradient techniques for learning-model [22] or model-reference [7, 24] adaptive control. It has been suggested that relay-correlation may lead to interactions between coefficients which are otherwise orthogonal [25], and it might be expected that the noise-bias would be greater for polarity-coincidence techniques. Since, however, the zero-crossings of the polarity-coincidence function and the true correlation function are the same for Gaussian inputs [15], even in the presence of noise, the noise-bias of null-seeking gradient techniques should be virtually independent of the method of correlation used.

Because of the practical importance of gradient techniques in both communication and control, it would seem worthwhile to devote some effort to a detailed comparison of the performance of various forms of hardware. The first stages of such a study and the initial results obtained are outlined in the next section.

Parameter Determination The problem selected for the initial study was the determination of the parameters of the first-order transfer-function:

$$1/(a_0 + a_1 s),$$

given the input/output waveforms of the noise-excited systems. This is an example of the general problem of finding the best (in some sense) weights, $\{w_i\}$, to approximate the output, Z, of some system by a linear combination of its inputs $\{X_i\}$. The error in satisfying the system relationship, E:

$$E = \sum w_i X_i - Z \tag{8}$$

is a function both of the estimated weights and the system state, and some functional of E is to be minimized.

Six techniques for varying the weights have been investigated:

1. Steepest Descent

$$\dot{w} = -\alpha E X_i \tag{9}$$

requiring analog multipliers for both the weights and their adaption.

2. Relay Correlation

$$\dot{w} = -\alpha \operatorname{sgn}(E) X_i \tag{10}$$

requiring analog multipliers (which may be simple motorized potentiometers) for the weights only.

3. Stochastic Relay Correlation

$$\dot{w} = -\alpha \operatorname{sgn}(E + \gamma) X_i \tag{11}$$

where γ is a random variable uniformly and symmetrically distributed over the range of E; this effectively linearizes the relay.

4. Polarity-Coincidence

$$\dot{w} = -\alpha \operatorname{sgn}(E) \operatorname{sgn}(X_i) \tag{12}$$

where w_i now takes discrete values and the integration is performed by a counter, so that no analog multipliers are required.

5. Stochastic Ternary

$$\dot{w} = -\alpha \operatorname{sgn}(E) \operatorname{sgn}(X_i)(1 + \operatorname{sgn}(|E| - \gamma))(1 + \operatorname{sgn}(|X_i| - \delta))/4 \tag{13}$$

this may be regarded as polarity-coincidence correlation with stochastic weighting according to the magnitude of the inputs (γ and δ are random variables uniformly distributed in the range of magnitudes of E and X_i respectively).

6. Stochastic Binary

$$\dot{w} = -\alpha \operatorname{sgn}(E + \gamma) \operatorname{sgn}(X_i + \delta) \tag{14}$$

this may be regarded as polarity-coincidence correlation with statistical linearization of the comparators (γ and δ are random variables symmetrically and uniformly distributed in the ranges of E and X_i respectively).

The computing arrangement for this last technique is shown in Figure 5 (one channel). If the 'high-frequency sawtooth' amplitude is reduced to zero this becomes an implementation of polarity-coincidence correlation (Equation 12). For intermediate values of the sawtooth amplitude the multiplication is proportional for small values of E and X_i, and bang-bang for large values—the partial linearization gives rise to 'dual-mode' operation.

Experimental Study There are four figures of merit which have to be determined for the various techniques:

(a) **Speed of Response** How rapidly are the weights adjusted in response to a step change. This is not simple to define since the equivalent gain of a relay is a function of the input amplitude. Hence, when E becomes large following a change in the system parameters, the speed of response of methods 1, 3, 5 and 6 is greater than that of methods 2 and 3. As the estimates converge and E becomes small, however, the speed of the first group decreases whilst that of the second pair remains constant. This cross-over shows clearly in the graph of Figure 6, comparing methods 1 and 2 in estimating a_0 with noise-free data.

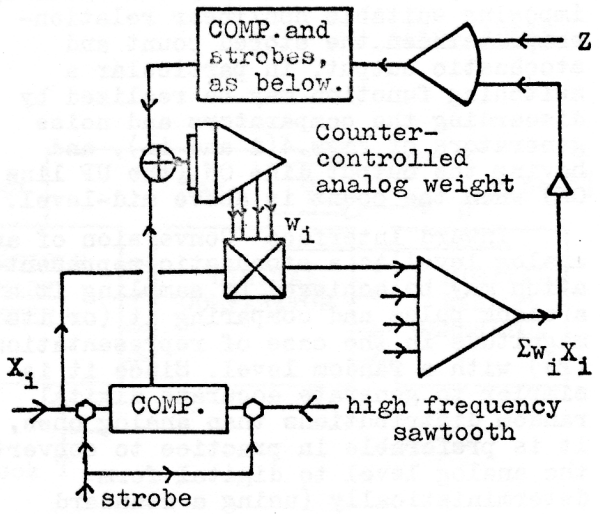


Figure 5 Stochastic Descent

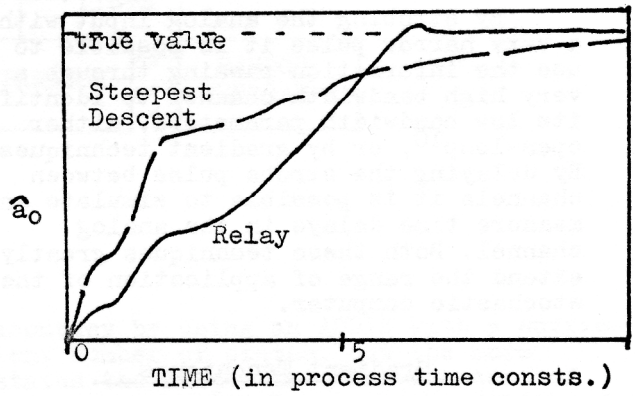


Figure 6 Speeds of Response

(b) **Bias Due to Noise on Data** All the techniques will underestimate the magnitude of the weights when uncorrelated noise is added to Z and each of the $\{X_i\}$ For the reason stated earlier, it was expected that this bias would not be greater for methods 3 and 4 than for the others, but this was subject to check under realistic conditions.

(c) **Interaction Between Weights** In order that a step-change in one parameter shall not cause transient changes in the estimates of the others, it is necessary to chose the $\{X_i\}$ to be orthogonal functions (relative to the functionals implied by equations (9) to (14)). a_0 and a_1 do multiply orthogonal functions with respect to methods 1,3,5 and 6, but it is not obvious that they remain orthogonal for 2 and 4 (although again the results of Reference 15 would suggest that they do under reasonable conditions).

(d) **Variance in the Final Estimates** Under noise-free conditions E tends to zero and the estimates reach fixed, stable values (apart from the one-unit relay 'chatter' in methods 2 and 4, and the stochastic variance in methods 3 and 6). Noise on the $\{X_i\}$, however, induces variance in the [wi] which increases with the gains, a, of the various methods. If the speeds of response are matched (in some sense) then the gain-varying methods (2 and 4) would be expected to have more variance in their estimates, as would also the stochastic binary techniques (3 and 6).

Results The gain of each method was adjusted so as to equalize their speeds of response (time taken to traverse 0.9 of the step) under high-noise conditions (uniformly-distributed noise, 0.2 of signal). The noise-bias in the estimate was measured under these conditions, together with the variance of the final estimate. The interaction between coefficients was measured under noise-free conditions by examining the effect of a step in one parameter on the estimate of the other. The overall time of convergence was set to be about 5 process time-constants.

It was found that:

The noise-bias was the same in every case (estimate about 0.8 of true value).

The interaction was the same in every case.

The variances of the final estimates were approximately in the ratios:

1.	Steepest Descent	1
2.	Relay Correlation	4
3.	Stochastic Relay Correlator	2
4.	Polarity Coincidence	10
5.	Stochastic Ternary	1
6.	Stochastic Binary	2

Hence the high α required to give the relay and polarity-coincidence techniques a sufficient speed of response leads to excessive gain in the converged condition and poor noise-rejection.

Adaptive Threshold Logic

Addition of a threshold element at the output of a steepest descent configuration:

$$Y = \begin{cases} +1 & if \quad \sum w_i X_i \geq \theta \\ 0 & if \quad \left|\sum w_i X_i\right| < \theta \\ +1 & if \quad \sum w_i X_i \leq -\theta \end{cases} \tag{15}$$

and consideration of two-level inputs only ($X_i = \pm 1$ or 0,1), converts it to an 'adaptive threshold logic' element. These have been expensively studied as Perceptrons [26], Adalines [27], Learning Matrices [28], and so on [29], and are basic components of most learning-machines and adaptive pattern-recognizers. The standard convergence proof [30] for an adaptive threshold logic element, given a sufficient range of examples of an input/output relationship which it is capable of realizing, uses a *reductio ad absurdum* argument demanding that the weight magnitudes be potentially greater than the minimum necessary for a solution to exist. An

alternative mode of adaption is possible in which an a priori bound is fixed for the magnitude of the weights:

let U_i^J be a set of binary vectors such that there exists some vector η_i:

$\sum \eta_i U_i^J > \theta$, for all J.

Given a sequence of vectors $X_i(n)$, frequently containing only U_J for all J, recursively construct a sequence of weight vectors.

$$w_i(n+1) = \begin{array}{c} w_i(n) \quad if \quad \sum w_i X_i > 0 \\ \alpha w_i(n) + (1-\alpha) X_i(n) \quad otherwise \end{array} \qquad (16)$$

Then it may be shown that $w_i(n)$ converges to a vector w_i such that:

$$\sum w_i U_i^J > 0$$

provided (l-α) is small compared with θ^2.

This result depends however on the continuity of the weights, and when these are both discrete and bounded not only do the convergence proofs fail but also counter-examples may be given showing limit-cycle behaviour rather than a solution. For example the set of binary vectors

A (1, 1, 1, -1)
B (1,-1,-1, 1)
C (-1, 1,-1, 1)
D (-1,-1, 1, 1)

may be identified by the weight vector:

W (1, 1, 1, 2),

and hence weights with the possible values w_i = -2, -1, 0. +1, +2, should be capable of converging to a solution. Given the repetitive cycle A B C D A B C D ..... however a limit cycle is formed:

W(0)		0	0	0	0
W(1)	A	1	1	1	-1
W(2)	B	2	0	0	0
W(3)	C	1	1	-1	1
W(4)	D	0	0	0	2
W(5)	A	1	1	1	1
W(6)	B	2	0	0	2
W(7)	C	1	1	-1	2
W(8)	D	0	0	0	2
	A				

.....

Consider now the equivalent device utilizing stochastic computing elements to realize equation (16) on average by stochastic interpolation. We have:

$$\sum W_i X_i > 0 \tag{17}$$

and if

$$\sum W_i(n) X_i(n) \leq 0 \tag{18}$$

then

$$W_i(n+1) \quad = \quad W_i(n) + \varphi_i X_i(n) \tag{19}$$

where φ_i is a random vector, $\varphi_i = 0, 1$. Consider now:

$$\sum (W_i - W_i(n+1))^2 = \sum (W_i - W_i(n))^2 - 2\sum \varphi_i (W_i - W_i(n) + X_i(n)/2) X_i(n) \tag{20}$$

combining (17) and (18) we have that, for some i,

$$(W_i - W_i(n)) X_i(n) > 0 \tag{21}$$

and hence from equation (20) there is a non-zero probability that the distance between the vectors at W_i and $W_i(n)$ will decrease. Thus, if the weights do not form a solution, during any cycle of the inputs there is a non-zero probability that the weight vector will move nearer an arbitrary solution vector. Hence, since the weights have a finite set of possible values, the probability of convergence tends uniformly to 1 with the number of training cycles.

The net result is that a stochastic adaptive-threshold-logic element requires less redundant states than the equivalent deterministic device.

Bayes Predictor for Binary Inputs

Statistical inference based on Bayes' 'Theorem' is fundamental to modern probability theory [31, 32, 33] and devices to implement various predictors based on the Theorem have been put forward many times [34, 35, 36] as basic components of learning machines.

Consider some event E whose occurrence (designated e(n)=1) is dependent upon which members of a set of events $\{E_i\}$ have occurred. Let the binary vector $[e_i]$ be such that $e_i=1$ if the event E. has occurred and $e_i=0$ otherwise. Then the probability that e(n) =1, given [ei(n)], is to be used to make the best prediction of E. The simplest utilization is that of maximum likelihood prediction, in which E is predicted to occur if:

$$p(e(n) = 1 \,|\, [e_i(n)]) > 0.5 \tag{22}$$

but more complex procedures may be used if further information is available, e.g. if the costs of predicting incorrectly one way or the other are known.

To estimate the conditional probability of E occurring for each possible combination of events from $\{E_i\}$ would not only require great storage capacity but also much experience, and in practice it is necessary to make certain simplifications which are equivalent to having the machine generalize from its experience. An estimating and predicting scheme based on Bayes Theorem together with certain assumptions about the nature of the events E and $\{E_i\}$ is developed in the next section.

Estimation and Prediction

Consider the likelihood ratio:

$$L(n) = p(e(n) = 1 \mid [e_i(n)]) / p(e(n) = 0 \mid [e_i(n)]) \tag{23}$$

By Bayes' Theorem this can be re-written:

$$L(n) = \frac{p([e_i(n)] \mid e(n) = 1)\, p(e(n) = 1)}{p([e_i(n)] \mid e(n) = 0)\, p(e(n) = 0)} \tag{24}$$

Now assume that the components, $e_i(n)$, are statistically independent under the occurrence of E, so that:

$$p([e_i(n)] \mid e(n) = 1) = \prod_i p(e_i(n) \mid e(n) = 1) \tag{25}$$

(where $\prod_i$ means 'the product for all i'),

and apply Bayes Theorem to the expanded terms:

$$L = \frac{p(e = 1)}{p(e = 0)} = \prod_i \frac{p(e = 1 \mid e_i)\, p(e = 0)}{p(e = 0 \mid e_i)\, p(e = 1)} \tag{26}$$

(where the argument 'n' has been dropped for clarity), which may be written:

$$L = L_0 \pi_i L_i \tag{27}$$

using an obvious notation.

Equation (27) implies that the quantity required in prediction, L, is the product of L_0 and the normalized likelihood ratios, L_i, of those events E. which have occurred. It is customary tb take logarithms, turning the product into a sum for easier computation; the quantity $\ln(L_i)$ is in fact the information in the event E_i relative to the event E [23]. Estimation of L_i, $\ln(L_i)$ or even the unormalized likelihood ratios is difficult, however, because they are nonlinear and unbounded functions; in practice no compatible estimation and prediction scheme using simple computing elements has been developed for the Bayes Predictor.

Stochastic Realization of Bayes Predictor

Estimation of the L and prediction based on them is achieved quite simply using stochastic computing elements. To each event E there corresponds a stochastic integrator estimating a probability p_i such that:

$$\frac{p_i}{(1 - p_i)} = L_i \tag{28}$$

A single integrator connected to the outputs of these implements equation (27) for prediction.

The integrators shown in Figure 7 are minor extensions of those in Figure 4(ii) in that they have multiple inputs, the counter incrementing only when they are all ON and decrementing when they are all OFF. The integrator estimating p_i receives inputs from a line which is ON when E has occurred; from the inverted output of the L_0 integrator estimating p(e=l); and from its own inverted output. Its HOLD line is ON only when the Estimate line is ON and E_i has occurred. The outputs from the integrators are gated according to the E_i which have occurred and feed an integrator which may be connected as a switching function if maximum likelihood prediction alone is required. Its output during prediction tends either to an estimate of the conditional

probability of E given the E_i which have occurred, or, in the case of a switching function, to the predicted E itself.

Thus the stochastic computer lends itself to a simple and economical realization of a Bayes Predictor for binary information.

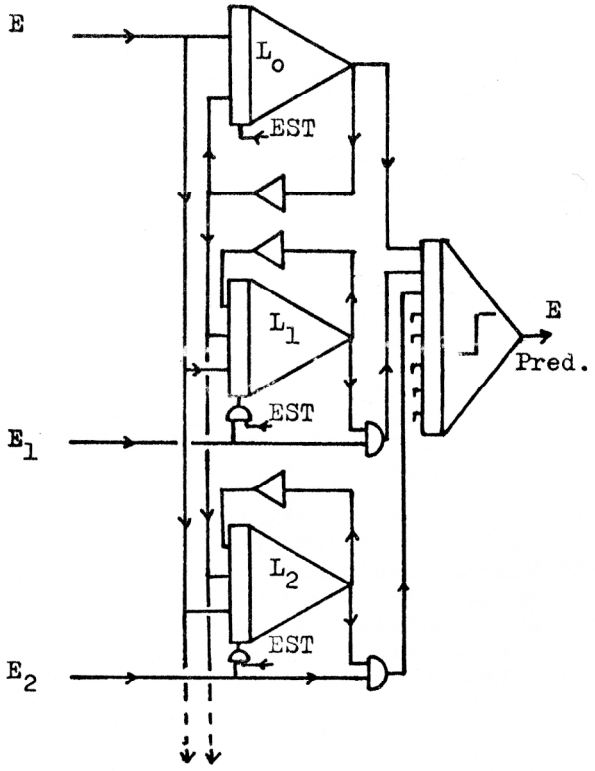


Figure 7 Bayes Estimator and Predictor

Markov Modelling

It has become conventional to introduce the abstract concept of a process through its state-transitions [37] rather than through the differential or difference equations which describe particular examples. To identify a real process on this basis, however, is not feasible, because the number of states is generally very large, the number of potential transitions very much larger, and hence both the storage capacity and time for identification required become astronomical. In practice it is necessary to make topological assumptions, such as continuity or linearity, which correspond to the generalization of experience of members of a class of transitions to all members of that class.

A Markov model of transitions between state-classes, however, may be made the basis of a very powerful and general identification technique, particularly if adaptive classification and selection of state-classes relevant to the control problem are also utilized. The computations involved are slow in the conventional digital computer, because they involve the repeated multiplication of large matrices. With stochastic computing elements, however, it is possible to establish a direct, parallel simulation of the state-transitions.

Consider the set of states $\{S_i\}$ (state-classes, policy-elements), and let the probability of a transition from S_i to S_j be p_{ij}. We may assume that:

$$\sum_j p_{ij} = 1 \tag{29}$$

by taking S_0 to be the class of states not included in $\{S_i\}$; p_{i0} is then the probability of going out of the state space considered, and p_{0i} the probability of coming into it at S_i from outside. Note that the transitions need not be first-order Markovian, but are assumed to be so for purposes of identification; the transition probabilities may appear to vary if this assumption does not hold.

The transition probabilities are estimated using the principle of the ADDIE (Fig.4(iv)), with one stochastic integrator for each p_{ij}. Prediction consists of direct simulation of the state-transitions, and a slight modification of the integrator of Figure 4(ii) is required since a transition from S_i can occur to one and only one state S_j. All the integrators for transitions from S_i form a single unit containing a counted for each S_i and a single random generator. The feedback from the output is built-in, and each unit has a set of inputs corresponding to transitions to S_j occurring, and a set of outputs corresponding to transitions to S_j being predicted.

In estimation, if a transition occurs from S_i to S_j, then the ESTimate (HOLD) line of the i'th integrator, together with its j'th input, are turned ON. In prediction, a flip-flop attached to each integrator unit determines which of the S_i is the current state. At a clock-pulse the output of this unit determines which flip-flop will be ON corresponding to the next state. By setting the i'th flip-flop ON and running the process through N clock-pulses several times, it is possible to estimate the probability of a transition from S_i to S_k within N steps (by counting the number of times that the k'th flip-flop is ON). Similarly it is possible to estimate the average path-length from S_i to S_k, and any other parameters or cost-functions of the process which may be required.

Conclusions

It is a common complaint that control theory has developed far beyond control practice. In fact the conceptual gap is accentuated out of all proportion to its true significance by the tremendous rate of growth of both control theory and electrotechnology—a temporal gap of even two years may make all the difference between technical failure and technical success.

Eight years ago we would have built a 'learning machine' with vacuum tubes and relays, and it would have been massive, unreliable and useless. Four years ago we would have built it with discrete semiconductor devices and it would have served as a demonstration of principle—the cost and size of a powerful machine would have been prohibitive. Today we have available integrated circuits containing several basic functional elements, such as flip-flops, on a single chip. In four years time we shall have complex arrays containing hundreds of gates and flip-flops interconnected as a complete system. The difficulty for the circuit manufacturer is not only in making these elements, but also in knowing what they should be. The requirements will come from the users, and control engineers may be expected to form a high proportion of these.

There are some general principles in the utilization of these new components which greatly affect the type of control system that may be realized: capacitors have a bulk far greater than equivalent digital storage elements and hence analog storage is wasteful and inefficient—on the other hand very high quality, low-drift, high-bandwidth amplifiers are available and their gains may be digitally controlled, the connections to a chip are its least reliable and most expensive feature and

should be minimized—equally cross-connections between the small, complex systems realizable in a single can are a major factor in the cost of fabricating a controller and must also be minimized both in number and in length—this implies that the circuit elements should be complete subsystems, and that data-transmission between them should be along a single wires the cost of the circuit elements is largely dependent on the number manufactured, and hence as few different types as possible should be used.

The advances in integrated circuit technology mean that feasible control practice will, in a few years time, have advanced beyond control theory—at least as it stands at present. It has been the objective of this paper to demonstrate that the gap between control theory and control technology is closing, and may even be expected to open in the opposite direction.

Acknowledgments

This work was done in close association with Dr. J.H. Andreae, (now at the University of Canterbury, N.Z.) and I wish to thank him for support, encouragement and ideas. My thanks are due also to S.T.L. for permission to publish this material.

References

l. Andreae, Gaines: 'A Learning Machine in the Context of the General Control Problem' IFAC Congress, 1966

2. Andreae: 'Learning Machines' Encyclopaedia of Information, Linguistics and Control, Pergamon, 1967

3. Gaines: 'Stochastic Computers' as (2)

4. Gaines: 'Stochastic Computing' AFIPS SJCC, 1967

5. Gaines: 'Application of Phase Computers to Automatic Tracking' IEE Conf. ATC 1967

6. Stear, Gregory: 'Capabilities and Limitations of Adaptive Technology' IFAC 1962, Rome.

7. Donalson, Kishi. in Modern Control System Theory Ed. Leondes, McGraw-Hill 1965

8. Lucky: 'Techniques of Adaptive Equalization' B.S.T.J., Feb 1966, 255-286

9. Kailaith: 'Optimum Receivers' in Information Theory Ed. Cherry Butterworth 1961

10. Cooper, *et al*: 'Wideband Digital Correlator' National Electronics Conference 1965

11. Becker : 'Two-Level Correlation' IRE Trans. EC10(4) 1961

12. Lundh :'A Digital Integrator' IEEE Trans. EC12(1) 1963

13. Korn, Korn: 'Analog and Hybrid Computers' 518-520, McGraw Hill 1965

14. Potts et al: 'Automatic Determination of Human and Systm Parameters' WJCC 1961

15. Veltman, Bos: 'Relay Correlator and Polarity Coincidence Correlator' IFAC Congr. 1963

16. Korn: 'Random-Process Simulation and Measurements' 136-139 McGraw Hill 1966

17. Jespers *et al*: 'Method to Compute Correlation Functions' Int. Symp. Inf. Th. 1962

18. Watts: ‘General Theory Amplitude Quantization’ IEE Mono.481M Nov. 1961
19. Widrow: ‘Study Rough Amplitude Quantization’ IRE Trans. CT3 1956
20. Nath, Mahalnabis: ‘Method for Statistical Linearization’ Proc. IEE 113(12) 1966
21. Pervozanskii: ‘Random Processes in Nonlinear Control’ Academic Press 1965
22. Margolis, Leondes: ‘Theory of Adaptive Control Systems’ IFAC Congr. 1960
23. Kullback: ‘Information Theory and Statistics’ Wiley 1959
24. Donalson, Leondes: ‘A Model Referenced Parameter Tracking Technique’ IEEET AppInd 68 1963
25. Young; ‘Determination of the Parameters of a Dynamic Process’ Rad. Electron. Eng. 29(6) 1965
26. Rosenblattt in Computer and Information Sciences Ed. Tou, Wilcox Spartan 1964
27. Widrow, Smith: as (26)
28. Steinbuch, Piske: ‘Learning Matrices and Applications’ IEEET EC12(5) 1963
29. Nilsson: ‘Learning Machines’ McGraw Hill 1965
30. Novikoff: ‘Convergence Proofs for Perceptrons’ Math. Th. Automata Wiley 1962
31. Savage: ‘The Foundations of Statistical Inference’ Methuen Mono. 1962
32. Watanabe: ‘Information Theoretic Aspects of Inductive Inference’ IBM Jour. 4(2) 1960
33. Middleton: ‘Communication and Statistical Decision Theory’ Electrotech 74 1964
34. Maron: ‘Design Principles of the Intelligent Machine’ Trans. IRE IT8(5) 1962
35. Minsky, Selfridge: ‘Learning in Random Nets’ as (9)
36. Fu : ‘Class of Learning Control Systems’ IFAC Symp. Sept 1965
37. Zadeh, Desoer: ‘Linear System Theory’ McGraw Hill 1963.