Proceedings of the 3rd Congress of the International Federation of Automatic Control (IFAC), London, (1966)
pp.342-9. [Paper 14.B. Session 14. Tuesday 21st June 1966]

A Learning Machine in the Context of the General Control Problem

B. R. Gaines (Psychological Laboratory, Cambridge)
J. H. Andreae (Standard Telecommunication Labs)

Introduction

Advances in automatic control theory may be seen to have two main objectives the synthesis of improved controllers for a given plant and the extension of the range of plants considered for control. Optimal control theory and its application to linear stems is an example of the former, whilst advances in the latter have been largely concerned with describing-function and quasi-linearization techniques for the linear approximation of non-linear but reasonably continuous plant.

It has become conventional so introduce the abstract concept of a plant through its state transitions (Zadeh and Desoer, 1963) rather than through the differential or difference equations which describe them in particular instances. However, the powerful and restrictive simplification that the plant should be linear is made immediately, so that the operational and matrix calculi developed for linear systems may be used to synthesize a controller. Essentially non-linear systems (Gibson, 1963) are approximated in the time or frequency domain by quasi-linearization or describing-functions, and those whose transitions are indeterminate through inadequate state or input description are treated as noisy or time-varying.

Application of the maximum principle to autonomous, completely identified plant leads to equations for the optimal, non-linear control policy, but the computational problems of solution under a variety of performance criteria are certainly not trivial (Boyadjieff *et al*, 1964). For linear plant with unknown parameters, on-line identification and synthesis by adaptive control has been investigated (Mishkin and Braun, 1961) and optimal control theory has been extended to take into account uncertainty about state variables or transitions (Rosenbrock, 1963; Feldbaum, 1963). Thus the emphasis has been upon the synthesis of optimal, non-linear controllers for linear or linearized plant, rather than upon control by any means of plant which may not satisfy the conditions of linearization.

Alternative approaches to control problems have been made under the guise of learning machines or artificial intelligence (Feigenbaum and Feldman, 1963), where the controller is often ridiculously sub-optimal for a given plant but is capable of controlling a wide range of logical as well as numerical environments. As the survey paper at the Second IFAC Congress (Pask, 1963) demonstrated, this work has been extensive and the games-playing and pattern recognition environments taken as plant for control have raised problems of a different nature from those of the linear or quasi-linear plant considered in automatic control theory. However, at the state transition level of the general control problem the differences are much less apparent and the schemes proposed for learning machines may be seen to be closely related to the adaptive controllers of automatic control. Comparison is generally difficult not only because of disparity between plant considered for control, but also through lack of coherence in work on artificial intelligence which has ramified without general direction.

In this paper the STeLLA learning scheme, which was described at the Second IFAC Congress (Andreae, 1963) in the context of games-playing and pattern manipulation environments, is examined both for the nature of its alternative simplification of the general control problem and for the relationship of its control strategies to those proposed in adaptive control. These strategies are exemplified by its behaviour in controlling a second-order, sampled-data, stochastic plant with bounded phase-space and transport-lag. This is a common type of plant which simulates the situation faced by one of the fully-commissioned adaptive controllers, the automobile driver!

Vehicle-Steering as an Environment

STeLLA, the Standard Telecommunication Laboratories Learning Automaton, is a storage limited, optimalizing controller which has been extensively simulated in a variety of plants or environments. For purposes of comparison with other controllers this paper will take the minimal-path aspect of a simple control task to illustrate the various strategies of adaptive policy formation used by STeLLA.

To make the plant itself meaningful we have simulated fairly closely the parameters and dynamics of an automobile being driven at constant speed, although the output in the form of a binary pattern and the on-off input to the steering wheel would not be relished by a human operator! STeLLA has to learn to steer a car travelling at 30 mph down the centre of a 40 foot wide straight road, using one of three possible control actions, wheel right, left or centre. Feedback information is presented as a binary pattern of 10 features, present or absent, relating to the quantized position and angle of the vehicle relative to the road; these are sampled at 200 msec intervals 50 msec before each steering action takes effect. Both position and angle are bounded in that the vehicle can never run back along or off the road and, should it hit the edge of the road, it ‘rebounds’ at an angle of 9°. Random noise in the control output in the form of indeterminacy and ‘skids’ plus a camber effect which turns the wheels away from centre add to the difficulty of control.

Since the plant is second-order, the phase plane portrait is particularly convenient for representation of the state-space and state-transitions. Data sampling and on-off control imply that trajectories are piecewise continuous and that out of each point of discontinuity can arise one of three arcs subject to slight random variation. In figure 1(b) a typical trajectory is shown as a position against time plot along the road and in 1(a) as a position against angle plot in the phase plane.

The binary representations of position and angle together form a 10 bit feedback pattern for the machine and the distinguishable states are shown by a grid over the phase plane, This quantization makes the state-transitions not merely stochastic but also indeterminate, in that transition probabilities are not state determined, although individual transitions cannot be distinguished from those of a time-varying stochastic system.

The demanded region marked in the centre of the phase plane corresponds to the vehicle being within 5 feet of the centre of the road and within 11° of straight. When the actual trajectory is within this region a binary signal given to the controller changes from 0 to 1, informing it that demand has been attained. This ‘reward’ signal is the only goal-oriented information given to STeLLA in order to effect control.

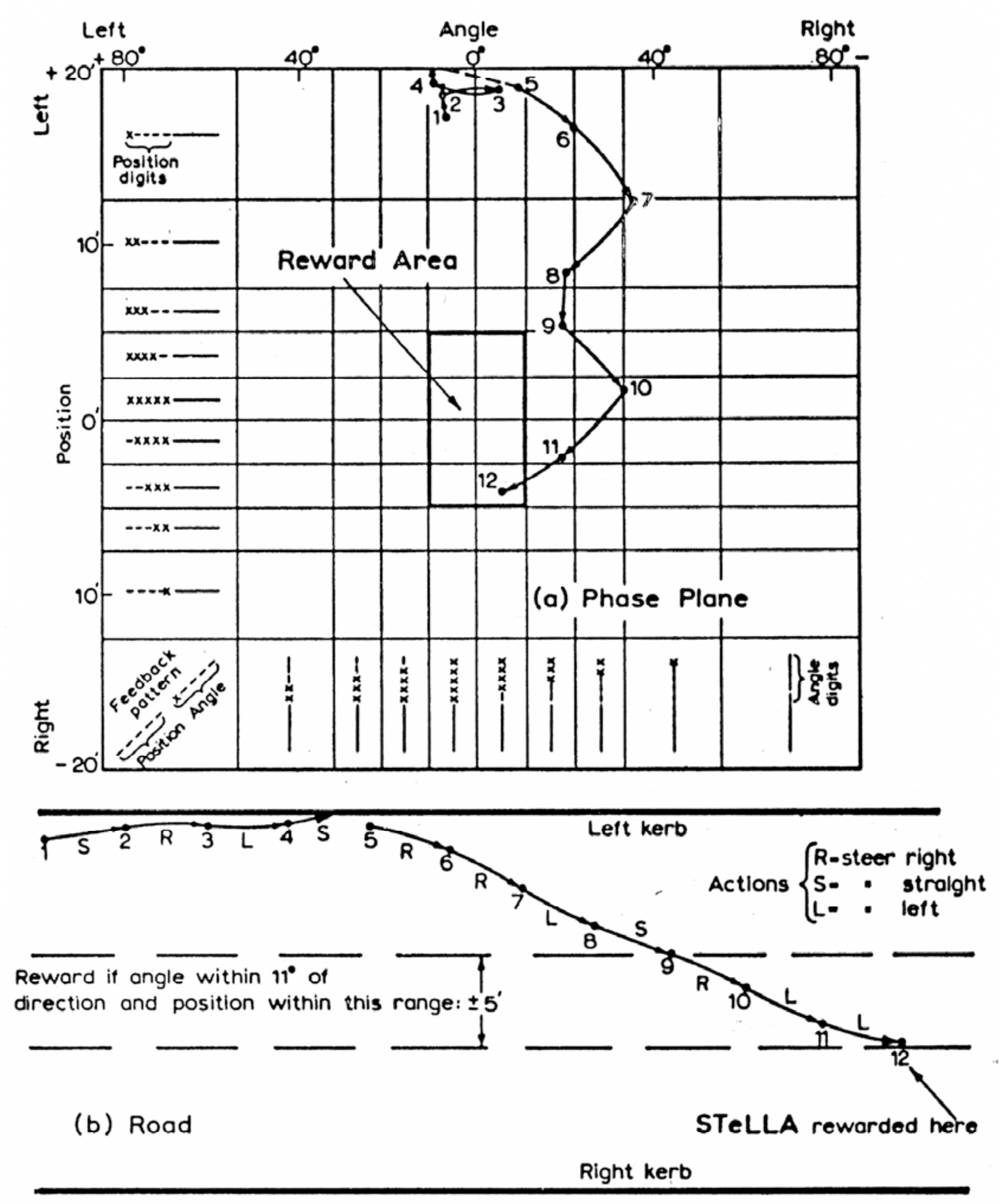


Figure 1. Car Steering: Search Trajectory to First Reward—(a) In phase plane (b) On road

The three actions are selected randomly, in this search trajectory. Reward on reaching position 12, which is within the reward area, causes STeLLA to store the last action to position 12 (Left), the feedback pattern (-XXXX---XX) sampled at position 11 and a sequence connection 'to reward'. The pattern is derived from the actual position 3/4 of the way from positions 10 to 11 because of sampling delay. The pattern is composed of 5 position digits (-XXXX) and 5 angle digits (---XX) which prescribe one of the rectangular areas in the phase plane. From position 4 a straight action took the vehicle into the kerb whence it was 'rebounded' on to the road in position 5.

Computer simulation: Carstella 11- 2 -1965 ICI/C/1043F/24-1. 40' wide road. 30mph.

Sampling interval: 0.2sec. Sampling delay: 0.05sec. Turning radius: 35'. Turning angle 14.3° 0.07°/ft camber +-0.14° indeterminacy +- 7.1° random skids with probability of 1 in 50 per interval.

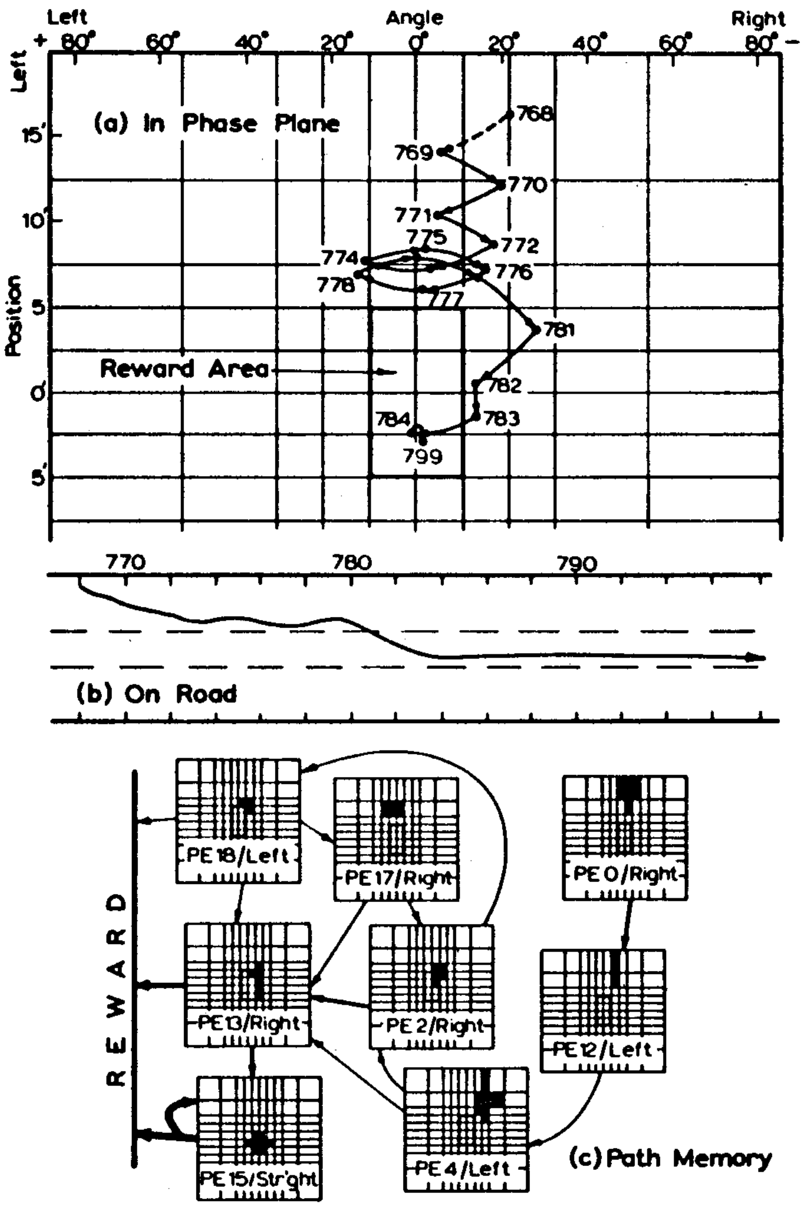


Figure 2. A Similar Trajectory After Learning

(a) In phase plane. The decision to steer left from positions 768 to 769 was taken with the help of the predictor. All actions from positions 769 on were decided by policy elements (PEs) of the policy in the path memory: 769 (PE 0); 770 (PE 12); 771,774,775,778,779, (PE 17); 772,773,777 (PE 18); 776, 781 (PE 4); 780 (PE 2); 783 (PE 13); 782,784...798 (PE15).

(b) On road.

(c) Part Of STeLLA's path memory at the start of the trajectory. Each PE stores an action, a pattern and one or more sequence connections. The pattern (including pattern digit weights) indicates conditions under which the action has been found successful. The conditions are equivalent here to areas in the phase plane and, instead of the pattern, we show for each PE the relevant area of the phase plane. See also figure 3(c). The successful consequences of performing the action of a PE under the conditions imposed by the patterns are stored as sequence connections from a PE to other PEs or to reward. Strongly confirmed connections are indicated here by thick arrows.

Computer simulation: Carstella B-3-1965 ICI/C/1043F/24.5.

STeLLA's Strategies

The purpose of STeLLA's strategies is to synthesize a control policy which maximizes the expected rate of receipt of the reward signal. This criterion is sufficient to define optimal solutions of both the minimal-path problem of constructing a trajectory utilizing the smallest number of control actions to get from the actual state to a demanded state and also the stability problem of remaining within the demanded region. In the vehicle-steering environment both problems arise but we shall emphasize the minimal-path aspect for purposes of comparison with other controllers.

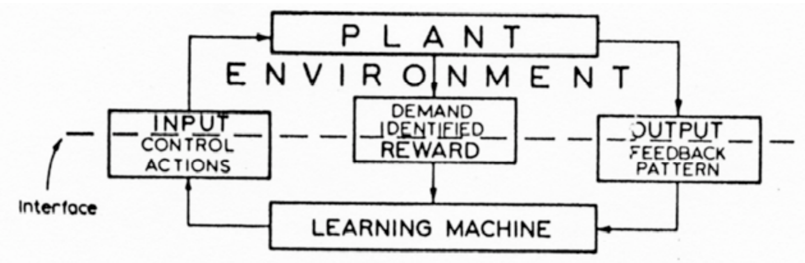
Given an on-off input to the 'plant' and sampled feedback, one form of optimal policy (Desoer and Wing, 1961) will consist of a division of phase space into cells ordered from demand, to which are attached control actions implemented when the feedback sample falls within the cell. A trajectory commencing in a cell n steps away from demand will have its next point of discontinuity in a cell (n-l) steps away so that the policy elements consisting of cell and control action form natural sequences to the demanded region. Quantization of the state-space for purposes of feedback causes transitions to become indeterminate but these may be regarded as generated by time-varying stochastic processes (Kalman, 1957), and the ordering of policy elements must then be based on the expected number of steps to demand.

STeLLA's strategy of sequential-policy formation is an algorithm for synthesizing such policies on-line, one step at a time from demand. Her strategy of state-generalization is an algorithm for varying the size of the cells. Her strategies of random and predictive search are means of manipulating the control actions in a random or model-directed manner when the feedback does not fall within an established cell so that it eventually comes to do so. Figure 3(b) shows schematically how the strategies appear as reaching forwards from the actual state in predictive search and working backwards from the demanded state in sequential-policy paths. In the following section these strategies are discussed in detail.

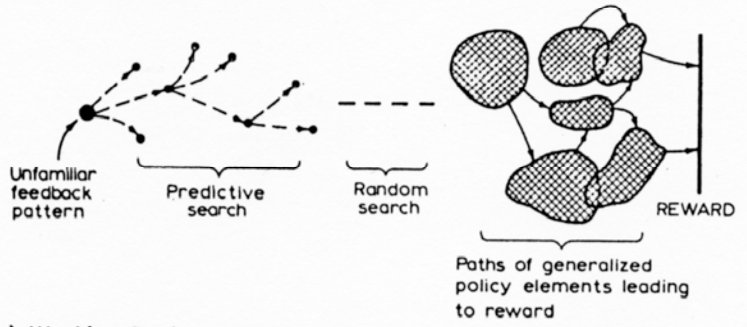
1. Search

Since nothing is known initially of the demanded state or the plant-transitions, random manipulation of control actions is the only policy which guarantees that demand will be reached given that the state-transition graph is strongly connected. Attempts to improve this strategy by restricting actions are apt to lead to trapping cycles which make the demanded state unobtainable (Bremermann and Salaff, 1963) and those which utilize feedback before the goal is reached may yield an even more inefficient procedure than random search (Friedberg, Dunham and North, 1959). It is possible to bias the random selection of actions in such a way as to improve efficiency, for example by decreasing the probability of the same control action occurring again for given feedback (Andreae, 1963), but this requires storage which might be better used elsewhere.

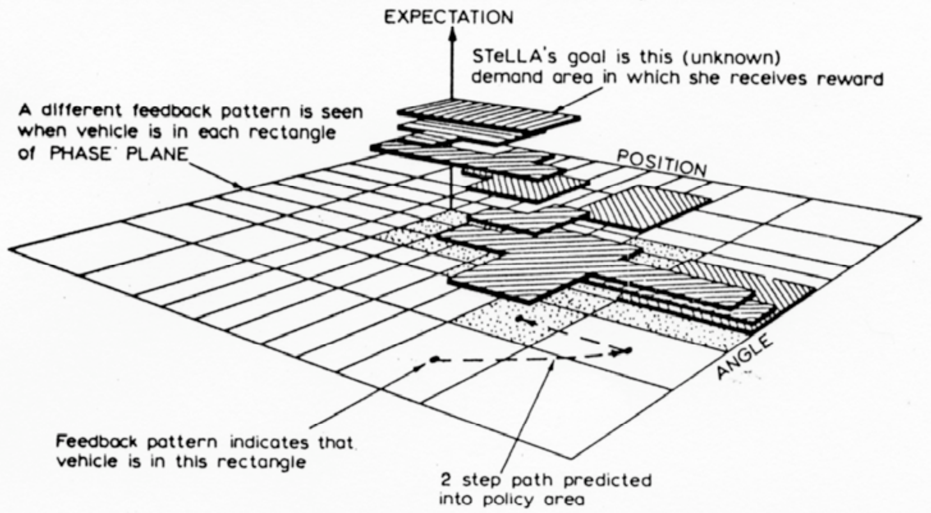
One trajectory which enters the demanded region under random search is shown in Figure 1. In the vehicle-steering plant it takes on average 8 seconds or 40 control actions for demand to be reached under this strategy, but variation about these figures is large. The use of any prior information to improve the search procedure would be advantageous, but in general such information may be false and a safety routine to take the controller back to random search is essential. This is therefore the basic strategy to which the controller reverts when none other is available.



(a) A Plant as an Environment of the Learning Machine



(b) Working Backwards from Reward by Sequential Policy Formation and Reaching Forwards to the Policy Region by Predictive Search


EXPECTATION
STeLLA's goal is this (unknown)
demand area in which she receives reward
A different feedback pattern is seen
when vehicle is in each rectangle
of PHASE PLANE
POSITION
ANGLE
Feedback pattern indicates that
vehicle is in this rectangle
2 step path predicted
into policy area

(c) Policy Elements Arranged in Ascending Order of Expectation to Reward for Policy of Fig. 3(c)

Figure 3. The Overall Strategy of STeLLA

The STeLLA path memory stores a policy comprising adaptive policy elements (PE). Each PE stores a control action, a feedback pattern with 'pattern digit weights' to determine a set of patterns equivalent by generalisation, and a number of sequence connections which anticipate on the basis of experience successful consequences of performing the prescribed action given one of the equivalent patterns. A 'successful consequence' is either the use of a PE with high expectation, or reward. The expectation, which is the estimated probability of reaching reward via the policy, sets the PEs in an ascending order of 'closeness to reward' and thereby ensures a convergent policy.

With a feedback pattern not covered by the policy, STeLLA attempts to project a path forwards into the policy region by predictive search using adaptive model reference as suggested in (c). In this diagram the policy of Figure 2(c) for the vehicle-steering environment of Figure 1 is shown as hill of PEs rising to a summit at reward.

If this fails, STeLLA reverts to random search

2. Sequential-Policy Formation

The minimum-time problem in a synchronous sampled-data, deterministic environment reduces to finding shortest paths between initial and final states on the graph of state-transition. In general only one optimal path may leave any state and, given the complete graph, a dynamic programming technique for finding the optimal policy could be used. This would determine all states which have a one-step transition to demand, and then all those states which have a one-step transition to these and so on (Desoer and Wing, 1961). Eventually optimal paths would have been mapped through all controllable initial states and, after rejecting redundant paths, the control policy could be formulated.

This computation of optimal policies back from demand has the disadvantage of generating a high proportion of irrelevant paths which cannot be discarded until she final stage (Westcott, 1965). Also the technique is not directly applicable to stochastic or indeterminate environments where an optimal solution, if one exists, must be obtained by approximation and iteration. The strategy of sequential-policy formation used by STeLLA is similar to dynamic programming but utilizes the information available through online policy-synthesis to generate paths which are relevant to the particular control task.

Under random search the plant will eventually be controlled to the demanded region and STeLLA will receive a reward signal informing her that demand has been met. The last feedback pattern received together with the control action performed is then stored as a policy element connected to reward, i.e. leading to demand. When this pattern is received again the attached action, having led directly to reward, is optimal if the environment is deterministic or indeterminate. Hence STeLLA implements the policy element, performing the attached action expecting to attain demand. At the same time the previous pattern and action are stored as a policy element connected to the first which is already connected to reward. Storage of policy elements one step at a time favours shorter paths and, by on-line iteration of this procedure, STeLLA builds up near-optimal paths terminating in reward.

Path formation not only synthesizes a control policy but also identifies the plant; the latter process may be refined to deal with stochastic or indeterminate plant by estimating the conditional probability of each stored transition both occurring and causing an increased expectation of reward. With this additional information STeLLA is able not only to choose between alternative paths by comparing expectations of leading to reward but also to adapt the policy to environmental changes by erasing those transitions which become of low probability. The probability estimates are decremented with lack of use of the policy element concerned so that the strategy is adaptive not only to actual changes in the plant but also to changing relevance of individual policy elements with improvement of the overall strategy. The expectation of reward gives an ordering of patterns along paths which enables the machine to determine whether implementation of a policy element has caused an advance towards reward, and hence ensures the convergence of trajectories generated by the policy. This is illustrated in figure 3(c).

3. State-Generalization

The strategy of sequential-policy, formation may be seen as a practical solution to the problem of identifying a plant whilst improving control over it, since it utilizes limited storage to identify that part of the environment relevant to the control problem and in so doing generates a control policy. Given the necessary storage, this strategy would be sufficient for control and, if there is no structure in the state-description, some technique of path-formation is all that can be used. In plant represented by linear systems, however, one of the most powerful properties is the implied metric of similarity upon the state space and one would expect a similar but weaker property in all but the most pathological state spaces of non-linear plant.

In a linear system a small change in state-vector for a given control action causes a small change in the succeeding state-vector so that an action which is optimal in one state will be nearly optimal in the neighbourhood of that state. In linear systems this property has the unwanted further implication of complete extension from local to global so that identification of any small region of the plant is sufficient to identify the entire plant. However, the concept of a metric neighbourhood, although developed in linear vector spaces, may be extended to non-linear systems without the implication from local to global this forms the basis of STeLLA's strategy of state-generalization.

At the interface (Figure 3(a)) feedback from the plant is coded for the controller and this code is assumed to carry a simple metric structure for measurement of similarity between feedback patterns. STeLLA stores with each pattern forming part of a policy element the parameters of its local metric initially set at some given value and uses a policy element not only when the feedback pattern is the same as the stored pattern but also when it is a sufficiently small distance away. The success or failure of this generalization, determined by whether the machine continues to advance along a path to reward, determines whether the stored parameters of the local metric are adjusted so as to increase or decrease the generalized region. The policy is thus adaptive and, although a good coding of the feedback will benefit the machine, a bad one will be rejected by it.

In the vehicle-steering environment the feedback pattern is a binary n-tuple (n = 10) representing the presence or absence of features of the position and angle of the vehicle across the road. A weighting n-tuple is stored with each pattern forming part of a policy element and is used to define the 'deviation' of any feedback pattern from the storm patterns each feature which differs between the patterns contributes its appropriate weight to the deviation. If the deviation is small enough the feedback pattern is regarded as the same as the stored pattern and the appropriate control action is implemented. If this generalization succeeds and the expectation of reward rises, the weights of the neglected features are decreased making similar generalizations more likely, but otherwise they are increased making them less likely. Should several policy elements have patterns with small enough deviations, one of them is selected for implementation by a random process weighted according to their expectation of reward.

4. Prediction by Model-Reference

In the minimal-path problem a trajectory is to be constructed from an actual state of the plant to a demanded state. So far the strategies considered have been largely concerned with the terminal boundary at demand and no strategy has been proposed to enable the machine to formulate a forward path meeting those from demand. However, some form of predictive control, in which estimates are made of the changes in state under various control actions, would enable the controller to 'explore' the region around the plant's actual state for states which are on stored paths to demand. Thus the search strategy could be directed so as to increase the probability of entering a region where the generalized sequential policy might be used.

Identification of state-transition would enable predictions to be made, but the storage required would be vast and unwieldy and better used for sequential policies. Instead the technique of adaptive model reference (Blandhol and Balchen, 1965) may be used and some assumed structure fitted to the state-transitions by statistical estimation. Even if this structure is not very realistic its local predictions may be usefully accurate. The use of a simple linear model for a complex plant with more poles or non-linearities is a basic technique of adaptive control (Mishkin and Braun, 1961) and, in a non-linear plant which cannot be thus approximated, some form of Bayesian predictor may be employed. The exact form again depends upon the assumed structure of state-description. In the vehicle-steering environment, where feedback is presented as a binary pattern, STeLLA uses logical functions of the feedback-pattern features as evidence from which to predict features of the succeeding pattern under a given control action.

In the simulation illustrated by the figures (1-3) the predictor is used to direct search and to control the extent of generalization by checking the anticipated transitions of the stored sequential policy. In directing search, the predictor projects a path forwards from the present state step-by step until its predictions become unreliable or until the projected path meets one stored as a sequential policy. In the former event STeLLA resorts to random search, but in the latter the appropriate control actions are implemented in an attempt to follow the predicted path.

Discussion

STeLLA may be seen as an attempt to synthesize a widely applicable and readily fabricated control scheme. The goal differs from that of synthesis techniques such as dynamic programming which emphasize the computation of control policies rather than their implementation or the gathering of knowledge for computation. Adaptive controllers which identify, compute and implement have generally been specific to a particular plant, whereas viability in a wide range of applications requires a non-specific control scheme which makes few assumptions about the plant and is readily fabricated. For STeLLA to assume that the actual plant transitions and demanded states are initially unknown helps to realize both aims since the controller has initially a simple homogeneous structure and the plant has only to satisfy the assumption that it is a strongly-connected stochastic automaton.

In real control problems prior information about the plant mast be used, where possible, to improve both the rate of adaption and the ultimate control policies of the adaptive controller. For example, the feedback of position and angle from the vehicle-steering plant was put into a Gray code for STeLLA since this most nearly matched the generalization metric to the topology of phase space. On one occasion the learning machine was ‘primed’ with an initial control policy to reduce the duration of search. Initial control policies may also be given to the controller by ‘training’ it along restricted trajectories. Increasing attention is being paid to the use of partial information about the plant (Clark, 1965; Kalaba, 1962; Astrom, 1965) and techniques such as *coding*, *priming* and *training* will undoubtedly be necessary for the practical application of adaptive control. It is customary to think of controllers in terms of optimality but, when the plant is indeterminate or time-varying and the controller itself is required to be widely applicable, such a concept loses much of its force. When fabrication and storage costs have also to be taken into account, one can only ask whether satisfactory control is possible and, if so, how much it will cost.

Acknowledgements

We should like to thank Messrs. D. R. Hill and O. E. Morgan for criticism and Imperial Chemical Industries Ltd., for computer time on their KDF9.

References

Andreae, J. H. Proc. 2nd IFAC Congress, Basle, 1963.

Astrom, K.J; J. Math. Anal. Applicatione 10 174 (1965).

Blandhol, E. and Balchen, J.G; Proc. 2nd IFAC Congress, Basle, 1965.

Boyadjieff, G., Eggleston, D., Jacques, M., Sutabutra, H. and Takahashi, Y.; J. Basio Engineering 86D 11 (1964).

Bremermann, H. J. and Salaff, S; tech, Rept., Univ. Of California, Nov. 1963.

Clark, J.M.C.; Paper 13. Advances in Automatic Control, Nottingham, 1965.

Desoer, C. A. and Wing, J; IRE Trans. AC-6 5, 111 (1961).

Friedberg, R.M., Dunham B. and North, J.H; IBM J.3 282 (1959).

Gibson, J.E; Non-linear Automatic Control, McGraw-Hill, 1963.

Feigenbaum, F. A. and Feldman, J; Computers and Thought, McGraw-Hill, 1963.

Feldbaum, A. A. Proc. 2nd IFAC Congress, Basle 1963.

Kalaba, R; Proc. Symp. Appl. Maths. 14, 173 (1962).

Kalman, R.E; Proc. Symp. Non-linear Cct Analysis, p. 273, Interscience, 1957.

Mishkin, E. and Braun, L; Adaptive Control, McGraw-Hill, 1961.

Pask, G; Proc. 2nd IFAC Congress, Basle, 1963.

Rosenbrock, H.H; Automatica 1 263 (1963)

Westcott, J.H; Paper S.3., Advances in Automatic Control, Nottingham, 1965.

Zadeh, L.A. and Desoer, C.A; Linear System Theory, McGraw-Hill, 1963.

SESSION 14: LEARNING SYSTEMS

Papers discussed: 14A by V. S. Pugachev; 14B by B. R. Gaines and J. H. Andreae; 14C by C. H. P. Brookes and B. Wise; 14D by B. Widrow; 14E by R. H. Raible and J. E. Gibson; 14F by V. N. Vapnik, A. Ya Lerner and A. Ya Chervonenkis; and 14G by M. A. Aiserman, E. M. Braverman and L. I. Rozonoer

P. H. Hammond (*Rapporteur*)—Research on learning systems originally developed out of a desire to study and imitate animal learning by the design of models using assemblages of logical elements which, in some sense, simulated nerve cells in the cerebral cortex.

Such models led to the development of many special purpose machines and computer programmes for the demonstration of learning principles. From these models, their catalytic effect on the thinking of engineers and mathematicians and their relation to conditioned reflexes and pattern perception have grown the present potential engineering applications to automatic control, adaptive filtering, and to the machine recognition of patterns.

Learning always implies information storage over periods that are long compared with the data-sampling rate or system time constants and the subsequent use of this stored data to improve performance. So-called learning systems that use extremum searching techniques to improve performance may require negligible storage capacity and are then extensions of error-corrected feedback systems and not true learning systems.

Because of the need to store operating data, learning requires long periods relative to system time constants while associative relations are accumulated. It is not surprising, therefore, that the most promising current applications of learning systems are to pattern recognition where a long training period can be tolerated, rather than to automatic control or adaptive filtering where such training might require uneconomic operation of industrial plant or data links for an extended time.

At the last IFAC Congress three papers were presented on this topic. The work on two of these has been extended in the meantime and is reported in two of the papers now before us.

In the present session, out of seven papers five are devoted primarily to control applications and two to pattern recognition algorithms. The only practical application dealt with is in the field of pattern recognition.

Paper 14B by B. R. Gaines and J. H. Andreae sets out to reconcile modern control theory with the quasi-empirical learning machine approach. The authors argue that when the general control problem is attacked by state transition techniques the differences between the learning machine approach and 'conventional' adaptive control are much less apparent.

By describing their automaton STELLA from a control theory viewpoint rather than from that of animal behaviour the authors show very clearly the similarities and differences between the two approaches. Nevertheless, it is still not possible to arrive at a quantitative comparison between the behaviour of such a learning system and that of a dynamic plant with dynamic programming, predictive control or other such control policy. In any control system for a specific task a comparison between the learning machine of this type and a more conventional controller is very difficult because precision and speed of response are sacrificed for versatility and ability to adapt, and these are conflicting requirements.

The authors could argue correctly that given the problem of automatically controlling a vehicle to drive along the middle of a road their automaton represents a good solution. This is true and provided that real problems can be found requiring such a versatile controller such systems have an obvious future. However, wherever it is possible to simplify the problem by, for example, providing guides down the roadway, this will be done to avoid the added complexity and storage capacity required for the versatile solution.

V. S. Pugachev, in Paper 14A, proposes a learning system based on statistical decision processes.

The example given by the author of a simple learning filter provides a useful standard of comparison with the well known Bayesian estimation result when the variance of average risk ϵ is of the form $\overline{(\hat{w}-w)^2}$.

Setting $D_T = \infty$ in the authors' equation (2.15), i.e. no teacher, leads to $1/\Delta_\lambda = 1/D_\lambda + N/2D_u$; compare the result for Bayesian parameter estimation.

The work described in this paper has close affinities with that of Feldbaum on dual control, and it would be of great interest to know of applications of the method to sampled data control with stochastic signals.

The paper by B. Widrow (14D) describes an extension of the well-known Adeline threshold logic units. Previous Adelines have learnt with a teacher in the sense that the desired response was presented to the adaptation machinery at each presentation of the input pattern. The present proposal is that the actual response be substituted for the desired response, i.e. feedback of the response takes place. The unit then becomes in a sense 'self-taught' and hauls its weighting elements up or down by means of its own bootstraps.

The noise filter example chosen by B. Widrow to study the performance of the bootstrap Adeline should be compared with V. S. Pugachev's approach by statistical decision theory. Note the common emphasis on the learning of appropriate estimators and the distinction between statistically optimal decisions which are not always perfect and the perfect decisions which do not lead to an optimal estimator.

Again, B. Widrow uses a Bayesian scheme to estimate the joint probabilities $p(\mathrm{O}, \mathrm{R})$, $p(\mathrm{O}, \mathrm{W})$, $p(\mathrm{A}, \mathrm{R})$, and $p(\mathrm{A}, \mathrm{W})$ in terms of conditional probabilities and first-order probabilities, where O, R is a decision which is optimal and right; O, W is a decision which is optimal and wrong; A, R is a decision which is antioptimal and right; and A, W is a decision which is antioptimal and wrong. In this way he arrives at an expression for rate of adaptation.

In the paper by C. H. P. Brookes and B. Wise the word 'learning' in the title is to be construed in a different sense from that used in the other papers, since very little information storage is used in the model-matching procedure described.

The authors argue that the best model for a linear plant in the sense of ease of adjustment is one in which the plant poles are simulated by appropriately placed real multiple poles, giving only one variable parameter for n plant poles. Plant and model order are equated by modelling the appropriate number of poles at the origin. Gain is the second variable parameter.

Note that the dynamic characteristics of a wide range of chemical processes with real poles can be fitted to the model

$$\frac{K \exp(-pT\hat{d})}{(1+pT_1)(1+pT_2)}$$

It would be interesting to know whether the presence of the time delay term would enable even a second-order model to match plants of much higher order using the criteria employed by the authors.

An important factor in learning models of the sort discussed in this paper is the stability and speed of the hill-climbing procedure employed to adjust the model parameters. The paper does not make clear the detailed nature of the iterative method which was used to obtain the results of Table 1, nor the time taken by the iterations relative to the time constant of the model.

R. H. Raible and J. E. Gibson, Paper 14E, discuss a simple but effective learning algorithm to improve a hill climber operating in a rapidly changing environment. The plant is provided with a basic hill-climbing controller which, even without higher level learning, makes some attempt to optimize the cost function.

The information which, in the ideal case, is accumulated in the store of the learning system is a set of co-ordinates of the points through which the locus of cost function minima pass (Fig. 2).

During the learning phase of the hill-climbing operation with changing environment a table is kept in a store addressed by sampled values of the couplet (α, I), the co-ordinates of points through which the system is passing (Fig. 1). These values form a short adaptative record, 10 samples long in the authors' examples. On terminating an adaptive record the value of α last recorded is stored in all the addresses of the adaptive record being terminated. Another record sequence is then started.

As each sample is taken of (α, I) the appropriate storage location is looked up. If a value of α is found in the store at that address a learned jump is made to the appropriate value of α. The values of I before and after the learned jump are compared and if I is degraded by the jump the step is retraced and the record corrected.

The method is shown applied to a hill-climbing system with environmental changes so rapid that the working point diverges widely from the locus of minima during such changes.

The scheme is attractive in being fairly easy to implement with an on-line computer and the learning process is shown to increase the speed of adaptation and improve the performance after a learning period.

Paper 14G by M. A. Aiserman, E. M. Browerman, and L. I. Rozonoer generalizes the potential function method of pattern recognition to the problem of curve fitting and extrapolation by an iterative technique. The method is strongly related to well-known stochastic estimation procedures.

The application of potential functions to pattern recognition has been the subject of several papers by the authors since their introductory paper in the 1963 IFAC Congress; it is further considered in this paper.

The paper ends with a block diagram (unfortunately not reproduced with the preprints), which shows how Rosenblatt's perceptron can be looked upon as employing a special case of the potential function technique.

The authors' work would appear to provide a good theoretical basis for pattern recognition algorithms. However, the ultimate proof of such an algorithm must be in its ability to recognize real patterns, and published experimental results will be looked for with interest.

V. N. Vapnik, A. Ya Lerner, and A. Ya Chervonenkis begin by discussing the problem of a plant A connected to a controller O which contains several levels of control (authors' Fig. 1, p. 14F.9).

The authors then turn to the use of such a hierarchy as a system for learning by imitation in which unit A is an automaton handling an input of randomly generated words. The same words are supplied to unit O which is required to learn to imitate A.

A so-called 'algorithm with full memory' is proposed. This is defined as the operating policy of O_2 such that machine O exactly imitates machine A. This algorithm would appear to state that A is an ideal teacher in V. S. Pugachev's sense. For this state Theorem 1 is given and the rather surprising result is arrived at that the time to learn is independent of the dimensionality of the state space of A.

Discussion then turns to the classification of patterns as a particular case of learning by imitation. The problem is to construct a hyperplane separating descriptions of given patterns using a training procedure. The authors propose for this purpose a 'generalized portrait', and give algorithms for the construction of this paradigm.

The work described on the 'generalized portrait' is a precis of several papers previously published by the authors and, perhaps because of inadequate translation, the present paper does not make the method at all clear. A particular example of the application of this method to a simple problem would help to clear up confusion here.

The practical example given by the authors appears to demonstrate that the method is powerful when applied to geophysical problems. However, from the brief account given it is not possible to understand how the identification of the different layers was accomplished or what precise meaning to place on the authors' Fig. 4 (p. 14F.9).

In the penultimate paragraph the term 'counting frame' is used. This could mean either general-purpose computer abacus or special-purpose instrument and serves to illustrate the sort of difficulty caused to the reader by translation problems.

In conclusion, the papers presented in this session show that some progress is being made in the development of learning systems, but this is perhaps slow relative to other areas of control theory and practice. The difficult problem of finding valid applications outside pattern recognition remains with us.

B. R. Gaines—Two main lines of research have been pursued since my paper was written. The first is concerned with the lack of any economical realization of learning machines with conventional components and computing techniques, and has led to the development of practical hardware in the form of a stochastic computer. The second is concerned with the very different approaches required in the synthesis of controllers should a 'learning machine' become available at reasonable cost; strategies for 'coding, priming and training', mentioned only briefly in the paper, then become very important.

The stochastic computer was developed at Standard Telecommunication Laboratories in response to the need for an active storage element in learning machines (and indeed any adaptive device) in which the stored value was stable over long periods, could be varied by small increments, and whose output could act as a 'weight' multiplying other variables. Since large numbers of these elements would be required in any practical system they had to be small and cheap. Conventional analogue integrators and multipliers do not fulfil requirements of stability and low cost, and unconventional elements such as electro-chemical stores and transfluxors are unreliable or require sophisticated external circuitry to make them usable. Semiconductor integrated circuits have advantages in speed, size, stability and cost, but their use in complex, multiplexed computing circuitry in the general-purpose digital computer obviates them.

The stochastic computer uses the same circuit elements as the digital computer, but has an entirely different representation of quantity in which analogue variables are represented by the generating probability of a Bernoulli sequence of logic levels (rather than as a binary coded word). With a linear mapping from quantity to probability all of the normal analogue computer operations of inversion, addition, multiplication, integration, and function-generation may be carried out by simple gates and counters. With non-linear mapping the complex equations of Bayesian prediction are simply realized with similar elements. The stochastic computer is a favourable alternative to conventional analogue and digital computers when simplicity and low cost are more important than very high speed or high accuracy; hybrid configurations in which all three are combined offer the greatest future promise.

If learning machines are to become commercially viable it will be not as special-purpose controllers for specific tasks, but rather as general-purpose 'learning-components' (of complexity at least as great as STELLA) with inputs, outputs, and a reward or performance-criterion channel. This component will be a 'black-box' whose internal structure cannot be interfered with directly, and which contains initially no task-oriented information. To synthesize a controller the engineer must connect the inputs and outputs of this box to a real or simulated plant, and feed the performance-criterion channel with appropriate signals informing the machine when desired situations are being realized in the plant. Thus the information which the engineer has about the plant and feasible control strategies will be used not to design a control policy, but rather to *code* information from the plant and control inputs to the plant in an optimal way for the learning machine—to *train* the learning-machine by giving a sequence of tasks increasing in difficulty—and to *prime* the learning machine by feeding it approximate information or control-strategies before or during its interaction with the plant.

Priming requires a channel of communication with the learning machine not hitherto available, and my co-author has concentrated on giving STELLA a mode of interaction with an internal environment which can be used either as a 'note-book' or as the basis for 'conversational' interaction. (This work is being continued at the Electrical Engineering Dept, University of Canterbury, Christchurch, N.Z.) Experiments on training have been carried out by myself at Cambridge, using both simulated learning machines and human operators as trainees. It has been demonstrated that a suitable control policy for a given plant may be impossible to learn by direct operation on that plant, but that if a graded series of similar plants ranging from a simple task to the final desired task are given, then learning is rapid and stable.

R. H. Raible—The following corrections to Paper 14E should be noted:

The authors' affiliations should be given in the following way:

R. H. Raible
University of Cincinnati
Cincinnati, Ohio, U.S.A.

and

J. E. Gibson
Oakland University
Rochester, Michigan, U.S.A.

Both authors were formerly at Purdue University, Lafayette, Indiana, U.S.A.

The relation that appears on page 14E.3 in the right-hand column under paragraph head I, should be:

$$\left|\frac{\partial I}{\partial \alpha}+\frac{\partial I}{\partial k}\frac{\mathrm{d}k/\mathrm{d}t}{\mathrm{d}\alpha/\mathrm{d}t}\right| < |S_L|$$

W. K. Taylor—The author of Paper 14D states that the system learns 'without a teacher'. A teacher could be thought of as an independent source of information that produces selective changes in the learning system parameters. With this definition it is difficult to imagine a learning system without a teacher of some kind, either internal or external and with weak or strong selective action.

In the paper, I suggest that the 'critic' is a teacher because it supplies 'win' or 'lose' information to the system and helps to change the weights. For biological systems that learn by trial and error the environment can be regarded as a teacher that eliminates unsuccessful organisms. The successful ones contain an internal teacher that correctly classifies the temporal and spatial patterns received from the environment as 'good' or 'bad'. This is similar to the 'win' or 'lose' classification provided by the blackjack teacher. Learning with such a two-state teacher becomes impossibly slow when the number of distinct responses increases. The teacher must then supply more information, but this is only a quantitative effect and it seems that there is no need to introduce a new terminology for teachers that supply different amounts of information, or reside inside or outside the system under consideration.

P. C. Young—In Section 5 of Paper 14G, under the heading 'Block diagram implementation', the authors discuss the application of their potential function techniques to the case where the reproducing function, $f(x)$, is adequately characterized by a finite number of coefficients $c_i(i = 1 \to N)$. They have also indicated in reference 3 that these techniques can form the basis for real-time process parameter estimation and self-adaptive control. My own work in this field has been concerned with the estimation of the coefficients of a continuous differential equation description of a dynamic process. It is assumed in this work that the form of these equations is known 'a priori' and that it is possible to express the relations between the unknown coefficients or parameters in the form of a general vector-matrix equation,

$$\Phi\mathbf{c}-\mathbf{y} = \boldsymbol{\xi} \quad . \quad . \quad . \quad \text{(D73)}$$

Considering this equation at a particular nth instant of time, one could write:

$$\Phi_n\mathbf{c}_n-\mathbf{y}_n = \boldsymbol{\xi}_n \quad . \quad . \quad . \quad \text{(D74)}$$

where Φ_n is a given $r\times i$ data matrix with elements, $\phi_{lm}(x^n)$; $\mathbf{c}_n$ is the $i\times 1$ vector of unknown parameters, c_i; $\mathbf{y}_n$ is a given $r\times 1$ data vector with elements y^n; and $\boldsymbol{\xi}_n$ is an $r\times 1$ error or noise vector.

It is further assumed that a statistical model for any time-variable characteristics of the parameter vector, $\mathbf{c}$, is available in the form of a vector difference equation,

$$\mathbf{c}_n = A(n, n-1)\mathbf{c}_{n-1}+\boldsymbol{\omega}_{n-1} \quad . \quad \text{(D75)}$$

where $A(n, n-1)$ is an $i\times i$ transition matrix and $\boldsymbol{\omega}_{n-1}$ is an $i\times i$ Gaussian disturbance with zero mean and covariance matrix, Q, which provides a statistical degree of freedom to the equation.

The apparent restriction of the latter assumption is not serious, for the model provided by equation (D75) is quite flexible. For instance, if the parameters are stationary then $A(n, n-1)$ is made a unit matrix and the random sequence, $\boldsymbol{\omega}$, is removed from the equation. Alternatively, if the parameters are expected to vary in some unknown manner, then the random sequence can be inserted and so provide the required degree of freedom. In this case, the covariance matrix, Q, could be chosen to match the expected maximum variation of the parameters. Finally, the model allows for a more sophisticated technique in which the elements of the transition matrix, $A(n, n-1)$, are estimated by means of a secondary 'learning' scheme which uses data obtained from the primary estimation stage.

Making use of the above assumptions, a recursive estimation algorithm for the parameters, c_i, can be obtained (1) by the minimization of a certain cost function consisting of a linear combination of quadratic forms in

(1) The observed error $\boldsymbol{\xi}_n$.

(2) The error between the actual value of $\mathbf{c}_n$ and the 'a priori' (i.e. prior to the new observation) prediction $\hat{\mathbf{c}}_{n\,|\,n-1}$ of $\mathbf{c}_n$ at the nth instant, given data and estimate at the previous $(n-1)$th instant.

The estimation algorithm is of the form,

$$\hat{\mathbf{c}}_n = \hat{\mathbf{c}}_{n\,|\,n-1}+\Gamma_n\Phi_n{}^T\{\mathbf{y}_n-\Phi_n\hat{\mathbf{c}}_{n\,|\,n-1}\} \quad \text{(D76)}$$

where, superscript T denotes matrix transpose and Γ_n is an $i\times i$ time variable weighting matrix with elements $\gamma_{lm}{}^n$. This matrix is generated by another recurrence relation derived with the help of equations (D75) and (D76).

Another algorithm at present under preliminary investigation is of the general form

$$\hat{\mathbf{c}}_n = \mathbf{c}_{n\,|\,n-1}+\Gamma_n\Phi_n \sin\{\mathbf{y}_n-\Phi_n\hat{\mathbf{c}}_{n\,|\,n-1}\} \quad \text{(D77)}$$

This particular type of algorithm is being considered because it provides a logical extension to the analogue 'least magnitude' method of parameter estimation used previously (2).

Equations (D76) and (D77) show a resemblance to certain of the equations which appear in the paper presented by Aiserman *et al.* However, this similarity is even more pronounced when it is realized that equation (D76) has convergence characteristics which liken it to the so-called methods of multidimensional stochastic approximation referred to by the authors in a footnote to Section 1 of their paper. As they point out, Tsypkin (3) has indicated the relation between their techniques and those of stochastic approximation. It is also interesting to note that equation (D76) together with the recurrence relation for the time-variable weighting matrix, Γ_n, also resemble the optimal filter equations of Kalman (4) which themselves can be derived by an extension of the original work of Gauss into the analysis of data corrupted by errors. These observations tend to confirm the suggestion made by the authors in a recent paper (5) that more general theories exist which cover the whole range of estimation (or 'learning') theory, from simple least squares regression analysis, through real time parameter estimation and optimal filter theory to stochastic approximation and the method of potential functions.

REFERENCES

(1) YOUNG, P. C. 'Real-time identification of dynamic processes' (to be published).

(2) YOUNG, P. C. 'The determination of the parameters of a dynamic process', *The Radio and Electronic Engineer* 1965 **29** (No. 6, June), 345.

(3) TSYPKIN, Ya. Z. 'Establishing the characteristics of a function transformer from randomly observed points', *Automatika i Telemekhanika* 1965 **26** (No. 11, November), 1947.

(4) KALMAN, R. E. 'New methods and results in linear filtering and prediction theory', *R.I.A.S. Rept* 61-1 1960 (Martin Co., Baltimore, Md).

(5) AISERMAN, M. A., *et al.* 'The Robbins–Monro process and the method of potential functions', *Avtomatika i Telemekhanika* 1965, **26** (No. 11, November), 1951.

Y. C. Ho—I will raise two points concerning related results which may interest the authors of Paper 14G.

Consider the problem of recovery of the function $f(x)$ from noisy measurements of the function value at randomly selected observation points. We may regard the problem as that of determining those values of s_i which minimize the sum of the squares of the difference between the observed value of the function, $\hat{y}_n$, and the computed value of the function based on the estimated $c_i(n)$, $\sum_{i=1}^{N} c_i(n)\phi_i(x_n)$, i.e. a least square fit problem. If we impose the further requirement that the least square fit is to be computed recursively, then we are quickly led to an algorithm for $c_i(n)$ very similar to that represented by equation (31) where r_n becomes a matrix that can be recursively calculated with little additional effort. Convergence under very mild assumptions can also be proved by using stochastic approximation or the law of large numbers. Experimentally, we have found that the least square algorithm converges faster than that by equation (31). Have the authors observed similar experiences in their simulation?

In practice, most $f(x)$ do not possess a finite expansion as represented by equation (5). On the other hand, for computational simplicity, one is forced to use a finite expansion as (5). The natural question then arises as to the asymptotic properties of the approximation when the potential function techniques is applied. It turns out that the following is true for algorithm defined by (3) and (13) as well as the least-square algorithm mentioned above:

$$\lim_{n\to\infty} c_i(n) = c_i^* \quad \text{with probability one}$$

where

$$c_i^* \text{ minimizes } E(f(x) - \sum_{i=1}^{N} c_i\phi_i(x))^2$$

In other words, the convergent finite expansions have mean square approximation property. This provides another justification for the use of the potential function method.

The above points are discussed in more detail in references (6) and (7).

REFERENCES

(6) HO, Y. C. and BLADON, C. C. 'On the abstraction problem in pattern recognition', *Cruft Lab. tech. Rept No. 476* (August) 1965 (see also expanded version in *National Electronic Conference Proceedings* 1966, to be published).

(7) KASHYAP, R. L. and BLAYDON, C. C. 'Recovery of functions from noisy measurements taken at randomly selected points and its application to pattern classification' (to be published in *Proc. Instn elect. Engrs* 1966).

I. H. Rowe—In Section 2 of Paper 14G, an unknown function $f(x)$ is reconstructed from random samples. Would the authors report on the success of the potential function method when used in numerical computation? Specifically:

(1) Is the rate of convergence, using an arbitrary form for the potential $K(x, y)$ as the authors suggest, rapid enough so that in general, vast numbers of sample points need not be taken?

(2) How does the number of sample points required for convergence grow as the dimensionality of x is increased? Does the number increase to the power of the dimension of x?

B. Widrow—W. K. Taylor's comments are well taken. The term 'learning without a teacher' could be misleading, but it has been used by a number of authors in the United States and Europe to describe several types of learning processes related to the 'bootstrap learning' technique. 'Learning without a teacher' has been used to describe the training of trainable networks when the desired network responses are not supplied with the presentation of each individual input signal vector. 'Learning with a critic' would perhaps be a more apt description. Learning with a teacher is always faster than learning merely with a critic; but the difference in learning speed is very great (greater than an order of magnitude) only when the bootstrap learning system is close to optimal. An unsupervised system that truly learned without a teacher would generally be useless to humans. At some point, the human must inject his performance criterion if he expects the learning system to do things that are useful *for him.*

M. A. Aiserman—In answer to I. H. Rowe, convergency rate of the algorithms of the potential functions method depends mainly on the 'bizarreness' of the function to be restored and to some extent on the reasonability of choice of the potential function $K(x, y)$. Indeed, convergency rate depends on the system of functions chosen for serial expansion and on the number of 'harmonics' actually present in the expansion representing the restored function. Usually, the length of the series grows with the growth of the space dimension.

Since the restored function is *a priori* unknown, only estimations of the type mentioned in the paper and experimental data are valid. In our experiments closed-form potential functions were used. In problems we solved it appeared that the convergency was practically achieved if the number of points that were shown was approximately one order higher than the dimension of the space.

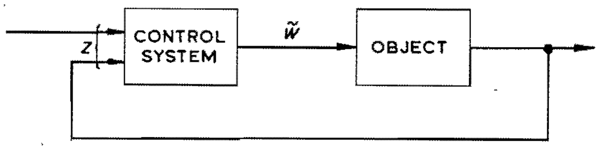


Fig. D47

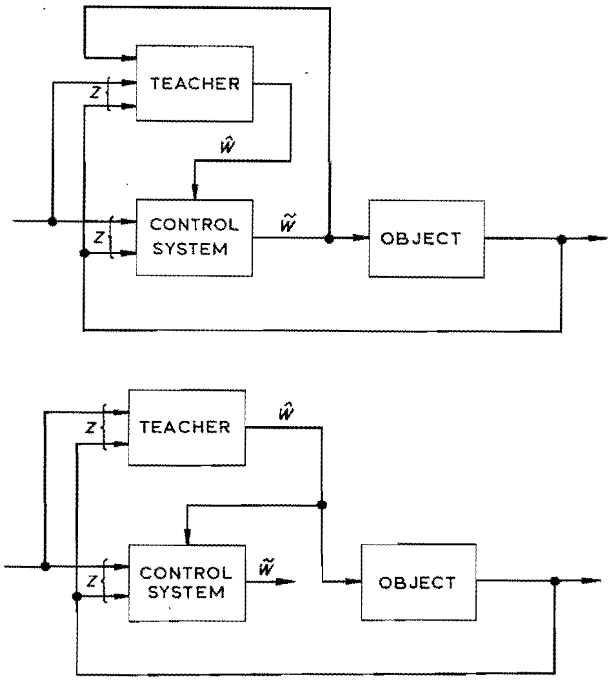


Fig. D48

V. S. Pugachev—A 'learning system' is the name given to a system which improves on the basis of external information received by it. When the system does not receive any supplementary external information over the normal input signals, it is called a 'self-learning system'. If the system receives, apart from input signals, supplementary external information, one calls it a 'system learning from a teacher'. In this case we call a 'teacher' any source of additional external information, be it a human operator, or another automatic system.

Two methods of teaching the automatic system are possible: teaching by demonstration, and teaching by evaluation of the system performance. Figs D47 and D48 show three types of teaching a system which is controlling a given object, O. In the given case the input signal of the control system Z represents a combination of all controlling signals and feedback signals.

Z is the input signal; W is the required output signal; $\tilde{W}$ is the actual output signal of the learning system; $\hat{W}$ is the output signal of the teacher. The algorithm of the teacher is determined in the general case by its decision function

$$\delta_y(\hat{w} \mid z, w, \tilde{w})$$

In the case when $\hat{W}$ is a finite-dimensional vector, δ_y represents a conditional probability density of the outside teacher signal $\hat{W}$ at any given realizations z, w, $\tilde{w}$ of the signals Z, W, $\tilde{W}$. When $\hat{W}$ represents a random time function, then δ_y represents a certain functional of $\hat{w}$, z, w, $\tilde{w}$, which characterizes the effect of the distribution of the teacher output signal probability on the results of learning. In the particular case of an ideal teacher, the decision function of the teacher represents an ordinary Dirac δ-function $\delta(\hat{w}-w)$. In the case of a real teacher, teaching the system by demonstration, δ_y does not depend on w, $\tilde{w}$. In the case of a real teacher, teaching the system by way of an evaluation of its actions, δ_y does not depend on w.

A required result of teaching may be, for example, an approximation of the decision function of the system to the required decision function $\delta(\tilde{w}/z)$. This, as a rule, represents the conditional probability density of the instantaneous value of the system $\tilde{W}$ output signal at any given realization z of the input signal Z. Since $\delta(\tilde{w}/z)$ necessarily depends on the unknown statistical characteristics of the input signal Z, and on the required output signal W, then in this case the teaching task appears to be the development of a suitable statistical evaluation of the required decision function $\delta(\tilde{w}/z)$.

Assuming, as before, that the unknown statistical characteristics of the input, the required output signals Z, W, and, consequently, the decision functions δ_y and δ represent determined functions, which depend on the final number of unknown parameters forming the vector λ; also assuming that

$$\delta_y(\hat{w} \mid z, w, \tilde{w}) = \Delta(\hat{w} \mid z, w, \tilde{w}, \lambda) \quad \text{(D78)}$$

$$R(z, \hat{w}, \tilde{w} \mid \lambda) = \int P_1(z, w \mid \lambda)\Delta(\hat{w} \mid z, w, \tilde{w}, \lambda)\, dw \quad . \; . \; . \quad \text{(D79)}$$

and assigning an *a priori* probability density $\alpha(\lambda)$ for unknown parameters, we obtain the following formula for the *a posteriori* probability density of unknown parameters:

$$\omega(\lambda) = c\alpha(\lambda) \prod_{i=1}^{N} R^i(z_i, \hat{w}_i, \tilde{w}_i \mid \lambda) \quad . \quad \text{(D80)}$$

where the upper index i takes into consideration that the functions P_1 and Δ, and consequently, R, may be different for different time periods, during which teaching signal realizations are introduced into the system. The corresponding formulae for self-learning, learning by demonstration, and learning by way of evaluating system actions follow from expression (D80). The extension of the general formula for the *a posteriori* probability density of unknown parameters for the case of teaching the automatic system by way of evaluating its actions, represents the first addition to the original report.

Having determined $\omega(\lambda)$, one can find the corresponding evaluation δ^* of the required decision function δ from any statistical criterion, and, in accordance with the decision function obtained in this way, one can develop the system algorithm, close to that required. In particular, one can find the optimal t.m.s. evaluation λ^* of the unknown vector λ, and accept as the evaluation δ^* the value of the function δ when $\lambda = \lambda^*$. The second addition to the original report is concerned with the extension of the theory for the case when the development of the system decision function close to that required becomes the aim of learning.

One of the most important problems of the theory of learning systems is the clarification of the extreme possibilities of learning. In order to solve this problem one has to find the optimal learning system (in the given sense) which, after learning, is the best amongst all systems which make use of the same information. It is then necessary to evaluate the quality of this optimal system after the learning period.

To find the optimal learning system with a given loss function $l(W, \tilde{W})$ (which may depend on unknown parameters λ) one can apply general methods of the theory of statistical solutions and statistical theory of optimal systems (8)–(10). For this it is sufficient to consider the totality of all signals received by the system in the process of learning, and the next input signal after the learning Z, as a single composite input signal Ξ. In the case of self-learning, Ξ represents the totality of learning realizations $Z_1, \ldots, Z_n$, and of the next realization Z of the input signal. In the case of system learning by demonstration, Ξ represents the totality of learning realizations $Z_1, \ldots, Z_n$, $\hat{W}_1, \ldots, \hat{W}_n$ of the input signal and of the teacher output signal and of the next realization Z of the input signal. In the case of system learning by evaluating its action, Ξ represents the totality of pairs of realizations $(Z_1, \tilde{W}_1), \ldots, (Z_n, \tilde{W}_n)$ of the input and output system signals, corresponding to those pairs of evaluations $\hat{W}_1, \ldots, \hat{W}_n$, given by the teacher, and of the next realization Z of the input signal.*

As a result one obtains the following algorithm of the optimal Bayes learning system: the output signal W^* of the optimal system is determined by minimizing the conditional risk

$$\rho(\Xi, W^*) = \iint l(w, W^*)\, K(w \mid Z, \lambda)\omega(\lambda \mid \Xi)\, \mathrm{d}w\, \mathrm{d}\lambda \quad \text{(D81)}$$

where

$$\omega(\lambda \mid \xi) = c\alpha(\lambda)P_2(z \mid \lambda) \prod_{i=1}^{N} R^l(z_i, \hat{w}_i, \tilde{w}_i \mid \lambda) \quad \text{(D82)}$$

and $K(w \mid z, \lambda)$ is the conditional probability density of the required output signal W at the given realization z of the input signal Z, and a given value of the vector λ. The formula (D82) differs from (D80) by the presence of the supplementary multiplier $P_2(z \mid \lambda)$. This indicates that the optimal Bayes learning system ought to continue to perfection by way of self-learning after the learning has been terminated.

This report suggests that an evaluation of the quality of the optimal learning system (which determines the limiting possibilities of system learning at the given conditions and with the given teaching information) can be made, according to the relative value ϵ of increasing the average risk, as compared with its average value (in respect to λ), which corresponds to the optimal Bayes system with full information (i.e. with the accurate knowledge of the true value of the unknown vector λ). The algorithm of the optimal Bayes learning system represents the third addition to the original report.

The basic task of any further development of the theory presented here is the research for simplified approximate algorithms of automatic learning systems which are close to optimal.

* *The problem of determining the optimal learning system was raised and solved for the first time for an individual case of recognition of two images by M. Ye Shaykin* (11).

REFERENCES

(8) Pugachev, V. S. 'Theory of random functions and its application to problems of automatic control', *Fizmatgiz* 1962.

(9) Pugachev, V. S. 'Statistical methods in automatic control', Works of the Second International Congress of IFAC. Optical systems. Statistical methods. *Nauka* 1965.

(10) Sysoyev, L. P. 'Evaluations of parameters of signals modulated by random processes', *Avtomatika i telemekhanika* 1965 **26** (No. 10).

(11) Shaykin, M. Ye. 'Optimal evaluations of signal parameters according to the final number of "learning" realizations', Works of the Third All-Union Conference on the Theory of Automatic Control. Odessa, 20–26th September, 1965.

V. N. Vapnik, A. Ya. Lerner, and **A. Ya. Chervonenkis**—Recently we succeeded in formulating one, in our opinion, rather general problem of teaching to which the problem of pattern recognition teaching and some problems of control teaching can be reduced.

We have proved, under some general limiting conditions, that this problem has a solution; have found a method of solution; have estimated teaching time for this method; and have finally constructed several teaching algorithms that can be processed on a digital computer. We termed this problem the problem of teaching a machine to perform extremal imitation.

Here is the statement of the problem: Let us assume that it is necessary to synthesize converter B which matches vector Y to each input vector X.

Let us assume also that to each pair of X, Y we can match a number g which describes qualitatively the data processing. Suppose that numbers describing the quality of data processing can be computed for any pair of X, Y with the aid of function $G_0(X, Y)$ about which we only know that it belongs to a class of functions G.

We also know that input vectors X behave independently, following a certain unknown but fixed distribution $P(X)$.

During the synthesis of converter B we can observe its operation and evaluate the operation of another converter —i.e. converter A, about which we only know that it performs sufficiently well. In other words, if for each input vector X converter B produces a vector Y such that it would receive a rating not below that of converter A when processing the same vector X, then the operation of converter B would be considered satisfactory.

It is necessary to construct a machine which, by observing its operation and by actively conducting experiments (learning), would guarantee, with a reliability not below the given one (e.g. $1-\eta$), that, in a fixed number of teaching steps, a converter B would be synthesized for which the probability that its operation would be rated (with the aid of function $G_0(X, Y)$) below that of converter A by a magnitude larger than ϵ, would be lower than the given number κ.

Or formally, through l teaching steps with a probability not below $1-\eta$, it is required that the following inequality be satisfied:

$$P\{G_0[X, A(X)]\} > G_0[X, \dot{B}(X)+E] < \kappa \quad \text{(D83)}$$

In other words, for the class of functions G it is required to construct a learning machine that whatever the operator of converter A, whatever the distribution of input vectors, and provided only that conversions are evaluated with the aid of function $g(X, Y) \in G$, the machine will synthesize, through l teaching steps, converter B for which (D83) is satisfied.

It was shown that the problem can be solved, provided only that the class of functions G is such that it can be spanned by an E-network (distance ρ is determined from

$$\rho = \max_{X, Y} |g_1(X, Y) - g_2(X, Y)|)$$

Furthermore, teaching algorithms are found for which length l of the teaching sequence is limited by $l \leqslant C(\kappa, \eta) \ln N . \ln \ln N$, where N is the number of cells of the E-network, $C(\kappa, \eta)$ is the constant independent of N, which was also determined.

However, generally speaking, the algorithms which we have found are not suitable for processing on a digital computer, because they are too universal. They are applicable in the case in which the class of functions G consists of functions that can either be processed poorly or not at all on a digital computer.

It is therefore natural to limit the class of functions G by constructing a teaching algorithm that can be processed on a digital computer. We have constructed such an algorithm for a certain class of extremal functions G.

The complete study was published in the journal *Automation and Telemechanics*, Nos 5 and 6, 1966. At present this algorithm is used in solving operational control problems.