SCR-641

Sandia Corporation

•••••••• **MONOGRAPH**

A MATHEMATICAL MODEL FOR AN ASSOCIATIVE MEMORY

by

G J Simmons

APRIL 1963

Issued by
Sandia Corporation,
a prime contractor to the
United States Atomic Energy Commission

Printed in USA. Price $2.50. Available from the Office of
Technical Services, Department of Commerce,
Washington 25, D. C.

SCR-641
MATHEMATICS AND COMPUTERS
TID-4500(19th Edition)

SANDIA CORPORATION MONOGRAPH

A MATHEMATICAL MODEL FOR AN ASSOCIATIVE MEMORY

by

G. J. Simmons

April 1963

ABSTRACT

A mathematical model for an associative memory is proposed which uses associative addressing and distributed storage. Associative addressing is accomplished by mapping from a space with relatively few dimensions (input variables) to the vertices of a binary-valued hypercube embedded in a much higher dimensional space. The dimension of the image space is chosen to be sufficiently great that a hyperplane can be passed through the origin such that the relative distances to the image points are the relative functional values which are to be stored. The distributed memory is achieved in the n-tuple representation of the hyperplane, since each element will in general be used in calculating the distance to many points (images), and hence in storing many functional values.

The second part of the paper is devoted to a novel technique formulated for solving the large linear systems which arise in such a problem and a proof of the convergence of such a procedure. Unfortunately, the basic form of an associative memory imposes the restriction that only a single linear expression be available at any one time, and that further its relation to other expressions not be known. This generally imposes a further restriction that the linear expressions be randomly drawn from the linear system and returned. Typically these systems have many more variables than equations.

The final portion of the paper is devoted to several examples of the behavior of the associative memory as simulated on a CDC 1604 computer, and to illustrative examples of the convergence properties of the algorithm proposed here to solve the associated linear systems.

ACKNOWLEDGMENT

It is a pleasant task to acknowledge the assistance of several investigators whose efforts have been vital to the research reported in this paper. The mathematical model for an associative memory was strongly influenced by the research of Mr. J. W. Smith of the Navy Management Office on his ADAP II, which he generously communicated in advance of publication. Much of the experimental work was made possible by the efforts of Dr. H. Everett and Miss B. J. Ellis of the Weapons Systems Evaluation Group of IDA, who were able to reduce what appeared to be an intractable mathematical scheme to a feasible one. Finally, there are the author's colleagues, Drs. L. W. Rook and W. L. Garner whose contributions appear unnoted throughout the paper. To Dr. Rook especially, as the codeveloper of the probabilistic associative memory, the author is indebted for many suggestions, criticisms and original ideas.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

A MATHEMATICAL MODEL FOR AN ASSOCIATIVE MEMORY

Introduction

Memory techniques which have so far been developed with computer technology all share the same limitations imposed by the requirement that the memory address be specified uniquely for storage or retrieval of data. In many problems these addresses are merely "dummy variables" with the only meaningful order being one derived from the data points themselves; examples of such operations are sorting, interpolating and catalog look-up. Largely because of the astronomical proportions of the computing task assumed by some real (and reasonable) problems in these and other closely allied areas, there has been an increasing interest in the type of memory evinced by all higher forms of animal life in which information is stored (and retrieved) on the basis of content. Storage systems with this modus operandi are generically classified as "associative memories."[1-6]

There are two fundamentally different conceptions of an associative memory. The first can be regarded as an exact memory (in the same sense that a conventional computer memory is exact) which generates a storage address from an examination of the data to be stored.[1-3] In this version, an item for entry is characterized by as many descriptors as possible (or as may be limited by the dimension of the memory) and the entry is stored "at the intersection" of these descriptors. The list of descriptors, if regarded as separate from the data word itself has been given the name associative criterion by Kiseda, et al.[2] Obviously, a single storage word may in general have a multiplicity of associative criteria, or alternatively, several words may have a common one. For an example, if a particular piece of information is filed jointly under a list of twenty properties or attributes (the associative criterion), and if another piece of stored information has nineteen of the attributes in common with it, then a query of these nineteen common properties would result in both pieces of information being recalled from memory and read out. This interpretation of an associative memory is concerned with exact data storage and retrieval (albeit the retrieval request is inexact and incomplete) and not with "interpolation" between stored data points.

---

[1]W. L. McDermid and H. E. Petersen, "A Magnetic Associative Memory System," IBM Journal **4**, pp. 59-62, 1959.

[2]J. R. Kiseda, H. E. Petersen, W. C. Seelbach, and M. Teig, "A Magnetic Associative Memory," IBM Journal **5**, pp. 106-121, 1961.

[3]R. R. Seeber and A. B. Lindquist, "Associative Memory with Ordered Retrieval," IBM Journal **6**, pp. 126-136, 1962.

[4]R. R. Seeber, "Cryogenic Associative Memory," National Conference of the Association for Computing Machinery, August 23, 1960.

[5]R. L. Boyell, "A Semantically Associated Memory," pp. 161-169 in Biological Prototypes and Synthetic Systems **1**, edited by E. E. Bernard and M. R. Kare, Plenum Press, New York, 1962.

[6]B. Widrow, "Generalization and Information Storage in Networks of Adaline 'Neurons'," pp. 435-461, in Self-Organizing Systems, 1962, edited by M. C. Yovits, et al., Spartan Books, Washington, D. C., 1962.

An interesting variation on the above approach is typified by Boyell's recent paper[7] in which each element of the associative criterion has a probability assigned to it. In this memory model one may consider the entries to occur at the nodes of a network in which the numerical value (p) assigned to each link represents the cumulative degree of association between the entries stored at the terminal nodes. This is still essentially the same system as that described above in which the storage location is determined by the associative criterion; however, the introduction of weighting factors for the properties which make up the associative criterion effects a more realistic association of stored items. In other words, simple memory recall is still uniquely determined; it is only when the memory is interrogated with a partial list of properties, so that several entries must be associatively compared, that the additional capability of this scheme is used.

The second conception of an associative memory is concerned almost exclusively with an ability to interpolate between stored data points. This does not imply that exact recall of stored data points is no longer desired, although this capability is generally compromised, but rather that the memory should have the ability to interpolate (or extrapolate) in a field of statistically related memory entries (learning experiences). Biological systems evince both types of memory processes (hence their common classification under the heading of associative memories); however, the first appears most often in an overlapping form as shown, for example, in the results of free association psychological tests. Apparently the most important design feature of the biological system is what is termed "distributed memory." Unlike a computer memory where a particular item is stored in a sharply defined location (which is equally true for the associative memories discussed above), the entries into a biological system are distributed through relatively extended volumes of the neural material, so that even partial extirpation cannot erase a strongly recorded memory trace.[8]

The concept of distributed memory has received a great deal of attention in the past few years, primarily because of the pioneer work of F. Rosenblatt on perceptrons.[9-11] The literature on perceptrons has become so large that a sizable bibliography is required to cover the papers in this field alone; however, the general approach set forth by Rosenblatt[9,10] is essentially applicable to the efforts of all workers in the

---

[7]R. L. Boyell, op. cit., p. 5.

[8]K. S. Lashley, "Brain Mechanisms and Intelligence: a quantitative study of injuries to the brain," University of Chicago, Chicago, 1929.

[9]F. Rosenblatt, "The Perceptron - a theory of statistical separability in cognitive systems," Cornell Aeronautical Laboratory Report No. VG-1196-G1, January 1958.

[10]F. Rosenblatt, "Principles of Neurodynamics - perceptrons and the theory of brain mechanisms," Spartan Books, Washington, D. C., 1962.

[11]F. Rosenblatt, "Perceptual Generalization over Transformation Groups," pp. 63-100 in Self-Organizing Systems, 1960, edited by M. C. Yovits and S. Cameron, Pergamon Press, New York, 1960.

field. Rosenblatt defines a perceptron to be "a set of signal generating units (or 'neurons') connected together to form a network [which is defined to be a 'brain model']... The logical properties of a perceptron are defined by:

1. Its topological organization (i.e., the connections among the signal units ['neurons'];
2. A set of signal propagation functions, or rules governing the generation and transmission of signals;
3. A set of memory functions or rules for modification of the network properties as a consequence of activity."[13]

The key point in the foregoing (and by implication in the research of other perceptron investigators) is that the device is a brain model intended to reveal through an appropriate neural modeling scheme insights into "the psychological functioning of a brain in terms of known laws of physics and mathematics, and known facts of neuroanatomy and physiology."[13] This has resulted in a very diffuse set of requirements on the overall system with primary emphasis on the neuron models (neuromimes) which are the building blocks of such systems.

All of the studies of neural net analogs, distributed memories, neuromimes, etc., which were referred to above (but not referenced) are direct outgrowths of the original neuropsychological theories of D. O. Hebb[14] and F. A. Hayek[15] in which the concept of equipotentiality investigated by Lashley[12] is developed into a neurophysiological model capable of explaining the phenomena of distributed memory traces. In this theory the "seat" of memory is the synaptic junction through which one neuron can affect another neuron. Probably the best summary of this theory is given in Hebb's original "neuro-physiological postulate": "When an axon of cell A is near enough to excite cell B and repeatedly and persistently takes part in firing it, some growth process or metabolic change takes place in one or both cells such that A's efficiency as one of the cells firing B, is increased."[14] While the modeling of such a scheme is undoubtedly in keeping with the objective of the study of brain function, the unbelievable complexity of the subject ($\approx 10^{10}$ neurons in the brain of man, each with $\approx 10^{3}$ synapses) so dwarfs any achievable model that it is difficult to see how any results (?) achieved with a model can be extrapolated to the human brain. However, it does not appear unreasonable to attempt the modeling of some phylogenetically early nervous systems.

The foregoing thumbnail sketch of efforts in the general area of associative memories has been given to indicate the basic objectives and procedures used by the various other investigators in this area. In this paper we shall be concerned primarily with the concept of a distributed memory, and exclusively with the second conception of the associative memory which results from such a scheme, i.e., a memory with interpolatory and extrapolatory capabilities.

---

[12] K. S. Lashley, op. cit., p. 6.

[13] F. Rosenblatt, Principles of Neurodynamics, p. 6.

[14] D. O. Hebb, "The Organization of Behavior - a neuropsychological theory," John Wiley and Sons, Inc., New York, 1949.

[15] F. A. Hayek, "The Sensory Order," University of Chicago Press, Chicago, 1952.

Consider a function $f(x_1 \ldots x_n)$ defined over n variables, where each of the variables is (for convenience) restricted to some discrete set of values, as specifying the environment or source of learning experiences for an associative memory. Then the set of values $\{f(x_1 \ldots x_n), x_1 \ldots x_n\}$ consisting of the complete specification of the location of the point in space and the value assumed by the function f at that point constitutes a single learning experience or data point for entry into the memory. It will be shown later that the value of the function need by only statistically defined, and that further the location of the point can be incompletely or ambiguously specified; however, for the moment it is convenient to consider the point and the functional value as well defined. The base vector of the function may be conveniently written as $(x_{1\alpha}\, x_{2\beta} \cdots x_{n\eta})$ where the second subscripts designate the specific value assumed by each of the variables out of its possible set of values. If we think of the space as defined by a linear array of symbols

$$(x_{11}\, x_{12} \cdots x_{1k_1})\,(x_{21}\, x_{22} \cdots x_{2k_2}) \cdots (x_{n1}\, x_{n2} \cdots x_{nk_n})$$

where each of the parentheses contains all of the values which the variable can assume, then a particular point in the discrete sensory space (S-space) is expressable as a vector which has a single 1 in each parenthesis, with all other entries in that parenthesis being zero. Thus, if we are concerned with the function at the point $(x_{12}\, x_{22} \ldots x_{n2})$ then the corresponding linear representation is:

$$(010\ldots0)\,(010\ldots0) \ldots (010\ldots0) \qquad (1)$$

It is perhaps useful to anticipate some later results at this point and note that a continuous independent variable is simply the limiting case as the number of partitions over the range of the variable becomes infinite, with the unit impulse becoming the Dirac delta function in the limit for a sharply defined value of the variable. Also, the case of the statistically specified value of the independent variable corresponds to a distribution of values among the partitions in the discrete case or over the range in the continuous case. The associative memory studies are not concerned with continuous variables, although the mathematical model suggests this generalization. The restriction to discrete variables is conditioned by an ultimate concern with real world problems, and by considerations of the process being modeled.

The primary problem in "addressing" data for entry into a memory, as has been pointed out earlier for both the normal computer memory and for living associative memories, is the generation of an address which can be meaningfully related to the information being stored. Without concerning ourselves at this point with what is to be done with an "address," or indeed with what is meant by storage, we can discuss one systematic scheme for generating a portion of the address. Consider two points in S-space represented by their linear vectors (in the same form as (1)). Ignoring for the moment the value assumed by the function at these points, the first question, and certainly the most natural one, is "How close are the points to each other?" In order to discuss "closeness" it is necessary to introduce a metric. Ideally the metric should have an intuitive appeal as a natural type of distance function in the space. The most commonly used metric, the Euclidean, fails in the present case because of the nature of the functions to which it is to be applied, and the use to which it is to be put. The way in which the variable ranges were defined for the base vector is suggestive of a frequency function and the metric which is required should give a measure of the agreement of these functions. The obvious course, which is not acceptable although this is not equally obvious, would be to convert the frequency functions to cumulative distribution functions over

the same range and then to use a Euclidean metric. Thus, in the case of well behaved X vectors where only a single unit impulse exists we find

becomes

One may now introduce the Euclidean metric for the cumulative distribution functions and at the same time preserve the essential intuitive meaning of the metric. Define D to be given by

$$D^2 = \sum_{i=1}^{n} \sum_{j=1}^{k_i} \left(F(x_{ij}) - F'(x_{ij}) \right)^2 \tag{2}$$

It is evident that this is a metric. The following example will illustrate why this construction is satisfactory for F(X) and not for X. Let the partition of a single variable in the vector V be given by

(0010000000)

and let the corresponding variable in the vector V′ be given by

(0000000001)

These code words are representations of the X's on some suitably chosen scale. Then a simple application of (2) with F(X) replaced by X would give a value 2, but this same value would be obtained for any other vector pair in which the vectors were not identical. On the other hand $D^2 = 7$, which is a measure of the separation of the code words in a perfectly normal manner.

The metric over the frequency functions which is suggested by the above argument is given by

$$d^2 = n - \sum_{i=1}^{n} \sum_{j=1}^{k_i} x_{ij} x'_{ij} \tag{3}$$

which has the simple interpretation of testing whether the functions are identical in each partition. Thus, by computing either D or d one can measure the closeness of one vector in S-space to another. Actually, a great deal more information is contained in D than in d; however, it has not been possible, with the present model, to make use of this additional information.

The base vector X, or the associative criterion as it is commonly called, contains adequate descriptive material to allow the classification and hence addressing of f(X) for storage. If it does not satisfy this requirement then it is an incomplete or ambiguous criterion, and neither the memory nor a human operator could perform an unambiguous associative classification. Assuming that the associative criteria are complete and that all of the data points for storage are available in parallel, then the storage procedure is very

simple. The data words are all examined to determine which of the criteria are nearest and in what order they lie. Addresses in the memory are then assigned accordingly, so that points which are near each other in the input space are near each other in the memory.

Unfortunately, learning experiences or data points for storage in a memory occur serially rather than in parallel so that it is not possible, in general, to compute directly the distance between all pairs of base vectors. In a normal computer memory this is solved by retrieving from storage the exact form of the entry and then computing D or d. It is this basic operation which makes sorting, collating, etc. assume such astronomical proportions for many practical problems. In the distributed memory, as we have already noted, this information is diffused throughout a large structure so that direct computation becomes impossible.

In the present model this problem is solved by computing the distance of each associative criterion from a large number of fixed reference points in the space, rather than by a direct comparison with other associative criteria which either have been or are to be addressed for storage. These fixed reference points (or vectors) are called association vectors, and the entire set of such points when represented in an ordered array is called an association matrix by analogy with the association matrix which may be derived from real neural nets, and which has been used by Rosenblatt and others in simulated neural net studies.

There are N elements, and hence N dimensions, in each of these association vectors, where

$$N = \sum_{i=1}^{n} k_i \tag{4}$$

The points in the input space, n-dimensional, were mapped onto the vertices of the N-dimensional binary valued hypercube situated in the vertex of the first octant in N-space. Accordingly in the model, each of the coordinates of the reference points is restricted also to be either a 0 or 1; i.e., the reference points are themselves vertices of this same N-dimensional hypercube. It is most convenient to consider these coordinates to have been assigned a value randomly, with the probability of a 1 being entered at any position being p. As will be shown in the following development this, or some equivalent restriction, is essential to the operation of an associative addressing procedure such as the one proposed here. However, this definition brings with it a difficulty which did not exist in the computation of the distance between a pair of associative criteria. There, since the vectors, X, were restricted to being well behaved, the distance was either 1, within a single partition of a variable, or 0 depending on whether the value of that variable was the same in the two associative criteria, or dissimilar, respectively. Now there is the possibility that a single partition could contain all 1's, for an example, in which case the distance function d would be 0 in that partition, irrespective of which associative criterion it was compared to. This weakness of the metric d in the enlarged space, i.e., the permissible S-space and the space made up of the reference points, has not prevented the model from working, as the following analysis will prove, but it does indicate that the system is operating on only a very small portion of the total information available. We will still define the distance in this enlarged space by d and (for the moment) ignore the degeneracies which can occur.

If there are R association vectors, then the association matrix is an N x R binary matrix, A, as defined earlier. The product of the base vector, considered as an N element row vector, and A, is a row vector with the R elements being the metric d^2 less n in each case. Call the elements of this product matrix s_i by analogy with the operation being modeled from the biological system, namely association of a set of stimuli with a set of neuron synaptic connections (and values).

$$S = X \cdot A \tag{5}$$

Now we will introduce a nonlinear operation which is analogous to the all-or-none response of a neuron to the integrated synaptic stimuli. A new (1 x R) row vector, with elements α_i will be generated from the S vector by forming the following logical decision for each element of S

$$\alpha_i = \begin{cases} 1 \text{ if } s_i \geq \delta \\ 0 \text{ if } s_i < \delta \end{cases} \tag{6}$$

Thus we have generated from the product $(X \cdot A)$ a binary (1 x R) row matrix which has a 1 in the i-th position if the particular base vector X is not more than $n - \delta$ distant from the association vector A_i. Again it is worthwhile emphasizing that for R sufficiently large the vectors α and α' specifying the distance of X and X′ from the R association vectors $A_1 A_2 \ldots A_R$, are at the same time adequate to define the distance between X and X′. This is true in spite of the weak definition of the metric in the extended space. Since this fact is a key element in the following development and use of this model, it is of interest to demonstrate its validity in the limit, for "R sufficiently large."

Following our earlier usage of these symbols the vectors in S-space will be assumed to have N elements grouped in n partitions or variable ranges in each of which a single one occurs. Now let A be the complete $N \times 2^N$ binary matrix in which every possible N bit binary word appears as a column of the matrix. The S matrix is then a (1×2^N) row matrix formed by multiplying A by X. If we consider two vectors, say X and X′, in S-space the distance between them is found from (3) to be

$$d^2 = n - X' X^T \tag{7}$$

The first step in demonstrating the conjecture made above that the weak metric is a complete one is to show that d^2 as given by (7) may be computed from S and S′ and nonspecific knowledge about S-space:

$$XA = S \text{ and } X'A = S'$$

form

$$X'A(XA)^T = S'S^T \tag{8}$$

or

$$X'AA^T X^T = S'S^T$$

The unusual properties of the complete binary matrix make it possible to greatly simplify the left hand side of (8) through a manipulation of AA^T

$$AA^T = a_{ij} \begin{cases} a_{ij} = 2^{N-2} & \text{if } i \neq j \\ a_{ij} = 2^{N-1} & \text{if } i = j \end{cases}$$

irrespective of any ordering of the A_i in A. Thus AA^T can be decomposed to the sum of two particularly simple matrices, i.e.,

$$AA^T = 2^{N-2} \begin{pmatrix} 1 & \cdots & 1 \\ \cdot & & \cdot \\ \cdot & & \cdot \\ \cdot & & \cdot \\ 1 & \cdots & 1 \end{pmatrix} + 2^{N-2} I \tag{9}$$

This formula may now be re-introduced into (8) to give the following very simple form

$$2^{N-2} X' \begin{pmatrix} 1 & \cdots & 1 \\ \cdot & & \cdot \\ \cdot & & \cdot \\ \cdot & & \cdot \\ 1 & \cdots & 1 \end{pmatrix} X^T + 2^{N-2} X' X^T = S' S^T \tag{10}$$

If one notes that the effect of the matrix operation in the left hand member is merely to tally twice the non-zero elements of X' and X^T and that this number is n, then (10) can be reduced and the resulting equation solved for $X'X^T$

$$X'X^T = \frac{1}{2^{N-2}} \left\{ S'S^T - n^2 2^{N-2} \right\} \tag{11}$$

Finally the distance between X and X′ is found to be

$$d^2 = n - \frac{1}{2^{N-2}} \left\{ S'S^T - n^2 2^{N-2} \right\} \tag{12}$$

where the right hand term is expressed only in terms of S, S′ and parameters descriptive of the general characteristics (n and N) of S-space as was desired. This proves that for some R, in this case $R = 2^N$, the metric defined by S is a complete one, that is to say, that the location of a point in S-space is unambiguously specified by the S vector.

The foregoing theorem is very nicely illustrated by the following example. Let S-space consist of two variables, with two possible values for each, so that n = 2, N = 4. Thus there are only four X vectors in S-space; i.e., we have a space consisting of only the four points:

$$X_1 = 0101$$

$$X_2 = 1001$$

$$X_3 = 1010$$

$$X_4 = 0110$$

The A matrix will then be a (4 x 16) matrix of the following type in which the order of the columns is immaterial.

$$A = \begin{pmatrix} 0000000011111111 \\ 0000111100001111 \\ 0011001100110011 \\ 0101010101010101 \end{pmatrix}$$

The corresponding vectors giving the distance of X_i from each of the association vectors (rows in A) are

$$S_1 = (0101121201011212)$$
$$S_2 = (0101010112121212)$$
$$S_3 = (0011001111221122)$$
$$S_4 = (0011112200111122)$$

In order to compute the distance between the various X_i using formula (12) one will require the terms $S_i S_j^T$. These are computed using the elements of the S_i array above and found to be:

$$S_{12} = 20 \qquad S_{23} = 20$$
$$S_{13} = 16 \qquad S_{24} = 16$$
$$S_{14} = 20 \qquad S_{34} = 20$$

where

$$S_{ij} = S_i S_j^T = S_i^T S_j$$

and

$$n^2 2^{N-2} = 2^2 2^2 = 16$$

Thus the distances between the X_i are

$$d^2_{12} = 1 \qquad d^2_{23} = 1$$
$$d^2_{13} = 2 \qquad d^2_{24} = 2$$
$$d^2_{14} = 1 \qquad d^2_{34} = 1$$

which are, of course, the results one obtains by applying (7). The main reason for exhibiting the above example (which could have been dispensed with since the theorem (12) was proven in general) was to generate a complete set of S vectors for a simple example so that the α vectors could be derived and examined. First assume $\delta = 2$ in formula (6). The α vectors are then:

$$\alpha_1 = 0000010100000101$$
$$\alpha_2 = 0000000001010101$$
$$\alpha_3 = 0000000000110011$$
$$\alpha_4 = 0000001100000011$$

Let $\alpha_{ij} = \alpha_i \alpha_j^T = \alpha_i^T \alpha_j$

so that

$$\alpha_{12} = 2 \qquad \alpha_{23} = 2$$

$$\alpha_{13} = 1 \qquad \alpha_{24} = 1$$

$$\alpha_{14} = 2 \qquad \alpha_{34} = 2$$

The order of these results is suggestive of the table of distances tabulated in the preceding discussion. Before this lead can be investigated a set of identities relating the α_i and X_i must be established.

First consider the form of the S matrix, $S = X \cdot A$. It will have integer elements with all integer values 0 thru n being represented. This can be decomposed into the following useful form

$$S = \sum_{i=1}^{n} S(i) \tag{13}$$

where each of the S(i) is the matrix whose elements are all either i or 0. This representation of S allows one to represent α analytically in the following form.

$$\alpha = \sum_{i=\delta}^{n} \frac{1}{i} S(i) \tag{14}$$

where δ is the threshold value chosen for the logical operation in (6).

The basic problem is: given α and α', is it possible to compute the distance between X and X', or in other words is the weak metric α complete in the extended space? Note that the value of the threshold δ against which α is computed is left unspecified, and hence can be considered as one of the parameters of the problem. Assume δ to be n; in other words the association vector must have a 1 in precisely every code position in which the X vector has a 1 if the corresponding α element is to be nonzero. As we have noted before, this result is not affected by the presence of extra 1's in the association vector, and in fact this excess (or overlapping) coverage will later prove to be the key to the operation of the model. α is now a single term and may be written as

$$\alpha = \frac{1}{n} S(n) \tag{14-a}$$

It is now possible to compute the number of nonzero elements in α rather simply. If there are n partitions with k_i discrete values possible in the i-th partition, and if the association matrix is the same complete binary matrix discussed before, the number of elements whose value is n in S is

$$\phi_o(n) = \prod_{i=1}^{n} 2^{k_i - 1} \tag{15}$$

If we consider another vector with (n - 1) of the partitions identical, with a single element different in a partition possessing k_j elements, then the number of $\alpha_i = n$ in the same relative positions of the two α vectors is

$$\phi_1(n) = \prod_{\substack{i=1 \\ i \neq j}}^{n} 2^{k_i - 1} \times 2^{k_j - 2} \tag{16}$$

This argument is simply extended to the case where the two X vectors differ in an arbitrary number, $p \leq n$, of partitions

$$\phi_p(n) = \prod_{\substack{i=1 \\ i \neq j}}^{n} 2^{k_i - 1} \prod_{j=j(1)}^{j(p)} 2^{k_j - 2} \tag{17}$$

It is possible to express X as a function of α as defined by (14-a), $\phi_o(n)$ as given by (15) and A, or more precisely A^T. We shall now prove the following theorem.

$$X = \left[\frac{\phi_o(n)}{2}\right]^{-1} \left\{\alpha A^T - \frac{\phi_o(n)}{2} (1 \ldots 1)\right\} \tag{18}$$

First consider the term αA^T. α_i is a one, if and only if, s_i is equal to n, which is to say that the binary cycles of the rows of A, assumed to be ordered as shown in the previous example for this argument (which correspond to a 1 in the elements of the X matrix) must all be positive, or equal to one in this case. Thus, the operation of matrix multiplication followed by the logical decision (6) is exactly analogous to a double logical extract operation. If we now reverse this operation and extract α against A^T, i.e., the logical operation

$$\sum_{i=1}^{n} \alpha_i A^T_{ij} = \beta_j \tag{19}$$

The rows of A which contributed to making s_i pass the threshold test will now be "in phase" with the variations of α_i; that is they will be positive in all cases when α_i is positive. All other rows will assume all possible binary arrangements while these are fixed and consequently will be positive and zero equally often. Thus the "in phase" rows will contribute a 1 for each 1 in α, or what is the same thing, if R is understood to represent a general "in phase" row of A and R^T is the corresponding column of A^T

$$\alpha \cdot R^T = \sum_{i=1}^{n} \alpha_i = \phi_o(n) \tag{20}$$

and similarly if r is a general "out of phase" row of A, and r^T is the corresponding column of A^T

$$\alpha r^T = 1/2 \sum_{i=1}^{n} \alpha_i = \frac{\phi_o(n)}{2} \tag{21}$$

But if we recall the definition by which the terms "in phase" and "out of phase" were introduced, namely that an in phase row of A is one which corresponds in position to a nonzero element of X and conversely for the out of phase rows, we see that the vector X is derivable from αA^T. αA^T is therefore a (1 x N) row vector which has $\frac{\phi_o(n)}{2}$ in every position in which X has a zero, and $\phi_o(n)$ in every position in which X has a one. Equation (18) is thus obtained by a trivial algebraic manipulation upon αA^T.

The original question which prompted this discussion can now be answered in the affirmative--for an appropriate choice of the threshold, δ, and for R sufficiently large (at least for the case $R = 2^N$ as we have demonstrated) X can be expressed as a function of its distance from each of the association vectors in terms of the weak metric α. Certainly if X and X′ can be expressed in this manner, the distance between them d^2, can also be expressed in terms of the same variables since d^2 is defined on X and X′. This is incidentally a much stronger result than the one which we set out to prove which only required that we be able to find d^2, not that we be able to solve uniquely for X and X′. The weaker case is interestingly enough much harder to prove; however, it is also much more valuable in the application of the construction to modeling real associative memories.

Having completed all of the preliminary formulations required to compute d^2 for any vector pair, X and X′, in S-space it is an interesting exercise to compute the formula for d^2 itself.

$$d^2 = n - \left\{\frac{2\alpha' A^T}{\phi_o(n)} - (1 \ldots 1)\right\}\left\{\frac{2\alpha A^T}{\phi_o(n)} - (1 \ldots 1)\right\}^T$$

This may be expanded into the following form which is more amenable to simplification and computation.

$$d^2 = n - N + \frac{4}{\phi_o^2(n)}\alpha' A^T A \alpha^T - \frac{2}{\phi_o(n)}(1 \ldots 1) A \alpha^T - \frac{2}{\phi_o(n)}\alpha' A^T \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \tag{23}$$

The two matrix expressions

$$(1 \ldots 1) A \alpha^T \text{ and } \alpha' A^T \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$$

may be reduced to their equivalent scalers. First by the argument of the previous section $\alpha' A^T$ is a (1 x N) row matrix which has the element $\frac{\phi_o(n)}{2}$ in every position in which X had a zero and the element $\phi_o(n)$ in every position in which X had a one. A similar statement applies to the column matrix $A \alpha^T$. The final matrix operations

$$(1 \ldots 1)X \text{ and } X \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$$

simply sum these elements so that one may summarize these results by

$$-\frac{2}{\phi_o(n)}(1 \ldots 1) A \alpha^T = -\frac{2}{\phi_o(n)}\alpha' A^T \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} = -(N + n) \tag{24}$$

which makes it possible to express (23) in the following simple form:

$$d^2 = (N + 3n) - \frac{4}{\phi_o^2(n)} \alpha' A^T A \alpha^T \tag{25}$$

which may itself be simplified even further by expanding A^TA into a summation of simple matrices; however, (25) suffices for the purposes of the present exposition.

The results which have been derived for the discrete well behaved S-space and their extension to continuous input spaces with statistically defined arguments can now be given a simple geometrical interpretation. To do this we shall require the concept of a Hamming weight[16] for the points in the extended S-space. The Hamming weight of a binary vector is derived from the extremely useful concept of the distance (Hamming distance) between a pair of binary code words introduced by R. W. Hamming[17] in discussing the error correcting capability of codes. It is defined to be the number of elemental positions in which the words differ, and hence is closely related to our metric D^2, if we replace "elemental position" in the definition by "partition." A single error in transmitting a binary code will obviously result in a code at a Hamming distance of 1 from the original code, and similarly a distance of d will correspond to d errors. The Hamming weight of an n-tuple V is defined to be its Hamming distance from the zero n-tuple.

Our restriction of the X vectors in S-space to "well-behaved" vectors can be stated equivalently as a restriction to code points with a Hamming weight of n. An essential point to note though is that not all code points with a Hamming weight of n are permissible X vectors. Thus there are N_H points with a Hamming weight of n and only T well behaved X vectors.

$$N_H = \binom{\sum_{i=1}^{n} k_i}{n}$$

$$T = \prod_{i=1}^{n} k_i$$

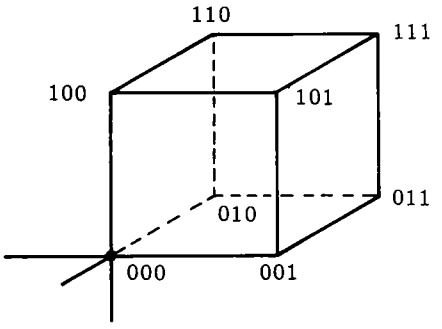
A very useful graphic representation of binary code words which we will use throughout this paper associates the vertices of the unit side hypercube in the first octant of n-space with the binary codes which

---

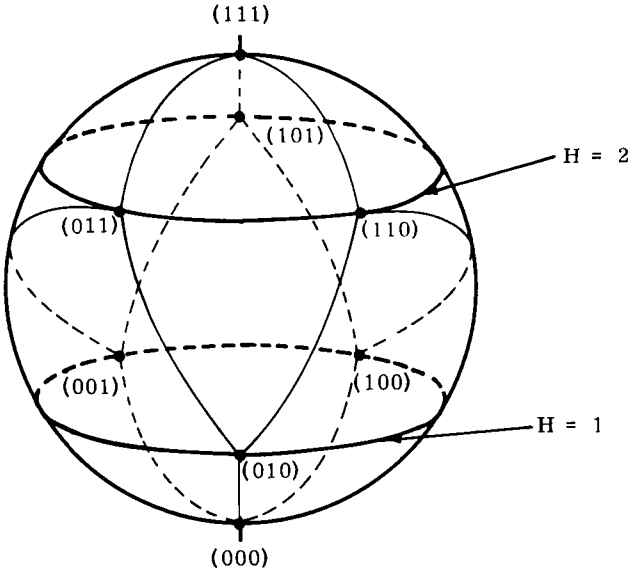
[16] W. W. Peterson, Error-Correcting Codes, John Wiley and Sons, Inc., New York, 1961.

[17] R. W. Hamming, "Error Detecting and Error Correcting Codes," Bell System Technical Journal, **29**, pp. 147-160, 1950.

are their n-tuple representations. Thus in 3-space we have

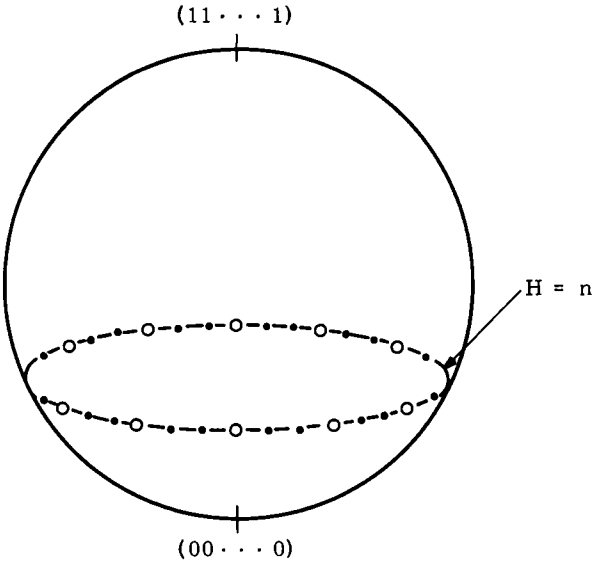


A hypersphere of radius $\frac{\sqrt{n}}{2}$ can be circumscribed about every such hypercube. The most important point for our geometric discussion though is that the circle drawn through all vertices of the same Hamming weight (points in the extended S-space) of the hypercube on the surface of the hypersphere will be equidistant from the poles of the sphere at (00 . . . 0) and (11 . . . 1). Obviously every point which can exist in the extended S-space lies on this hypersphere. We can now project these hyperspheres on an ordinary sphere in 3-space (or even a circle in 2-space) in which case there will be n-1 circles around the sphere corresponding to Hamming weights of n-1, n-2, . . . 2, 1. Thus the simple projection of the 3-variable hypercube on the sphere yields a representation as shown in the following figure.



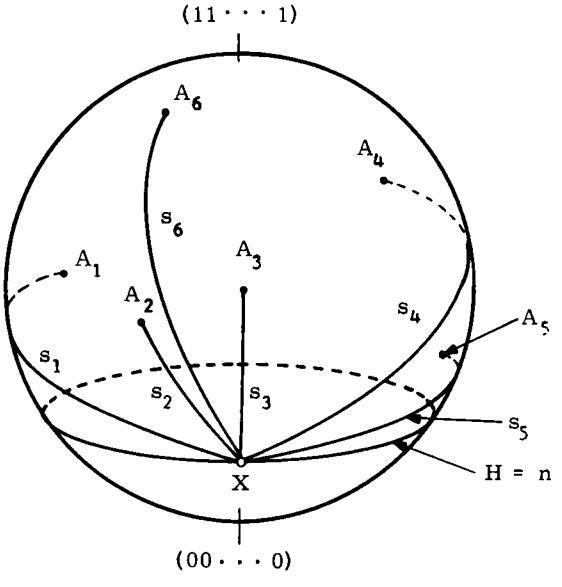
Similarly for higher dimensional projections we get n-1 circles, each with $\binom{n}{w}$ points on it of Hamming weight, w. In general, disregarding the points which do not have a Hamming weight of n, we have as the

input S-space (well behaved X vectors) projection on the sphere:



Where the symbol o represents permissible X vectors and the symbol · represents other points which have a Hamming weight of n.

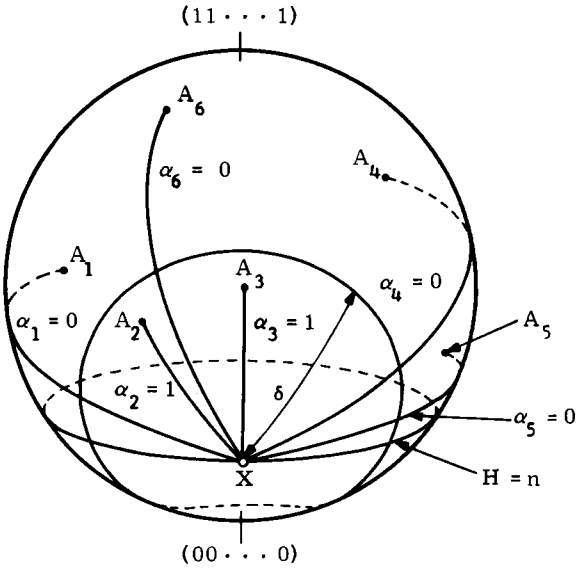
Obviously the distance, d, from any such point, X, could be computed. This is precisely the S-vector discussed above, i.e.,



The nonlinear thresholding operation consisted of testing each such distance against a reference δ to see if it was greater than or equal to the reference.

$$\alpha_i = \begin{cases} 1 \text{ if } s_i \geq \delta \\ 0 \text{ if } s_i < \delta \end{cases}$$

Recalling the way in which s was formed, i.e., $s_i = X \cdot A_i$, we see that s is largest for the closest reference points. Thus the threshold operation can be geometrically represented by constructing about X as a center a hypersphere of radius δ and determining which of the reference points lie inside or on this sphere. The α_i corresponding to these points will be assigned a value of 1 by the threshold test. The intersection of the hypersphere about X with the hypersphere on which X occurs is a hypersphere of dimension (n - 1) as is shown in the following projection on a sphere in 3-space.

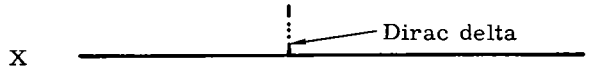


In the case discussed in the text all of the points on the hypersphere were used as reference points (complete binary matrix) and consequently the distribution of reference points on the surface was a uniform one. The completeness proofs, which were based on a $\delta = n$, were equivalent to proving that if we knew which points lay in the hemihypersphere centered on X, we knew X unambiguously.

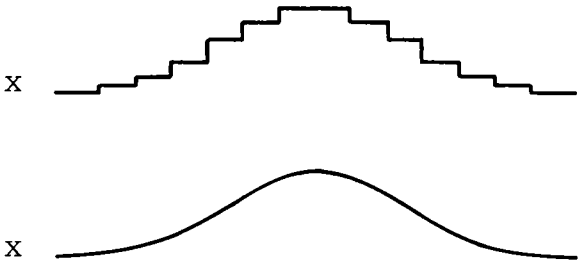
This completes the geometrical description of the mapping from N dimensional hyperspace in which the extended S-space is embedded to the R dimensional memory space or M-space. By appealing to the geometric concepts just developed it now becomes feasible to demonstrate the consequences of relaxing the rather stringent requirement that the X-vector be well behaved.

In analogy to the simple representation of the well behaved X-vector introduced earlier

We can represent the limiting case, as the partitioning of the variable range [a, b] becomes infinitesimally small

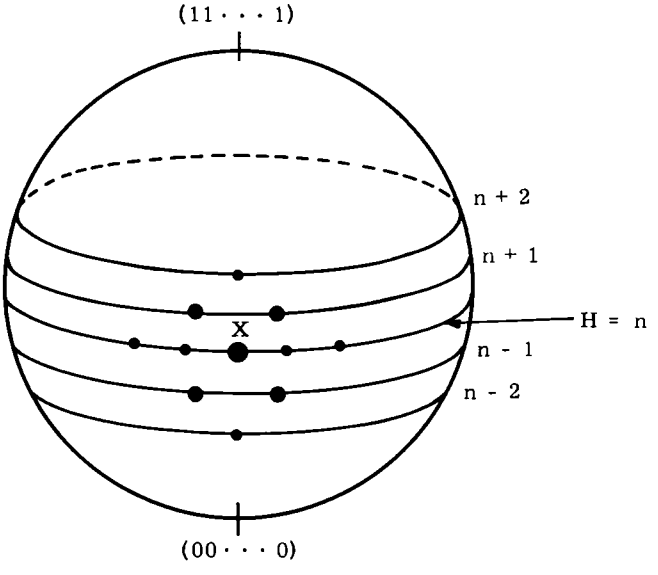


The two statistically specified variables (discrete and continuous) behave as follows:



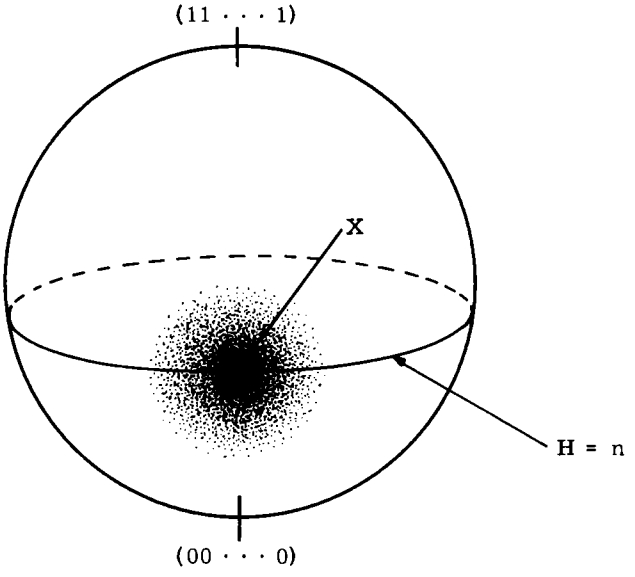
The geometric representation of the projection of the various X vectors derived above is fairly straightforward. The first new type introduced above, the well defined discrete value in a continuous range, merely results in the permissible X-vectors mapping into a dense set of points on the circle H = n in the projection onto the hypersphere.

The statistically specified discrete variable, though, is quite interesting. What this does is associate a probability p_{ij} with the i-th element of the j=th partition. Therefore, a probability $P_{(i1)(j2)\ldots(\eta n)}$ is associated with every permissible point in X-space, which may incidentally be zero. This probability is the likelihood, given X as the input, that the image of the argument is the point $(0\ldots i\ldots 0)(0\ldots j\ldots 0)\ldots(0\ldots \eta\ldots 0)$. For the first time in this discussion the points in the input S-space to which measurements of distance are to be made are not confined to the circle H = n, but lie spread over adjacent circles. This is shown schematically in the following figure.



where the symbols ●, •, and • represent some scale of descending probabilities of occurrence. Thus distance must be computed to all members of this set which have nonzero probabilities of occurrence. This procedure can be extremely complicated for even small problems (few dimensions and discrete levels in the input space).

The last case, the statistically specified continuous variable, is a direct extension of the above in which the distribution of points about X becomes a dense set.



In the above figure the likelihood of this behavior occurring is indicated by the shading of the region surrounding X.

The metric used for the computation of distances to the various reference points is the same for the statistically defined discrete case as in the foregoing analysis. The extension to the continuous case in terms of integrals rather than summation is obvious. Other than this methodological change the associative addressing scheme remains the same as that discussed before.

The foregoing discussion indicates the manner in which the techniques developed here could be extended to other input spaces; however, for the balance of this paper we shall concern ourselves exclusively with discrete well behaved input vectors and their attendant geometry. This restriction is conditioned by an ultimate concern with real world problems, and realizable systems.

In view of the lengthy digression to prove that the metrics d^2 and D^2 were complete in the extended space and to interpret these results geometrically, it is probably desirable to summarize the reasoning and results which have led to this point. We began with the generation of a meaningful address for the storage of a data word from the data itself. It was pointed out that if all the data could be made available in parallel (at the same time) a comparison of each word to every other word would allow a classification into collections which were similar in the sense that they were "near" each other. Unfortunately, this is not feasible, and it is the resulting series recall problem which plagues digital computers on this type of problem. Our interpretation of the existing neurophysiological evidence of the mechanisms of memory trace formation has convinced us that nature has instead evolved the technique of fitting the data word against a very large number of sloppy templates, and the fact that such a scheme works in the living associative memory is an indication that, in at least one type of system, redundancy can compensate for

sloppiness in classification. At this point we introduced the concept of an association matrix and an operation by which we measured the distance between each point in this "association space" and the input points in the embedded S-space, rather than measuring directly the proximity of the input points among themselves. The association vectors were literally our templates and the distance function D^2 was the measure of fit. For a particular case it was shown that this metric was complete for the extended space, that is that any classification of the input data words which could have been made by examining all of the inputs in parallel could be made serially by sequential fitting of the words to the association vectors. At this point we emulated nature and made our test of the quality of fit to the templates as weak as possible, i.e., a single bit of information is retained from each template test, and again we were able to show that for at least one space, which was based on reasonable assumptions, this metric was also complete.

Distributed Storage

In the foregoing analysis of a possible associative addressing technique we specifically restricted attention to that fraction of the data word which specified the location of the data point in S-space. This base vector, X, was shown to preserve its metric relationship to other base vectors under a linear transformation followed by a threshold operation which transformed X into a new vector, α. It is this α which we propose to use as an associative address in storing or retrieving data from an associative memory. The most important qualitative point in this procedure is that the transformation maps points from the n-dimensional S-space into points in a 2^N dimensional space. This tremendous increase in the number of points in space is the basis for the storage of the balance of the data word.

In general, if we consider an n-dimensional space we can position an $(n-1)$ dimensional hyperplane through this space so that the perpendicular distances to n arbitrary points (which are assumed to constitute a basis for the space) from the plane may be arbitrarily specified. The restriction that the points must constitute a basis is equivalent to requiring linear independence of the base vectors from the origin of the system to the points. It is this latter statement which is the essential restriction for the weaker case in which fewer than n points are being fitted to a hyperplane in n-dimensional space.

Let the equation of the hyperplane be given in the form

$$XQ = d$$

where the X and Q are nonzero, n-tuples. Then the equivalent normal form is [18]

$$XL = p$$

where

$$\ell_i = \frac{Q_i}{||Q||}$$

are the cosines of the direction angles of the hyperplane and p is the length of the normal from the origin of the n-space to the hyperplane. This normal, which is of course unique, may be most easily expressed

[18] D. M. Y. Sommerville, An Introduction to the Geometry of N Dimensions, Dover Publications, Inc., New York, N.Y., 1958.

as an n-tuple, with one free parameter, ρ.

$$N = \rho L$$

The ℓ_i are now the direction cosines of the normal. With the notions just developed it is now an easy matter to express the distance d_i of an arbitrary point $\alpha_{(i)}$ to the hyperplane.

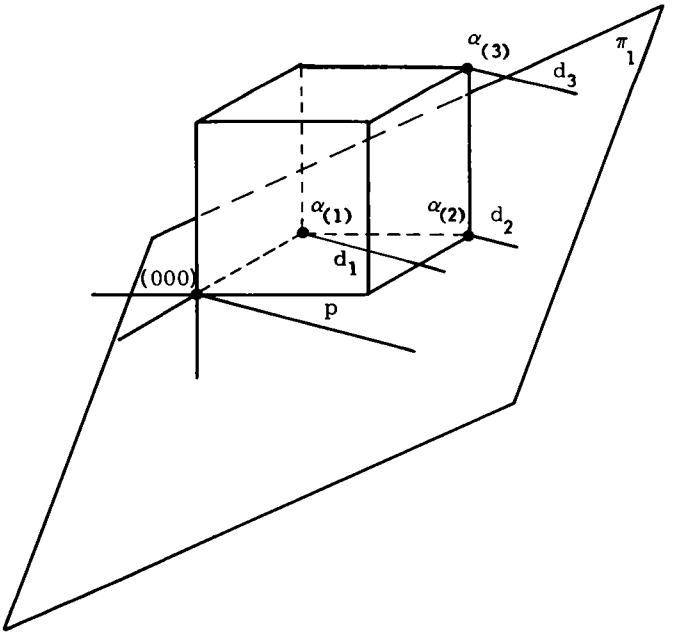
$$d_i = p - \alpha_{(i)} \cdot L$$

But this is a linear system, in general nonhomogeneous, of the form

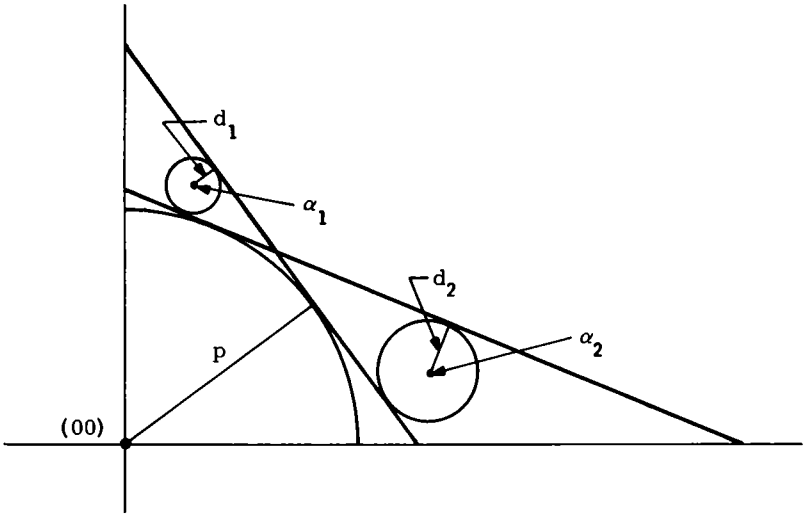
$$\alpha \cdot L = G$$

where G is the n-tuple of elements $(p - d_i)$. Since a solution of such a system for arbitrary G is possible if and only if the elements of the linear system $\alpha \cdot L$ are linearly independent, we arrive at the earlier restriction that the vectors to the points, $\alpha_{(i)}$, must be linearly independent and therefore, in the case in which there are n-points, must constitute a basis for the space.

As was shown earlier in the description of the formation of S-space and of the procedure by which points in it are mapped into points in A-space, these image points are restricted to be vertices of the n-dimensional binary valued hypercube positioned in the vertex of the first octant of the space. Actually the restrictions were much tighter than this; however it suffices to consider any of the vertices of the hypercube in the following development. First we have just shown that a hyperplane can be found such that it lies at a specified, but arbitrary, distance from each of n linearly independent points in n-space, as shown in the following example in 3-space.



Obviously π_1 is not the only such plane, and in fact may not be the only such plane at a distance p from the origin, as the following example in 2-space illustrates.



This is not too crucial to the foregoing analysis since any of these π_i planes would constitute a solution to the linear system; however, in the following sections in which the planes are being solved for by an iterative procedure the possibility of several solutions (hyperplanes) which are nearly the same, may lead to instability in the solution; i.e., the algorithm cannot determine which of two adjacent hyperplanes it is in (or near) and hence oscillates between them. This difficulty can be taken care of by removing some of the unnecessary degrees of freedom in the solution. Instead of considering a hyperplane at an arbitrary distance p from the origin, which lies at arbitrary but specified distances from the $\alpha_{(i)}$, let us consider a hyperplane through the origin which lies at arbitrary relative distances from the $\alpha_{(i)}$. In other words find Q such that

$$\frac{\alpha_{(i)} \cdot Q}{\alpha_{(j)} \cdot Q} = \frac{d_i}{d_j}$$

Recalling that the $\alpha_{(i)}$ are vertices of the n-dimensional hypercube, we see that this is satisfied by letting

$$d_i = \frac{\alpha_{(i)} \cdot Q}{||Q||}$$

which is the quantity we propose to use in storing f(X); i.e., let

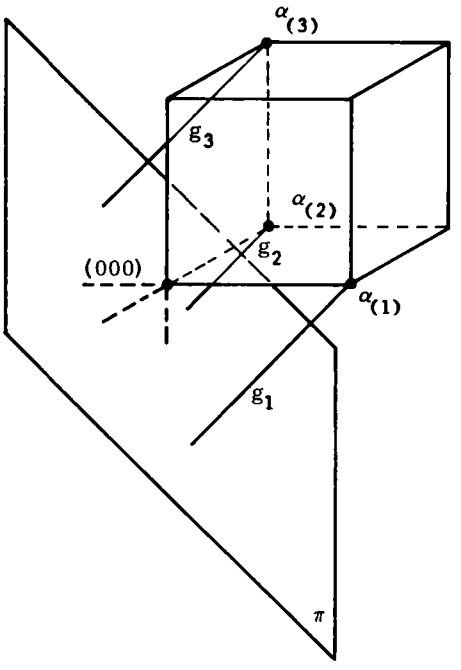
$$f(X_{(i)}) = d_i \, ||Q|| = \alpha_{(i)} \cdot Q$$

We shall hereafter refer to the n-tuple Q as the hyperplane with the understanding that what is actually meant is that Q is the coefficient array of

$$XQ = 0$$

which is a linear equation in n unknowns and hence is the equation of the hyperplane.

It is now possible to construct, in the instance where n linearly independent points are being fitted, a single plane through the origin with the required relative distances, say g_i. The following figure shows this for the examples used earlier from 3-space.



If fewer than n-points are to be "stored" using this technique, then there are correspondingly infinitely many solutions for each free parameter so introduced. The essential point though is expressed by equation (28), namely that the function to be stored at a point $\alpha_{(i)}$ is given by the normal distance to the hyperplane Q times the norm of Q, i.e., $\alpha_{(i)} \cdot Q$.

First we shall investigate the existence of Q(s) such that (28) can be satisfied and then we shall concern ourselves with algorithms by which a Q may be found. In the well behaved input vectors which were considered in the previous section, only a single nonzero element appears in each partition; thus the total points in S-space which need be considered is

$$T = \prod_{i=1}^{n} k_i \tag{29}$$

This may be contrasted to the total number of vertices of the N-dimensional hypercube of

$$T_s = 2^N \tag{30}$$

Use of the complete binary matrix A will demonstrate the existence of solutions in the limit (for R sufficiently large) although we shall be primarily concerned with the relaxation of these assumptions later on. Equation (30) actually gives the number of dimensions of the space into which S-space is transformed by A, so that the complete binary hypercube in this association or A-space has a total of vertices.

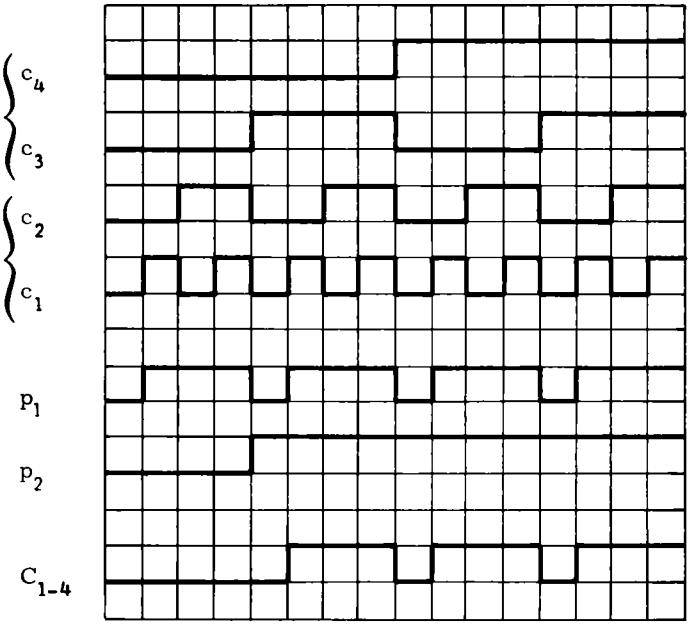
$$T_A = 2^{2^N} \tag{31}$$

Fortunately the transformation is one-to-one so that only T out of the 2^{2^N} possible vertices can have pre-images. Formula (15) gives the number of nonzero elements in α, and hence the Hamming weight of an α vector under the above assumptions. Since only a very small fraction of the 2^{2^N} vertices have pre-images, and since the Hamming weight is $\phi(n)$, the actual number of variables involved in specifying Q may differ significantly from the 2^N possible. For an example, the α vectors computed earlier, under the assumptions that n = 2 and N = 4, were of the form

$$\alpha_1 = \epsilon_5 + \epsilon_7 + \epsilon_{13} + \epsilon_{15}$$
$$\alpha_2 = \epsilon_7 + \epsilon_{11} + \epsilon_{13} + \epsilon_{15}$$
$$\alpha_3 = \epsilon_{10} + \epsilon_{11} + \epsilon_{14} + \epsilon_{15}$$
$$\alpha_4 = \epsilon_6 + \epsilon_7 + \epsilon_{14} + \epsilon_{15}$$

where the ϵ_i comprise the natural basis for E_{2^N}. $\phi_o(n) = 4$ in each case, but only nine of a possible sixteen $\left(2^{2^2}\right)$ variables appear. As was noted earlier the mapping into A-space is one-to-one (again keeping in mind our stated assumptions and restrictions) so that there are exactly four equations in nine unknowns in this system, and hence infinitely many solutions. Or what is the same thing: given any four values of f(X), say $f(X_i)$ corresponding to the derived α_i, infinitely many eight-dimensional hyperplanes in a 9-space can be found such that α_i is $f(X_i)$-distant, etc. Obviously if the above statement holds for a sub-space of T_A, it holds for T_A itself.

We shall now investigate the number of variables and equations in general, as functions of the partitioning of S-space, and whether this system is a compatible one. There are only T pre-images in S-space due to the assumption of a fixed Hamming weight of n, and since the transformation by the A matrix is one-to-one, there are exactly T α vectors, or equivalently T linear equations involving some selection of the possible 2^N variables in A-space. The only real problem then is to determine how many of these variables will be involved. As we have already noted, the occurrence of 1's in α_i is very closely correlated with the binary cycles which can contribute to the particular α. Unlike the previous problem where the problem was the unique inversion of the logical threshold operation to re-compute the unique X_i associated with an α_i, here we are concerned with just which periods of the whole binary cycle scheme can produce a 1 in some α. This may be rephrased to ask: just which periods, of the binary cycles in the A matrix (which has been assumed ordered simply for convenience in exhibiting the cycles) have at least one member positive among the cycles associated with each partition in any particular time interval? The following figure, based on the earlier example of n = 2, N = 4, should illustrate this very well.



The above figures are of course precisely the composite rhythmic structures made famous by Joseph Schillinger[19,20] in his researches on the mathematical basis of the arts and the mathematical theory of musical composition. c_i represents the individual binary cycles which we have referred to before; however, p_1 and p_2 are the result of applying an inclusive logical 'or' to the elements of each partition. Incidentally it makes no difference if the partitions are chosen in some other order; obviously C_{1-4}, or C_{1-N} in the general case, will be the same irrespective of any ordering of the k_i. C_{1-N} is the result of forming a logical "and" among the p_i; and the positive or true statements in C_{1-N} correspond to periods when some α_i would have a 1 occurring. Thus, in this simple example, as we could have ascertained by counting the ϵ_i which appeared in α_1 through α_4, there are nine variables involved. In general, if there are n variables, each having k_i partitions, the number of variables in A-space which will appear as elements in permissible α vectors is

$$T_{\alpha} = \prod_{i=1}^{n} \left((2^{k_i} - 1) \right) . \tag{32}$$

If this is not immediately apparent, it may be seen by the following consideration of the previous example. Obviously the period of p_1 will be 2^{k_1}, with a single 0 in position t_1 and 1's in the balance of the period. By the same token the period of p_2 will be $2^{k_1+k_2}$ with the first 2^{k_1} positions filled with zeros and 1's in the balance of the period. This same construction could be continued over all of the partitions. It is now a simple matter to compute the value assumed by C_{1-N} in the time period t_j:

$$C_{1-N}(t_j) = p_1(t_j)p_2(t_j) \ldots p_n(t_j) \tag{33}$$

[19] Joseph Schillinger, The Mathematical Basis of the Arts, Philosophical Library, New York, 1948.

[20] Joseph Schillinger, The Schillinger System of Musical Composition, edited by Lyle Dowling and Arnold Shaw, Carl Fischer, Inc., New York, 1946.

and by producting this with the (1×2^N) column matrix, all of whose elements are 1's, we get (32) as desired.

Another example which is not so symmetric as the previous one may clarify these results. Let $n = 3$ with $k_1 = 2$, $k_2 = 3$, and $k_3 = 3$. In this case the Schillinger cycle, C_{1-8}, is:

$$37 \quad \underline{3131313131313531313131313} \tag{34}$$

in which the bracketed expression is repeated six times in all. The underlined numerals are positive, i.e., correspond to occurrences of variables in some α_i. Thus for this particular example $T = 18$ whereas the number of variables which appear in the α vectors is 147 or $T_\alpha = 7 \times 7 \times 3$. The reason for choosing an asymmetric example was so that we could exhibit the effect of reordering the partitions. Assume $k_1 = 3$, $k_2 = 2$, and $k_3 = 3$. In this case the Schillinger cycle, C_{1-8}, is

$$41 \quad \underline{71717971717} \tag{35}$$

in which the bracketed expression is repeated six times in all. Again $T_\alpha = 147$. There is only one other possibility in this simple example: $k_1 = 3$, $k_2 = 3$, and $k_3 = 2$. In this case the Schillinger cycle, C_{1-8}, is

$$73 \quad \underline{7171717171717971717171717} \tag{36}$$

in which the bracketed expression is repeated three times in all. As predicted by the general formula (32) the number of variables in all cases was 147, a number which dwarfs the number of equations. However, it is a significant reduction from the 256 variables potentially available in A-space.

Formula (32) guarantees that for the case considered in the example, there will always be many more variables than equations, and hence infinitely many nontrivial solutions. Or what is the same thing, there are infinitely many hyperplanes in A-space which are $f(X_i)$-distant from the point α_i for all of the T points possible in S-space. Now that we are assured of the existence of solutions, we shall be concerned with an algorithm which will produce at least one solution.

An Algorithm for the Solution of the Associated Linear Systems

The system of T equations in T_α unknowns could, if it were available, be solved by several conventional methods. Unfortunately the data points in S-space are presented serially so that only a single α equation is available to the associative memory at any particular time. Thus any solution will have to be generated by a sequential operation on each equation. For solving linear systems there is only one technique in use which has this essential property, i.e., sequential manipulation of one equation at a

time. This is the "relaxation method" devised by Gauss and revived and popularized by R. V. Southwell and his colleagues.[21,22] Basically, this procedure depends on the successive minimization of a set of "residuals" associated with the family of linear equations being solved. Let the linear system be of the form

$$\sum_{i=1}^{n} \alpha_{ij} X_i = f(X_i) \tag{37}$$

then the residuals, R_i, are defined to be

$$f(X_i) - \sum_{i=1}^{n} \alpha_{ij} X_i' = R_i \tag{38}$$

where the primed values of X_i indicate some estimate of the actual vector X. The most commonly employed algorithm used in relaxation of the family given by (37) depends on computing all of the residuals, R_1, $R_2 \ldots R_T$, for some assumed value of X. The largest of these residuals is then chosen and a change of the elements in the assumed X is made so that this residual is caused to become zero (or near zero) and a new family of residuals is computed for this new value of X'. This procedure is repeated until all residuals are within a tolerable bound of zero, at which time the value of X' is considered to be the approximate solution for X. The above description of the relaxation method is based on the most commonly used algorithm, but not the only one used! Other techniques depend on choosing the residual which requires the greatest change in X' for its liquidation; over-relaxation so that the algebraic sign of the residual being liquidated is changed; and under-relaxation, in which only a part of the residual is liquidated. We do not intend to discuss relaxation methods, as such, and the foregoing was intended only to show the relationship of the algorithm used in the present model to an existing procedure for the solution of a system of linear equations.

In each of the algorithms mentioned above for use with relaxation techniques, the entire system of linear equations was assumed to be available at one time so that all of the residuals could be computed and a decision made, based on the values of these residuals, as to the next step in the process. In the model of the associative memory under discussion only one equation is available at a time, so that its residual can be either partially or totally liquidated, without, however, knowing the effect of this operation on the other residuals.

From equations (28) and (38) we may write the whole linear system, in residual form, as

$$f(X_{(j)}) - \alpha_{(j)} \cdot Q' = R_j \tag{39}$$

---

[21] R. V. Southwell, Relaxation Methods in Engineering Science - a treatise on approximate computation, Oxford University Press, Oxford, 1940.

[22] F. S. Shaw, An Introduction to Relaxation Methods, Dover Publications, Inc., New York, 1953.

where $X_{(j)}$ is the j-th point chosen in S-space (not a component of X) and $\alpha_{(j)}$ is the image of $X_{(j)}$ under A in A-space (again not a component of α). If R_j is to be liquidated, and if no information is available concerning the relative value of a variable (i.e., the effect of a particular variable on all other residuals) which is certainly true in the restricted system under consideration here, and if further all of the coefficients are identical (1 in this case), then the readjustment of Q′ to a new value may be best accomplished by dividing the correction equally among all components. What we have said is that we have no information on which to base a selective adjustment of the variables in Q′, and hence we will impartially adjust all of them by the same amount. If R_j is to be totally liquidated, then the computational algorithm takes the form:

$$\left.\begin{aligned} \Delta &= \frac{R_j}{H_\alpha} \\ Q'' &= Q' + \Delta \alpha^T \end{aligned}\right\} \qquad (40)$$

where H_α is the Hamming weight of α, and the primes indicate successive approximations of Q. For a partial liquidation of R_j, the algorithm assumes a similar form:

$$\left.\begin{aligned} \Delta &= k\frac{R_j}{H_\alpha} \\ Q'' &= Q' + \Delta \alpha^T \end{aligned}\right\} \qquad (41)$$

where $0 < k < 1$, and the other variables are the same as before. Obviously (40) and (41) can be combined into a single expression by letting k have the range (0, 1), however the special case of total liquidation of R_j in (40) is of such interest that it seems worthwhile considering it as a separate rule. This is also true since only the value $k = 1/2$ is normally considered for partial liquidation.

With the new estimate for the vector Q obtained from either (40) or (41), a new residual will be computed using the next set of coefficients, α_i, when they are generated. We shall be concerned therefore with the solution of a system of linear equations by relaxation when the sequence in which the equations are relaxed is random. Obviously there will be serious questions of convergence with such a procedure. The following section is devoted to a discussion of these general problem areas.

Convergence Proof for the Proposed Algorithm

The convergence of a computational procedure, such as the one which has been outlined here, is basically a statistical problem dependent on the particular sequence of learning points chosen in S-space. A proof of the convergence of the procedure will consist of a demonstration that the method defines a solution in the limit as the learning sequence is allowed to increase without bound, irrespective of the order in which individual learning points are chosen. We shall show this in the present section with some weakening assumptions which we have been unable to remove.

First we shall require the following lemma concerning the average value, $\overline{\alpha_i \alpha_j^T}$, of the products of the α vectors taken over all possible pairings: $i \neq j$. Actually we shall be concerned with

$$\overline{\frac{\alpha_i \alpha^T j}{\phi_o(n)}} = \Psi \tag{42}$$

which is of course the average of the cosines of the angles between the various vectors since $\phi_o(n) = \alpha_i \alpha_i^T$ for every i so that $|\alpha_i||a_j| = \phi_o(n)$. Let $\gamma = \overline{\alpha_i \alpha_j^T}$, $i \neq j$; then we shall prove

$$\gamma = \frac{\phi_o(n)}{2^n(T-1)} \left\{ \prod_{i=1}^{n} (k_i + 1) - 2^n \right\} \tag{43}$$

where all of the symbols have been introduced and defined in the preceding sections. By formula (16) the number of α elements in common, α_i, between the α vectors which are identical in (n - 1) partitions, and which consequently differ in a single partition which may be assumed to have k_j elements is

$$\phi_1(n) = \prod_{\substack{i=1 \\ i \neq j}}^{n} 2^{k_i - 1} \times 2^{k_j - 2} \tag{16}$$

Similarly formula (17) defines the extended formula for $\phi_p(n)$ when the base vectors differ in p elements (and hence partitions for well behaved vectors). It follows from the way in which $\phi_p(n)$ was defined that

$$\alpha_i \alpha_j^T = \phi_p(n) \tag{44}$$

for some p, $1 \leq p \leq \phi_o(n)$

This equation is the basis of the following computation of γ.

The number of ways a vector X, consisting of n partitions of k_i discrete values each, can differ in exactly one partition from another vector is

$$N_1 = \frac{k_1 k_2 \ldots k_n}{2} \left\{ \sum_{i=1}^{n} (k_i - 1) \right\} \tag{45}$$

or introducing T from equation (29) we get

$$N_1 = \frac{T}{2} \sum_{i=1}^{n} (k_i - 1) \tag{45-a}$$

By the same argument the number of ways this same vector X could differ from another vector in exactly two partitions is found to be:

$$N_2 = \frac{T}{2} \sum_{\substack{i=1 \\ i \neq j}}^{n} \sum_{j=1}^{n} (k_i - 1)(k_j - 1) \tag{46}$$

This technique can be successively reapplied to find N_3, N_4 . . . until finally we obtain for N_n

$$N_n = \frac{T}{2} (k_1 - 1)(k_2 - 1) \ldots (k_n - 1) \tag{47}$$

But now consider what we have derived in the above formulas. Obviously, all possible pairings (in which order within a pairing does not matter) of the X vectors have occurred in the above enumeration. But the total number of ways that such pairs can be formed is simply $\binom{T}{2}$, i.e., the arrangements of T items (well behaved vector points in S-space) taken two at a time. But this must then be equal to the sum of N_i.

$$\binom{T}{2} = \sum_{i=1}^{n} N_i \tag{48}$$

If the N_i are replaced by their equivalent forms, (45), (46), . . . (47), we obtain

$$\frac{T(T-1)}{2} = \frac{T}{2}\left\{\sum_{i=1}^{n}(k_i - 1) + \sum_{\substack{i=1 \\ i\neq j}}^{n}\sum_{j=1}^{n}(k_i - 1)(k_j - 1) + \cdots + (k_1 - 1)(k_2 - 1)\cdots(k_n - 1)\right\} \tag{49}$$

The bracketed expression in (49) reduces trivially from its expansion into symmetric functions to

$$\left\{k_1 k_2 \cdots k_n - 1\right\}$$

which is of course identically (T - 1), and (48) is thereby proven in general.

As was pointed out in the passing comment above, the expansion of (49) into symmetric functions of the k_i was the basis for the proof of (48); it is also the basis for the proof of our lemma (43). Define a family of symmetric functions, S_i, on the k_i as follows:

$$\left.\begin{aligned}
S_0 &= 1 \\
S_1 &= \sum_{i=1}^{n} k_i \\
S_2 &= \sum_{\substack{i=1 \\ i\neq j}}^{n}\sum_{j=1}^{n} k_i k_j \\
S_3 &= \sum_{\substack{i=1 \\ i\neq j\neq \ell}}^{n}\sum_{j=1}^{n}\sum_{\ell=1}^{n} k_i k_j k_\ell \\
&\vdots \\
S_n &= k_1 k_2 \cdots k_n
\end{aligned}\right\} \tag{50}$$

Then the $N_i(n)$ derived above (the parenthetic (n) is used with N_i to identify the particular partitioning) have the following simple form in terms of the S_i, where n is the upper bound to the i indices;

$$N_i(n) = \frac{T}{2}\sum_{j=0}^{i}(-1)^j \binom{n-i+j}{j} S_j \tag{51}$$

The result expressed by formula (51) may be conveniently illustrated by the following example for $n = 4$, where only the factor under the summation in the right hand term of (51) is shown.

i					
4	S_4	$-S_3$	$+S_2$	$-S_1$	$+1$
3		S_3	$-2S_2$	$3S_1$	-4
2			S_2	$-3S_1$	$+6$
1				S_1	-4

Incidentally, the proof of the preceding paragraph depended on forming the sum of this array by columns, i.e., on reversing the order of summation in (51) with S_i replaced by the equivalent multiple sums over the k_i. The primary reason for exhibiting the above array though is to illustrate the following point. The array is obviously missing a $+1$ in the right hand column which would correspond to including the case N_o. We shall add and subtract such an element so the array can be completed. Now referring to the results of formula (17) we can write

$$\gamma = \binom{T}{2}^{-1} \frac{T}{2} \sum_{i=1}^{n} N_i \phi_i(n)$$

But $\phi_i(n) \equiv \frac{1}{2^i} \phi_o(n)$

from (17), so that we may finally note that

$$\gamma = \binom{T}{2}^{-1} \frac{T}{2} \sum_{i=1}^{n} \left\{ \frac{\phi_o(n)}{2^i} \left(1 - \frac{1}{2}\right)^{n-i} S_i - 1 \right\} \tag{52}$$

But this can be simplified to

$$\gamma = \frac{\phi_o(n)}{2^n(T-1)} \sum_{i=1}^{n} (S_i - 2^n) \tag{53}$$

which can be rearranged by the use of (50) into the desired form

$$\gamma = \frac{\phi_o(n)}{2^n(T-1)} \left\{ \prod_{i=1}^{n} (k_i + 1) - 2^n \right\} \tag{43}$$

The function Ψ which is the form which occurs naturally in the following analysis is given by (42) and is

$$\Psi = \frac{1}{2^n(T-1)} \left\{ \prod_{i=1}^{n} (k_i + 1) - 2^n \right\}$$

The example used earlier had $n = 2$; $k_1 = k_2 = 2$ and hence $\phi_o(n) = 4$ and $T = 4$, so that

$$\gamma = \frac{4}{4 \cdot 3} \left\{ 3 \cdot 3 - 4 \right\} = \frac{5}{3}$$

which value may be easily checked by averaging the $\alpha_i \alpha_j^T$ tabulated for the example.

We shall require a notation suitable for indicating an assumed ordering of the sequence of memory entries as reflected by the approximating functions, Q, R, f and Δ, used in the solution of the linear system. Since the ordering appears first in the X, let $X_{(j)}$ represent the j-th X to be chosen (not the j-th component of X, X_j). Then we may define $\alpha(X_{(j)})$ or simply $\alpha_{(j)}$ as the j-th associated α function. If Q were known, then by the definition of Q

$$f(X_{(j)}) = (X_{(j)}) \cdot Q$$

but since we are concerned with an approximation of Q, this expression can be replaced by the following generalization:

$$F(X_{(j)}) = \alpha(X_{(j)}) \cdot Q_{(j-1)}$$

or

$$F_j = \alpha_{(j)} \cdot Q_{j-1} \tag{54}$$

where we have introduced ordinary subscripts wherever there was no possibility of confusion. The residuals as defined by equation (39) may now be represented

$$R_{(j)} = f(X_{(j)}) - F(X_{(j)})$$

or

$$R_{(j)} = f_{(j)} - \alpha_{(j)} Q_{j-1} \tag{55}$$

The liquidation correction, Δ, is then given by

$$\Delta_j = \frac{k R_{(j)}}{H_a} \tag{56}$$

which becomes in our illustrative example with the complete binary matrix A and well behaved vectors X

$$\Delta_j = \beta R_{(j)}; \quad \text{where } \beta = \frac{k}{\phi_o(n)} \tag{57}$$

Finally the Q_j are defined recursively by

$$Q_j = Q_{j-1} + \Delta_j \alpha_{(j)}^T \tag{58}$$

The preceding paragraph was repetitious of material presented in the previous section; however, it introduced a notation suitable for the representation of the sequential behavior of the various functions used in relaxing the linear system. In this section we shall be concerned with the asymptotic behavior of Q_n as n becomes very large (and hence with the asymptotic behavior of the other functions as well). We shall first exhibit the behavior of Q_i for small i and then deduce from the simple form of these functions the general form of Q_i as i becomes large. It was assumed in the preceding section that Q_o is taken to be

the origin of A-space in all instances. The following sequence then shows the evolution of Q_i for Q_o through Q_3

$$Q_o = \{0\}$$

$$F_1 = \{0\}$$

$$R_{(1)} = f_{(1)}$$

$$\Delta_1 = \beta f_{(1)}$$

$$Q_1 = \beta f_{(1)} \alpha_{(1)}^T$$

$$F_2 = \beta f_{(1)} \alpha_{(2)} \alpha_{(1)}^T$$

$$R_{(2)} = f_{(2)} - \beta f_{(1)} \alpha_{(2)} \alpha_{(1)}^T$$

$$\Delta_2 = \beta f_{(2)} - \beta^2 f_{(1)} \alpha_{(2)} \alpha_{(1)}^T$$

$$Q_2 = \beta f_{(1)} \alpha_{(1)}^T + \beta f_{(2)} \alpha_{(2)}^T - \beta^2 f_{(1)} \alpha_{(2)} \alpha_{(1)}^T \alpha_{(2)}^T$$

$$F_3 = \beta f_{(1)} \alpha_{(3)} \alpha_{(1)}^T + \beta f_{(2)} \alpha_{(3)} \alpha_{(2)}^T - \beta^2 f_{(1)} \alpha_{(3)} \alpha_{(2)} \alpha_{(1)}^T \alpha_{(2)}^T$$

$$R_3 = f_{(3)} - \beta f_{(1)} \alpha_{(3)} \alpha_{(1)}^T - \beta f_{(2)} \alpha_{(3)} \alpha_{(2)}^T + \beta^2 f_{(1)} \alpha_{(3)} \alpha_{(2)} \alpha_{(1)}^T \alpha_{(2)}^T$$

$$\Delta_3 = \beta f_{(3)} - \beta^2 f_{(1)} \alpha_{(3)} \alpha_{(1)}^T - \beta^2 f_{(2)} \alpha_{(3)} \alpha_{(2)}^T + \beta^3 f_{(1)} \alpha_{(3)} \alpha_{(2)} \alpha_{(1)}^T \alpha_{(2)}^T$$

$$\begin{aligned} Q_3 = {} & \beta f_{(1)} \alpha_{(1)}^T + \beta f_{(2)} \alpha_{(2)}^T + \beta f_{(3)} \alpha_{(3)}^T - \beta^2 f_{(1)} \alpha_{(2)} \alpha_{(1)}^T \alpha_{(2)}^T \\ & - \beta^2 f_{(1)} \alpha_{(3)} \alpha_{(1)}^T \alpha_{(3)}^T - \beta^2 f_{(2)} \alpha_{(3)} \alpha_{(2)}^T \alpha_{(3)}^T \\ & + \beta^3 f_{(1)} \alpha_{(3)} \alpha_{(2)} \alpha_{(1)}^T \alpha_{(2)}^T \alpha_{(3)}^T \end{aligned}$$

The Q_i functions as developed above may now be summarized in the following form.

$$Q_0 = \{0\}$$

$$Q_1 = \beta f_{(1)} \alpha_{(1)}^T$$

$$Q_2 = \beta \sum_{i=1}^{2} f_{(i)} \alpha_{(i)}^T - \beta^2 f_{(1)} \alpha_{(2)} \alpha_{(1)}^T \alpha_{(2)}^T$$

$$Q_3 = \beta \sum_{i=1}^{3} f_{(i)} \alpha_{(i)}^T - \beta^2 \sum_{j>i}^{3} \sum_{i=1}^{3} f_{(i)} \alpha_{(j)} \alpha_{(i)}^T \alpha_{(j)}^T + \beta^3 f_{(1)} \alpha_{(3)} \alpha_{(2)} \alpha_{(1)}^T \alpha_{(2)}^T \alpha_{(3)}^T$$

It is a simple matter to show that the form by which the Q_i are represented above is preserved for larger values of i. This formalism is independent of the actual functions which appear in the learning sequence. As a matter of fact the simplifications which result from assigning specific values to the functions in the sequence are the basis of our weakened convergence proof. The following expression then is a completely general representation for Q_m

$$Q_m = \beta \sum_{i=1}^{m} f_{(i)}\, \alpha_{(i)}^{T} - \beta^2 \sum_{j>i}^{m} \sum_{i=1}^{m} f_{(i)}\, \alpha_{(j)}\, \alpha_{(i)}^{T}\, \alpha_{(j)}^{T}$$

$$+ \beta^3 \sum_{\ell>j}^{m} \sum_{j>i}^{m} \sum_{i=1}^{m} f_{(i)}\, \alpha_{(\ell)}\, \alpha_{(j)}\, \alpha_{(i)}^{T}\, \alpha_{(j)}^{T}\, \alpha_{(\ell)}^{T} \qquad (59)$$

$$+ \cdots + (-1)^{m+1} f_{(1)}\, \alpha_{(m)}\, \alpha_{(m-1)} \cdots \alpha_{(2)}\, \alpha_{(1)}^{T}\, \alpha_{(2)}^{T} \cdots \alpha_{(m)}^{T}$$

Using this approximate value for Q we can compute the estimated value of $f(X_{(j)})$ from (54) once we know (or assume) the sequence in which the α values occur.

In our test for convergence we shall make use of a pair of weakening assumptions, which, however, do not vitiate the proof. First we shall assume that $f_{(1)} = 1$ and $f_{(i)} = 0$ for all $i \neq 1$. This is equivalent to asking that Q be the hyperplane passing through all of the $\alpha_{(i)}$, $i \neq 1$, and lying at unit distance from $\alpha_{(1)}$. The essential assumption in introducing this argument is, that since the device can solve the homogeneous systems

$$\alpha_{(i)} \cdot Q = 0 \qquad i \neq j$$

$$\alpha_{(j)} \cdot Q = 1 \qquad 1 \leq j \leq m$$

where each choice of a j determines a new linear system, it can also solve the superposition of these when the assumptions are relaxed. This is unproven at the present and is one of the main problems to be studied further. The second assumption, which is actually only a specification of a value for a variable, is the decision to completely liquidate the residuals at each stage of the relaxation procedure. This is equivalent to assuming k to be 1. Under these assumptions it is an easy task to compute the approximating functions F_j, at any point in the learning (or storage) cycle.

It is important to note that in formula (59) the parenthetic subscripts indicate relative order of occurrence in the learning sequence, and not necessarily disinct points. For an example, every point could be the same without in any way violating the assumptions. There is no loss of generality involved in assuming $\alpha_{(1)}$ to be the α corresponding to the single X with a nonzero function f(X), for no change in Q_i will result from the initial exposure of the device to any number of points for which $f(X_i) = 0$, since Q_0 is assumed to be $\{0\}$. Now assume that $\alpha_{(i)}$ occurs η times in the learning sequence. If the sequence is sufficiently large that all of the α can be assumed to have occurred with the same frequency, then it is possible to rewrite (59) in a greatly simplified form. To do this we shall drop the sequential notation and use simple subscripts to indicate distinct entities (not elements of the various functions). In other words

we shall assume a learning sequence in which m is sufficiently large so that each X_i can be assumed to have occurred η times. This assumption makes it possible to analyze Q_i without reference to the probability description of the learning sequence itself.

First we shall expand the Q_i function about the $\eta\ \alpha_{(1)}$. To do this requires two steps: in the first we expand (59), which is expressed only in terms of the sequence of learning points, in terms of the n distinct functions and secondly we shall express these functions in terms of their functional dependence on $\alpha_{(1)}$. Obviously there is a contribution of $\frac{\eta}{\phi_o(n)}$ from the first term of (59) under our assumption of $m = \eta n$. Similar simple algebraic manipulations define the results of transforming the other terms of (59); therefore we shall only record the results.

$$\beta \sum_{i=1}^{m} f_{(i)}\, \alpha_{(i)}^T = \frac{\eta}{\phi_o(n)}\, \alpha_1^T \tag{60-a}$$

$$-\beta^2 \sum_{j>i}^{m} \sum_{i=1}^{m} f_{(i)}\, \alpha_{(j)}\, \alpha_{(i)}^T\, \alpha_{(j)}^T = \frac{\eta^2}{\phi_o^2(n)} \sum_{i=2}^{n} \alpha_i\, \alpha_1^T\, \alpha_i^T - \frac{\binom{\eta}{2}}{\phi_o(n)}\, \alpha_1^T \tag{60-b}$$

$$\beta^3 \sum_{\ell>j}^{m} \sum_{j>i}^{m} \sum_{i=1}^{m} f_{(i)}\, \alpha_{(\ell)}\, \alpha_{(j)}\, \alpha_{(i)}^T\, \alpha_{(j)}^T\, \alpha_{(\ell)}^T = \frac{\eta^3}{\phi_o^3(n)} \sum_{j=2}^{n} \sum_{i=2}^{n} \alpha_j\, \alpha_i\, \alpha_1^T\, \alpha_i^T\, \alpha_j^T$$

$$+ \frac{\eta\binom{\eta}{2}}{\phi_o^2(n)} \sum_{j=2}^{n} \alpha_j\, \alpha_1^T\, \alpha_j^T + \frac{\eta\binom{\eta}{2}}{\phi_o^3(n)} \sum_{j=2}^{n} \left(\alpha_i \alpha_1^T\right)^2 \alpha_1^T + \frac{\binom{\eta}{3}}{\phi_o(n)}\, \alpha_1^T \tag{60-c}$$

etc. for other higher order terms in (59). If now we gather terms with similar sums as coefficients, we can collapse this representation into the form referred to above as an expansion about $\alpha_{(1)}$. Obviously we get one term involving only α_1^T:

$$C_1 \alpha_1^T = \left\{ \frac{1}{\phi_o(n)} \sum_{i=1}^{\eta} (-1)^{i-1} \binom{\eta}{i}\, \alpha_1^T \right\} \tag{61}$$

or

$$C_1 = \frac{1}{\phi_o(n)}$$

and similarly a term involving only $\sum_{i=2}^{n} \alpha_i\, \alpha_1^T\, \alpha_i^T$ etc. for the other sums.

Finally we may write Q′ in the form:

$$
\begin{aligned}
Q' = {} & \frac{\alpha_1^T}{\phi_o(n)} - \frac{1}{\phi_o^2(n)} \sum_{i=2}^{n} \alpha_i \, \alpha_1^T \, \alpha_i^T \\
& + \frac{1}{\phi_o^3(n)} \sum_{\substack{j=2 \\ j \neq i}}^{n} \sum_{i=2}^{n} \alpha_j \, \alpha_i \, \alpha_1^T \, \alpha_i^T \, \alpha_j^T \\
& + \frac{1}{\phi_o^3(n)} \sum_{i=2}^{n} \alpha_1 \, \alpha_i \, \alpha_1^T \, \alpha_i^T \, \alpha_1^T \\
& - \frac{1}{\phi_o^4(n)} \sum_{\substack{\ell=2 \\ \ell \neq j \neq i}}^{n} \sum_{j=1}^{n} \sum_{i=2}^{n} \alpha_\ell \, \alpha_j \, \alpha_i \, \alpha_1^T \, \alpha_i^T \, \alpha_j^T \, \alpha_\ell^T \\
& - \frac{1}{\phi_o^4(n)} \sum_{\substack{j=2 \\ i \neq j}}^{n} \sum_{i=2}^{n} \alpha_1 \, \alpha_j \, \alpha_i \, \alpha_1^T \, \alpha_i^T \, \alpha_j^T \, \alpha_1^T \\
& + \frac{1}{\phi_o^5(n)} \sum_{p=2}^{n} \sum_{\substack{\ell=1 \\ p \neq \ell \neq j \neq i}}^{n} \sum_{j=1}^{n} \sum_{i=2}^{n} \alpha_p \, \alpha_\ell \, \alpha_j \, \alpha_i \, \alpha_1^T \, \alpha_i^T \, \alpha_j^T \, \alpha_\ell^T \, \alpha_p^T \\
& + \frac{1}{\phi_o^5(n)} \sum_{\ell=2}^{n} \sum_{\substack{j=1 \\ \ell \neq j \neq i}}^{n} \sum_{i=2}^{n} \alpha_1 \, \alpha_\ell \, \alpha_j \, \alpha_i \, \alpha_1^T \, \alpha_i^T \, \alpha_j^T \, \alpha_\ell^T \, \alpha_1^T \\
& + \cdots
\end{aligned}
\tag{62}
$$

Although it is true that Q′ approaches the form of (62) asymptotically as η becomes arbitrarily large, it is necessary to specify just how many of the sums in (62) are completed for a given, finite, η. It is a violation of our earlier assumption that η is a large number; however, assume for the moment that η is 1 and that each variable has occurred exactly once. In this case obviously only the first two terms are filled completely, and with each successive increase of (n - 1) entries an additional term is completed. It is not essential to this argument that the terms be ordered within the cycles, but only that they occur once in each cycle. We therefore let η become large, in which case approximately $\eta + \left[\frac{\eta}{n-1}\right]$ sums are completely filled out in (62) since each term will have occurred approximately η times.

The first question concerns the behavior of $\alpha_1 \cdot Q'$ as η becomes large. The result should be asymptotic to 1. To see that this is indeed so, consider the terms obtained by forming the product of α_1 and Q′ as given by (62). The first term will be identically 1 by the definition of $\phi_o(n)$. The second term

will be identical to the fourth term with the sign reversed, the third term will likewise be identical to the fifth term with a sign difference. In a similar manner the r-th term will be identical to the (r + 3)-rd term with a difference of sign so that the product terms disappear by pairs. The discrepancy above between the second and fourth terms is due to the fact that the term which would have been the third term was absorbed into the first term coefficient as was shown in (61). Thus, if we can show that the individual terms go to zero in absolute value we will have proven

$$\lim_{\eta \to \infty} \alpha_1 \cdot Q' = 1$$

as was desired.

As was noted above, if the memory storage sequence is terminated after η cycles, then in general the first $\eta + \left[\frac{\eta}{n-1}\right]$ terms of (62) are completed; i.e., all of the indicated terms under the summation have been generated. The next term then dominates the error (which fact must be shown) and has a simple upper bound in terms of the Ψ which we have defined earlier. First consider the manner by which the terms of (62) were formed from (60). The terms of the $\left(\eta + 1 + \left[\frac{\eta}{n-1}\right]\right)$-st term will involve $2\left\{\eta + \left[\frac{\eta}{n-1}\right]\right\}$ α_i, no adjacent pair of which have the same index i. Since each α_i occurs twice in the sum, once on the left as α_i and once on the right as α_i^T, this implies a freedom of choice among (n - 1) items at each step.

The elements of the $\left(\eta + 1 + \left[\frac{\eta}{n-1}\right]\right)$-st sum are characterized by having all possible forward transpositions of the α indices. Thus, for an example, if we assume α_i to consist of the set $\{\alpha_1, \alpha_2, \alpha_3\}$, at the end of the second memory storage cycle the fourth term, i.e., $\left(2 + 1 + \frac{2}{3-1}\right)$, will be

$$-\frac{1}{\phi_o^4(n)}\left\{\alpha_1 \alpha_3 \alpha_2 \alpha_1^T \alpha_2^T \alpha_3^T \alpha_1^T + \alpha_2 \alpha_1 \alpha_2 \alpha_1^T \alpha_2^T \alpha_1^T \alpha_2^T + \alpha_2 \alpha_1 \alpha_3 \alpha_1^T \alpha_3^T \alpha_1^T \alpha_2^T + \alpha_2 \alpha_3 \alpha_2 \alpha_1^T \alpha_2^T \alpha_3^T \alpha_2^T\right\} \quad (63)$$

or (from the assumption of simple cyclical ordering for small values of η) all of the forward transpositions of the index sequence 1 (23) (12) subject to the restraints that no adjacent pair of indices may be the same, and that the first element must be a 1. Obviously this is only a part of the entire fourth term, since four more distinct α_i product terms can be formed subject to the above restrictions. The essential observation though, is that the terms of (63), when producted with an α_i, have an average value less than or equal to γ^4, i.e., Ψ^4 upon dividing through by $\phi_o^4(n)$.

The next thing which must be determined is just how many elements are under the $\left(\eta + 1 + \left[\frac{\eta}{n-1}\right]\right)$-st summation in general.

This is a combinatorial problem, for which no simple solution is available. We shall first solve the general problem, and then use these solutions in the present investigation of convergence. The most general statement of the combinatorial problem is the following: Given n distinct items in some order, let the symbol $(12 \cdots n)$ represent the assumed sequence, and also let the sequence be repeated η times followed by any fraction of a sequence in the same order; i.e., symbolically

$$(12 \cdots n)_1 (12 \cdots n)_2 \cdots n)_\eta (12 \cdots h) \quad (64)$$

How many distinct u-tuples can be formed by forward transpositions among the elements of this sequence subject to the restrictions that no adjacent pair of elements may be the same, and that the first element must be a 1? Let H be the number of elements in (64); i.e.,

$$H = n\eta + h$$

and let ${}_{H}C_{u}$ be the symbol representing the number of u-tuples which can be formed from this sequence of H symbols subject to the above restrictions; then ${}_{H}C_{u}$ satisfies the following partial difference equation.

$${}_{H}C_{u} = {}_{H-1}C_{u} + {}_{H-1}C_{u-1} - {}_{H-n}C_{u-1} \tag{65}$$

That this is so may be seen from the way in which new terms are formed (under the summation). All terms with $(u - 1)\alpha_i$ in the preceding stage (memory entry) will gain an additional element, the new α, except for those which end in the same α which will be the number of terms with $(u - 1)\alpha_i$ occurring n stages (entries) earlier (i.e., exactly one cycle earlier). Also the preceding expression for Q is added to this new liquidation factor so that the number of terms with $u\ \alpha_i$ in the preceding stage must be added. But this set of generating rules is precisely what is expressed in (65). (65) is a very special form of an n-th order partial difference equation which can be solved by Boole's technique. As a final expression for ${}_{H}C_{u}$, we find:

$${}_{H}C_{u} = \sum_{i=0}^{I} (-1)^{i} \binom{H - in}{u - i} \binom{H - i(n-1)}{i} \tag{66}$$

where

$$I = \text{Min.} \left\{ \left[\frac{H}{n} \right], \left[\frac{H - u}{n - 1} \right] \right\}$$

The results of formula (66) are well illustrated by an example, both for numerical values and for the generation scheme itself. Using the same example used earlier, n = 4 -- so that a full cycle length is 4 and the completion cycle with which sums are filled out in (62) is 3. The following array give the ${}_{H}C_{u}$ for $H \leq 9$, and correspondingly for $u \leq H$ in each case.

H	0	1	2	3	4	5	6	7	8	9
0	(1)									
1	1	1								
2	1	2	1							
3	1	(3)	3	1						
4	1	3	6	4	1					
5	1	3	8	10	5	1				
6	1	3	(9)	17	15	6	1			
7	1	3	9	23	31	21	7	1		
8	1	3	9	26	50	51	28	8	1	
9	1	3	9	(27)	66	96	78	36	9	1

The circled figures represent the first point in the sequence at which a particular term is completed and illustrates a result which was deduced earlier from the generating scheme; namely, that the u-th sum of (62) when completed has $(n - 1)^u$ terms under the summation sign. To illustrate the generation of row H = 9 in the above array by (66) construct the following sets of numbers:

$$\begin{array}{cccccccccc} 1 & 9 & 36 & 84 & 126 & 126 & 84 & 36 & 9 & 1 \\ & -6 & -30 & -60 & -60 & -30 & -6 & & & \\ & & 3 & 3 & & & & & & \end{array}$$

and sum the columns (summation indicated in ((66))) to get:

$$\begin{array}{cccccccccc} 1 & 3 & 9 & 27 & 66 & 96 & 78 & 36 & 9 & 1 \end{array}$$

In the preceding section we developed a formula for ${}_H C_u$, the number of terms under an arbitrary sum of (62), at any point in the learning sequence. We shall now use this result to complete the convergence proof. The sequence of values:

$$\left\{ {}_u C_u,\ {}_{u+1} C_u \cdots {}_{(n-1)u-1} C_u \right\}$$

is monotonically increasing by (66), and is bounded by

$${}_{(n-1)u} C_u = (n - 1)^u \tag{67}$$

as was noted above. For our purposes this bound is adequate. We set out to determine how many elements appeared under the $\left(\eta + 1 + \left[\frac{\eta}{n - 1}\right]\right)$-st summation in (62) at the completion of η memory storage cycles. We have found this number to be:

$$\eta\, n^{C\left(\eta + 1 + \left[\frac{\eta}{n - 1}\right]\right)}$$

Thus the $\left(\eta + 1 + \left[\frac{\eta}{n - 1}\right]\right)$-st term, say E_η, is bounded by:

$$E_\eta \leq \left\{ (n - 1)\Psi \right\}^{\left(\eta + 1 + \left[\frac{\eta}{n-1}\right]\right)} \tag{68}$$

The bracketed expression must be less than one if $\alpha_1 \cdot Q'$ is to converge to 1. From (43) we may write:

$$(n - 1)\Psi = (n - 1)\left\{ \frac{\prod_{i=1}^{n} (k_i + 1) - 2^n}{\prod_{i=1}^{n} 2k_i - 2^n} \right\} \tag{69}$$

The bracketed term on the right hand side of (69) is a maximum obviously when the values of k_i are a minimum so that the effect of the additive constant in the multipliers in the numerator is emphasized. By the manner in which the variables in S-space were defined, i.e., as a linear array of discrete states, k_i is at least 2. Therefore consider all of the k_i to be exactly 2 for the moment. In this case:

$$(n - 1)\Psi = (n - 1)\left(\frac{3^n - 2^n}{4^n - 2^n}\right)$$

which gives the following values for small n:

n	$(n-1)\Psi$
1	0
2	5/12
3	19/28
4	13/16
5	211/248
6	3325/4032
7	6177/8192

which shows the maximum of $(n-1)\Psi$ at $n = 5$. Thereafter the function behaves asymptotically like

$$(n-1)\left(\frac{3}{4}\right)^n$$

which goes to zero as n goes to infinity. We have therefore proven that

$$\lim_{\eta\to\infty} E_\eta = 0,$$

using the bound on E_η given in (68). Before we can conclude that

$$\lim_{\eta\to\infty} \alpha_1 \cdot Q' = 1$$

we must still show that the $\left(\eta + 1 + \left[\frac{\eta}{n+1}\right]\right)$-st sum of (62) dominates the other incomplete sums. To do this, observe that the bound on the next term will have the same argument as the bracketed expression for E_η, which we have shown to be less than one, raised to a higher power, etc. Thus, the error in the estimation of f_1 incurred by terminating the memory storage sequence after η cycles is less than E_η as defined by (68). Therefore, we may finally conclude that $\alpha_1 \cdot Q'$ does indeed converge to 1 with η as was to be shown.

The foregoing section completes half of the convergence proof; in this section we must prove:

$$\lim_{\eta\to\infty} \alpha_z \cdot Q' = 0 \qquad z \neq 1$$

Fortunately, only a few revisions are required in the argument of the preceding proof to adapt it to this case also. In forming (62) we expanded (59) about the term $\alpha_{(1)}$; a similar expansion can be made about $\alpha_{(z)}$, or simply α_z in our assumed ordered sequence. We shall require an alternate expression of (62) in which the terms are unexpanded about α_1.

$$Q' = \frac{\alpha_1^T}{\phi_o(n)} - \frac{1}{\phi_o^2(n)} \sum_{i=2}^{n} \alpha_i \alpha_1^T \alpha_i^T + \frac{1}{\phi_o^3(n)} \sum_{j=1}^{n} \sum_{\substack{i=2 \\ j\neq i}}^{n} \alpha_j \alpha_i \alpha_1^T \alpha_i^T \alpha_j^T$$

$$- \frac{1}{\phi_o^4(n)} \sum_{\ell=1}^{n} \sum_{\substack{j=1 \\ \ell\neq j\neq i}}^{n} \sum_{i=2}^{n} \alpha_\ell \alpha_j \alpha_i \alpha_1^T \alpha_i^T \alpha_j^T \alpha_\ell^T + \cdots \tag{70}$$

The above form was implicit in the derivation of (62). Now expand each of the above terms about α_z, just as we expanded them about α_1 earlier, to get:

$$Q' = \frac{\alpha_1^T}{\phi_o(n)} - \frac{1}{\phi_o^2(n)} \sum_{\substack{i=2 \\ i\neq z}}^{n} \alpha_i \alpha_1^T \alpha_i^T - \frac{1}{\phi^2(n)} \alpha_z \alpha_1^T \alpha_z^T$$

$$+ \frac{1}{\phi_o^3(n)} \sum_{j=1}^{n} \sum_{\substack{i=2 \\ j\neq i \\ j\neq z}}^{n} \alpha_j \alpha_i \alpha_1^T \alpha_i^T \alpha_j^T + \frac{1}{\phi_o^3(n)} \sum_{\substack{i=2 \\ i\neq z}}^{n} \alpha_z \alpha_i \alpha_1^T \alpha_i^T \alpha_z^T \tag{71}$$

$$- \frac{1}{\phi_o^4(n)} \sum_{\ell=1}^{n} \sum_{\substack{j=1 \\ \ell\neq j\neq i \\ \ell\neq z}}^{n} \sum_{i=2}^{n} \alpha_\ell \alpha_j \alpha_i \alpha_1^T \alpha_i^T \alpha_j^T \alpha_\ell^T - \frac{1}{\phi_o^4(n)} \sum_{\substack{j=1 \\ j\neq i \\ j\neq z}}^{n} \sum_{i=2}^{n} \alpha_z \alpha_j \alpha_i \alpha_1^T \alpha_i^T \alpha_j^T \alpha_z^T + \cdots$$

When $\alpha_z \cdot Q'$ is formed, the first term is cancelled by the third term. It is at this point that the expansion about α_z differs from the expansion about α_1. For as we pointed out in the earlier discussion of the expansion about α_1 this third term was absorbed into the first term in that one case. In exactly the same manner as before, the second term will be cancelled by the fifth; the fourth by the seventh; and, in general, r-th term will be identical to the (r + 3)-rd term with a difference of sign so that the product terms disappear by pairs. The error term will be exactly the same as the one developed in the preceding analysis of the convergence of $\alpha_1 \cdot Q'$ to 1. Therefore, we may conclude that

$$\lim_{\eta \to \infty} \alpha_z \cdot Q' = 0 \qquad z \neq 1$$

as was desired.

The arguments of the two sections just completed prove the weakened convergence of the sequential relaxation technique used by the associative memory. It is not a very useful proof to show that the technique converges in the limit with η since n itself may already be a large number; however, we have proven that the device can behave analytically as expected. The next problem (which is much more difficult and important) is to investigate the behavior of the device under weaker assumptions.

Summary of Mathematical Model

With the completion of the convergence proof, it is now possible to give a very succinct description of the mathematical model for an associative memory developed in this paper. In view of the length of some of the arguments which have been used, it seems especially desirable to summarize the essentials of this model before going on to some intuitive statements concerning its functioning and some empirical results demonstrating its behavior.

A function (commonly a decision function) f(X) is defined over an n-dimensional S-space, in which each of the variables is restricted to assume one of a discrete set, k_i in number, of values. This

restriction to "well-behaved" vectors is equivalent to imposing a constant Hamming weight of n on the base vectors X_i. Each of these vectors is considered to be a linear array in which all possible values of each variable appear, so that a particular vector X_i will have a single entry in each variable range corresponding to the value that particular variable assumes in X_i. Obviously, the number of elements in the X vector is N.

$$N = \sum_{i=1}^{n} k_i \tag{4}$$

A metric S is defined by

$$S = X \cdot A \tag{5}$$

where A is an (N x R) matrix whose columns are vertices of the binary-valued hypercube in S-space, which in its entirety is an N-dimensional space. These points, A_i, constitute a set of R fixed reference points in S-space. The metric S defines the distance of X from each of the A_i, and is a complete metric. A much weaker metric, α, is defined on S by the logical operation

$$\alpha_i = \begin{cases} 1 & \text{if } s_i \geq \delta \\ 0 & \text{if } s_i < \delta \end{cases} \tag{6}$$

where δ is a threshold value; $0 \leq \delta \leq n$.

It was shown that for some A (i.e., for a sufficiently large number of fixed reference points) there exists at least one (and in general infinitely many) hyperplane, Q, in the A-space in which α is defined, such that the distance at which the point α, whose pre-image as defined above is X, lies from Q is the function f(X) multiplied by a suitable normalizing constant. Such a hyperplane is generated by the iterative procedure:

$$\Delta_j = k \frac{R_{(j)}}{H_\alpha} \qquad 0 < k \leq 1 \tag{56}$$

where

$$R_{(j)} = f_{(j)} - \alpha_{(j)} \cdot Q_{j-1} \tag{55}$$

and H_α is the Hamming weight of the vector $\alpha_{(j)}$ whose pre-image is S-space is $X_{(j)}$. And Q_j is defined by the recursive relation

$$Q_j = Q_{j-1} + \Delta_j \alpha_{(j)}^T \tag{58}$$

A special note was made of the case k = 1 corresponding to the total liquidation of the residuals $R_{(j)}$. In all cases it is assumed that the first estimate for Q is the origin of A space.

Thus we may finally conclude that the output of this particular model of an associative memory, (at a time when Q′ is the estimate for Q) upon being interrogated with the address (associative criterion) X is

$$F(X) = \alpha(X) \cdot Q' \tag{54-a}$$

Experimental Investigations

The model for an associative memory developed in the foregoing analysis is impractical to realize since the complete binary matrix ($R = 2^T$) quickly assumes astronomical proportions for what are still small problems. This difficulty is overcome by making use of a remark made in the course of proving the completeness of the α metrics, namely that it suffices to have a sufficiently dense uniform coverage of the hypersphere with reference points, A_i. In the problems to be discussed in this section this is accomplished by selecting the vertices of the inscribed hypercube with a uniform probability distribution. A later paper is to be devoted to an analysis of this probabilistic model; however, for the purposes of the present discussion we will merely state the defining parameters and analyze the resulting memory performance.

The first question to be answered, and the one which will occupy the greater portion of this section, concerns the behavior of the "solutions" produced by the algorithm proposed for solving the large linear systems generated by an associative memory. We are especially interested in the behavior of the Q vector in cases where free parameters are involved so that infinitely many solutions are possible. Which solutions are generated? How rapidly does the system converge? Does it converge monotonically? These and other related questions we shall investigate by studying a sequence of contrived linear systems devised to illustrate various features of the algorithm. In these first experiments no associative memory is used; rather a linear system is constructed which corresponds to the α vectors which could have been produced by an associative memory.

Experiment No. 1

The following simple linear system of three equations in four variables was programmed for the IBM 1620, to be solved by the relaxation of single elements of the system without regard for the relative magnitudes of the residuals involved.

$$\left.\begin{array}{llllll} \alpha_{(1)} = & 1 & 1 & 0 & 1 \\ \alpha_{(2)} = & 0 & 1 & 1 & 0 \\ \alpha_{(3)} = & 0 & 1 & 1 & 1 \end{array}\right\} \mathscr{L}(\alpha)$$

$$\left.\begin{array}{ll} f_{(1)} = & 12 \\ f_{(2)} = & -8 \\ f_{(3)} = & 0 \end{array}\right\} C$$

In a case such as this, where so few elements are involved, there is no meaning to questions concerning the continuity of f. Thus there can be no meaningful estimation of solutions for points not yet introduced, i.e., no extrapolation of solutions. The general solution of

$$\mathscr{L} \cdot Q = C$$

is given in reduced cononical form by:

$$\begin{array}{cccc} q_1 & q_2 & q_4 & a \\ \left(\begin{array}{cccc} 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{array}\right. & & & \left.\begin{array}{c} 12 \\ -8 \\ 8 \end{array}\right), \end{array}$$

or

$$\begin{aligned} q_1 &= 12+a \\ q_2 &= -8-a \\ q_3 &= a \\ q_4 &= 8 \end{aligned}$$

A cyclical iterative sequence was chosen for this test; i.e., $\alpha_{(1)}\ \alpha_{(2)}\ \alpha_{(3)}\ \alpha_{(4)}\ \alpha_{(1)} \cdots$. The Q vectors which were generated are given below. The cycle number is given in parenthesis.

	4.0000	4.0000	0.0000	4.0000
	4.0000	-2.0000	-6.0000	4.0000
(1)	4.0000	-.6666	-4.6666	5.3333
(2)	5.1111	-.9259	-6.0370	6.9629
(3)	5.3950	-1.0514	-6.4465	7.4979
(4)	5.4478	-1.1263	-6.5741	7.7005
(5)	5.4405	-1.1774	-6.6179	7.7954
	.			
	.			
	.			
(10)	5.3645	-1.2928	-6.6574	7.9502
	.			
	.			
	.			
(34)	5.3333	-1.3333	-6.6666	7.9999

The last result, $Q_{(34)}$, is a perfect solution to the linear system within the round off which was established in the computer program. The convergence of the solution appears to be very slow. However, there are three factors which account for this which we can control in the linear systems generated by associative memories. These are:

1. A very large "overlap" of the α elements; i.e., 60% of the elements are shared between any pair of α vectors. This introduces an excessive amount of "destructive interference" in the augmentation. In linear systems generated by actual associative memories this can be controlled by varying δ.

2. As we noted earlier, the number of elements in the linear system or equivalently the number of points in the input space is so low that continuity of the function isn't meaningful, and cannot aid in convergence therefore. This will not be true of the problems to which associative memories will be applied in general.

3. A very dense α vector, i.e., 2/3 of all positions are filled with a 1. This compares very poorly with the theoretical optimum of 0.02 to 0.05. In the case of linear systems generated by associative memories this can be controlled by varying δ and p, where p is the probability used in choosing vertices of the hypercube to be reference points.

Experiment No. 1 then has demonstrated the character of the convergence of the algorithm for a very small problem in which the solution had a single free parameter. The convergence was monotonic, and the resulting solution could be made arbitrarily precise.

Experiment No. 2

This experiment was designed to demonstrate several properties of the iterative procedure which could not be shown with the extremely simple linear system used in experiment No. 1, and at the same time to use a system simple enough to permit easy visualization of the results. The objectives of this experiment are:

1. The investigation of a linear system which has several free parameters, and of the effect on the convergence of the iterative procedure as this system is truncated until it finally becomes determinate.
2. The influence of the particular sequence in which data points are chosen for storage, i.e., the sequence in which elements of the linear system are relaxed. The effects of selecting the elements randomly is especially important since this corresponds most closely to the procedure imposed on an associative memory by the information environment in which it operates.
3. The solution to a system with free parameters is determined by the point in hyperspace from which the iterative procedure starts, Q_o. The proof of this is left to the paper on the probabilistic associative memory; however, the results of the second part of this experiment at least indicate this type of behavior. The third question to be investigated then concerns the influence of selecting an initial point, Q_o, other than $Q_o = (0, 0, \cdots, 0)$.

The fundamental linear system, $\mathscr{L}_1(\alpha)$, investigated in this experiment consists of four equations in eight unknowns. The α vectors (coefficient arrays) are as follows:

$$\left.\begin{array}{llllllll} \alpha_{(1)} = 1 & 1 & 0 & 1 & 0 & 1 & 1 & 0 \\ \alpha_{(2)} = 0 & 1 & 1 & 0 & 1 & 1 & 0 & 1 \\ \alpha_{(3)} = 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 \\ \alpha_{(4)} = 0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 \end{array}\right\} \mathscr{L}_1(\alpha)$$

The functional values will of course be the same for $\mathscr{L}_1(\alpha)$ and for the truncated systems.

$$\left.\begin{array}{l} f_{(1)} = 1 \\ f_{(2)} = 3 \\ f_{(3)} = 2 \\ f_{(4)} = 0 \end{array}\right\} C$$

The general solution of

$$\mathscr{L}_1(\alpha) \cdot Q = C$$

is given in reduced cononical form by:

$$\begin{array}{cccccccc} q_1 & q_2 & q_3 & q_4 & a & b & c & d \\ \left(\begin{array}{cccccccc} 1 & 0 & 0 & 0 & 1/3 & -1/3 & 0 & 1/3 \\ 0 & 1 & 0 & 0 & 1/3 & 2/3 & 0 & 1/3 \\ 0 & 0 & 1 & 0 & 2/3 & 1/3 & 0 & 2/3 \\ 0 & 0 & 0 & 1 & -2/3 & 2/3 & 1 & -2/3 \end{array}\right. & & & & & & & \left.\begin{array}{c} 2/3 \\ 5/3 \\ 4/3 \\ -4/3 \end{array}\right) \end{array}$$

or

$$\left.\begin{array}{l} q_1 = \dfrac{4 - 2a + 2b - 2d}{6} \\ q_2 = \dfrac{10 - 2a - 4b - 2d}{6} \\ q_3 = \dfrac{8 - 4a - 2b - 4d}{6} \\ q_4 = \dfrac{-8 + 4a - 4b - 6c + 4d}{6} \\ q_5 = a \\ q_6 = b \\ q_7 = c \\ q_8 = d \end{array}\right\} Q$$

The first of the truncated systems, $\mathscr{L}_2(\alpha)$, has four equations in six unknowns and is obtained from $\mathscr{L}_1(\alpha)$ by deleting the two rightmost columns of the α array.

$$\left.\begin{array}{cccccc} 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{array}\right\} \mathscr{L}_2(\alpha)$$

The general solution to

$$\mathscr{L}_2(\alpha) \cdot Q = C$$

may be obtained from the general solution to $\mathscr{L}_1(\alpha) \cdot Q = C$ by setting c and d equal to zero and deleting q_7 and q_8 in Q.

The second of the truncated systems, $\mathscr{L}_3(\alpha)$, has four equations in four unknowns and is obtained from $\mathscr{L}_1(\alpha)$ by deleting the four rightmost columns of the α array.

$$\left.\begin{array}{cccc} 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{array}\right\} \mathscr{L}_3(\alpha)$$

This system is determinate and has as its solution the values assumed by q_1, q_2, q_3, and q_4 when a, b, c, and d are set equal to zero and by deleting q_5, q_6, q_7, and q_8 in Q.

The general solutions derived in the preceding paragraph make it possible to interpret the results obtained in solving the various linear systems, $\mathscr{L}_1(\alpha)$, $\mathscr{L}_2(\alpha)$ and $\mathscr{L}_3(\alpha)$ using the iterative procedure. First we shall consider a solution for $\mathscr{L}_1(\alpha)$ obtained by a cyclical (1 2 3 4) choice of the α vectors. Q_o = (0, 0, · · · , 0) in all examples unless it is specifically given as some other point. $Q_{(10)}$ is:

(10) 0.2979 0.9149 0.2128 -0.2979 0.7447 0.3830 -0.2979 0.7447

which is the correct solution to four significant figures as may be checked by substituting q_5 = a, q_6 = b, etc. into the general solution for $\mathscr{L}_1(\alpha)$. Thus the iterative procedure is seen to converge more rapidly than it did for the system of experiment No. 1. This general improvement could have been predicted since the overlap between adjacent elements was decreased from 60% to 25% and the density of the α vectors was dropped from 66-2/3% to 56-1/4%. The system is still too small to allow anything to be said about the continuity of the function.

It would be desirable to exhibit the behavior of the solutions as a function of the number of relaxations; however, the magnitude of the Q vector makes it impractical to simply tabulate the Q vectors as was done for experiment No. 1. The residuals provide some measure of convergence, since the absolute values of the residuals must go to zero if the Q is to be asymptotic to a true solution. The following list exhibits the residuals obtained for each cycle for $f_{(2)}$. The other residuals show precisely the same behavior as the iteration develops, so this single tabulation is loosely descriptive of the system convergence.

η	$R_{(2)}$
1	0.600000
2	0.147000
3	0.039375
4	0.011761
5	0.003900
6	0.001396
7	0.000520
8	0.000195
9	0.000072
10	0.000025
11	0.000008
12	0.000002
13	0.000000

We shall develop later a measure of convergence for use with larger systems where even the residuals provide only a very complex indication of the overall convergence; however, for the moment this listing of R is at least indicative of the general convergence of the procedure.

$\mathscr{L}_1(\alpha)$ was also solved using a random selection (equal probabilities) among the α vectors. In this case it required fifty cycles, as compared to thirteen in the cyclical case, to cause all residuals to become zero to the sixth decimal place. The effects of various cyclical sequences and random sequences will be explored in more detail using $\mathscr{L}_2(\alpha)$.

$\mathscr{L}_2(\alpha)$ was solved using the same (1 2 3 4) sequence used in solving $\mathscr{L}_1(\alpha)$. The solution Q is given by

$$(11) \quad 0.3499 \qquad 1.0750 \qquad 0.4250 \qquad -0.7000 \qquad 1.2250 \qquad 0.2750$$

This is a solution correct to four significant figures. The residuals for $f_{(2)}$ will again be used as indicators of convergence, although they are not as well behaved for $\mathscr{L}_2(\alpha)$ as they were for $\mathscr{L}_1(\alpha)$.

η	$R_{(2)}$
1	1.013888
2	0.334490
3	0.078800
4	0.002322
5	-0.011734
6	-0.009301
7	-0.005059
8	-0.002256
9	-0.000853
10	-0.000265
11	-0.000057
12	-0.000001
13	0.000000

The change in sign, and the apparent oscillation in the raw residual values is in general characteristic of these systems and explains why a raw residual provides a poor indicator of convergence for larger linear systems where several competing residuals may generate several changes of sign and possible oscillations in the course of converging to a solution.

$\mathscr{L}_3(\alpha)$, which is determinate, was also solved using the cyclical sequence. Since there is a unique solution, the question of interest here is how quickly did the procedure converge to this solution. The answer is that a solution, accurate to the fifth decimal place, was obtained on the eleventh cycle. $R_{(2)}$ is an even more difficult to interpret indicator of convergence here than with $\mathscr{L}_2(\alpha)$; however, it is tabulated below for comparison.

η	$R_{(2)}$
1	0.944444
2	0.143518
3	-0.072145
4	-0.074620
5	-0.038373
6	-0.013308
7	-0.002517
8	0.000529
9	0.000770
10	0.000424
11	0.000154
12	0.000032
13	0.000005
14	0.000001
15	0.000000

The results obtained on these three related linear systems indicate a general agreement with those predicted by the weakened convergence proof given in the main body of this paper, i.e., that convergence is essentially a function of the number of relaxation cycles and independent of the number of free parameters.

In the case where free variables exist in the solution, the general theory does not indicate which solution the procedure will converge to. It is somewhat surprizing therefore to find that the solution point, Q, is dependent only on the initial point, Q_o. Linear system $\mathscr{L}_2(\alpha)$ was solved using the following sequences for the selection of the α vectors:

I	1 2 3 4
II	1 3 2 4
III	1 4 3 2

and in each instance the procedure converged in approximately the same number of cycles to the same Q.

0.3499 1.0750 0.4250 -0.7000 1.2250 0.2750

One might conjecture that this is attributable to the initial point being $\alpha_{(1)}$ in each case. The following sequences were also used.

IV	2 3 4 1
V	3 2 1 4

and the same terminal value for Q was found. These sequences are all characterized by having every element occur with equal weight in simple sequences of length n. To show that this is not the explanation, the following two sequences were used, in each of which the occurrence of the elements is weighted.

VI	1 4 1 3 2 4 3 1 2 4
VII	3 2 4 3 1 2 4 1 4 1

Sequence VII is sequence VI cyclically permuted three steps to the left. In each case the procedure still converged to the same Q, although somewhat more slowly, nineteen cycles for four decimal place accuracy in Q. Finally one might conjecture that since these sequences are all cyclical, the final value of Q is dependent on the cyclical exposure scheme. $\mathscr{L}_2(\alpha)$ was solved several times using random sequences for relaxation, and the procedure converged in every case to the same Q. As was noted earlier the random relaxation technique required approximately fifty cycles to achieve four decimal place accuracy. $\mathscr{L}_1(\alpha)$ was investigated under similar circumstances with the same results.

The foregoing results would suggest that the final solution point is dependent on the initial point, Q_o. The following two test cases illustrate this. In the first case the system was initialized to start at the point

$$Q_o' = (1, 2, 5, 1, 0, 4).$$

The procedure converged in this case to a Q of:

1.000000 1.500000 1.500000 -2.000000 -0.500000 0.500000

regardless of the relaxation sequence chosen. In the second case the system was initialized to start at the point

$$Q_o'' = (0.5,\ 1.5,\ 1.5,\ 1.5,\ 0,\ 0)$$

in which case the procedure converged to a Q of:

0.700000 1.650000 1.350000 -1.400000 -0.050000 -.050000

One can show that this is an inherent property of the solution algorithm; however the foregoing example has illustrated this basic property of the procedure. The general proof is to be given in the analysis of the probabilistic associative memory in another paper.

Experiment No. 3

The extremely simple linear systems used in experiments No. 1 and No. 2 were discussed in exhaustive detail to show as clearly as possible the behavior of the iterative procedure itself, i.e., to indicate the answers to questions concerning: convergence, the effect of varying the sequence in which the elements are relaxed or indeed of using random relaxation and the effect of starting from differing values of Q_o. The remaining experiment with linear systems have just one objective: to show convergence for large systems. The present experiment is a step in that direction, and will allow us to introduce a running measure of convergence which will prove to be invaluable in comparing algorithms, etc. For these larger systems it becomes impractical to tabulate Q, and indeed pointless to do so. Instead we shall discuss only the convergence of the iterative procedure.

In this experiment a set of forty linear equations (with binary coefficients) in forty unknowns, and a half set of these equations were solved. It was desirable that these equations all have precisely the same Hamming weight to make them more nearly like the α vectors generated by the associative memory. To accomplish this the following scheme was employed. Each of the possible distinct combinations of two 1's and three 0's were assigned a code number: 0, 1, $\cdots$, 9. A random number generator was used to generate 320 random digits, each block of eight digits corresponding to a single linear element, and the appropriate combination was substituted for the decimal code digits to yield the system of forty random binary coefficients and forty linear elements. The first five elements are given in octal form below to indicate the general character of the linear system.

1 5 1 2 1 4 4 1 6 1 1 3 2 0
0 6 4 5 4 1 4 2 4 3 1 3 0 4
1 2 4 4 3 5 1 2 3 0 0 7 2 0
0 5 1 0 5 4 5 4 2 1 4 5 1 0
1 1 0 5 1 5 0 5 1 1 3 0 1 4

The statistically predicted overlap for such a system is 16%. This compares very well with the actual overlap of 15.8%. We thus have a system with a precisely known Hamming weight, 16 or a density of 40%, and a nearly statistically perfect overlap. One point still has to be insured -- the compatibility of the entire linear system. To insure this we assumed a $\hat{Q}$ and computed the functional values to be $\alpha_{(i)} \cdot \hat{Q}$, thus guaranteeing the linear system to be a compatible one, even if the α's failed to be linearly independent (as they did). The $\hat{Q}$ was chosen to be

$$\hat{Q} = (\hat{q}_1, \hat{q}_2, \cdots, \hat{q}_{40})$$

where

$$\hat{q}_i = i \qquad i = 1, 2, \cdots n$$

The function f computed in this manner has a mean of

$$\bar{f} = \frac{n(n+1)}{2} p = 328$$

and a standard deviation of

$$\sigma = \frac{n+1}{2} \sqrt{np(1-p)} = 63.5$$

This system of forty compatible equations in forty unknowns is the basis for the first portion of experiment No. 3.

The following indicator of convergence was defined:

$$E = \frac{1}{T} \sum_{i=1}^{n} \frac{|R_{(i)}|}{|\alpha_{(i)} \cdot \hat{Q}|}$$

defined over a cycle, or in the case of random sequences, over n successive elements. E is in some sense a mean absolute percentage of error in estimating $f_{(i)} = \alpha_{(i)} \cdot \hat{Q}$, and consequently is a useful indicator of convergence. It is important that $\bar{f}$ be well removed from 0, and that the σ be small enough to not produce large fluctuations in the percentages. The generating scheme described above was chosen to satisfy all of these requirements.

The 40 x 40 system was solved, first using a simple cyclical sequencing, and then using random sequencing. The convergence for the cyclical sequence is shown in Figure 1 and for the random sequence in Figure 2. These plots also show one of the empirically discovered features of this algorithm; namely that a plot of the logarithm of the mean absolute percentage error versus the logorithm of the cycle number is approximately a straight line. This is qualitatively true of all the systems which have been solved to date. Again it is worthwhile to note that the only essential effect of using a random sequence is to make the procedure converge at approximately half the rate of the cyclical sequence.

The second portion of this experiment consisted of solving the first twenty elements of the preceding linear system using both a simple cyclical sequence and a random sequence for relaxation; i.e., a linear system of twenty equations in forty unknowns. E for the cyclical sequence is plotted in Figure 3 and for the random sequence in Figure 4.

Experiment No. 3 was planned to demonstrate the behavior of the sequential relaxation technique when applied to a larger, and therefore more interesting, linear system, and to introduce E as a running index of convergence. Advantage was taken of the opportunity to further illustrate the comparative behavior of cyclical and random relaxation sequences.

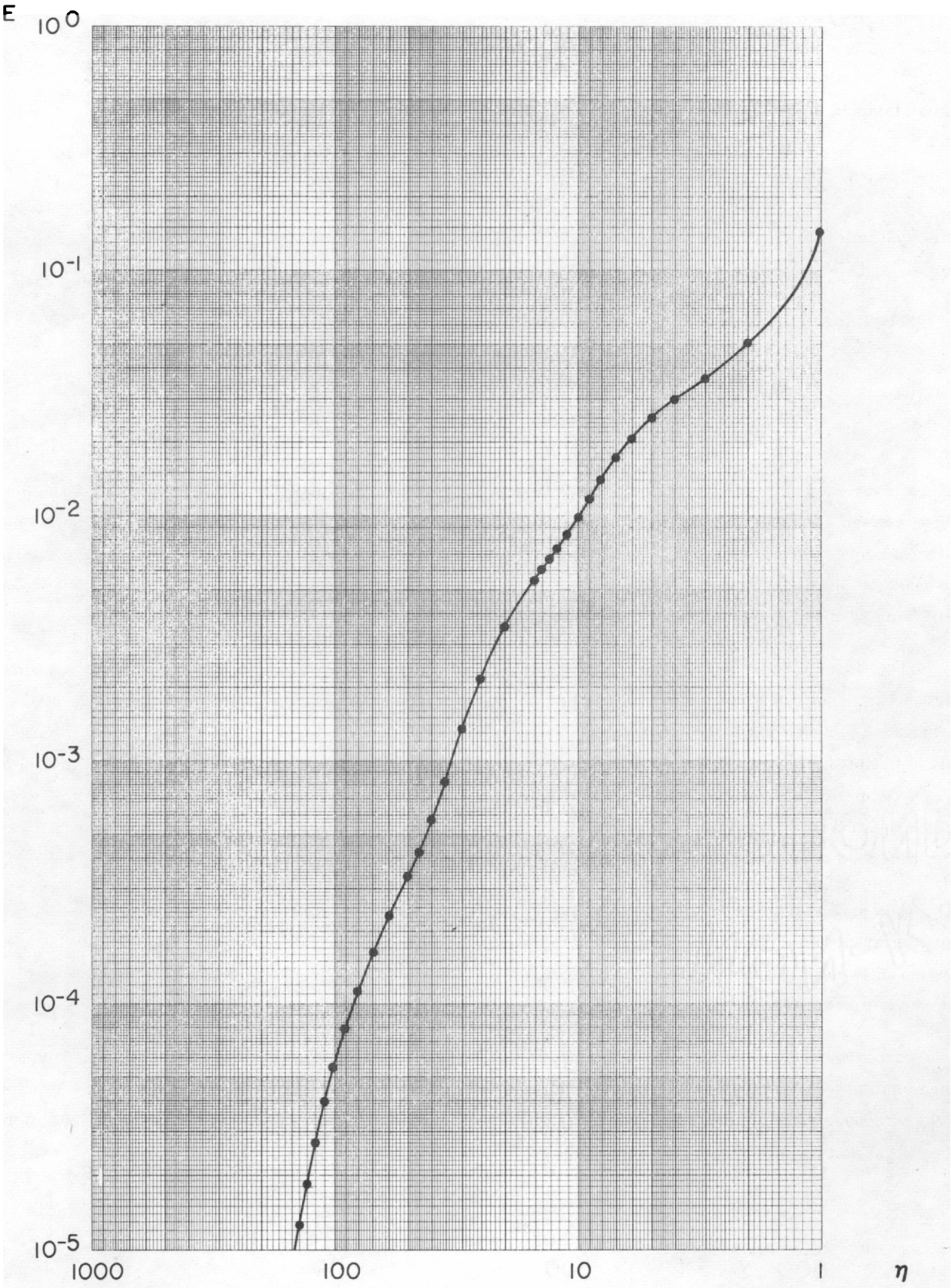


Figure 1. Convergence of a 40 x 40 linear system: cyclical sequence

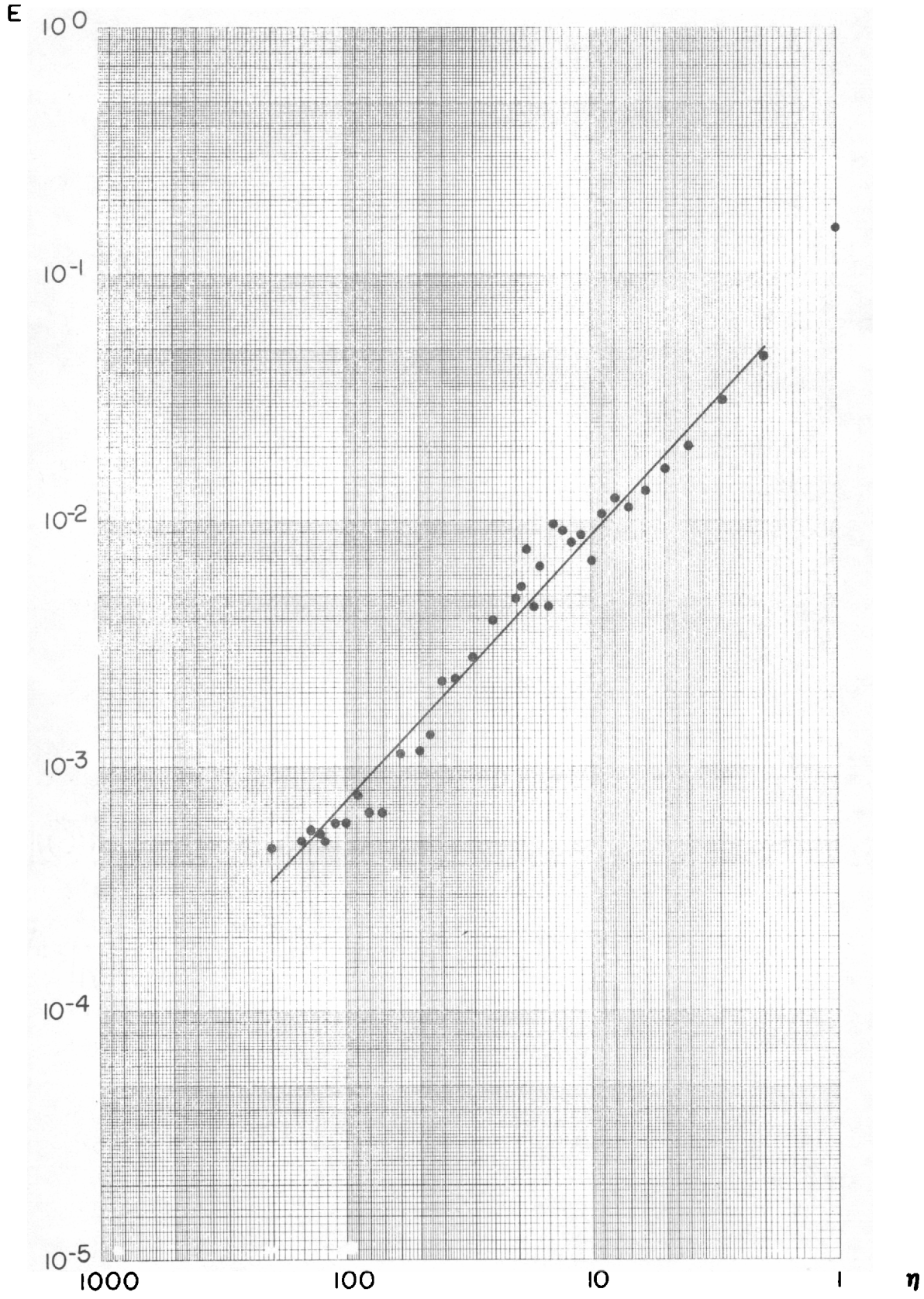


Figure 2. Convergence of a 40 x 40 linear system: random sequence

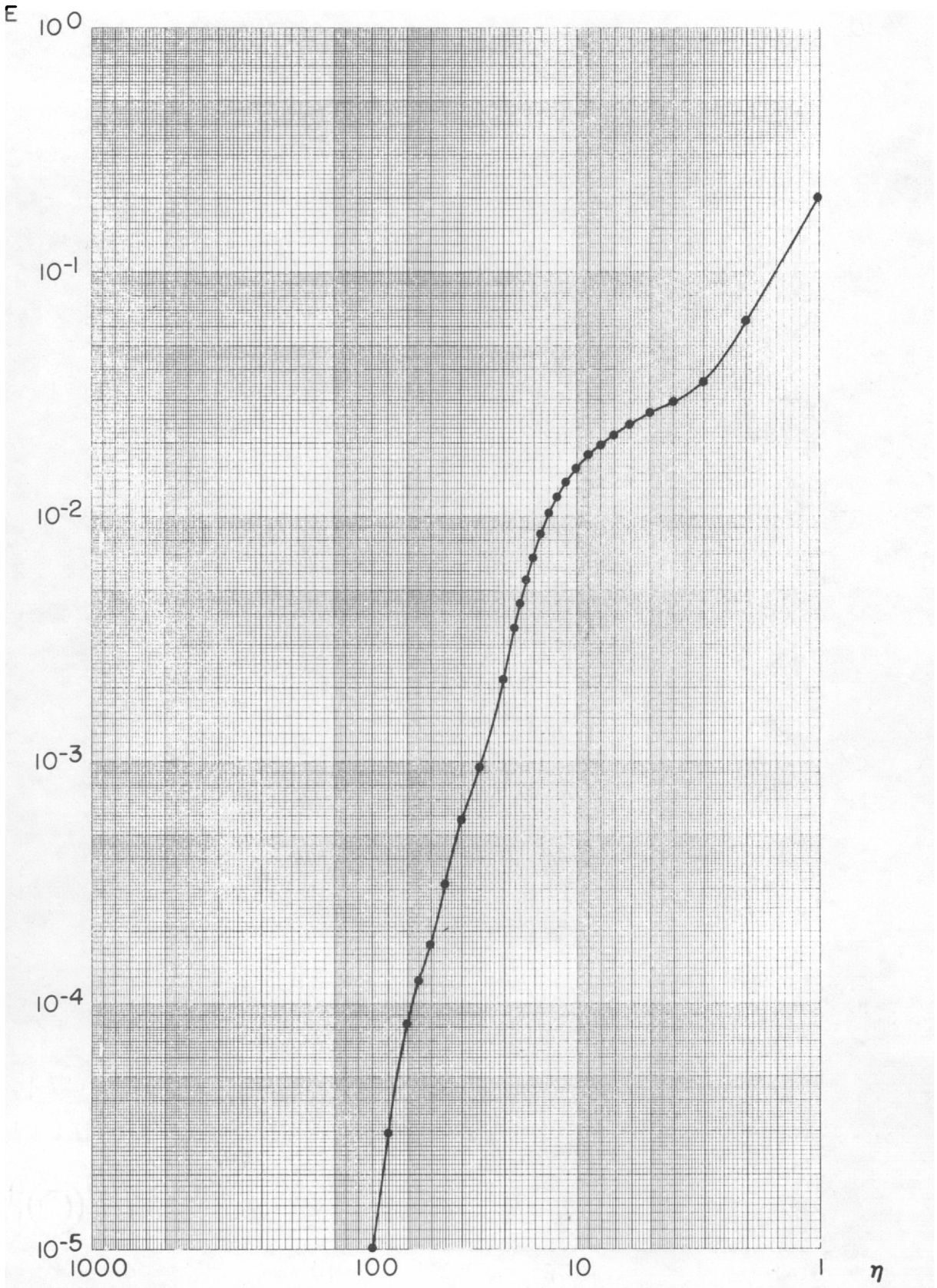


Figure 3. Convergence of a 20 x 40 linear system: cyclical sequence

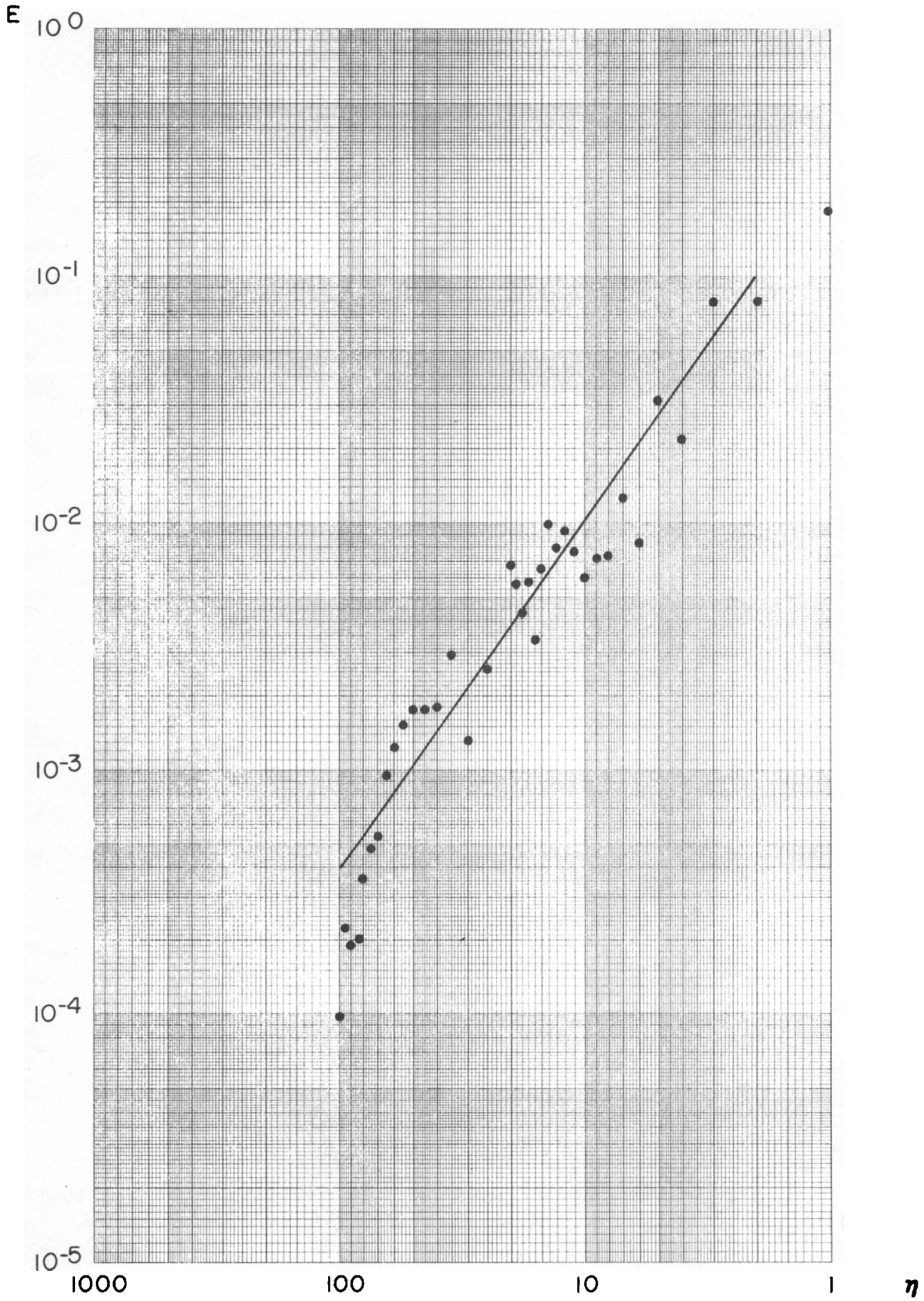


Figure 4. Convergence of a 20 x 40 linear system: random sequence

Experiment No. 4

The objectives of this experiment are basically the same as those which prompted experiment No. 3; however, there are two differences:

1. The linear system has been increased from 40 x 40 to 100 x 100 or from 20 x 40 to 50 x 100.
2. The coefficients have been directly assigned by the random number generator so that the Hamming weight is only statistically defined.

The α elements are generated by a random number generator with probability of having a 1 in any single position being 0.2. The mean Hamming weight of the α vectors therefore is

$$\overline{H}_\alpha = pn = 20$$

with a standard deviation of

$$\sigma_\alpha = \sqrt{\overline{H}_\alpha(1 - p)} = 4.$$

After the α elements were generated by the above scheme, the functional values were assigned using the same procedure described in experiment No. 3. The mean functional value then is given by

$$\overline{f} = \frac{n(n + 1)}{2} p = 1010$$

with a standard deviation of

$$\sigma = \frac{n + 1}{2} \sqrt{np(1 - p)} = 202$$

In this experiment only the cyclical sequence for relaxation was used. Figure 5 shows a plot of the convergence of the system of one hundred equations in one hundred unknowns as a function of the number of cycles through which the system has been relaxed.

Just as was done in experiment No. 3, we also solved the first half of the linear system, i.e., fifty equations in one hundred unknowns to determine the effect of having just as many free parameters as there are dependent variables. Figure 6 shows the convergence of this system, again using a cyclical sequence.

The linear system investigated in experiment No. 4 corresponds more closely to the systems which will be generated by the probabilistic associative memory, whereas the system of experiment No. 3 corresponds more closely to those generated by a complete binary matrix. In the latter case, the Hamming weight of all α elements is exactly the same, while in the probabilistic case the Hamming weight is only statistically specified. One would expect the behavior exhibited in this case to extrapolate directly to larger linear systems and thence to the linear systems generated by the associative memory. The next two experiments show this to indeed be the case.

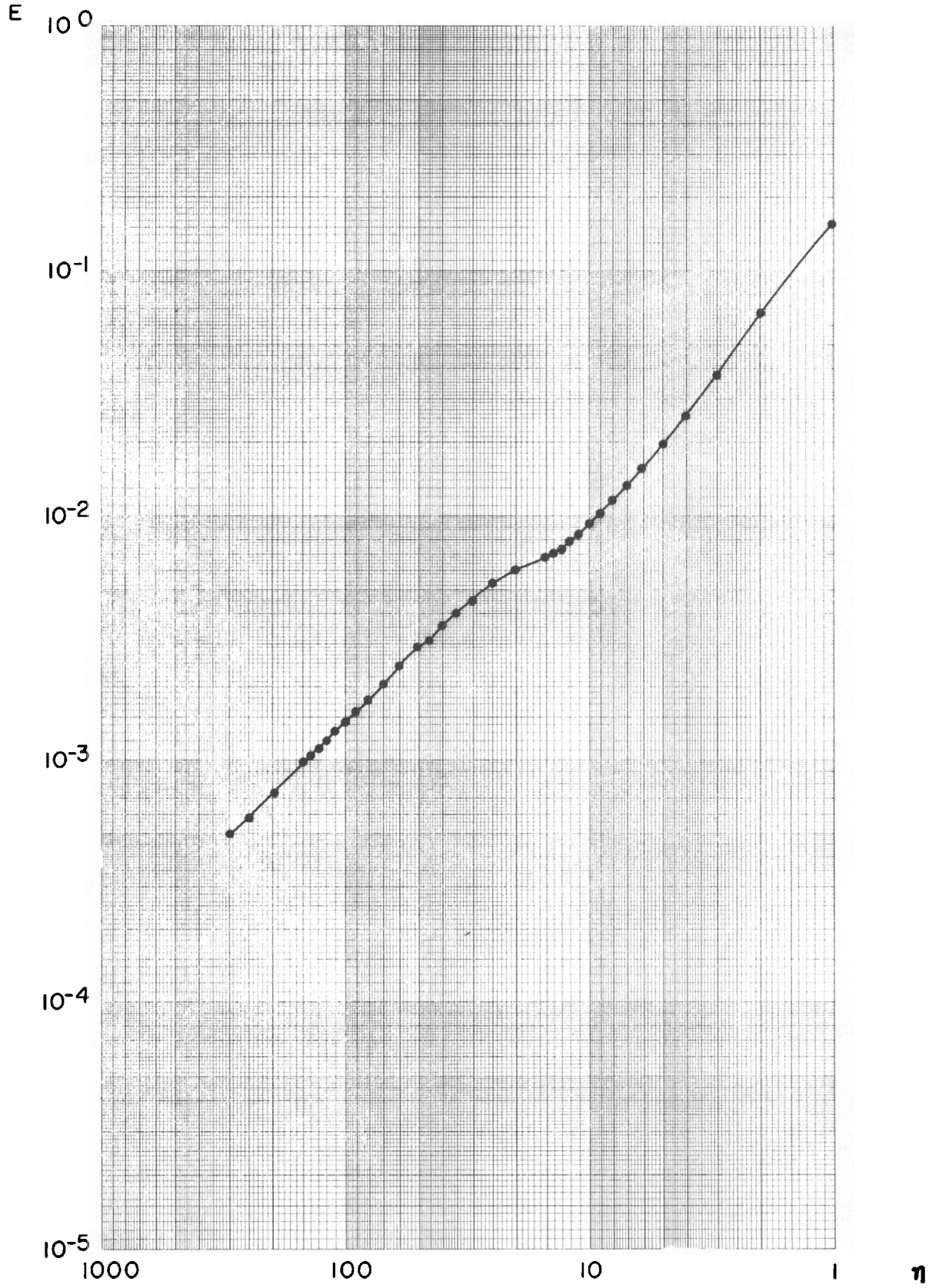


Figure 5. Convergence of a 100 x 100 linear system: cyclical sequence

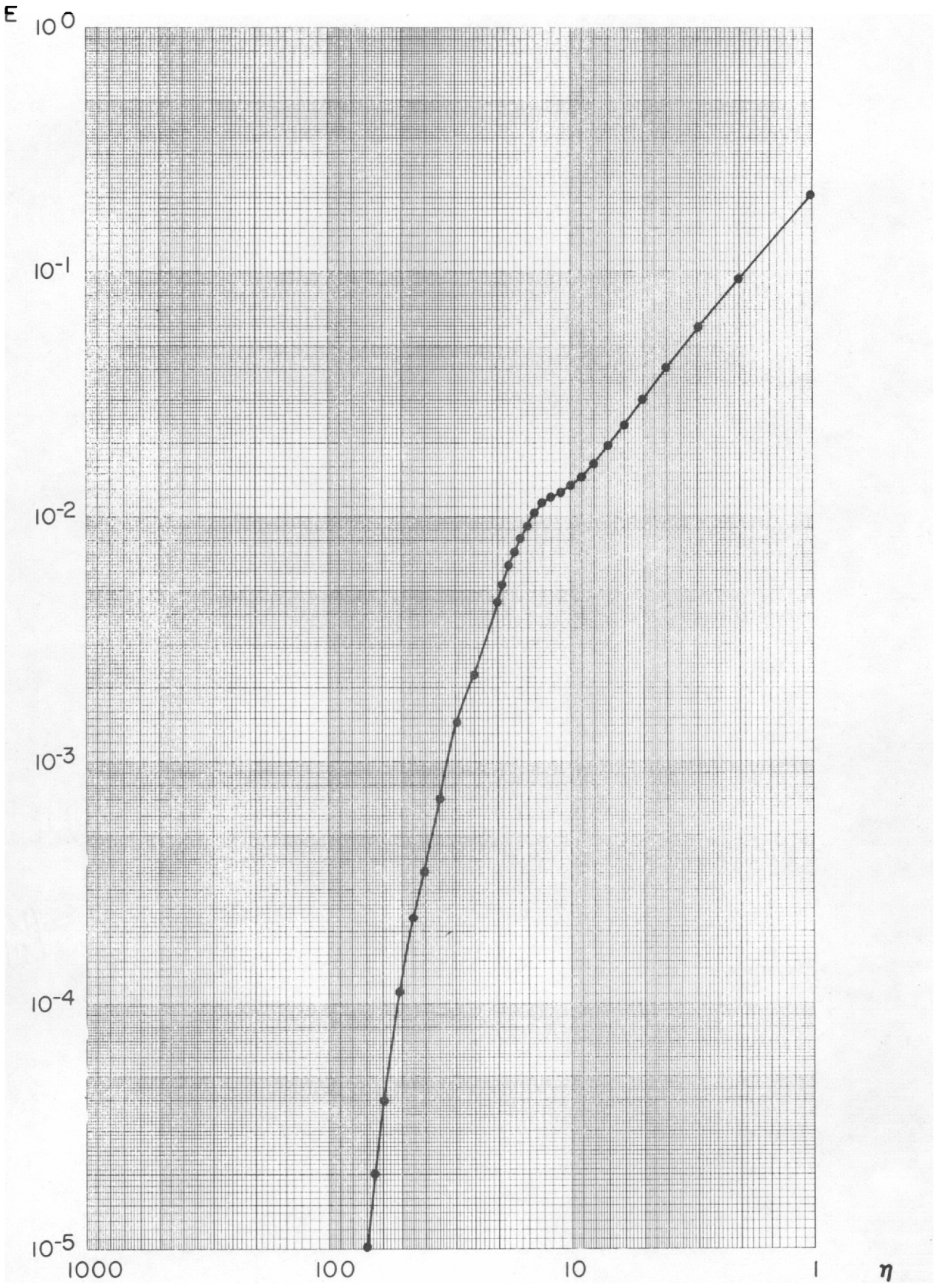


Figure 6. Convergence of a 50 x 100 linear system: cyclical sequence

Experiment No. 5

Since the linear systems generated by complete associative memories will characteristically have many more free parameters than there are dependent variables, the systems investigated in this and the next experiment are both of this type. The purpose of these two experiments is simply to increase the size of the linear systems to a point comparable to those which would be generated by interesting associative memories.

The linear system dealt with in this experiment has 250 equations in 500 unknowns, which provides an impressive problem for solution by any standards. The α vectors were generated by the same procedure used in experiment No. 4 and compatibility of the system was assured again by assigning functional values as in experiment No. 3. The resulting α vectors have a mean Hamming weight of 100 (using $p = 0.2$) and a standard deviation of Hamming weight of 8.94. The functional values have a mean of

$$\overline{f} = 25,050$$

with a standard deviation of

$$\sigma = 2,240$$

The convergence behavior of this linear system, when solved using cyclical relaxation, is shown in Figure 7.

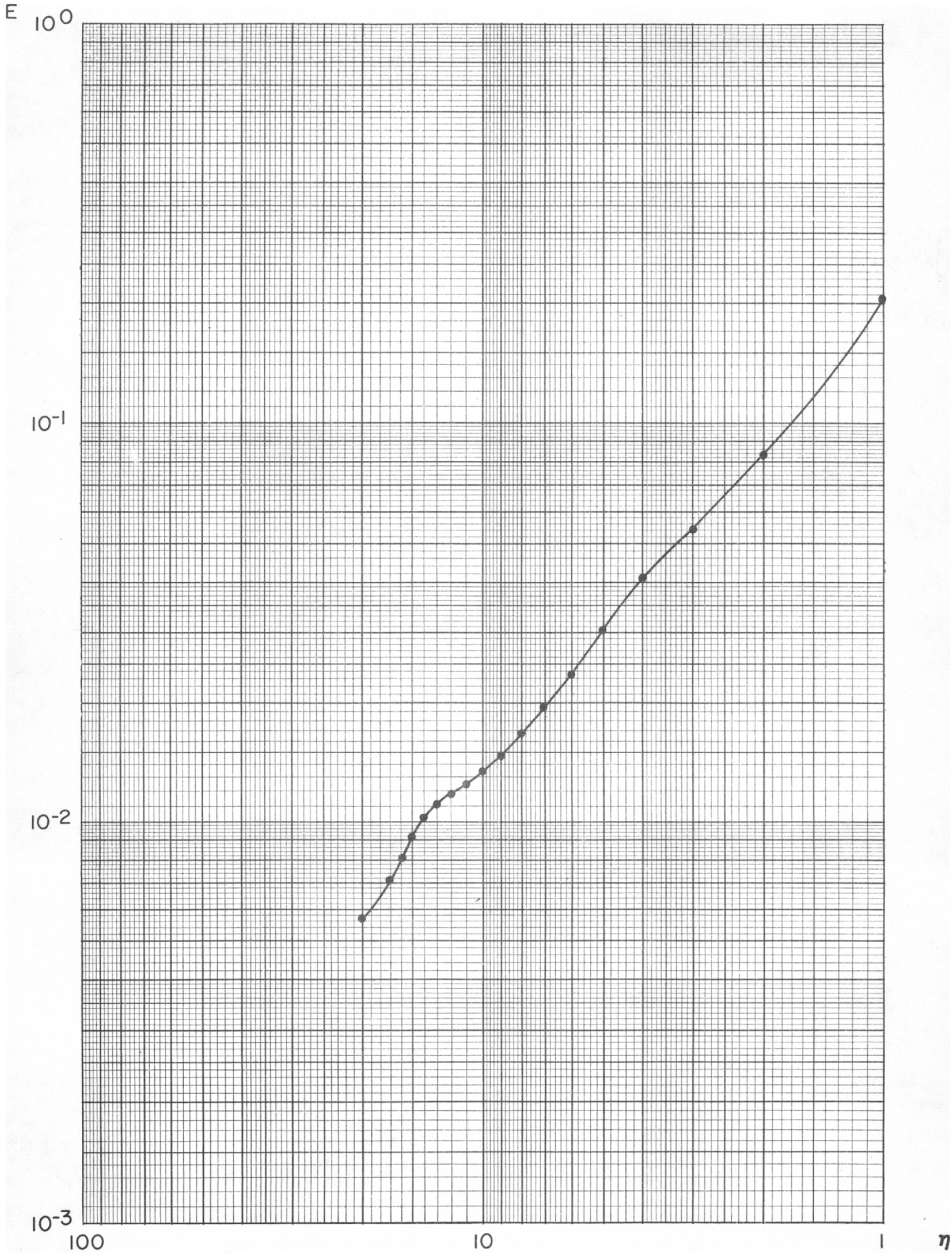


Figure 7. Convergence of a 250 x 500 linear system: cyclical sequence

Experiment No. 6

This last experiment (to be described in this paper) on linear systems differs from the preceding experiment only in the size of the linear system considered -- 250 equations in 1000 unknowns. The generating procedure for the α elements and the method of assigning the functional values to insure compatibility are the same as those already discussed. The essential parameters descriptive of this problem then are:

size	250 x 1000
p	0.2
$\overline{H}_{\alpha}$	200
σ_{α}	12.65
$\overline{f}$	100, 100
σ	6, 331

The convergence behavior of this linear system, when solved used cyclical relaxation, is shown in Figure 8.

The test problems investigated in this series of six experiments were selected to show the essential properties of solutions generated by nonselective relaxation, i.e., relaxation which is nonselective in the sense that the set of residuals is not examined to determine which linear element is to be relaxed, and also in the sense that the coefficients are not considered in selecting an element (or elements) to be relaxed. In addition to illustrating these general behaviorial characteristics, the size of the systems being solved was gradually increased until they were comparable in magnitude to those generated by interesting associative memories. With these comments we leave the subject of the nonselective relaxation technique and consider instead actual associative memory studies.

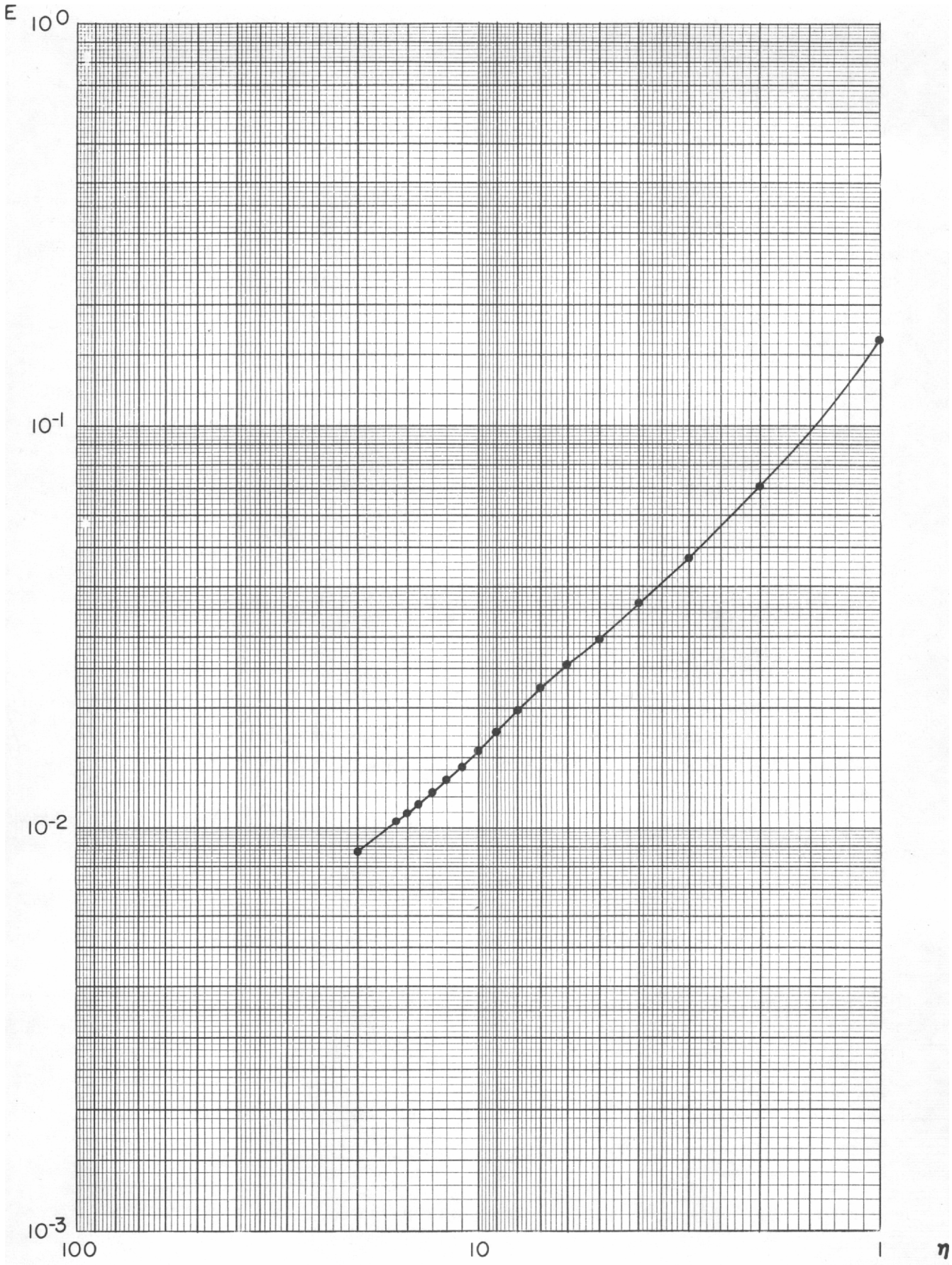
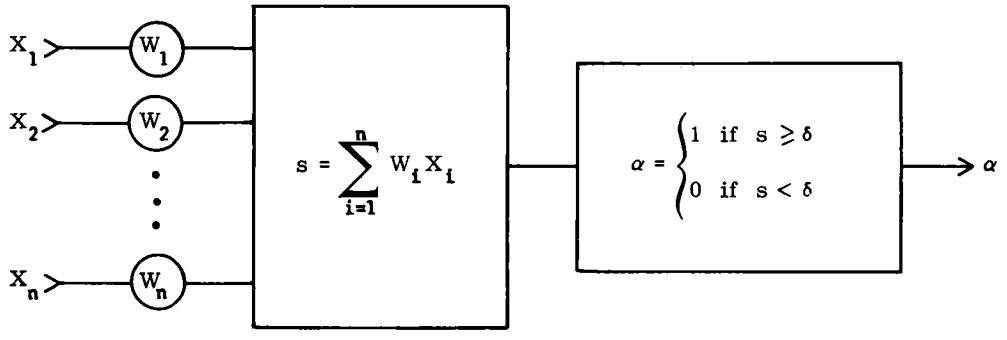


Figure 8. Convergence of a 250 x 1000 linear system: cyclical sequence

Experiment No. 7

One of the more interesting classes of decision functions are those generated in treating the general problem of pattern recognition. The trivially simple example with which we introduce the experiments dealing with the associative memory is such a function, and was chosen because it, like the trivial linear system of experiment No. 1, is small enough to be easily treated conventionally and at the same time because it illustrates one of the unique features of the associative memory.

We may obviously consider the points in the input space, X, to be "patterns," where the pattern elements are the variables and the decision function f(X) is a binary valued function which assigns the pattern to a category or to its complement. The question of when an arbitrary pattern classification can be achieved by a threshold logic element has been exhaustively investigated and reported in the literature so that we need only summarize the single point required for this experiment. The usual threshold logic element is represented as:



where α and the X_i are binary variables and the W_i are suitably limited weighting functions. If the W_i are all assumed to be either 0's or 1's, as occurs in our specification of a reference point on the surface of the n-dimensional hypersphere, then the s value and the associated α generated by the threshold logic element are the same as the s metric and the associated α element generated by the associative memory. Let the X_i be the elements of our well behaved X vector; then the question one wishes to answer is, just which possible combinations of X_i can be classified into the same category, i.e., $\alpha = 1$. The answer to this problem is well known: any grouping of the X_i such that the set which is to be assigned to a single category can be separated from the other possible points by a single hyperplane can be achieved by a threshold logic element. To see that this is true note that the function generated by the element,

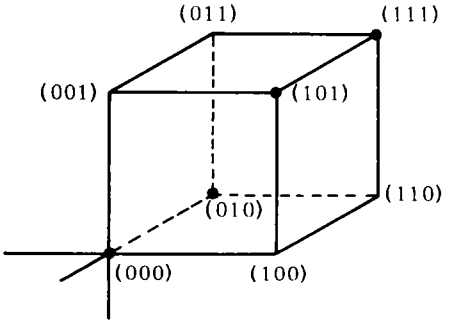
$$W_1 X_1 + W_2 X_2 + \cdots + W_n X_n = s,$$

is an arbitrary hyperplane in the space where the input points exist. Without saying anything about the procedures by which the W_i might be found, it is sufficient for our purposes to note that such an element does produce a single cutting hyperplane.

The rather lengthy introduction of the preceding paragraph was necessary in order to demonstrate the unusual feature of the simple example to be described here. We wish to define a simple class of patterns which defeat classification by a threshold logic element and which still have an intuitive appeal as patterns. The following scheme meets both of these criteria. Consider an n bit binary word, $(X_1 X_2 \cdots X_n)$, and define a parity function $\delta(X)$ to be the number of parity changes in the word, i.e., transitions from an element which is 0 to a next element which is a 1, or vice versa. Our rule then is the following: if $\delta(X)$ is even we assign X to the class, i.e., $f(X) = 1$, and if $\delta(X)$ is odd $f(X) = 0$. This will necessarily assign exactly half of the elements to the class since we are summing over the alternate binomial coefficients which occur as the multiplicities of the various possible values for $\delta(X)$. For an example, for $n = 3$ we get the following classification of patterns:

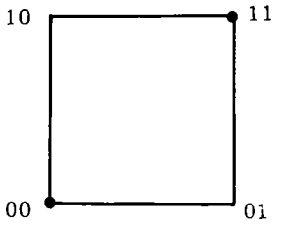
X	$\delta(X)$	f(X)
0 0 0	0	1
0 0 1	1	0
0 1 0	2	1
0 1 1	1	0
1 0 0	1	0
1 0 1	2	1
1 1 0	1	0
1 1 1	0	1

which maps on the hypercube as follows:



The preceding example is obviously one which cannot be classified with a cut in 3-space; however, if this problem were mapped using a complete binary matrix the above points would be embedded in a 64-dimensional space so that classification by a single cut would be possible.

The example used in this experiment is the two dimensional case of the above pattern generation scheme, i.e.,



which will map into a region of a 16-dimensional space using the complete binary matrix. For $\delta = 1$ we get as the linear system

$$\begin{array}{lcccccccccccccccc}
\alpha_{(1)} & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \\
\alpha_{(2)} & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\
\alpha_{(3)} & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\alpha_{(4)} & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1
\end{array}$$

and for $\delta = 2$ we get

$$\begin{array}{lcccccccccccccccc}
\alpha'_{(1)} & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
\alpha'_{(2)} & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\
\alpha'_{(3)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\
\alpha'_{(4)} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1
\end{array}$$

where the input X points were chosen to be

$$\begin{aligned}
X_{(1)} &= 0 \quad 0 \\
X_{(2)} &= 0 \quad 1 \\
X_{(3)} &= 1 \quad 0 \\
X_{(4)} &= 1 \quad 1
\end{aligned}$$

The functional values are

$$\begin{aligned}
f_{(1)} &= 1 \\
f_{(2)} &= 0 \\
f_{(3)} &= 0 \\
f_{(4)} &= 1 \ ,
\end{aligned}$$

which of course applies to both sets of α vectors. This then is the simple example of a pattern recognition problem which we wish to treat using an associative memory with a complete binary matrix.

There are two ways in which the output of the associative memory can be interpreted. First, the numerical value estimated by the memory as the function associated with a particular address may be assigned to the nearest permissible value for the function, i.e., a 1 or 0. Secondly, it can be required that the memory learn the function to some arbitrary accuracy. As would be intuitively expected, the first task is much the easier of the two. After only six cycles the memory unerringly assigns the function to the proper class for the first family of 2 vectors. However, twenty-eight cycles are required to produce an answer accurate to three significant figures when one requires that the memory produce the actual functional

value. Since the system is so small, the running index used with the larger systems to indicate convergence is not very useful here; instead we shall tabulate the residual for the first few cycles.

η	$R_{(1)}$	$R_{(2)}$	$R_{(3)}$	$R_{(4)}$
1	1.0000	-0.8333	-0.2083	1.1180
2	0.0295	-0.8000	-0.3562	0.9414
3	0.2575	-0.7319	-0.4501	0.7919
4	0.3911	-0.6482	-0.4997	0.6633
5	0.4591	-0.5606	-0.5149	0.5519
6	0.4823	-0.4756	-0.5051	0.4555
7	0.4756	-0.3971	-0.4781	0.3727
8	0.4499	-0.3296	-0.4403	0.3019
9	0.4129	-0.2654	-0.3966	0.2420
10	0.3702	-0.2127	-0.3506	0.1918
.				
.				
.				
28	0.0031	0.0035	-0.0022	-0.0034

Similar results are obtained in solving the α' system, although convergence is greatly enhanced by a lower Hamming weight of the α vectors. The system converges so rapidly for this δ that under the first rule of assigning the classification to be the nearest permissible functional value the classification is unerringly made after the second cycle. However, the computation of the exact functional value is not significantly better than in the preceding case. The residuals for some cycles are given below.

η	$R_{(1)}$	$R_{(2)}$	$R_{(3)}$	$R_{(4)}$
1	1.0000	-0.5000	-0.3750	1.1875
2	0.1406	-0.5703	-0.5214	0.5107
3	0.4182	-0.3341	-0.3809	0.2529
4	0.2942	-0.1783	-0.2290	0.1301
5	0.1711	-0.0934	-0.1273	0.0675
.				
.				
.				
28	0.0000	-0.0017	-0.0111	0.0056

This experiment was intended to show the behavior of an associative memory with a complete binary matrix on a simple pattern recognition problem of a type which cannot be handled by a threshold logic element. Thus there is no α element which fires uniquely with the classification function. In spite of this fact, the system converged to a good solution by either of two standards: either by assigning the functional value generated by the associative memory to the nearest permissible functional value or by requiring the memory to actually generate the required functional values.

It is of some interest to note the general reduction in matrix size which could have been made in the above matrices with no loss in generality, actually with no change in functioning, for this problem. If one notes that columns 4, 8, 11, 13, 14, 15, and 16 are all identical for the α vectors, then they may be

replaced by a single equivalent column. Similarly, the first column which is all zeros may be deleted without affecting the linear system. Therefore an equivalent system is the following:

$$
\begin{array}{lcccccccc}
\alpha_{(1)} & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 \\
\alpha_{(2)} & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 \\
\alpha_{(3)} & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \\
\alpha_{(4)} & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1
\end{array}
$$

A similar reduction for α' may be made. The point of this comment is that the complete binary matrix introduces a high degree of redundancy in the formation of the α vectors, and that a much smaller matrix suffices.

The preceding discussion of experiment No. 7 has been made at such length because it is only in a small memory that the detailed behavior can be shown, or indeed that a complete binary memory can be realized. The following experiments are all made with probabilistic matrices of such size that the running index of convergence is the only parameter which it is feasible to exhibit.

Experiment No. 8

This experiment, like experiment No. 7, is concerned with an application of the associative memory model to a problem in pattern recognition. However, unlike the problem treated in the previous experiment, this one while simple is certainly not trivial. The experiment will be described in two stages: first the procedure by which the patterns are generated will be described and analyzed, and then the associative memory which was used and its functioning will be discussed.

It is desirable in order to simplify the analysis of the results that the pattern generation scheme assign approximately half of the possible inputs to the class of patterns and half to the class of nonpatterns. It is also desirable that the procedure have an intuitive appeal as a pattern generator, that is that the patterns should be recognizable by some relatively simple logical test. The following satisfies both of these criteria. Consider a four-element mosaic arranged as a 2 x 2 square, where each element can assume any one of a discrete range of values: i.e.,

X_{11}	X_{12}
X_{21}	X_{22}

where the X_{ij} designates the particular value assumed by the ij-th element. The pattern which is tested for in this experiment consists of a bar, either vertical or horizontal, where both the bar and the background are subject to noise. This is accomplished by the following logical test:

$$f(X) = b_h \oplus b_v$$

where

$$b_h = \begin{cases} 1 & \text{if } \min(X_{11}, X_{12}) > \max(X_{21}, X_{22}) \\ & \text{or } \min(X_{21}, X_{22}) > \max(X_{11}, X_{12}) \\ 0 & \text{otherwise} \end{cases}$$

$$b_v = \begin{cases} 1 & \text{if } \min(X_{11}, X_{21}) > \max(X_{12}, X_{22}) \\ & \text{or } \min(X_{12}, X_{22}) > \max(X_{11}, X_{21}) \\ 0 & \text{otherwise} \end{cases}$$

This scheme has the intuitive appeal, if one interprets the range of possible values of X as a grey scale, of detecting a noisy pattern on a noisy background. This definition of a pattern is unnecessarily stringent, since in the case of a black and white mosaic it says that only perfect patterns can be recognized; however, for the moment it suffices as a generation scheme.

The probability of patterns, i.e., bars, versus nonpatterns is simply calculated. We shall define the probability in general, and then compute the values for the particular case under consideration here. Let the total number of elements in the mosaic be n, without regard for their relative orientation, and let the number of elements in a "pattern" or figure be m. Also let the number of discrete values (steps in the grey scale) available to X be g, then we can compute the total number of figures possible in a given orientation, N_f. The general rule for the recognition of a figure is that

$$\min (F) > \max (B)$$

where F designates the figure elements and B designates the background elements, i.e., the lightest element in the figure is to be darker than the darkest element in the background. Thus a contrast enhancing procedure which converted all elements of a field to black or white depending on whether they were greater than or equal to the minimum value in a potential figure or less respectively would reconstruct a black figure on a white background with no degradation, had the input mosaic pattern actually been in the class of figures. N_f is then given by:

$$N_f = \sum_{i=1}^{g} (i - 1)^{n-m} \left[(g + 1 - i)^m - (g - i)^n \right]$$

N_f is the number of possible distinct patterns according to the above definition in a particular orientation. We must also consider the possible orientations, as for an example the vertical and horizontal bars in the pattern-generation scheme used in this experiment. Let θ be the number of possible distinct orientations for the pattern on the retina; then the probability that a particular combination of mosaic values is a pattern, P_f, is given by:

$$P_f = \frac{\theta N_f}{g^n}$$

In the present example n = 4, m = 2, g = 48 and θ = 2 (horizontal and vertical bars). N_f is then calculated by

$$N_f = \sum_{i=2}^{48} (i - 1)^2 \left[(49 - i)^2 - (48 - i)^2 \right]$$

$$N_f = \sum_{i=2}^{48} (2i - 3)(49 - i)^2$$

$$N_f = 848,632.$$

and P_f is given by:

$$P_f = \frac{2 \times 848,632}{48^4} = 0.3197$$

The probability that an arbitrary four-bit pattern, with a grey scale of 48 steps, is recognizable as a bar is 0.3197, which provides an adequate density of patterns for the purposes of the present experiment.

The associative addressing matrix used in this experiment had an effective size of 9600 columns, each of which was 192 bits in length. The actual matrix is 1/48-th this size, 192 x 200, but a technique devised by Dr. H. Everett of WSEG makes it possible to use this smaller matrix in such a way as to be equivalent to the larger memory. The small matrix is filled with ones and zeroes, using a random number generator, with the probability of a one being entered in any bit position being a prechosen value p: in this example p = 0.2. The innovation which he introduced to extend this matrix is to use the matrix and each of its cyclical permutations (on column partitions) as a part of the associative addressing matrix. The cyclical permutations are restricted to a single variable range, i.e., within a partition. This precludes the problem of degeneracy which would arise if the random fill of one partition were tested as a template against two variables. Thus if we consider a ng x c basic matrix, the extended matrix will be ng^2 x c in size. This procedure has made it practical to run problems using the present size matrix, whereas the brute force generation, storage, and manipulation of matrices of this size would have been impractical even on large high-speed computers.

As was noted above, the matrix was filled with a probability p = 0.2. The threshold was selected to be $\delta = 3$. Using these parameters it is possible to define the statistical behavior of the memory in generating the associative addresses. The probability that any particular α element is a one, P_δ, is given by

$$P_\delta = \sum_{i=3}^{4} \binom{4}{i} (0.2)^i (0.8)^{4-i} = 0.0256$$

Consequently the mean Hamming weight of the α vectors is

$$\overline{H}_\alpha = 245.76$$

with a standard deviation of

$$\sigma_\alpha = 15.48.$$

This says that the number of α elements which will be operated on in any single relaxation cycle is between 215 and 277 with a 95% likelihood, which is a satisfactory description of the extent of the alteration in Q introduced by relaxation. In the course of this experiment the actual Hamming weights were computed and examined to determine whether they were statistically described by the above relations as a second order check on the behavior of the permuted memory. The agreement indicated the memory to be behaving precisely as predicted.

The associative memory was shown a sequence of 400 patterns constructed by using a random number generator to assign values to the X_{ij}. Whether a particular mosaic input was a pattern or not was determined by applying the logical test described earlier. It is worth noting that these 400 points were drawn

from an environment of $5,308,416,(48^4)$, possible mosaic configurations. The scores of the device are tabulated below, as accumulated over periods of fifty exposures.

Cycle	Number of "patterns" in the cycle	Pattern errors	Nonpattern errors
1	30	27	2
2	34	10	12
3	25	4	16
4	31	8	6
5	33	2	11
6	29	3	10
7	26	4	8
8	26	7	4
Totals	234	65	69

The results tabulated in the preceding paragraph summarize quite well the operation of the associative memory as a pattern recognizer. The fact that roughly the same number of patterns and nonpatterns have been misclassified, out of a population of 234 patterns and 166 nonpatterns, is an indication of unbiased performance in the classification of mosaic figures. These results are shown graphically in Figure 9 where the raw scores, accumulated over cycles of fifty exposures, and a smoothed curve are plotted versus cycle number.

After the first 330 points, i.e., one point for each sixteen thousand in the environment, a cumulative score for the next one hundred points was found. Seventy-nine of the one hundred were assigned to the correct category. The X^2 for this result is

$$X^2 = \frac{(43 - 52 \cdot 0.3197)^2}{52\,(0.3197)(0.6803)} = 60.73$$

which provides a general measure of the quality of the predictions made by the memory, i.e., in the sense that it shows the almost negligible likelihood of the classification having been achieved by chance ($p < 10^{-5}$).

Admittedly the patterns used in this experiment were very simple; still the convergence of the associative memory to the "decision function" on the basis of a very small exposure to the environment is impressive. This experiment on pattern recognition demonstrates the general behavior of the associative memory on such problems, and concludes our discussion of discrete decision function type problems. The next two experiments are concerned with continuous functions, the storage of which is the primary function of the associative memory.

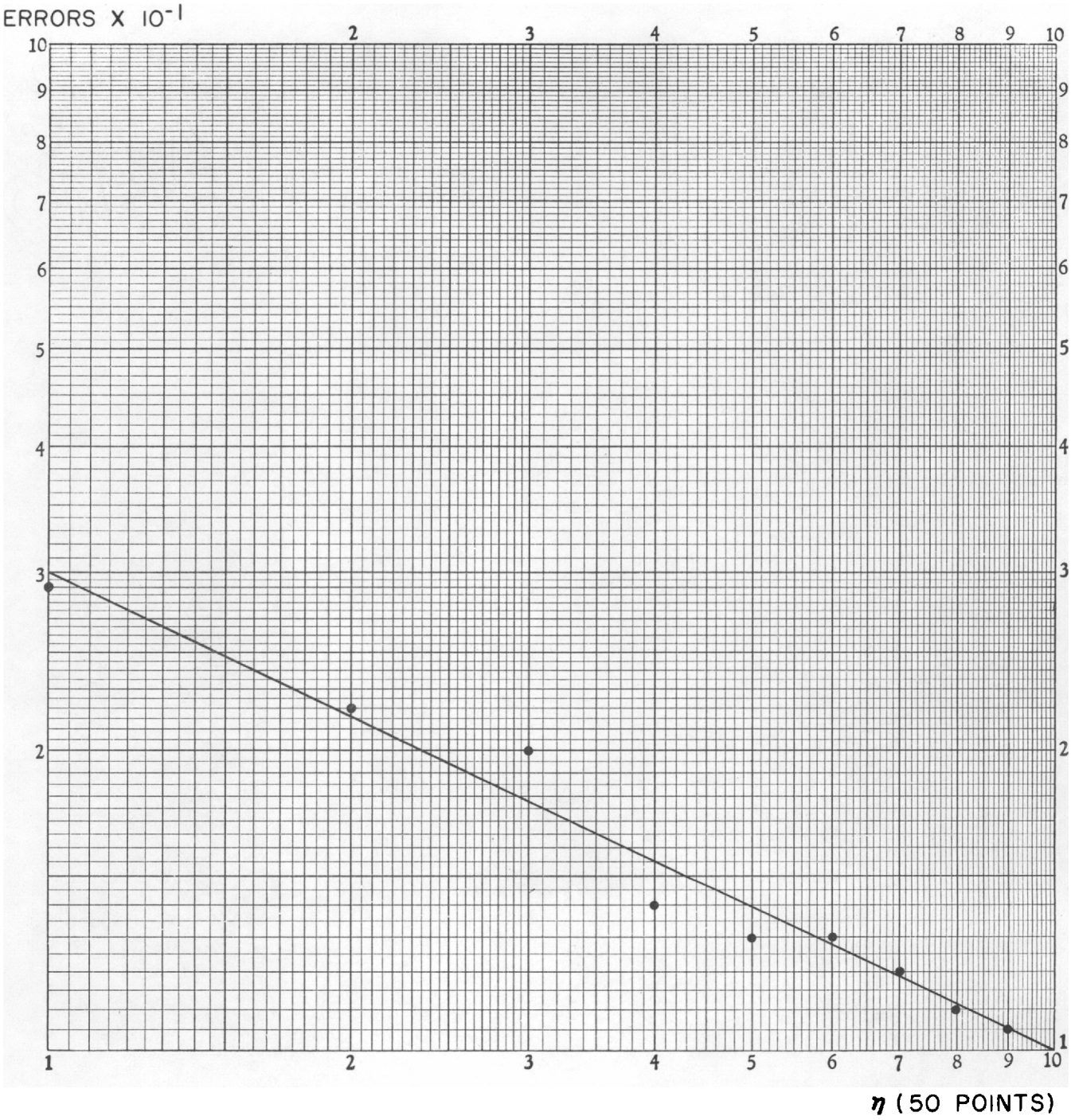


Figure 9. Scores of the pattern recognition program on cycles of 50 points each

Experiment No. 9

The associative memory model was constructed to permit storage of functional values in a distributed manner so that the relative nearness of input points would result in a corresponding overlap of memory elements and therefore a type of quasi-linear interpolation or extrapolation in the memory space. The convergence proof was dependent on the number of α elements being greater than the number of points in the input space. Unfortunately this is rarely feasible, so one of the most important questions to be asked when using such a device concerns the compatibility of the linear system generated when the number of equations greatly exceeds the number of variables. A complete analysis of this problem will be given in the paper on the probabilistic associative memory; however the following experiment provides a measure of the significance of this phenomena.

The simplest continuous function which could be considered would be a nonzero constant. The restriction that the constant be nonzero is essential, since the iterative procedure would never find a solution, other than the trivial $Q_o = (0, 0, \cdots, 0)$, to the homogeneous linear system otherwise. In the present experiment we are concerned with the behavior of the associative memory when applied to a continuous function defined over a large input space; $T >> R$. A constant f(X) is perfectly acceptable in this case; therefore let

$$f(X) = 1$$

for all input points X. Also let X be defined over four variables where each is partitioned into 48 discrete steps. This generates the same size input space as that used in the preceding experiment on pattern recognition -- 5,308,416 points. The problem then can be stated very simply: to what degree are the 5,308,416 equations in 9,600 unknowns generated by the associative memory compatible. The "solution" of such an over-determined system may seem to be a hopeless task; still the number of actual equations is vanishingly small compared to the total number possible, W;

$$W = 2^R = 2^{9600} \cong 10^{2890}$$

This fact does not lessen the difficulty of generating a compatible system, but rather indicates the possibility of success.

Figure 10 shows the quality of convergence obtained in this greatly over-determined system. The raw residuals, $(f(X) - \bar{f}(X))$, are plotted versus the learning points. Since the functional value is constant, $f(X) = 1$, this raw data plot provides a graphical representation of the unsmoothed percentage error. The system converges very well for the first one hundred or so points, and then improves only slightly in the next three hundred points. Although the graph does not extend beyond the first four hundred points, a thousand points in all were run with little or no improvement in the estimation beyond the first two hundred points. The final mean absolute error of 5.5% represents the "built in" incompatibility of the linear system generated by the associative memory. Various subterfuges for minimizing this effect, while preserving the basic associative addressing scheme can be conceived; however the simple model proposed in this paper has the limiting feature illustrated by this experiment.

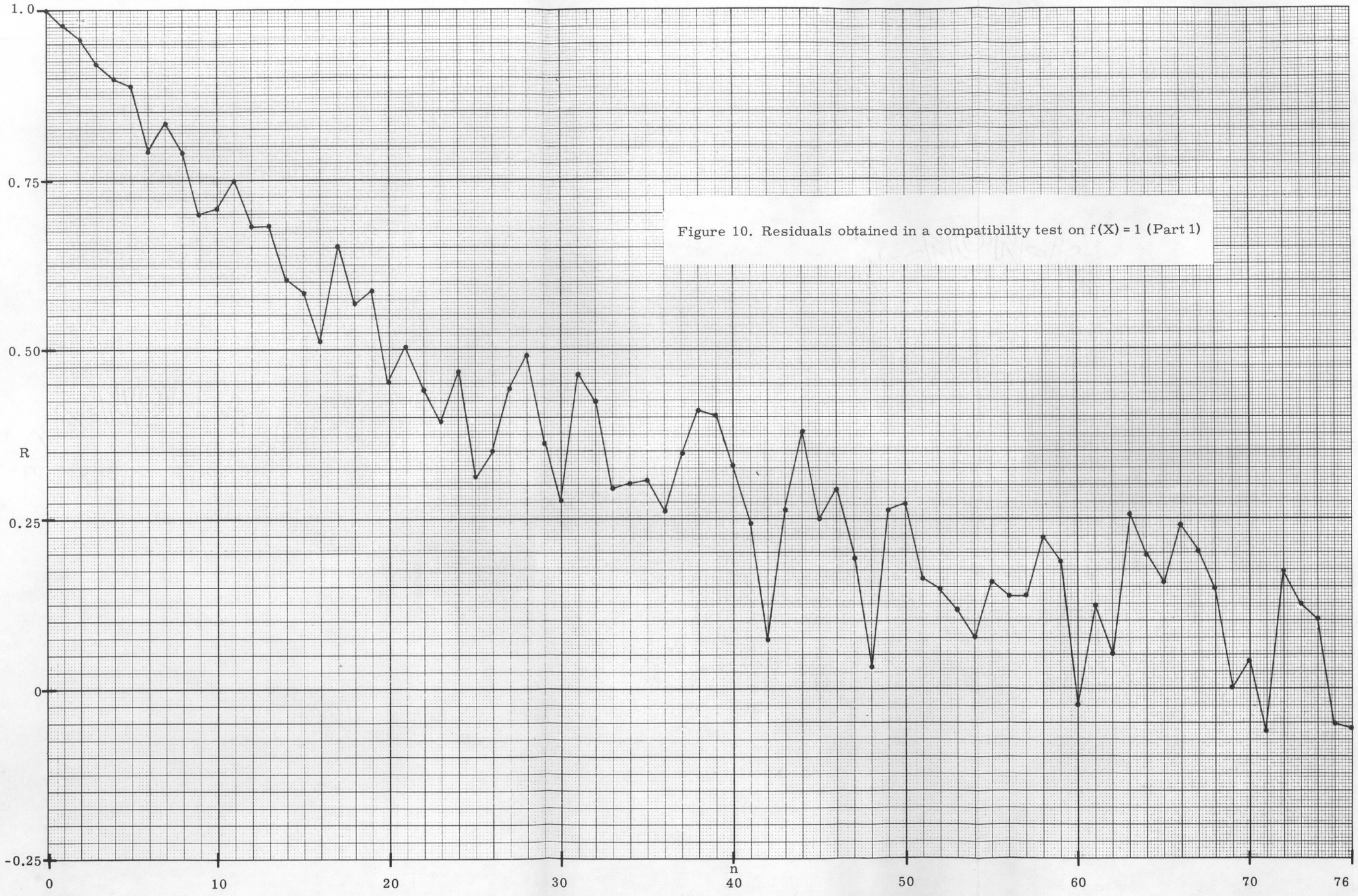


Figure 10. Residuals obtained in a compatibility test on f(X) = 1 (Part 1)

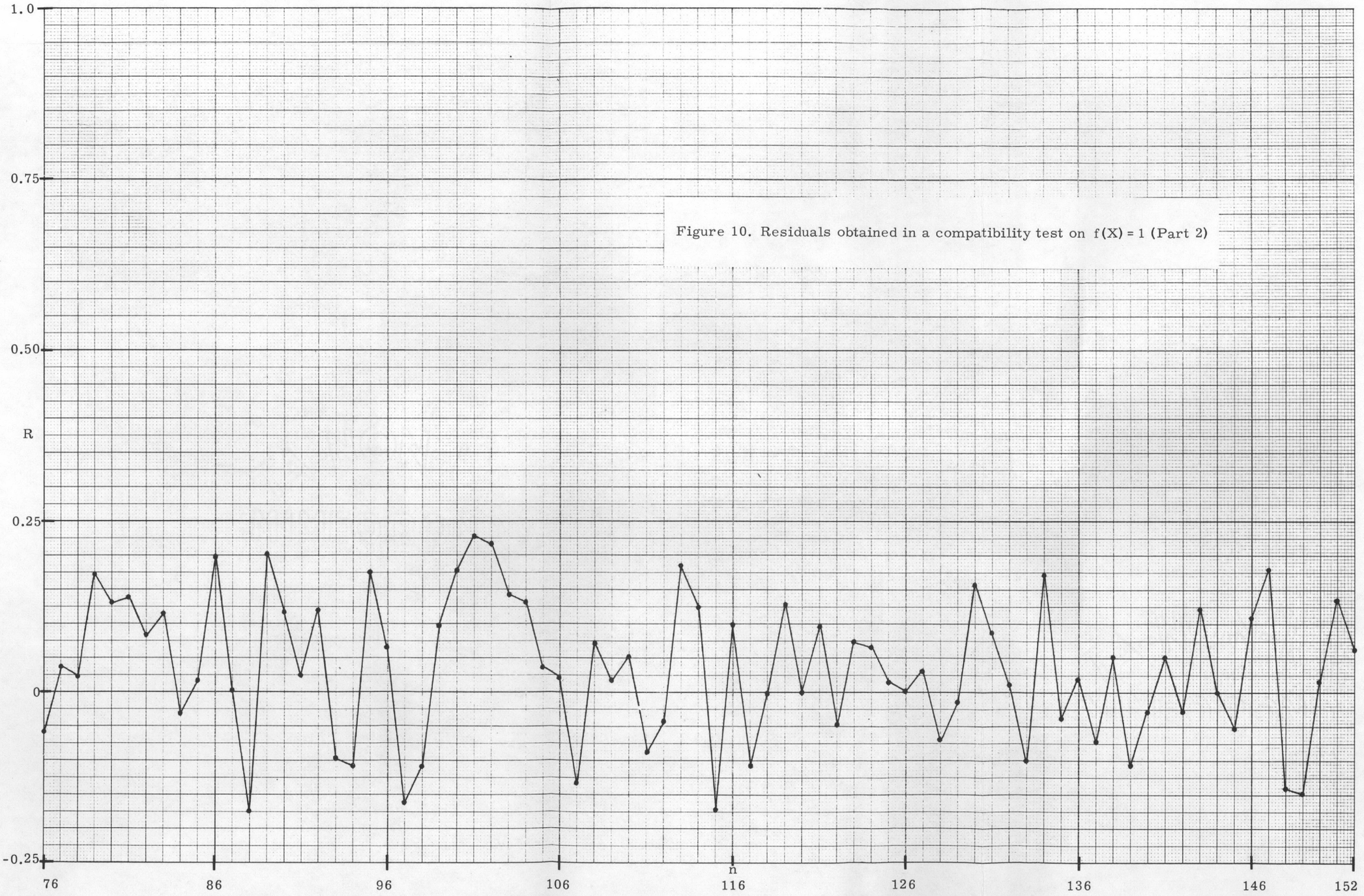


Figure 10. Residuals obtained in a compatibility test on $f(X) = 1$ (Part 2)

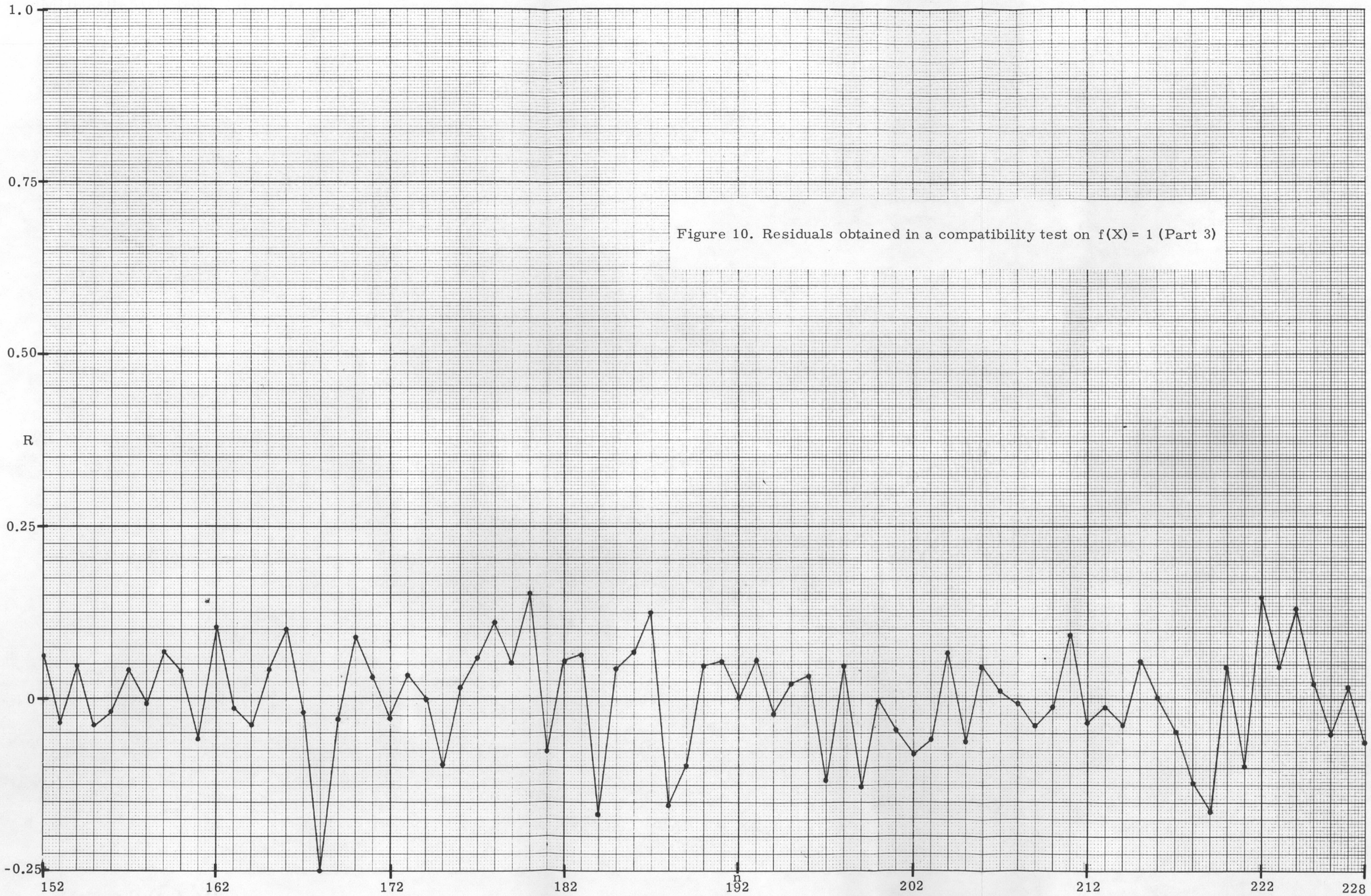


Figure 10. Residuals obtained in a compatibility test on f(X) = 1 (Part 3)

Experiment No. 10

This experiment, the last to be reported in this paper, is devoted to an investigation of the behavior of the associative memory when used to store a simple algebraic function originally used by Smith[23] in his studies of a perceptron-like computer model, ADAP II. This particular function was chosen because it embodied several special features which allow a testing of the associative memory functions and because it permits a qualitative comparison of Smith's results and those achieved using the associative memory. The "Smith function" is given by

$$F_s = X_1 X_2 + 2X_3 + 0 \cdot X_4$$

which has a mean of

$$\overline{F}_s = \frac{n_1 n_2}{4} + n_3$$

where n_i is the number of nonzero steps in the discrete range of the i-th variable. In the case where all variables have the same range one obtains the simpler expression

$$\overline{F}_s = \left(\frac{n}{2} + 1\right)^2 - 1$$

which is used in estimating the q_i when evaluating the convergence of the iterative procedure. For test purposes this function has several advantages. It involves a cross product term which has proven to be a very difficult function for perceptron-like (single layer perceptrons) devices to learn. It has a term which has a zero coefficient and hence, although this variable enters into the associative addressing, cannot affect the functional value. Finally the basic form is a linear summation of terms, a functional form most easily stored in the quasi-linear associative memory. In all of the tests which make up this experiment $n = 9$, i.e., $X_i = (0, 1, \cdots, 9)$.

First we shall describe a series of experiments, the interpretation of which presupposes results from an analysis of a probabilistic associative memory, in just enough detail to indicate the reasonableness of the parameters used in the main experiment to be shown here. The matrix is 192 x 9,600 with the Everett scheme for permuting a base matrix being applied to a basic 192 x 200 array. This associative addressing scheme will be invariant through all of the individual tests which make up this section, although the density with which the matrix is filled will be varied from test to test.

[23] J. W. Smith, "ADAP II - an adaptive routine for the LARC computer," unpublished report (Navy Management Office) communicated by the author, September 1962.

The first test has as its essential parameters, values of

$$p = 0.2$$
$$\delta = 3$$

which gives a mean Hamming weight of

$$\overline{H}_{\alpha} = 261.12$$

The convergence plot for this test is shown in Figure 11. E is defined in a slightly different manner here than before. In the case of the contrived compatible linear systems where E was first introduced, the functional values were tightly clustered about the mean and bounded away from zero so that no difficulty could be encountered in defining E to be the average absolute percentage error. Here the function is very broadly spread, and furthermore not bounded away from zero; therefore the following related definition for E is used in discussing this experiment.

$$E = \frac{1}{D} \sum_{i=1}^{D} \frac{\left| R_{(i)} \right|}{\overline{F}_s}$$

where D = 60 in this experiment.

The second test has as its essential parameters values of

$$p = 0.2$$
$$\delta = 2$$

which gives a mean Hamming weight of

$$\overline{H}_{\alpha} = 1735.68$$

The convergence plot (E versus cycles of D points each) for this test is shown in Figure 12. The substantially poorer convergence in this test as compared to the previous one is qualitatively explained by saying that there is excessive destructive interference in the Q augmentation steps due to the excessive density of the α vectors. One out of every six elements is being changed on each cycle. On the other hand, the function F_s is a rapidly varying one, although continuous, and ranges between 0 and 120 with a mean value of 29.25; therefore it is only to be expected that an excessive overlap of the α vectors will prove to be self destructive of the stored information. The poor convergence (and apparent oscillation) shown in Figure 12 is indicative of this type of difficulty.

This test is intended to show the opposite effect of that demonstrated in the previous test. There the problem arose from the Hamming weight being excessive due to the threshold being set too low; in this experiment we get a very small Hamming weight by setting the threshold too high. The defining parameters for this case are

$$p = 0.2$$
$$\delta = 4$$

which gives a mean Hamming weight of

$$\overline{H}_{\alpha} = 15.36$$

The convergence plot for this test is shown in Figure 13. The convergence here is roughly comparable to that found in the previous test but for different reasons. In this case the α vector is so sparsely filled that adjacent X points map into nearly the same α vector and consequently produce the same type of destructive interference seen before, but for different reasons.

We will state without proof or further justification that the optimum parameters are (in the sense of minimizing both types of destructive interference in the memory fill operation)

$$0.2 < p < 0.3$$

and a δ chosen such that

$$0.02 < P_{\delta} < 0.05.$$

With small problems, either few variables or small sets of discrete values, these conditions may be difficult to meet; however both criteria are satisfied in the present experiment by choosing as essential parameters

$$p = 0.2$$
$$\delta = 3$$

which gives a P_{δ} of

$$P_{\delta} = 0.0272.$$

The balance of this experiment will be concerned with a test drawn under these conditions.

The following test is actually the crux of this whole series of experiments since it demonstrates the actual operation of the associative memory. This test is concerned with the memory's performance on the F_s function using the parameters given in the preceding paragraph. The convergence plot for this particular test has already been given in Figure 11; however the detailed behavior of the memory is lost in a smoothed measure of convergence such as E. Since the primary function of the associative memory is the recall (or estimation) of functional values it is highly desirable to present the memory output in a form where its performance can be conveniently judged. The following three figures are a demonstration of the actual performance of the memory.

A random number generator was used to assign random values (0 - 9) to X_1, X_2, X_3, and X_4 and the corresponding F_s was calculated using these random variables. The opaque backing sheets show these actual functional values beginning at the 500-th, 1000-th and 4000-th data points. The clear plastic overlay shows in each case the predicted values generated by the associative memory when queried with the α vector corresponding to the argument of F_s. Even without reference to the convergence plot of Figure 11, it is readily apparent in comparing Figures 14, 15, and 16 that the memory has converged to the random functional in a very striking manner. In this test the memory was still being updated, i.e., each new data point was being entered at the same time as the test was being conducted.

Two questions which occur naturally when one considers an associative memory are;

1. How will it perform in a new field of data points, if forced to estimate without new data being entered?
2. What is the span of its memory; that is how well will it do if shown the same set of data points with which the memory was originally filled, compared to a new field of data points?

Both of these questions have been investigated. The memory was "filled" by drawing 3000 data points at random as described before, and then by setting the relaxation coefficient to zero, $k = 0$, further changes in the Q vector were prevented. First a sequence of random points, presumably new ones, were used to query the memory. The results of this test are shown in Figure 17, where again the computed actual functional values are shown on the opaque backing sheet and the associative memory outputs are shown on the clear overlay. $R = 0.10444$ which indicates that the terminal performance of the associative memory while in the learn cycle, $R = 0.10248$, is probably a measure of its storage accuracy over the entire functional space. The second question actually answers this by showing that the detailed memory of the actual data points entered in the memory has been sacrificed for a type of distributed functional memory for the entire functional space. Figure 18 shows this phenomena in a very striking fashion. The points immediately preceding the end of the storage period are all queried, with the unexpected result that the memory has a detailed retentivity period of perhaps 500 points, or half a cycle, and that beyond this the individual data points have been lost individually and have been assimilated instead in a collective form as a "smoothed" function. Figure 18 again shows the actual functional values on the opaque backing sheet and the output of the associative memory on the clear plastic overlay.

In conclusion then, the series of tests which made up experiment No. 10 has demonstrated the behavior of the associative memory on an actual nontrivial problem. The data could just as well have been drawn from any number of real life problem situations; however it seemed appropriate to discuss a well-behaved analytic function in this paper to permit the reader to readily judge the performance of the memory.

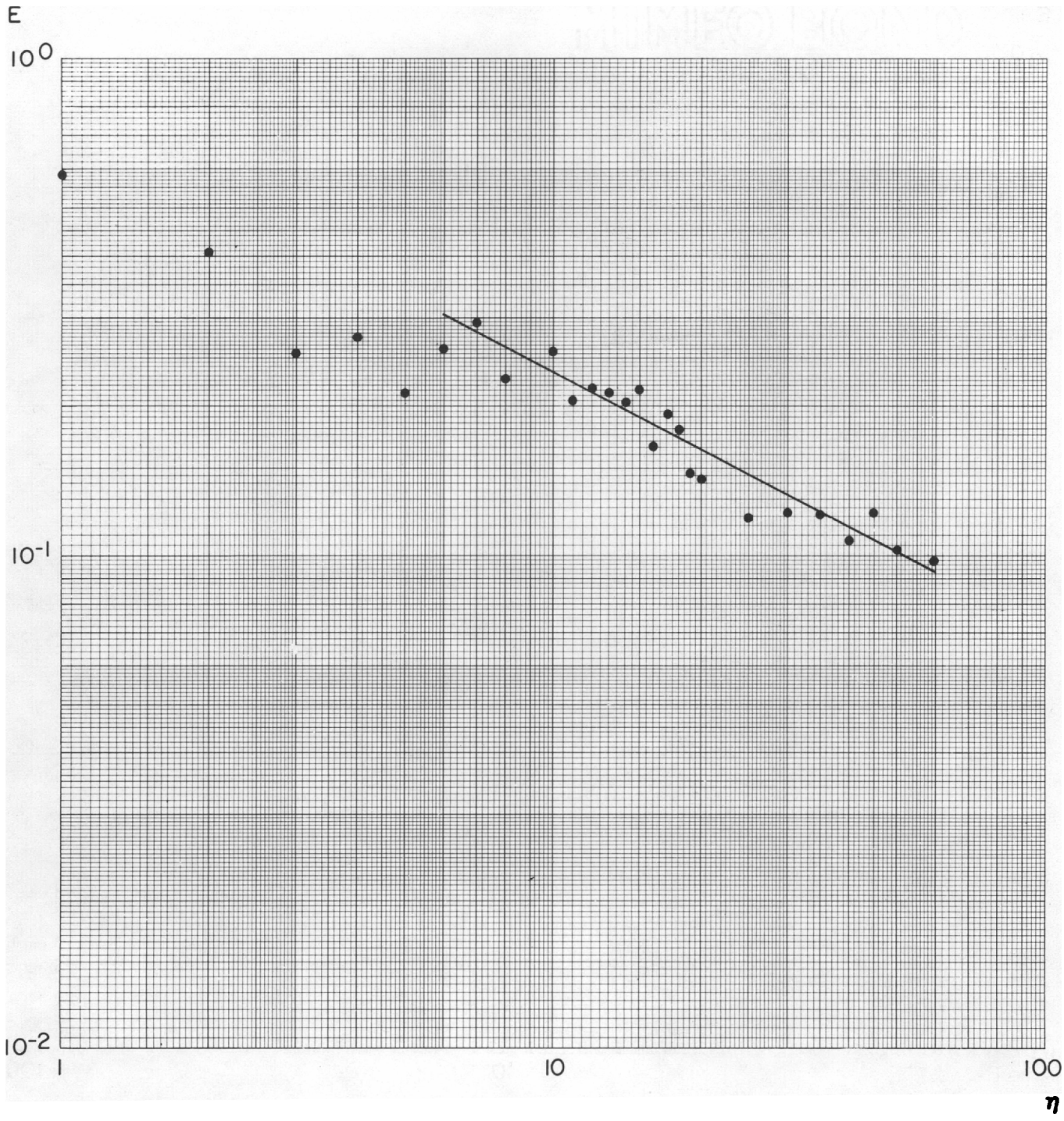


Figure 11. Convergence for F_s with $p = 0.2$; $\delta = 3$

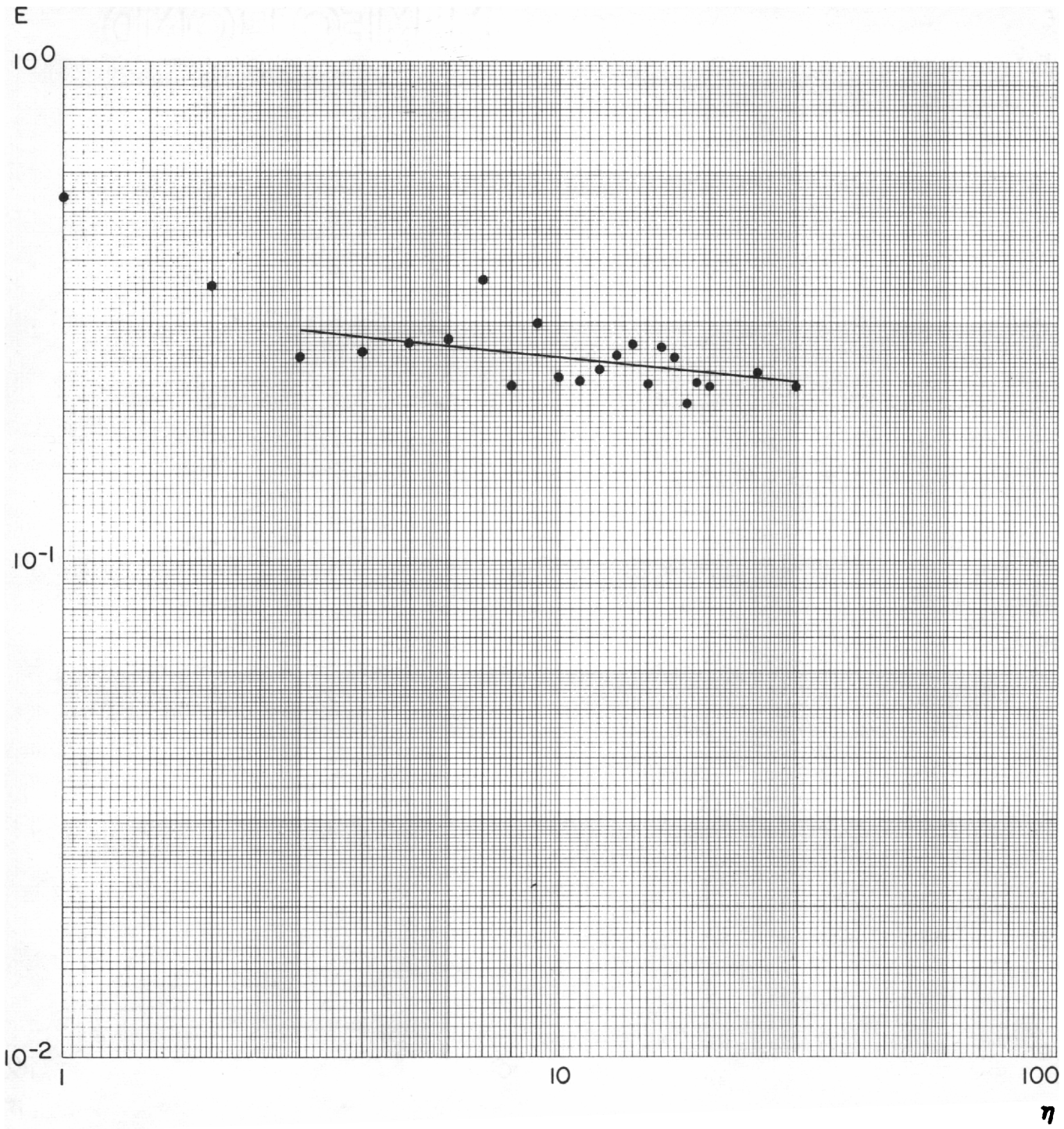


Figure 12. Convergence for F_S with p = 0.2; δ = 2

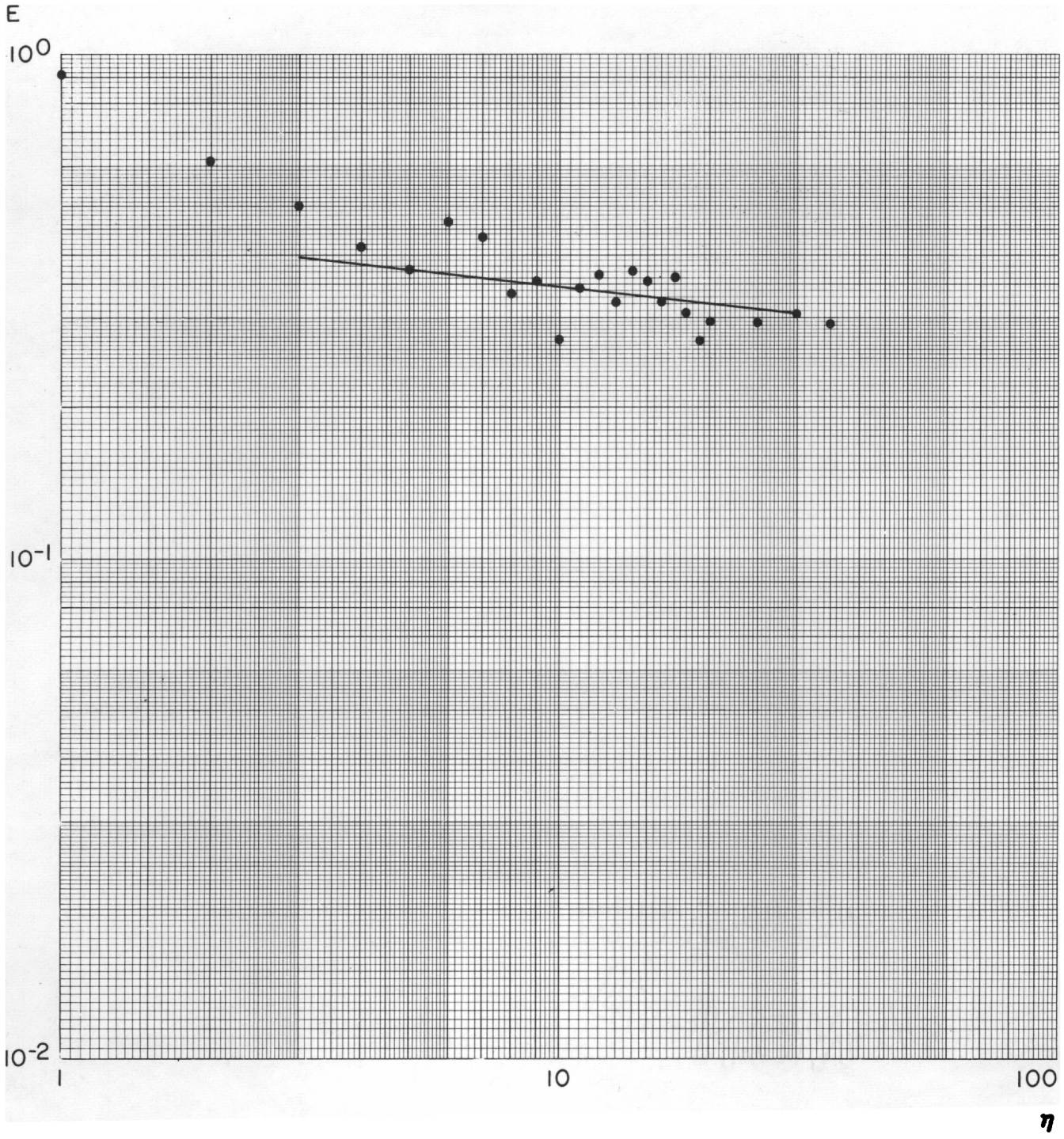


Figure 13. Convergence for F_S with p = 0.2; δ = 4

09-100

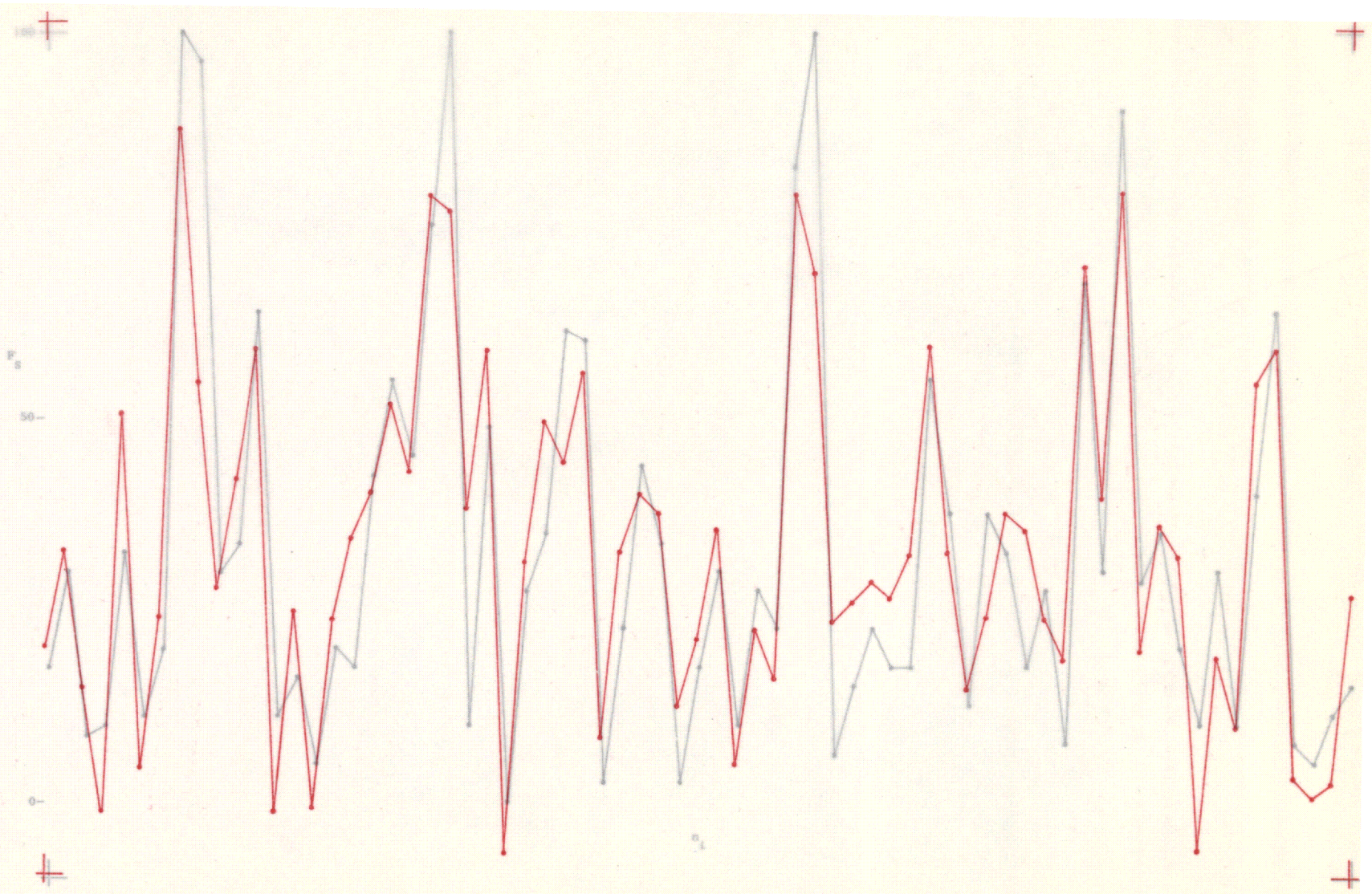


Figure 14. Predicted versus actual values of F_S, starting at (i) = 500

103-104

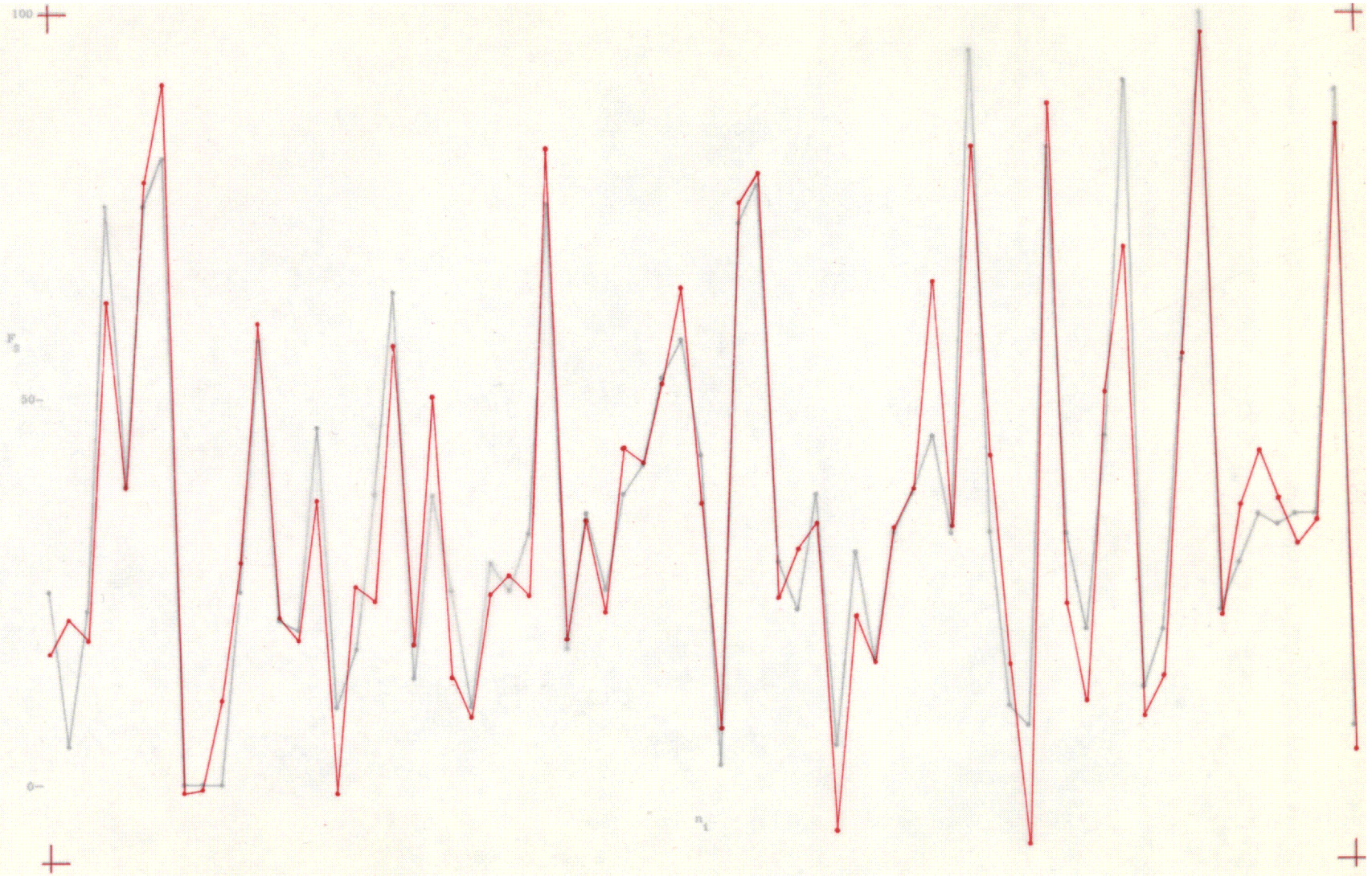
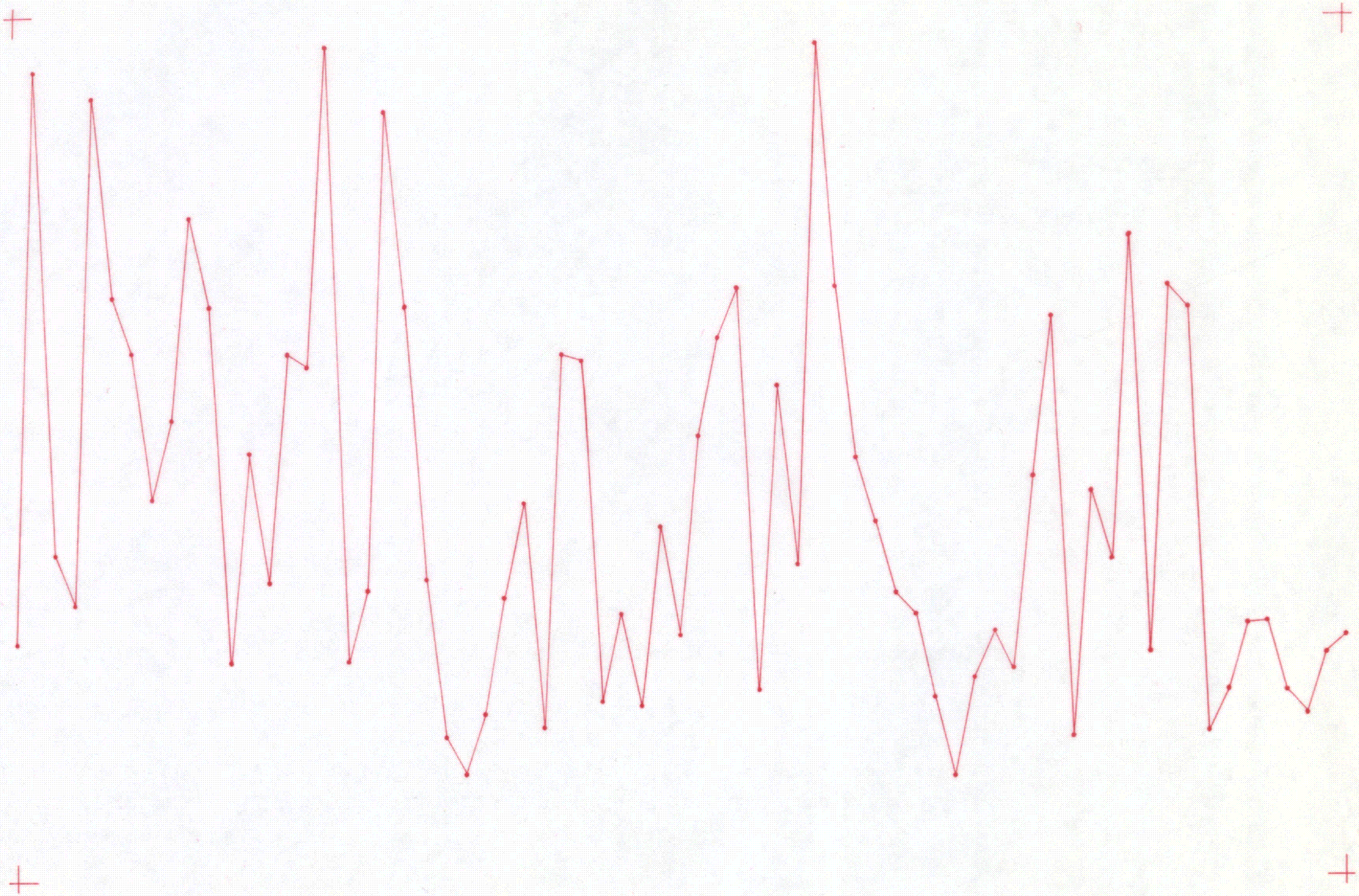


Figure 15. Predicted versus actual values of F_S, starting at (i) = 1800



107-108

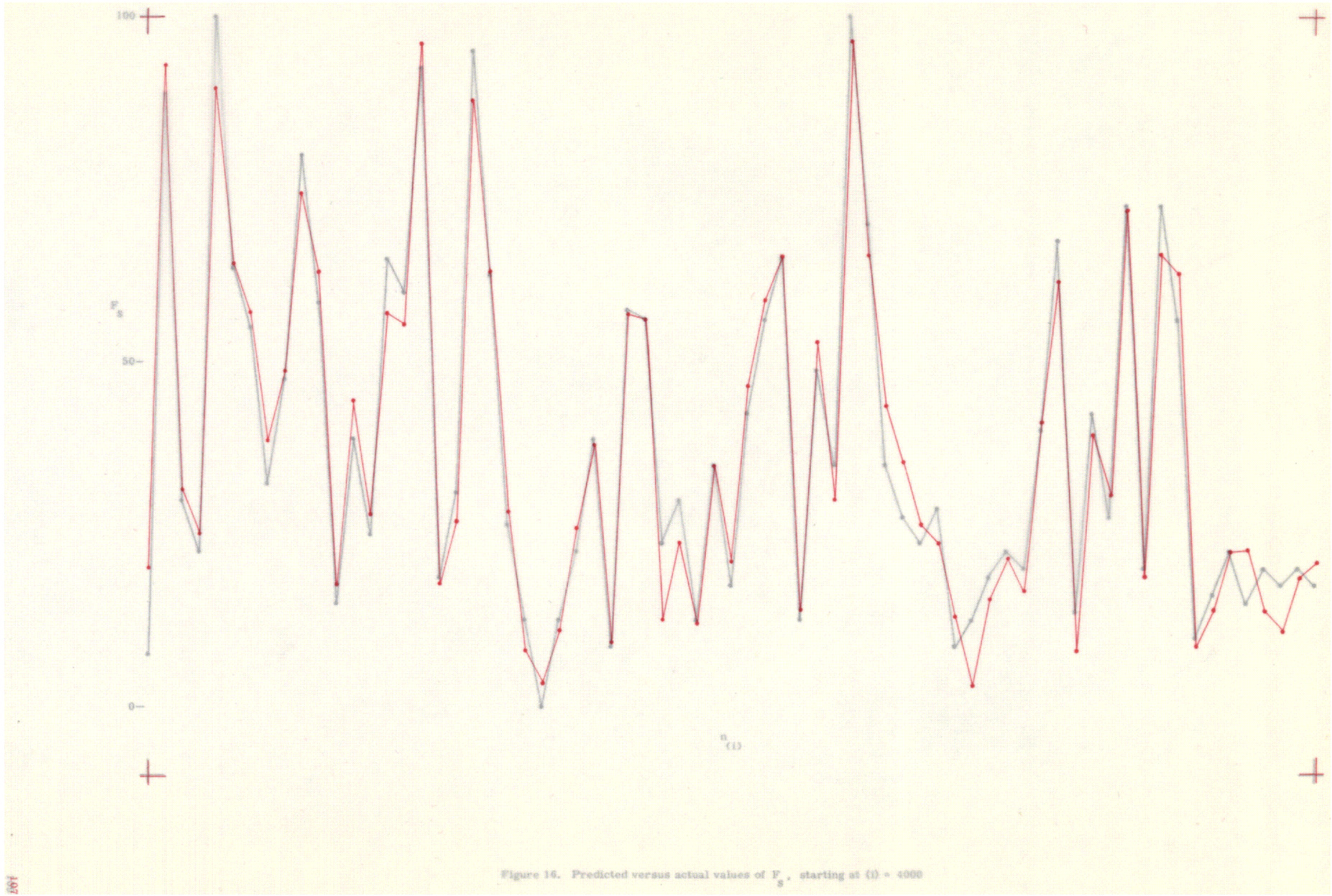


Figure 16. Predicted versus actual values of F_s, starting at (i) = 4000

111-112

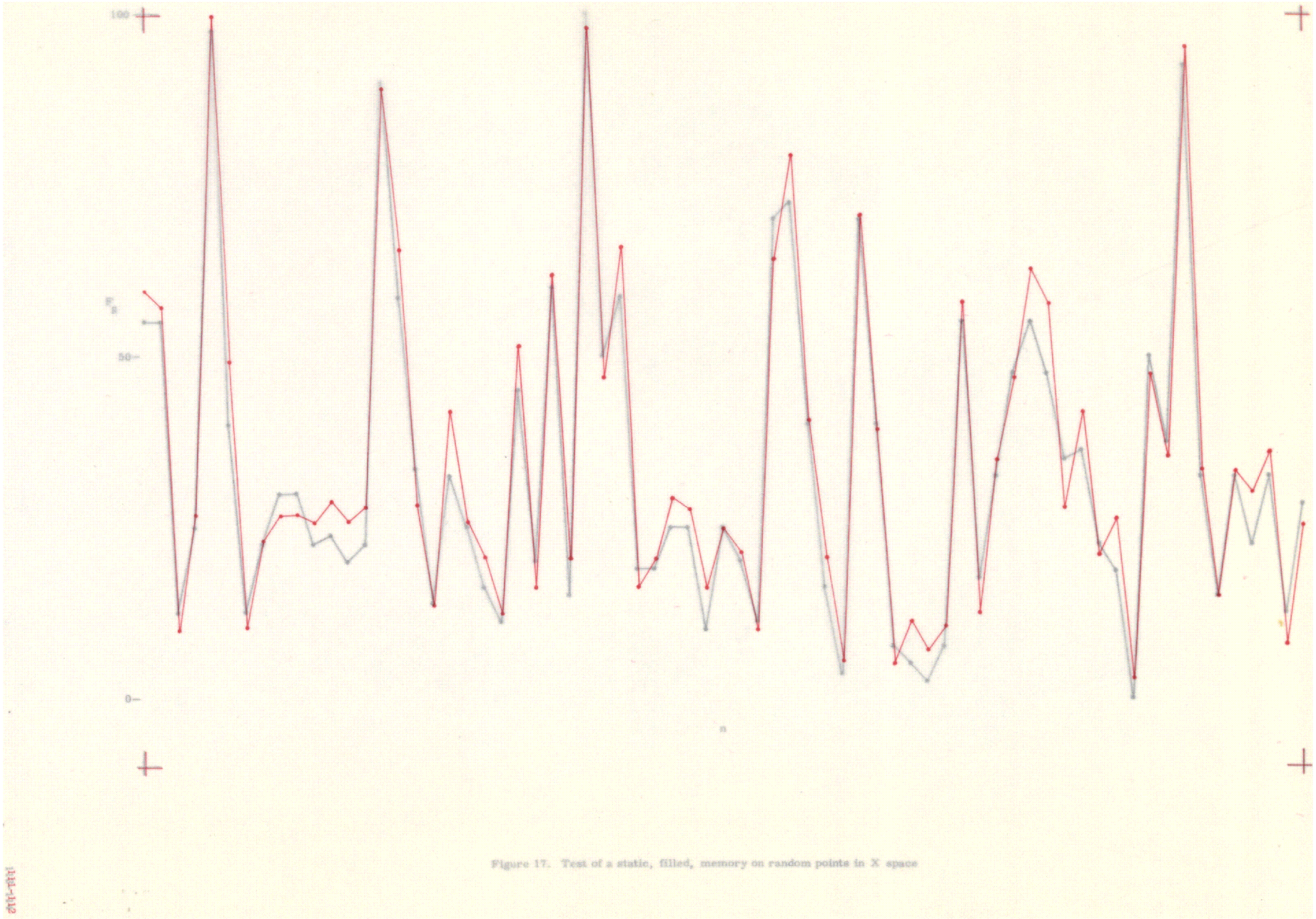
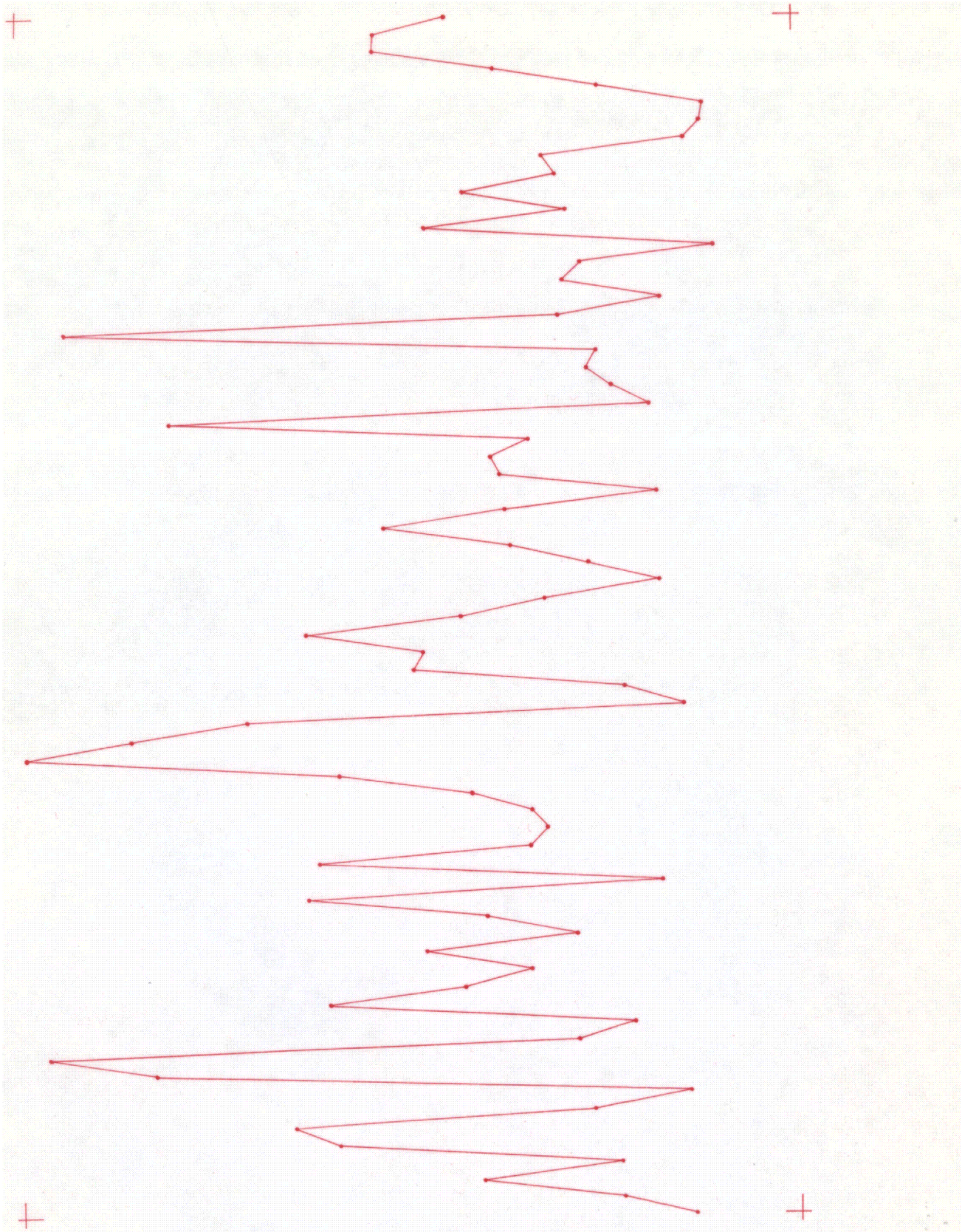


Figure 17. Test of a static, filled, memory on random points in X space



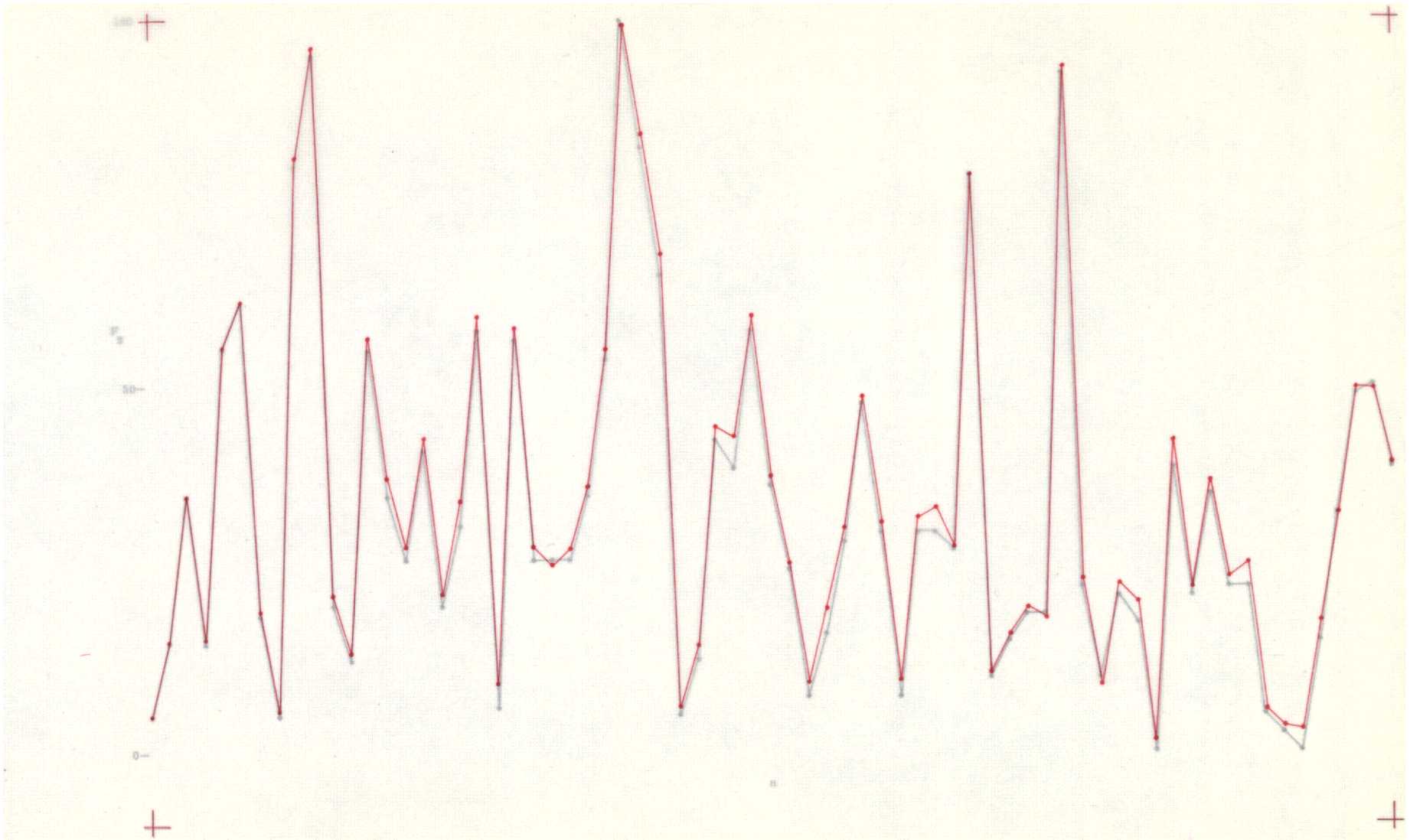


Figure 18. Test of the survival of detailed memory entries (memory span)

TID-4500(19th Edition)
MATHEMATICS AND COMPUTERS

No. of copies	Distribution
614	UC-32

Issued by
Technical Information Division
Sandia Corporation
Albuquerque, New Mexico