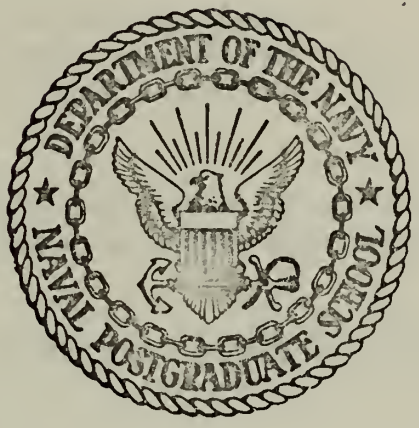
A COMPUTER-GRAPHICS SEPARATION ALGORITHM FOR PATTERN CLASSIFICATION AND CLUSTER ANALYSIS

Gilbert Paul Lauzon

NAVAL POSTGRADUATE SCHOOL
Monterey, California



THESIS

A COMPUTER-GRAPHICS SEPARATION ALGORITHM FOR PATTERN CLASSIFICATION AND CLUSTER ANALYSIS

by

Gilbert Paul Lauzon

Thesis Advisor: B. O. Shubert

September 1973

Approved for public release; distribution unlimited.

A Computer-Graphics Separation Algorithm for Pattern Classification and Cluster Analysis

by

Gilbert Paul Lauzon
Lieutenant, United States Navy
B.S., Tufts University, 1967

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
September 1973

ABSTRACT

A separation algorithm applicable to the pattern classification and cluster analysis of n-dimensional ($n > 2$) data is presented. The algorithm reduces the dimensionality of the problem by projecting each point into a plane. This plane is presented to the user on a computer graphics console screen. The operator picks a point on the screen with a lightpen and chooses a "direction of movement" to achieve or increase separation, thereby causing an iteration of the algorithm. Each iteration is in fact a reorientation of the plane into which the data points are projected. Iterations continue until satisfactory separation is achieved. The algorithm is not restricted by the dimensionality of the data, nor are any distributional assumptions required. Results from six case studies indicate that the algorithm is a useful tool for the analysis of multi-dimensional data.

TABLE OF CONTENTS

I. INTRODUCTION

The separation algorithm presented in this paper is a development of an idea of Professor Thomas M. Cover of Stanford University. It provides a tool for classifying multidimensional data sets. Two classification methods can be handled by the program which implements the algorithm; these two methods are pattern classification and cluster analysis. Meisel [4] discusses and contrasts these two processes. He describes pattern classification as the "process of developing on the basis of a finite set of labelled samples, a decision rule with which we can classify a point in the pattern space corresponding to an unlabelled sample." Cluster analysis on the other hand is described as a process for "describing and locating the discernible subsets of which a set of sample points of reasonable complexity is usually formed."

Meisel notes that the difference between pattern classification and cluster analysis is that the samples for the former are labelled, whereas in the latter they are unlabelled. Thus it appears that the question of whether a particular problem should be approached as a pattern classification or cluster analysis problem depends on the use to be made of the results and on the nature of the data available. In the pattern classification problem the initial data set is divided into two (or more) subsets which are identified. Each member of the data set is labelled as being included in exactly one of these subsets. The decision rule developed

typically takes the form of a function which has an argument in the form of a data point. When input with the values of elements of different subsets, the function assumes values in disjoint ranges. For example if the elements of a two subset problem are identified as X and Y respectively, and the discriminant function is f, possibly f(X) would be negative for all X; and f(Y) would be positive for all Y. The discriminant function is then used to classify unlabelled points Z as being in either the X or the Y subset.

In a cluster analysis problem, the initial data set has no previously defined subdivisions. The purpose of the analysis is to decompose the given set into a group of subsets. The number of subsets to be found can be specified in advance or unknown. Utilizing similarities among the elements in the original data set, assignment to the subsets is made.

Both pattern classification and cluster analysis techniques can be described as being direct or indirect. In an indirect process, some criterion function is used in the construction of the discriminant function or is used to define the quality of clustering desired. A direct process has no such criterion function; the formulation of the discriminant function (or clusters) is accomplished by making immediate use of the data points. The algorithm developed in this paper is direct in nature.

It may be that for a particular problem, both pattern classification and cluster analysis techniques are appropriate. For example in a pattern classification problem with two subsets forming the data set, the subsets might be inspected separately using cluster analysis. Alternatively

after successful usage of clustering techniques, a pattern classification algorithm might be employed to define discriminant functions for the clusters.

The separation algorithm presented in this thesis is implemented by a computer program which utilizes the graphics edit mode of operation provided by the XDS9300 digital computer combined with the Adage AGT-10 graphics console. The man-machine interface is emphasized. The basic concept of the algorithm becomes apparent when a comparison is made with the concept of Fisher's Linear Discriminant. Duda and Hart [3] present Fisher's Linear Discriminant as a method for reducing the dimensionality of a pattern recognition problem. By projecting the points of the n-dimensional pattern space onto a straight line in the pattern space, the dimensionality of the problem is reduced from n to one. They further point out that the projections of the points may fall on the line in a very confused manner. By changing the position of the line in the n-space, an orientation is sought in which the projections of the data points will display separation. The achievement of this orientation, if possible, is the goal of the discriminant analysis.

By way of comparision, the algorithm presented here also represents a dimension reduction approach; however, the projection of the data points is made onto a plane vice a line. The dimension reduction is then from n to two. If n is less than or equal to two, this algorithm is not appropriate; more straightforward methods employing scatter plots, etc., can be employed. As in the Fisher method, the algorithm proceeds as a

series of reorientations of this plane in the n-space. Following each new orientation the program presents the projections on the screen of the graphics console. In a pattern recognition problem with two subsets, each point's projection is represented by a video "Y" or "X". In a cluster analysis each projection is presented as a "Y". With the XDS9300/AGT-10 combination in the graphics edit mode, the user decides whether or not to make an iteration, and thus reorient the plane. The decisions if and how to iterate are based on the situation presented on the console screen. The importance of the man-machine interface should be clear. The iterative process will be described in greater detail later. Note that the decision of how to iterate and when to terminate rests with the user. He determines when adequate separation has been attained.

II. A PREVIOUS COMPUTER-GRAPHICS APPLICATION

This discussion of a report by Herman Chernoff entitled "THE USE OF FACES TO REPRESENT POINTS IN n-DIMENSIONAL SPACE GRAPHICALLY" 1 is presented as an example of the type of man-machine graphics approach to multidimensional data analysis previously proposed. Chernoff uses each n-dimensional vector corresponding to a data point to mathematically determine the shape and features of a human face which is output by a CALCOMP plotter. Chernoff's hypothesis is that points belonging to the same class or cluster will be identified by the user due to the similarity of the faces they generate. The user constructs the clusters by grouping the faces into sets which show common features. As an illustrative example, Chernoff applies his method to a data set of eighty eight-dimensional measurements of nummulitid specimens from the Eocene Yellow Limestone Formation of Northwestern Jamaica. (This data set will also be used later as an illustrative example for this paper.) Chernoff's results indicated the existence of three clusters in the data. These results correspond exactly to the results obtained by Wright and Switzer [6] who employed a different cluster analysis scheme to the same data set.

In constructing faces Chernoff can handle up to eighteen dimensional data to determine such features as the shape and spacing of the eyes, size and curvature of the mouth, and the shape of the head. Clustering obtained by Chernoff's method may depend on which components

of the data vectors control the various facial features; it is not clear how this assignment should be made before the problem is started. Chernoff dismisses any criticism of the eighteen dimension limitation, saying that any further increase in dimensionality could be overcome by adding features to the faces. This is an interesting point; it would seem that if too many features were added to the faces, the decision process would be more complicated. Chernoff notes that "when the eyes are very small, the pupils become hard to detect." Also he notes that some information is lost when the two ellipses which form the head are not differentiable and the face is circular. Certainly, within any moderate range of dimensional values, and as noted by Chernoff, this method "provides a promising approach for a first look at multivariate data which is effective in revealing rather complex relations not always visible from simple correlations based on two-dimensional linear theories."

III. DISCUSSION OF THE ALGORITHM

As indicated in Section I the separation algorithm developed in this paper is applicable to both the pattern classification and the cluster analysis problems. The concept of operation of the algorithm in each of these applications is the subject of the following paragraphs.

For the pattern classification problem the analysis is conducted on an n-dimensional data set which is divided into two subsets. These subsets might represent two classes of objects, P_1 and P_2. Each object would be characterized by an n-vector (or pattern) of measurements. Each element of the pattern is referred to in the literature as a feature. The goal of the pattern classification analysis is to determine a function (called the discriminant function) $f: R^n \rightarrow R^1$, where R^n is real n-space, which can be used to assign an unlabelled pattern Z to either P_1 or P_2. A scheme for assigning Z might be:

$$f(Z) < 0 \Rightarrow Z \in P_1$$

$$f(Z) > 0 \Rightarrow Z \in P_2$$

If a discriminant function of reasonably low complexity can be determined for the two classes, the classes are separable; if no such function is found, the classes are inseparable. Since the discriminant function is constructed using samples from each of the classes, it must be determined that the separation between the classes is sufficient to warrant the use

of the discriminant function in making class assignments. For example, for two classes to be linearly separable, the convex hulls of the two sets of projections should be disjoint.

For purposes of illustration and discussion, suppose the initial data set has N members in P_1 and M members in P_2. The P_1 subset is read by the computer as Y vectors; thus the program reads N vectors Y_i, $i = 1,N$. Similarly the P_2 subset is read as a set of X vectors, X_j, $j = 1,M$. The dimension n of these vectors should be greater than two. (If n is two or less the problem can be handled graphically.) The algorithm proceeds to the dimension reduction task as follows. The user chooses two non-zero vectors A and B in Euclidean n-space. A linear transformation is established as $T_{A,B}(Z) = (A \cdot Z, B \cdot Z)$ where $A \cdot Z = \sum_{i=1}^{N} a_i z_i$ is the inner product and $Z \in R^n$. This transformation which is a projection of the point Z into the plane defined by the vectors A and B is the algorithm's vehicle for dimension reduction. The projection is not generally orthogonal. The transformation is applied to each member of P_1 and P_2; the two sets of coordinate pairs are then scaled. In order to be presented on the screen of an AGT console, a coordinate pair (x,y) must have values $-1 \leq x,y \leq 1$; it is to obtain pairs within this range that the scaling is done. The scaling is accomplished by searching through both sets of projection pairs to find the maximum absolute value of the first coordinates (called TMAX) and the maximum absolute value of the second coordinates (called SMAX). Then the typical $(A \cdot X, B \cdot X)$ becomes $(A \cdot X/TMAX, B \cdot X/SMAX)$, and the typical $(A \cdot Y, B \cdot Y)$ becomes $(A \cdot Y/TMAX, B \cdot Y/SMAX)$. Thus the

range of values for all pairs of the problem is as required by the AGT console. The resulting A-B plane is graphically presented to the user on the AGT screen. It is the user's decision based on information gained from the screen to decide how to proceed. In all but the most extraordinary case the initial picture will show some intermingling of the X's and Y's. The heuristic argument which follows provides a reasonable guide for proceeding.

The user is seeking a presentation which depicts the X's and Y's in disjoint concentrations. Frequently the initial view of the A-B plane will yield an indication of where these concentrations will develop. If this is not the case, and provided there is a solution, such an indication should appear after a sequence of "random" iterations. The user then picks the X or Y which is deepest in the opposing concentration and "moves" it toward its proper concentration. The user implements these decisions with the box of sixteen function switches and the lightpen attached to the console. The positions of the function switches and the lightpen relative to the graphic console are indicated in Figure III-1. The user makes his choice by depressing the function switch marked "PICK X" or "PICK Y" and placing the tip of the lightpen over the X or Y. Figure III-2 shows a user at the console ready to pick a point with the lightpen. The point seen by the lightpen partially vanishes; that is, a Y might become a $\vee$ and an X, a Y. When part of the desired point disappears, the user completes the choice by depressing the switches marked "COMPLETE CHOICE" and "END EDIT". Ready to move, the

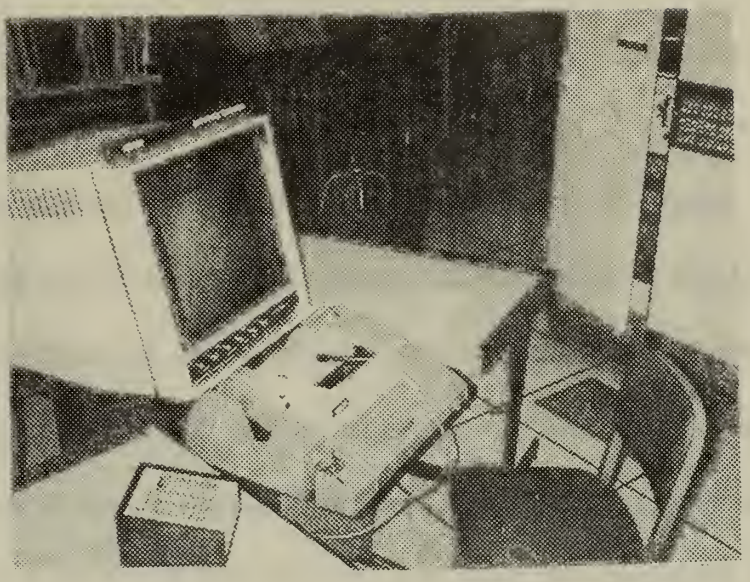


Figure III-1. AGT CONSOLE

Figure III-2. OPERATOR SITTING AT AGT CONSOLE

user depresses one of four switches (LEFT, RIGHT, UP, or DOWN) to indicate the desired direction of motion. Figures III-3 and III-4 give an indication of what would occur on two iterations. In Figure III-3 the user picks the Y indicated and iterates with a move to the right. Then in Figure III-4 he picks an X and iterates with a move to the left. The iteration step consists of the modification of the A and B vectors, the recomputation and rescaling of the projections, and finally the representation of these projections on the AGT screen. The modified values of the A and B vectors depend on the point picked and the direction of motion chosen. If the point picked is the projection of data point Z, the new A and B vectors will be as follows:

Direction of Move	NEW A	NEW B
LEFT	A-Z	B
RIGHT	A+Z	B
UP	A	B+Z
DOWN	A	B-Z

The new A and B vectors are used in the recomputation of the T transformations.

Having achieved the desired separation, the user causes the line printer (XDS9300 output device) to print a six by six inch reproduction of the projections in the A-B plane. Using this printout he may fit a suitable discriminant function between the two concentrations. In Figure III-5 an output with the discriminant function is represented. The discriminant function f(Z) in this case is a straight line, and its equation

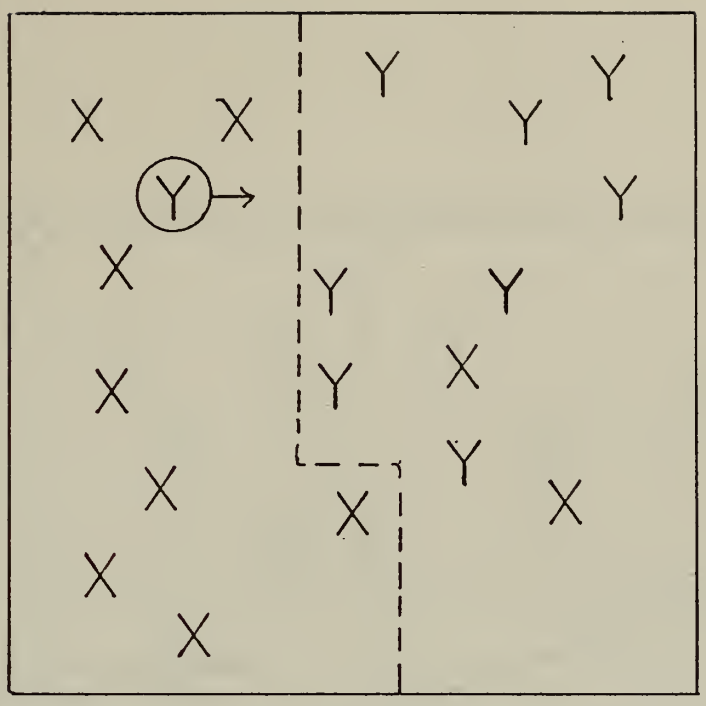


Figure III-3. Iteration by picking a Y and moving it right

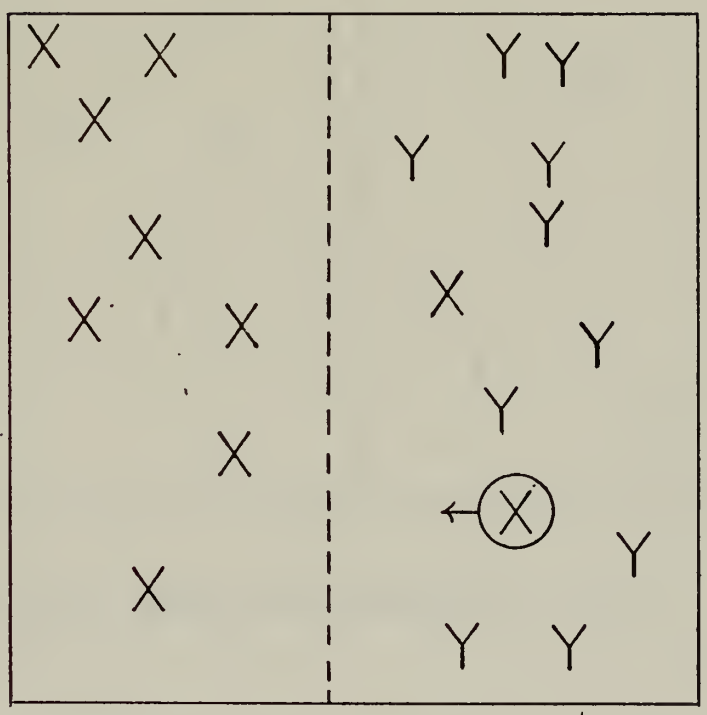


Figure III-4. Iteration by picking an X and moving it left

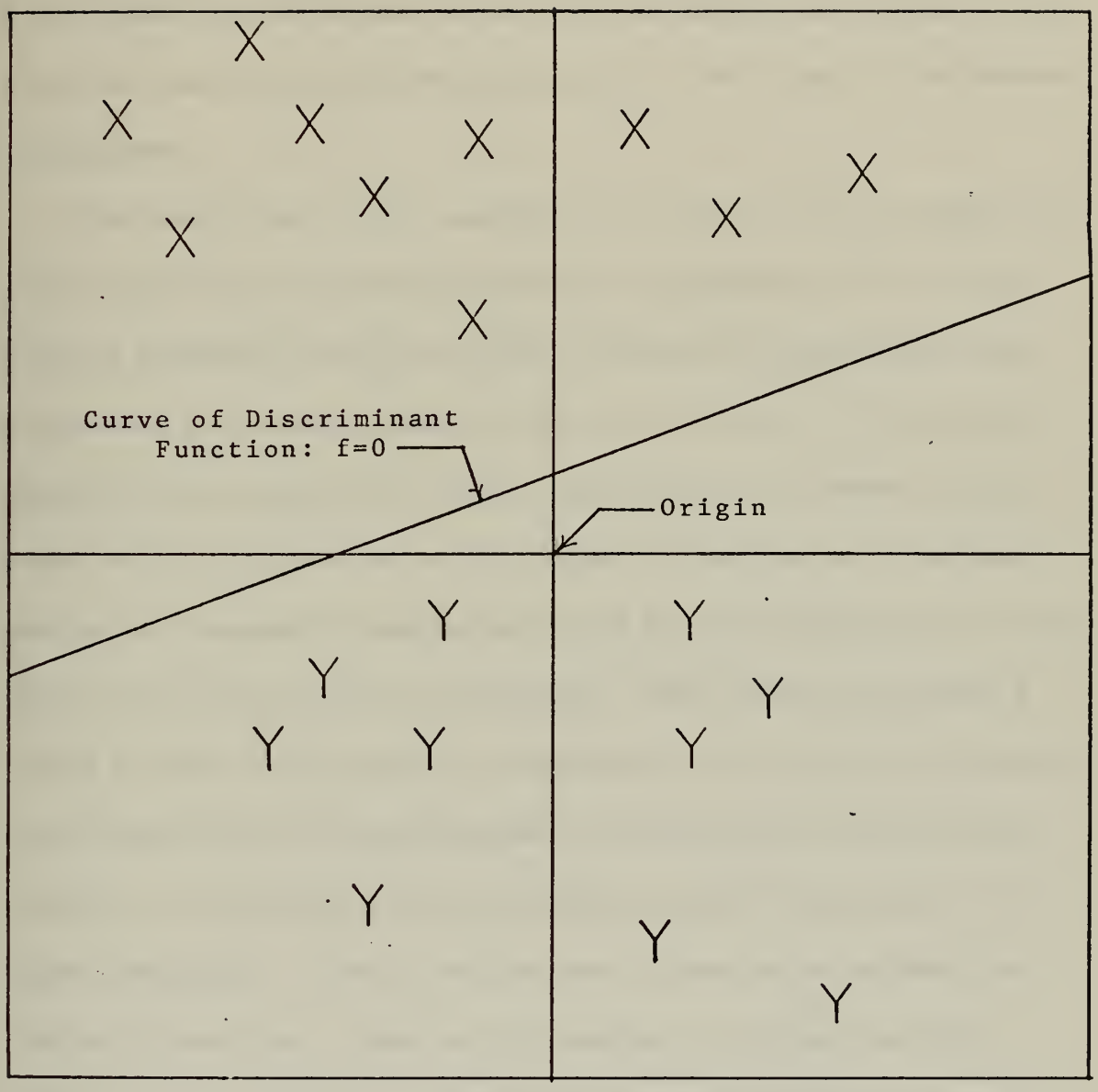


Figure III-5. Output with discriminant function

would be:

$$f(Z) = \alpha\,(A \cdot Z/TMAX) + \beta\,(B \cdot Z/SMAX) + \gamma$$

where α, β, and γ are scalars. The vectors A and B and the scaling factors TMAX and SMAX have values corresponding to the printout. With the specification of the discriminant function, the pattern classification is completed.

The employment of the separation algorithm for cluster analysis is very similar to that described for pattern classification. For a cluster analysis problem the initial set of data points is not broken down into subclasses; the program reads all patterns as Y vectors. The iterative operation of the cluster separation is the same as the pattern classification except that there are no X's present. When the user considers that he has displayed a satisfactory set of clusters, he can have the line printer output the six by six inch picture. The output also provides a listing by point of the projection coordinates. Thus the cluster membership of each point can be determined. As in the pattern classification problem, no criterion function is provided to direct the procedure or to signal termination. It rests with the user to make value judgments to conduct the analysis. There is no guarantee of a unique clustering scheme for a given data set; comparisons between different schemes should provide a great deal of information.

As a final comment on the verbal description of the two types of operations just presented, it should be noted that in reality the program does not "move" any points. Instead, it is reorienting the plane into

which the points are projected by modifying the A and B vectors. Separation is achieved when the A-B plane is properly oriented.

Mathematically the progress of the algorithm can be traced through a series of steps as shown below. Suppose the initial data set consists of k-dimensional vectors Y_i, i=1,N and X_j, j=1,M. In the cluster analysis problem M=0. With the non-zero k-vectors A and B specified, the algorithm proceeds:

1) Center the data sets by calculating a k-vector $C=(1/M+N)(\sum_{i=1}^{N} Y_i + \sum_{j=1}^{M} X_j)$, and use C to center the X's and the Y's as follows:

$$X'_j = X_j - C, \quad j=1,M$$

$$Y'_i = Y'_i - C, \quad i=1,N$$

2) Calculate the projection $T_{A,B}$ for all X and Y:

$$T_{A,B}(X'_j) = (A \cdot X'_j, B \cdot X'_j) \quad j=1,M$$

$$T_{A,B}(Y'_i) = (A \cdot Y'_i, B \cdot Y'_i) \quad i=1,N$$

3) Calculate the scaling parameters TMAX and SMAX

$$TMAX = \max(\max_j |A \cdot X'_j| , \max_i |A \cdot Y'_i|) \qquad \begin{matrix} i=1,N \\ j=1,M \end{matrix}$$

$$SMAX = \max(\max_j |A \cdot X'_j| , \max_i |B \cdot Y'_i|)$$

4) Scale the $T_{A,B}$ projections

$$T_{A,B}(X'_j) = (A \cdot X'_j/TMAX, B \cdot X'_j/SMAX) \quad j=1,M$$

$$T_{A,B}(Y'_i) = (A \cdot Y'_i/TMAX, B \cdot Y'_i/SMAX) \quad i=1,N$$

5) At this point M video X's and N Y's appear on the AGT screen, if a Y is picked and a right move is selected, a new A=A+Y is calculated. Thus at this point, either a new A vector or a new B vector is found and control shifts to step 2.

The five steps listed above describe the activity in the program at each iteration.

IV. MOTIVATION OF THE ALGORITHM

The attraction of the approach of this algorithm for separating data into its classes has a dual basis. Study of the iterative operation of the algorithm reveals as the first part of this basis the manner in which the correlation within data classes is exploited in seeking separation. The second grounds for confidence is the similarity of the algorithm's method to that proposed in the Perceptron Convergence Theorem. This similarity will be utilized to present a convergence proof for the algorithm. Each of these supporting concepts will be discussed in the following paragraphs.

If the data set to be analyzed via the algorithm is separable, the set can be considered to be made up of a number of subsets. For illustrative purposes, suppose the data set consists of two subsets. The assumption is made that there is a higher correlative measure within these subsets than between them; it is this difference which has value in the attainment of separation. If the two subsets are labelled X and Y, then without considering the scaling operation, it is clear from the discussion of the last section that the algorithm maps these sets onto a plane as

$$(A \cdot X_j, B \cdot X_j) \quad \text{for } X_j \in X$$

$$(A \cdot Y_i, B \cdot Y_i) \quad \text{for } Y_i \in Y$$

Now suppose an iteration is made by picking an X and moving to the right. It has been seen that the new A is in fact $A+X_k$. The new mappings are

$$((A+X_k)\cdot X_j, B\cdot X_j) = (A\cdot X_j + X_k\cdot X_j, B\cdot X_j), \text{ for } X_j \in X$$

$$((A+X_k)\cdot Y_i, B\cdot Y_i) = (A\cdot Y_i + X_k\cdot Y_i, B\cdot Y_i), \text{ for } Y_i \in Y$$

The difference in the mappings (i.e., the separation gained) is seen to be in the first coordinates as the difference between $X_k \cdot X_j$ and $X_k \cdot Y_i$. Standard correlation concepts indicate that if the X set is highly correlated, while the correlation between the X and Y sets is low, then the product $X_k \cdot X_j$ will be greater than $X_k \cdot Y_i$. Clearly if some Y_m is picked the difference in the mappings caused by the iteration will be the difference between $Y_m \cdot Y_i$ and $Y_m \cdot X_j$. In general the degree of separation gained on a single iteration will tend to be small; however, it is the goal of the algorithm to combine the incremental gains in separation of a sequence of iterations to obtain the desired distinction between classes.

To perceive the similarity of the algorithm to the method of the Perceptron Convergence Theorem, one has only to consider the following version of this work as presented by Minsky and Papert [5]:

"The Perceptron Convergence Theorem"

Consider the following program which the vector notation $A\cdot\Phi$ is $\sum a_\phi \phi(x)$ (in place of our usual notation)

START Choose any value for A

TEST Choose an X from $F^+ \cup F^-$

If $X \in F^+$ and $A\cdot\Phi > 0$ go to TEST

If $X \in F^+$ and $A\cdot\Phi \leq 0$ go to ADD

If $X \in F^-$ and $A\cdot\Phi < 0$ go to TEST

If $X \in F^-$ and $A\cdot\Phi \geq 0$ go to SUBTRACT

ADD Replace A by A + Φ (X)

Go to TEST

SUBTRACT Replace A by A - Φ (X)

Go to TEST

We assume until further notice that there exists a vector A* with the property that if $X \in F^+$ then $A^* \cdot \Phi(X) > 0$ and if $X \in F^-$ then $A^* \cdot \Phi(X) < 0$. The perceptron convergence theorem then states that whatever choice function is used in TEST, the vector will be changed only a finite number of times."

The variable A of the above theorem corresponds to the A of the present algorithm; there is no B vector in the Perceptron theorem approach. The Perceptron method can be considered to be a more direct method than that presented for the algorithm here since it specifies that only misclassified points will be operated on. Naturally this approach is allowable but not required in the present work. If it is assumed that, in employing the algorithm, the user conforms to the modus operandi of the Perceptron Convergence Theorem, it is possible to prove that if the data is linearly separable the algorithm will attain separation within a finite number of steps. The following proof is a modification of the method employed by Duda and Hart [3]; it shows that, provided there exists an orientation of the A-B plane in which the X subset is linearly separable from the Y subset, this orientation can be attained with a finite number of iterations. Suppose the situation in the A-B plane is as indicated in Figure IV-1. Let b=B/SMAX and a=A/TMAX. Then

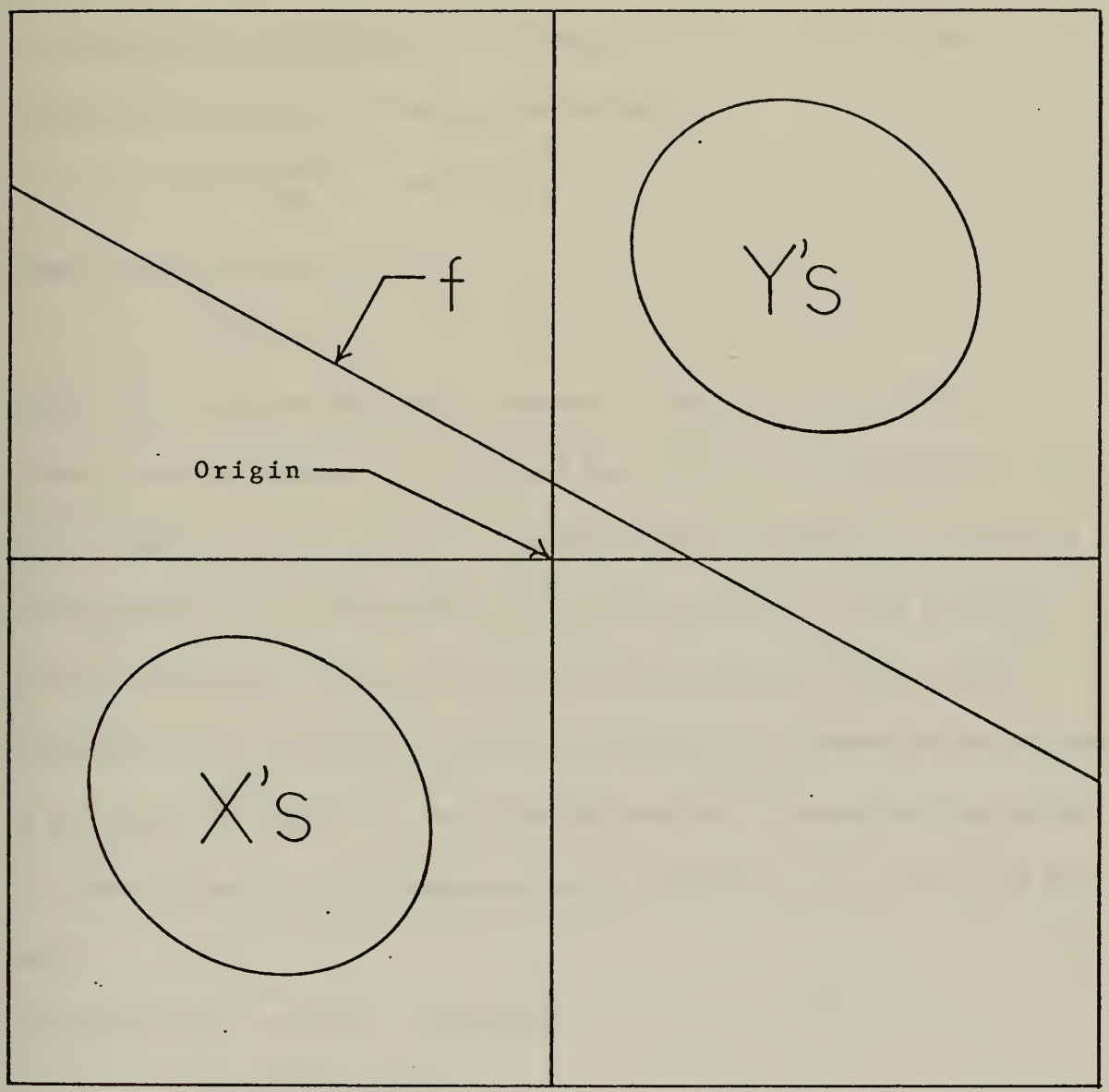


Figure IV-1. Orientation of the A-B plane in which the X and Y sets are linearly separated

$$b \cdot Z - m(a \cdot Z) + w_o = f(Z)$$

where m is the slope of f, w_o is the y intercept, and Z is an X or Y. Then

$$f(Z) = (b-ma) \cdot Z + w_o$$

To simplify this expression the following n+1 dimensional (where n is the dimension of A,B, and Z) vectors are defined:

$$T = \begin{pmatrix} b-ma \\ w_o \end{pmatrix} \quad \text{and} \quad S = \begin{pmatrix} Z \\ 1 \end{pmatrix}$$

Then clearly we can write

$$f'(S) = T \cdot S$$

Then f'(S) is a hyperplane which passes through the origin of the n+1 dimensional "S" space. As Duda and Hart point out, the addition of the $n+1^{st}$ dimension as a one to the n-dimensional Z preserves all distance relationships among the points of the original data set. The S vectors are all in the same n-dimensional subspace of the n+1 space. This translation into n+1 space reduces the problem from examining the vectors a and b and the scalars w_o and m to the problem of inspecting the vector T. From Figure IV-1 it is apparent that $f'\begin{pmatrix} X \\ 1 \end{pmatrix} < 0$ for all X, and $f'\begin{pmatrix} Y \\ 1 \end{pmatrix} > 0$ for all Y.

Now defining a variable U such that:

$$U = \begin{pmatrix} -X \\ 1 \end{pmatrix} \text{ for all X}$$

$$U = \begin{pmatrix} Y \\ 1 \end{pmatrix} \text{ for all Y}$$

the problem is confined to discussing $f'(U) = T \cdot U > 0$ for all U. The behavior of the user is such that if he picks a Y such that $f'\begin{pmatrix} Y \\ 1 \end{pmatrix} < 0$, he causes the program to iterate with T = T+Y. Similarly if he picks X such

that $f'(\overset{X}{1}) > 0$, he iterates with T = T-X. In either case the iteration can be considered as picking a U and iterating with T = T+U. The iterations are conducted only when the function as formed misclassifies the U picked.

Let T_k be the value of the T vector on the k^{th} iteration; U_k is the point picked for the k^{th} iteration; and e is a positive scalar. Also $\hat{T}$ is the "solution" vector T. Then

$$(T_{k+1}-e\hat{T}) = (T_k-e\hat{T})+U_k \qquad (a)$$

$$||T_{k+1}-e\hat{T}||^2 = ||T_k-e\hat{T}||^2 + 2(T_k-e\hat{T})\cdot U_k + ||U_k||^2 \qquad (b)$$

However $T_k \cdot U_k < 0$. Thus

$$||T_{k+1}-e\hat{T}||^2 \leq ||T_k-e\hat{T}||^2 - 2e\hat{T}\cdot U_k + ||U_k||^2 \qquad (c)$$

Since $\hat{T}\cdot U_k$ is positive, the second term on the right of equation (c) will dominate the third if e is sufficiently large. In particular let $r=\max_i U_i$ and $s=\min_i \hat{T}\cdot U_i$ for $i=1,\ldots,m_1+m_2$ (assume there are m_1 X's, and m_2 Y's). Then

$$||T_{k+1}-e\hat{T}||^2 \leq ||T_k-e\hat{T}||^2 - 2es + r^2 \qquad (d)$$

Now if $e=r^2/s$

$$||T_{k+1}-e\hat{T}||^2 \leq ||T_k-e\hat{T}||^2 - r^2 \qquad (e)$$

Thus the squared distance from T_k to $e\hat{T}$ is reduced by at least r^2 at each iteration. After K iterations

$$||T_{k+1}-e\hat{T}||^2 \leq ||T_1-e\hat{T}||^2 - Kr^2 \qquad (f)$$

Since the squared distance can not become negative, it follows that the sequence must terminate after no more than K_o iterations, where

$$K_o = ||T_1-e\hat{T}||^2/r^2$$

Thus the number K_o provides an upper bound on the number of iterations required to transform the initial vector T_1 to the solution vector $\hat{T}$.

V. DISCUSSION OF THE COMPUTER-GRAPHICS PROGRAM

The construction of the program formulated to implement the separation algorithm will be discussed in this section, primarily to demonstrate what operations are performed on the data during the execution of the program. Instructions for the use of the program are provided in Appendix II.

The program is written in FORTRAN IV as modified for the XDS9300 computer installation at the Naval Postgraduate School. It employs the library graphics program "MAD" to allow use of the AGT graphics console with function switches and lightpen to alter the graphics display and to provide printed output via the line printer of the display. The main program can be considered to consist of four parts:

1) Set Up

2) Projection Calculation And Display

3) Display Modification

4) Output

In the set up portion storage is established in the 9300 to handle arrays based on the dimension and size of the set or sets, if there are two, of data. The data is read into the computer via two formatted READ statements. The first data set is labelled Y and the second is X (for the pattern classification problem). If a cluster analysis problem is to be conducted the variable M (number of members in the second set) is set

to zero. The program then reads the data in as one group of Y's. The only modification made directly to the data follows. The data as a whole is centered. For example in three dimensions, three averages DAV(1), DAV(2), and DAV(3) are computed over the elements of the members of the population of the X's and Y's. Then each $Y=(y_1,y_2,y_3)$ is modified to $(y_1-DAV(1),y_2-DAV(2),y_3-DAV(3))$ and similarly $X=(x_1,x_2,x_3)$ is translated to $(x_1-DAV(1),x_2-DAV(2),x_3-DAV(3))$.

The final inputs of the set up section are the initial values of the A and B vectors. The program then proceeds into the calculation of the projection coordinate pairs of each Y and X; the second part of the program now starts.

Through a succession of DO-loops the projections of the Y's and X's are calculated. The AGT screen is set up to display two dimensional points (G,H), where $-1 \leq G,H \leq 1$. This necessitates that the projections be appropriately scaled at each iteration. The scaling is accomplished by searching through all pairs for the maximum absolute values of each of the two coordinates. The first is called TMAX, the second, SMAX. Then if a point Y maps into a projected point (YY1,YY2), the point displayed on the AGT screen will have coordinates (YY1/TMAX,YY2/SMAX). The program then proceeds to display the set of the projections generated from the Y's followed by the projections generated from the X's. With the presentation of these points, the user using the function switches and the lightpen can pick an X or Y and cause an iteration by picking the direction of shift (LEFT, RIGHT, UP, or DOWN); this occurs in the third portion of the program.

If the operator chooses to make another iteration by picking an X or Y on the screen and specifying a direction, the program identifies the data vector corresponding to the chosen point and using the centralized version of this data point, calculates the new A or B vector. A loop is then made back to the beginning of the second section. There are options in addition to making an iteration open to the user. He can cause the program to loop back to the first section for inputting new values for the A and/or B vectors. He does this by depressing the switch "NEW A&B". By pushing the function switch "REGRESS", he can return to the display previous to the present. Finally by pushing "OUTPUT" he can cause the program to proceed to section four.

In section four the program fills a sixty-one by thirty-seven dimensional array with symbols corresponding to the present AGT view. This array prints out as a six by six inch square. The square is outlined by asterisks except for four zeros indicating the intercepts of the coordinate axes with the edge of the square. The points within the square appear as an X, 0, J, 8, or E. The X and 0 correspond respectively to the position of an X or Y projection in the array cell. An 8 corresponds to two or more Y's in a cell; an E implies two or more X's. A J indicates that a combination of X's and Y's occupies the cell in question. When the printout is complete, control is shifted to section three to allow the user to decide what to do next.

It is clear that the only modification to the data is the centralization to deviations about the means performed in section one. The projection

of an n-dimensional point, for example, Y, has coordinates $(A \cdot (Y-\bar{Y})/TMAX, B \cdot (Y-\bar{Y})/SMAX)$. As mentioned earlier, this section of the thesis is only intended to describe the operation of the program and to depict the relation between an initial data point and its eventual projection.

The compilation time for this program is approximately eight minutes. Of course, the total running time for the program varies with the dimensionality of the problem, the size of the data set(s), and with the number of iterations required by the user. The running time per iteration in a six dimensional problem with two hundred and eighty data points was found to be approximately twenty seconds. A user can expect to spend between thirty and forty-five minutes making iterations on a problem of this magnitude. If there is no reasonably clear-cut separation between the sets of a pattern classification problem or between the clusters of a cluster analysis, the running time may be considerably longer.

VI. EXAMPLES AND APPLICATIONS

As a means of verifying the proper operation of the separation algorithm in the graphics program, four data sets were generated for pattern classification analysis as follows:

1) Two three-dimensional cubes

2) Two six-dimensional cubes

3) Four six-dimensional cubes

4) Four dimensional sphere within a four-dimensional shell

Each of these data sets possessed geometric qualities which could easily be perceived, thus allowing concrete evaluation of the operation of the program on multidimensional data. Before addressing each data set individually, a few general comments are appropriate. Each set consists of two differentiable subsets labelled X and Y. The format of the outputs is as described earlier, with points being output as X, 0, 8, E, or J. The vector C in the discriminant functions is the centering vector defined in Section III. Coordinate axes are drawn in. The points picked for iteration are circled, and the direction for the iteration is indicated by an arrow extending from the circle.

The tuples of numbers which make up each subset were generated on the IBM 360 as uniformly distributed random numbers over appropriate ranges. In each case the number of elements per subset depended on the dimensionality of the problem; for n dimension each subset should

have at least 2^n members. This lower bound is set to insure some reasonable degree of definition for the subsets in the n space. The particulars of each data set and the results of the sample runs will now be discussed sequentially.

CASE 1. The first problem to which the algorithm was applied involved the separation of two three dimensional unit cubes. Both the X and Y subsets consisted of thirty-three tuples. If $X=(x_1,x_2,x_3)$ then x_i for i=1,3 was generated as a random number, uniform (0,1); if $Y=(y_1,y_2,y_3)$ the y_i for i=1,3 was generated uniform (1,2). Thus the X subset can be thought of as being drawn from the unit cube with one vertex at the origin and lying in the three-space. The Y subset was drawn from the unit cube displaced diagonally from the origin by the X cube. The two cubes have the vertex (1,1,1) as a point in common; however in accordance with the requirement that the sets be disjoint, (1,1,1) was considered to be a possible Y value.

Figures VI-1-1 through VI-1-3 show a three step sequence in the application of the algorithm. In Figures VI-1-1 and VI-1-2 the Y (indicated as 0's) and X subsets are mixed; however satisfactory separation is obtained in Figure VI-1-3. The values of the A and B vectors are as follows:

Figure VI-1-1	A(1)=-.5507	B(1)=-1.1837
	A(2)= .5069	B(2)= -.7223
	A(3)=-.1261	B(3)= 1.4448

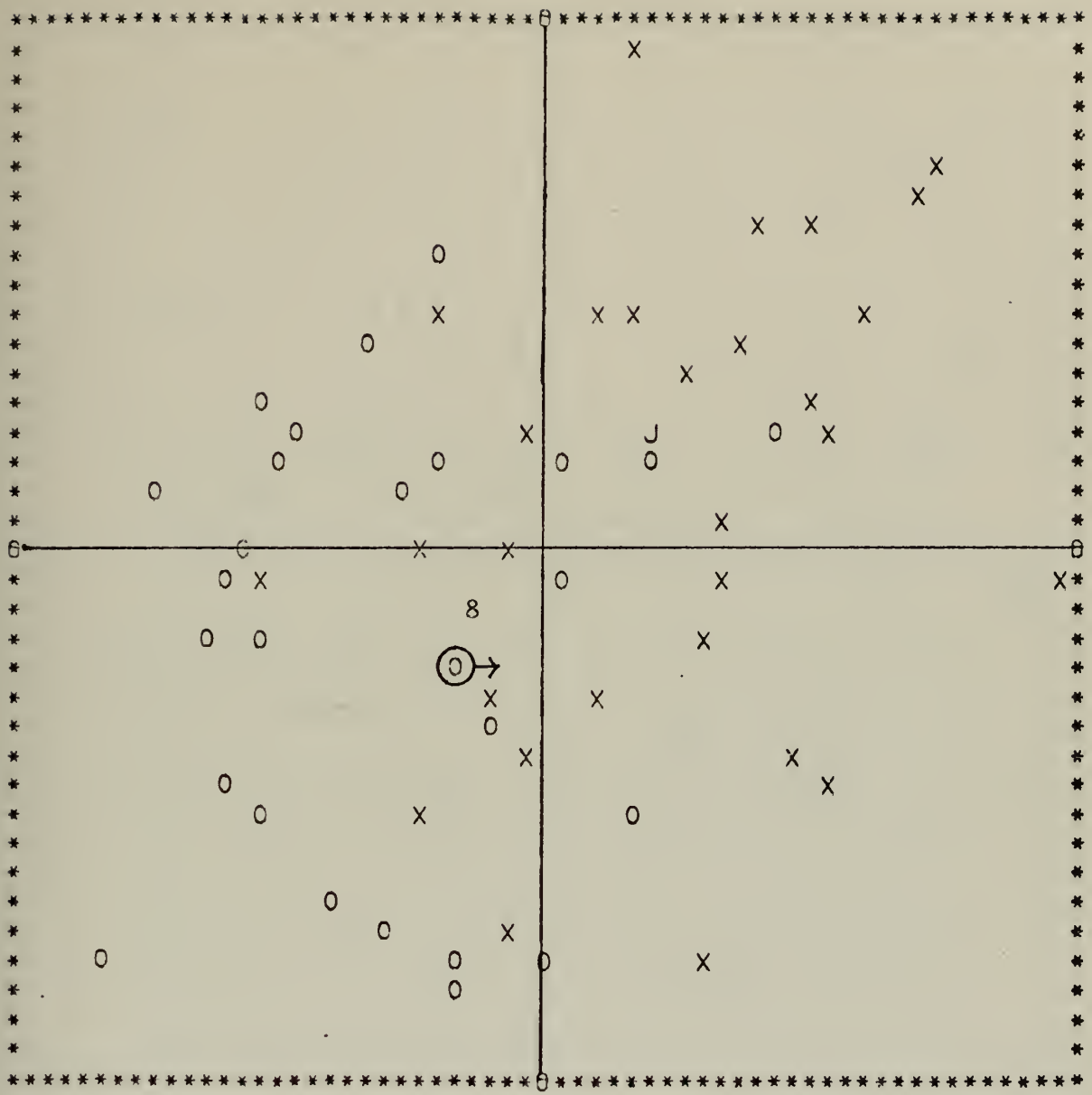


Figure VI-1-1. CASE 1, FIRST STEP

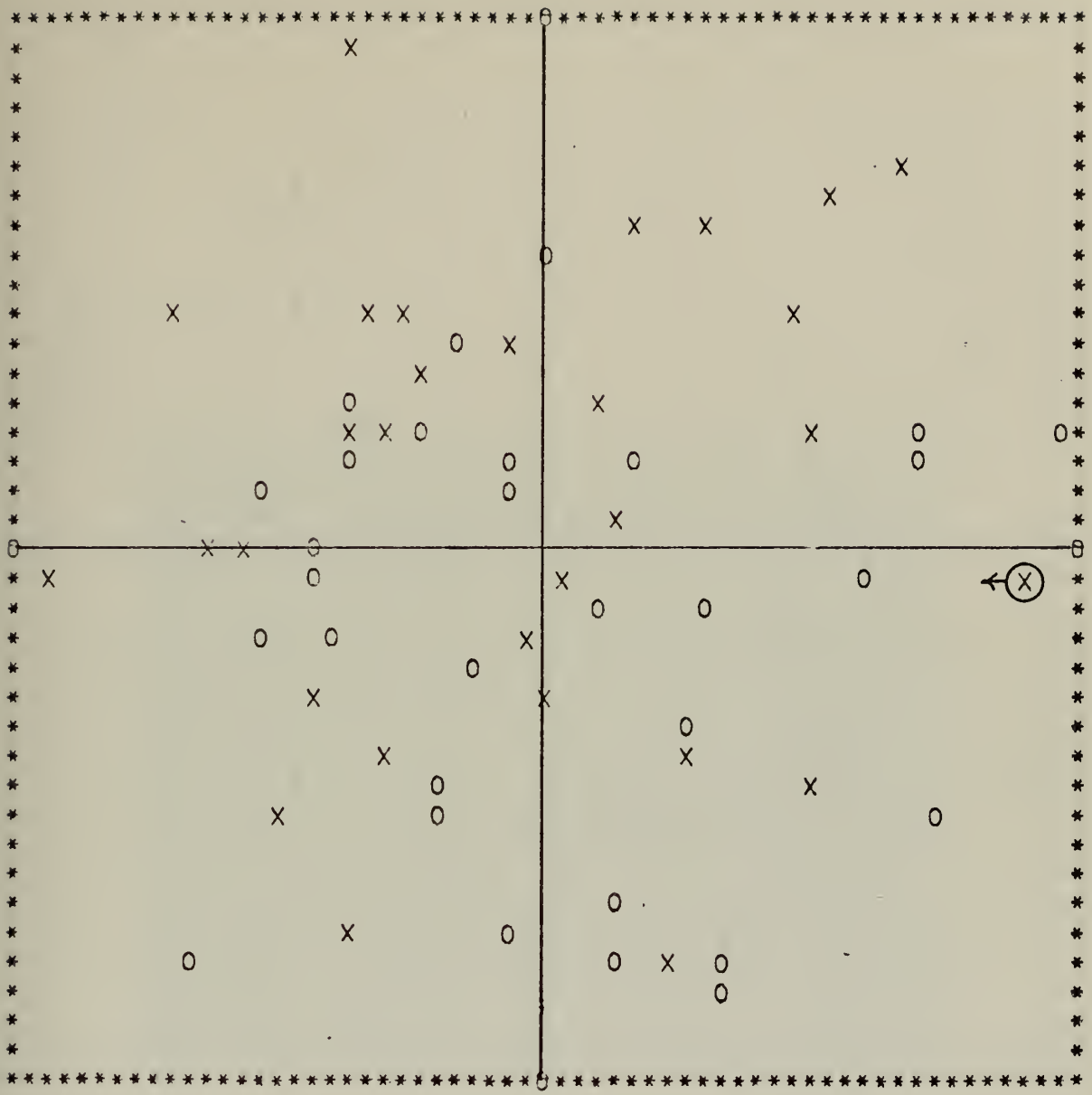


Figure VI-1-2. CASE 1, SECOND STEP

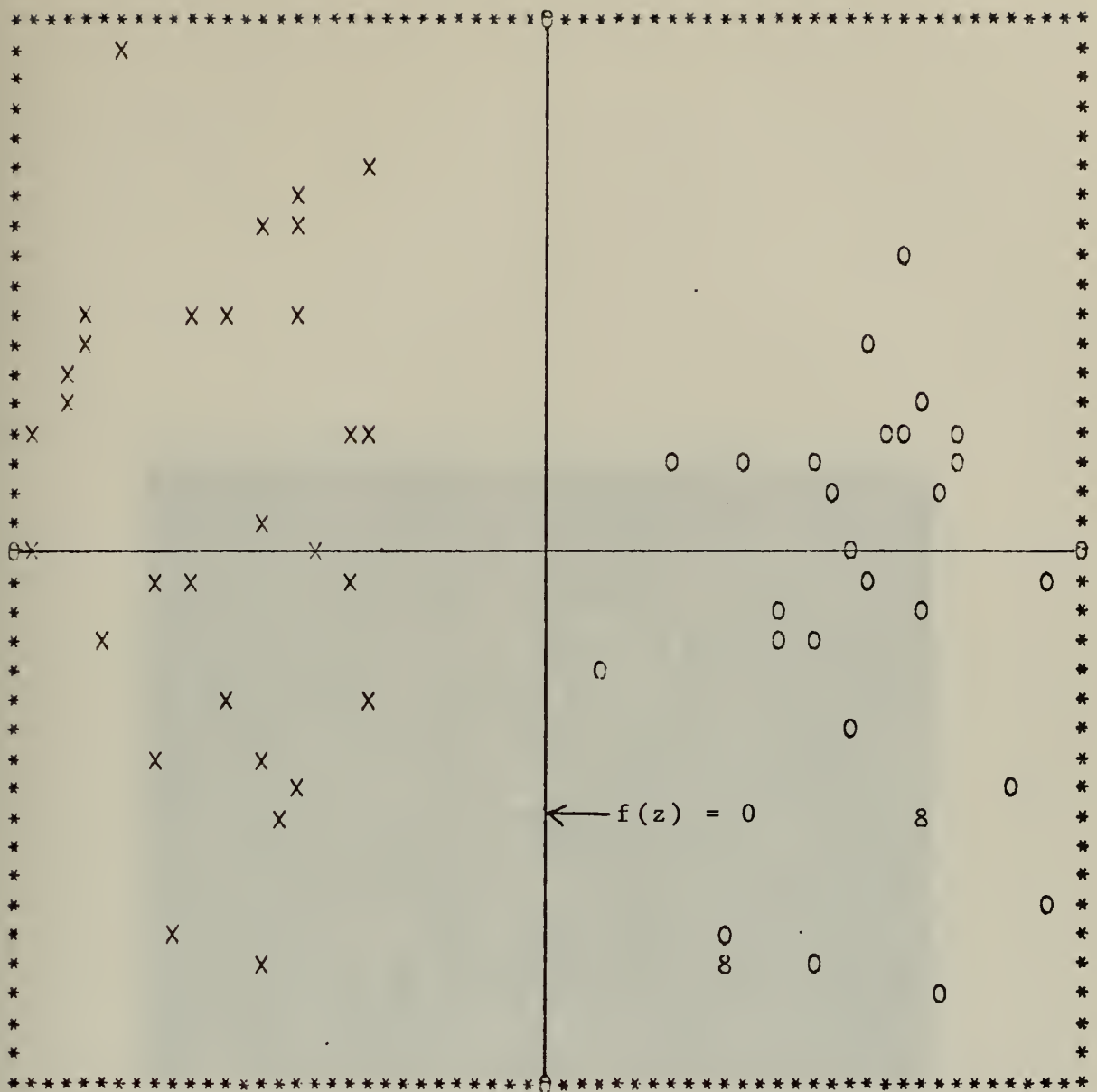


Figure VI-1-3. CASE 1, THIRD STEP

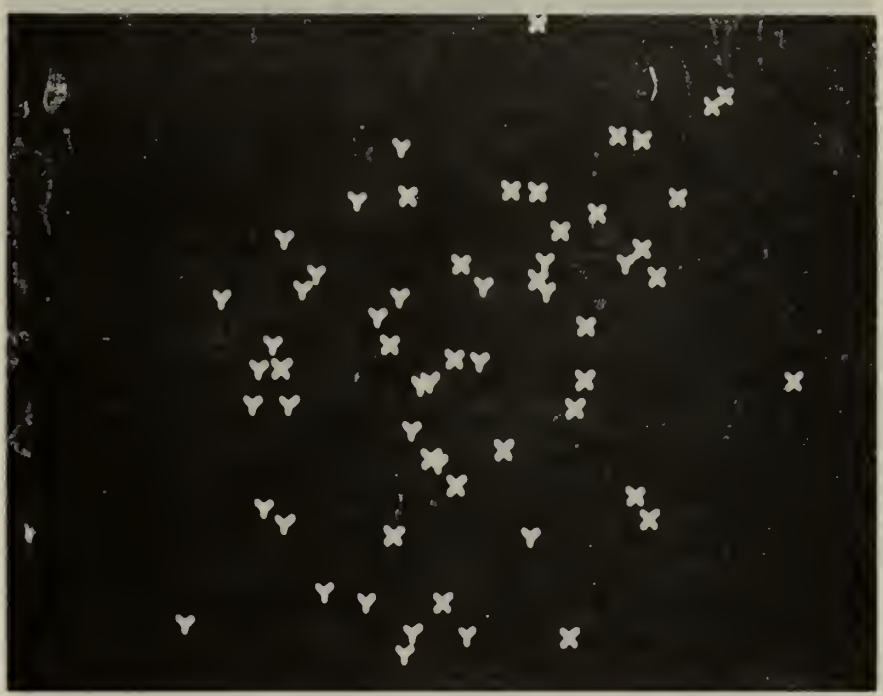


Figure VI-1-4. PHOTO OF CASE 1, FIRST STEP

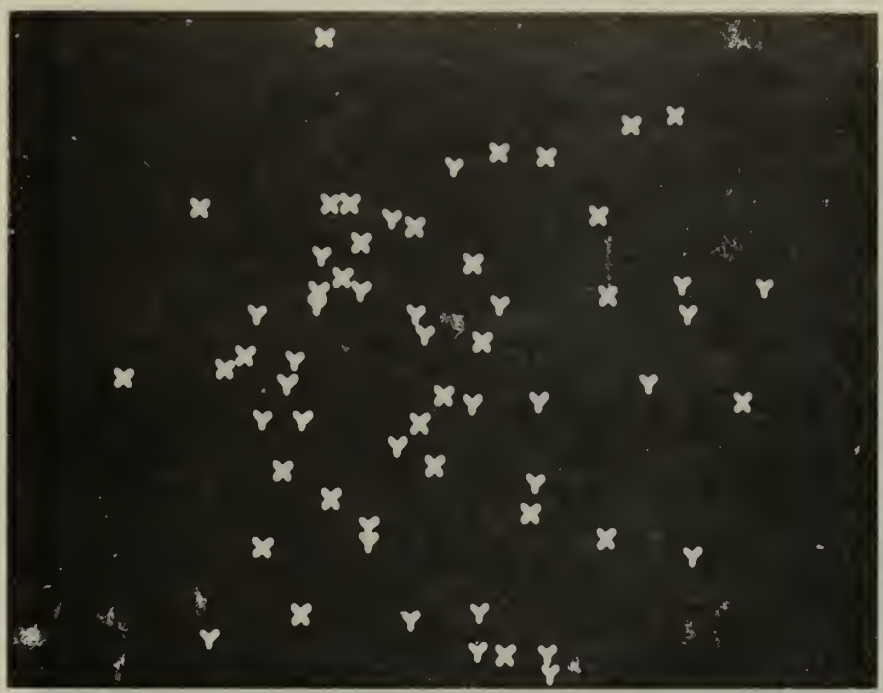


Figure VI-1-5. PHOTO OF CASE 1, SECOND STEP

Figure VI-1-6. PHOTO OF CASE 1, THIRD STEP

Figure VI-1-2	A(1)=-.3283	B(1)=-1.1837
	A(2)= .5668	B(2)= -.7223
	A(3)=-.0842	B(3)= 1.4448
Figure VI-1-3	A(1)= .5713	B(1)=-1.1837
	A(2)= .5229	B(2)= -.7229
	A(3)= .6769	B(3)= 1.4448

From Figure VI-1-3 a candidate discriminant function can be clearly seen as

$$f(Z) = A \cdot (Z-C)/TMAX$$

where A has the component values listed for Figure VI-1-3 above, and TMAX=1.5198. Thus if $f(Z) > 0$, Z is classified as a Y; if $f(Z) < 0$, Z is classified as an X.

Figures VI-1-4 through VI-1-6 are provided to show the degree of correlation between the view on the AGT screen and the resulting printed output. Figures VI-1-4, VI-1-5, and VI-1-6 correspond respectively to the outputs designated Figures VI-1-1, VI-1-2, and VI-1-3; the high degree of similarity between the related figures is clear. The results of this comparison can be readily extrapolated to the remaining problems of this section and to applications of the algorithm in general; therefore, photographic evidence for the remaining cases need not be included.

CASE 2. For the second application the dimensionality of the problem increased from three to six. The Y and X subsets were composed of seventy-six tuples generated from unit cubes oriented in six-space in the same fashion as the unit cubes of case one were in three-space.

Thus a sample from the Y subset would be $Y=(y_1,y_2,y_3,y_4,y_5,y_6)$ where $\frac{1}{2}\leq y_i \leq 2$ for i=1,6; and for X the correspondence would be $X=(x_1,x_2,x_3,x_4,x_5,x_6)$, $0\leq x_i \leq 1$ for i=1,6. Figures VI-2-1 through VI-2-3 depict the separation algorithm applied to this problem. Note that except for the greater number of data points involved, the complexity of the problem is not increased from the three dimensional problem of case one. The A and B vectors for these three views are as follows

Figure VI-2-1	A(1)= 1.3448	B(1)= .0868
	A(2)= .3701	B(2)=-1.4109
	A(3)=-1.9594	B(3)= -.7244
	A(4)= .7525	B(4)=-1.0185
	A(5)= 1.3047	B(5)= 2.9777
	A(6)= -.7075	B(6)= .4225
Figure VI-2-2	A(1)= 1.3448	B(1)= .6450
	A(2)= .3701	B(2)= -.5899
	A(3)=-1.9594	B(3)= .0692
	A(4)= .7525	B(4)= -.3409
	A(5)= 1.3047	B(5)= 3.0208
	A(6)= -.7075	B(6)= .7001
Figure VI-2-3	A(1)= 1.3448	B(1)= .7298
	A(2)= .3701	B(2)= .4801
	A(3)=-1.9594	B(3)= .5716
	A(4)= .7525	B(4)= .5345
	A(5)= 1.3047	B(5)= 3.0897
	A(6)= -.7075	B(6)= .7335

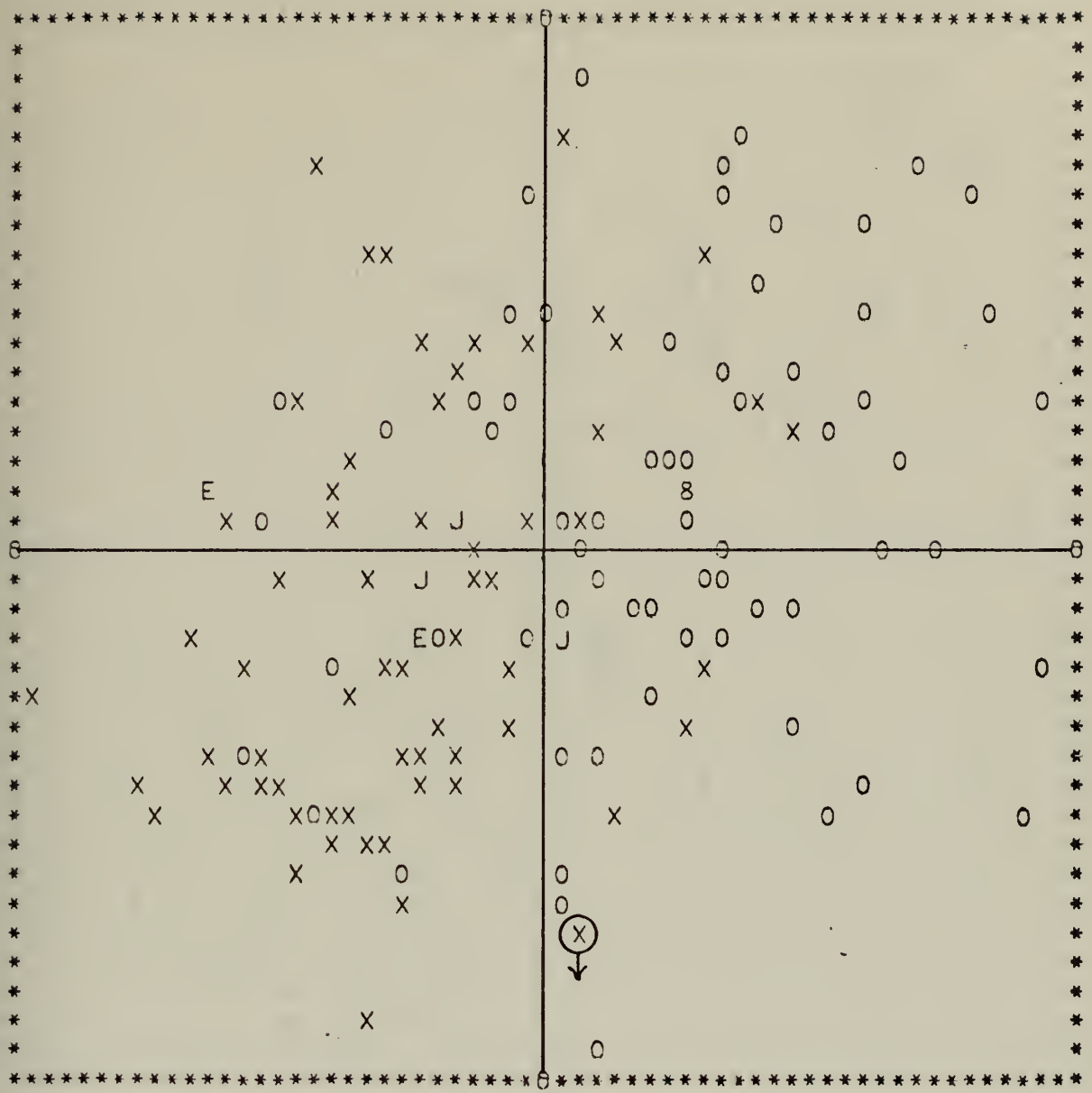


Figure VI-2-1. CASE 2, FIRST STEP

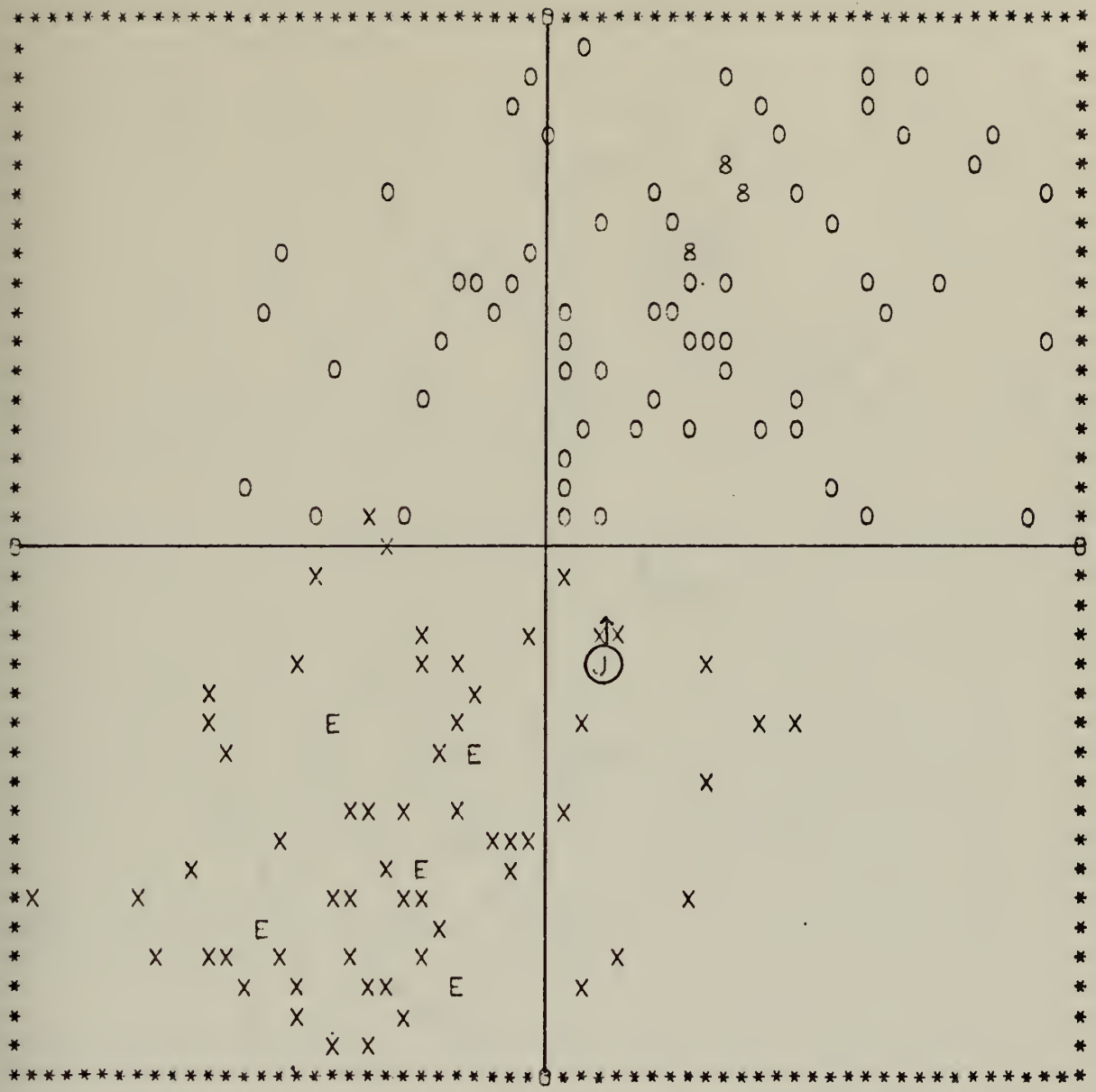


Figure VI-2-2. CASE 2, SECOND STEP

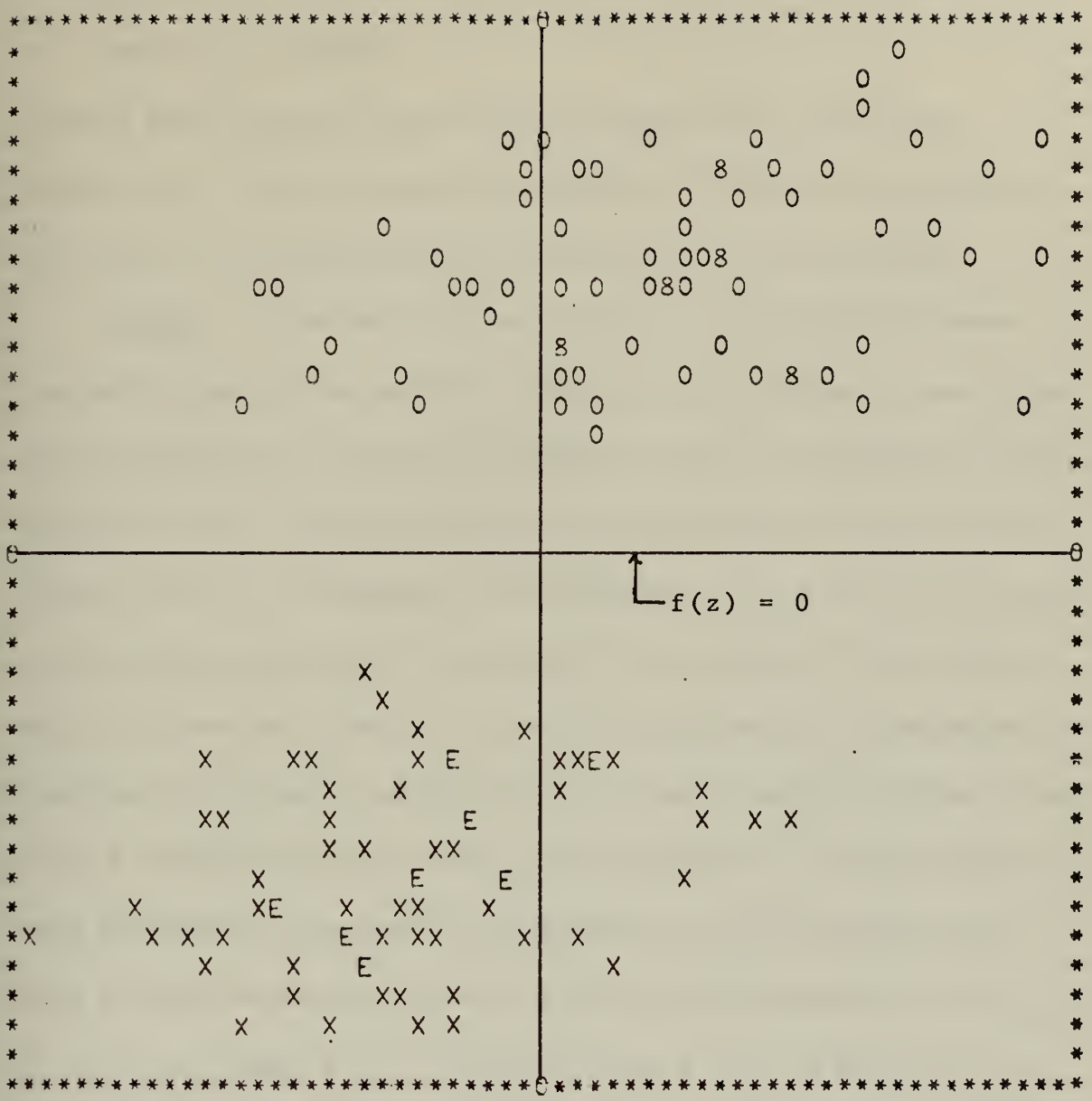


Figure VI-2-3. CASE 2, THIRD STEP

From Figure VI-2-3 a reasonable discriminant function would be

$$f(Z)=B\cdot(Z-C)/SMAX$$

where B has component values listed for Figure VI-2-3 above and SMAX=3.1940. Thus a vector Z for which $f(Z) > 0$ would be assigned as a Y; if $f(Z) < 0$ the result would be classification of the Z as an X.

CASE 3. In the last problem, separation was obtained between two sets of six-dimensional data. The aim of the problem in case three is to complicate the situation by generating two six-dimensional subsets, labelled Y and X, each of which can further be described as consisting of two subsets. The purpose of the application is to see if the algorithm would cause a separation to appear not only between the Y and X sets, but also within these sets. In a sense this problem is a combination of a pattern classification application and a cluster analysis problem. The Y and X sets each consist of one hundred and forty-six tuples; these sets were additionally generated as two subsets of seventy members each. Thus the first seventy points in the Y subset were generated as $Y=(y_1,y_2,y_3,y_4,y_5,y_6)$ where $0 \leq y_i < 1$ for i=1,4 and $1 \leq y_i \leq 2$ for i=5,6. In the second subset of Y's, $0 \leq y_i \leq 1$ for i=1,6. The first seventy X's were typically $X=(x_1,x_2,x_3,x_4,x_5,x_6)$ where $0 \leq x_i < 1$ for i=1,5 and $1 \leq x_6 \leq 2$; for the second X subset, $0 \leq x_i < 1$ for i=1,4, $1 \leq x_5 \leq 2$, and $0 \leq x_6 < 1$. As in case two the increased complexity of the situation is reflected immediately by the number of points involved; the computational aspects are unaffected. Once again only three figures are required to display separation. The A and B vectors assume values as indicated

below:

Figure VI-3-1	A(1)= .0503	B(1)= .1174
	A(2)= -.3371	B(2)= -.5137
	A(3)= .5022	B(3)= 1.4294
	A(4)= 1.1334	B(4)= -.5413
	A(5)= .6375	B(5)= .6996
	A(6)= 1.5314	B(6)=-1.2121
Figure VI-3-2	A(1)= 1.6166	B(1)= .1174
	A(2)=-1.7057	B(2)= -.5137
	A(3)= .6496	B(3)= 1.4294
	A(4)= 1.2731	B(4)= -.5413
	A(5)= .9450	B(5)= .6996
	A(6)= 1.9902	B(6)=-1.2121
Figure VI-3-3	A(1)= 1.6166	B(1)=-1.5022
	A(2)=-1.7057	B(2)=-2.1664
	A(3)= .6496	B(3)= 1.4117
	A(4)= 1.2731	B(4)= -.3975
	A(5)= .9450	B(5)= .6192
	A(6)= 1.9902	B(6)=-1.4552

The discriminant function will obviously be of a more complicated nature, judging from the appearance of Figure VI-3-3. The following function appears to be satisfactory:

$f(Z)=(A\cdot(Z-C)/TMAX+B\cdot(Z-C)/SMAX)(A\cdot(Z-C)/TMAX-B\cdot(Z-C)/SMAX)$

where A and B assume values indicated for Figure VI-3-3, TMAX=7.3396,

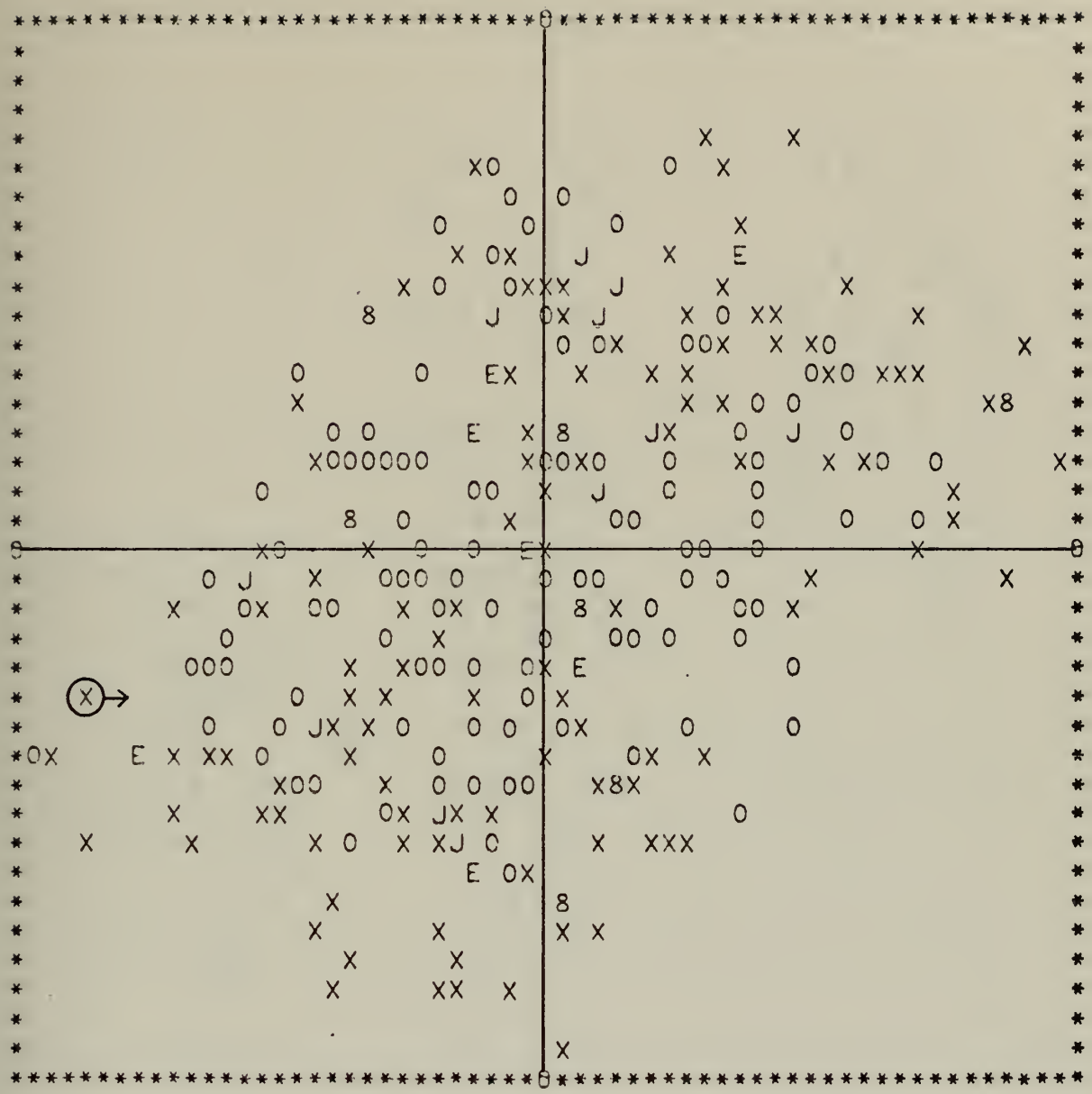


Figure VI-3-1. CASE 3, FIRST STEP

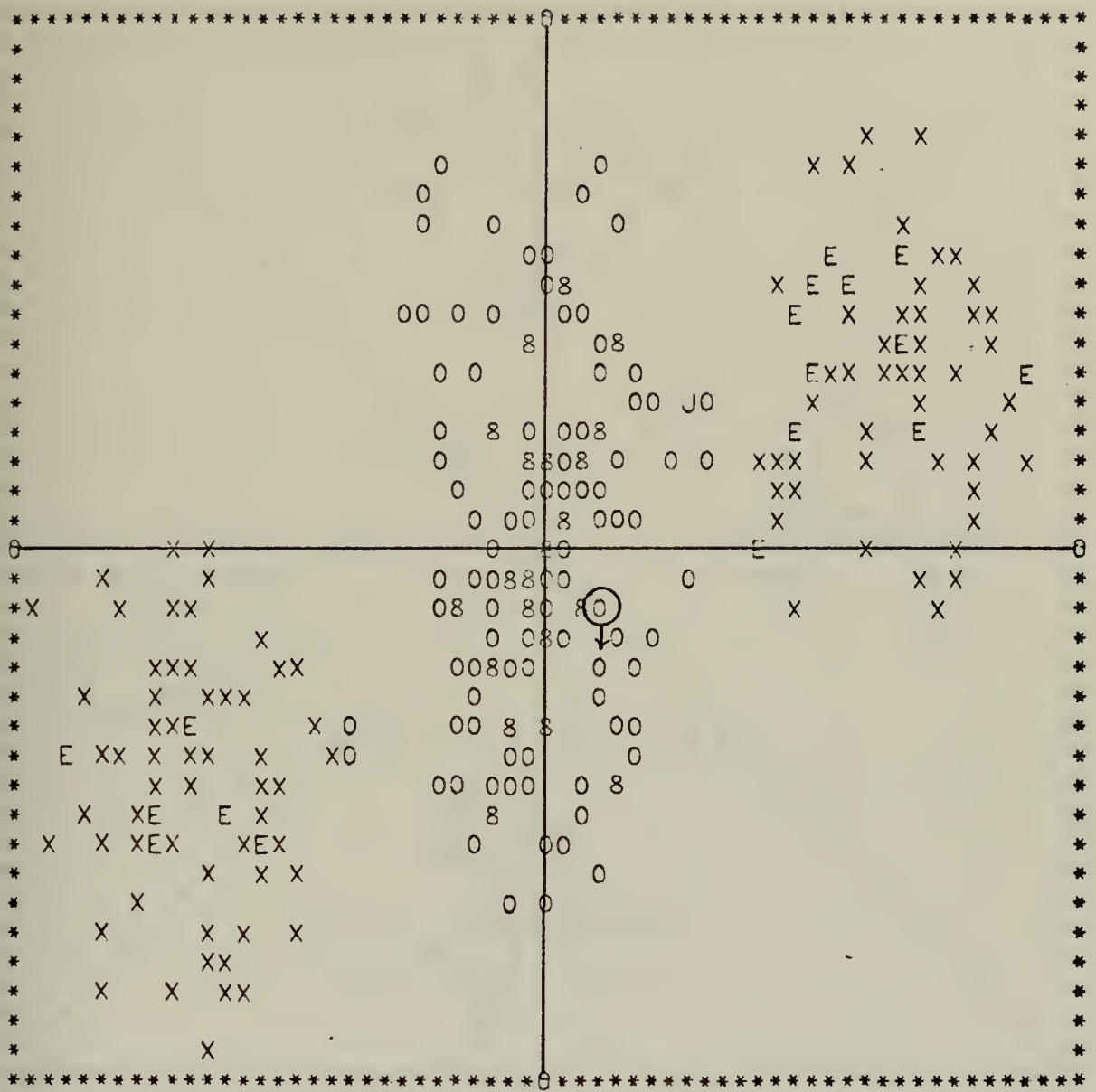


Figure VI-3-2. CASE 3, SECOND STEP

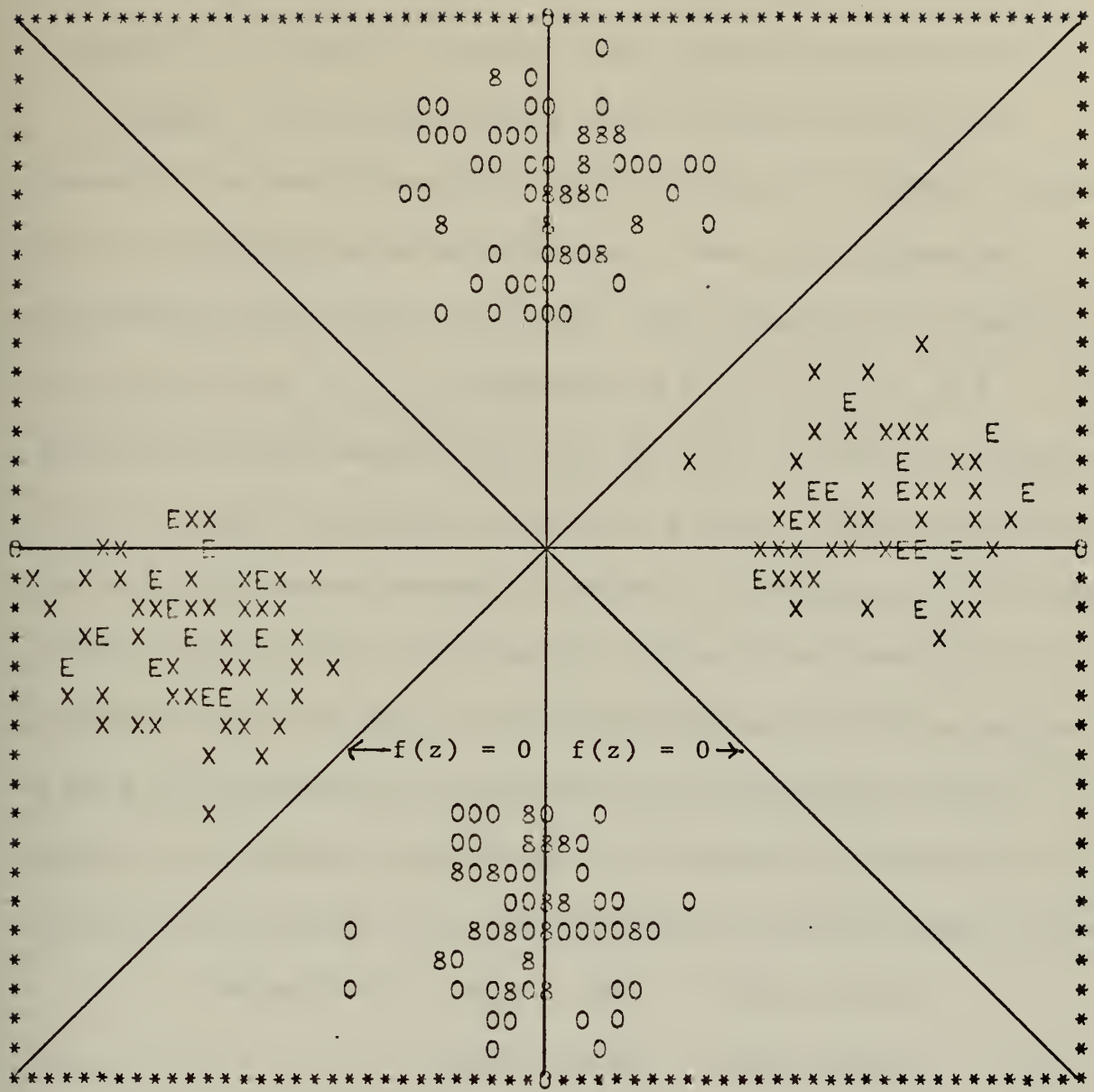


Figure VI-3-3. CASE 3, THIRD STEP

and SMAX=7.5366. For this problem, $f(Z) > 0$ would result in Z being classified as an X; $f(Z) < 0$ naturally would result in Y classification.

CASE 4. In the first data sets of this section the points were generated from multi-dimensional cubes. In case four a different orientation is used, this time in four dimensions. Twenty-four tuples are generated for each of the X and Y sets. The Y set generated so that a typical point $Y=(y_1,y_2,y_3,y_4)$ satisfies $2 \leq y_1^2 + y_2^2 + y_3^2 + y_4^2 \leq 4$. Similarly an X point satisfies $x_1^2 + x_2^2 + x_3^2 + x_4^2 \leq 1$. Thus the geometric "picture" is that of a sphere surrounded by a shell. For this application, two sets of figures are provided to emphasize the importance of the initial values assigned to the A and B vectors. Figures VI-4-1 through VI-4-5 display the results of making an unfortunate choice of initial values for A and B. Thirty iterations were carried out from this initial choice; however, no significant improvement on the situation as indicated by Figure VI-4-5 was made. The values of the A and B vectors were as follows:

Figure VI-4-1	A(1)= .5780	B(1)=- .5000
	A(2)= .5780	B(2)=- .5000
	A(3)= .5780	B(3)= 1.0000
	A(4)= .0000	B(4)= 1.0000
Figure VI-4-2	A(1)= .6680	B(1)=- .5000
	A(2)=- .2623	B(2)=- .5000
	A(3)=- .0208	B(3)= 1.0000
	A(4)= .0900	B(4)= 1.0000

Figure VI-4-3	A(1)= .6680	B(1)=- .0280
	A(2)=- .2623	B(2)= .0917
	A(3)=- .0208	B(3)= .4002
	A(4)= .0900	B(4)= .7390
Figure VI-4-4	A(1)= .6480	B(1)=- .0280
	A(2)= .1963	B(2)= .0917
	A(3)=- .7196	B(3)= .4002
	A(4)= .6110	B(4)= .7390
Figure VI-4-5	A(1)= .6680	B(1)=- .0280
	A(2)=- .2623	B(2)= .0917
	A(3)=- .0208	B(3)= .4002
	A(4)= .0900	B(4)= .7390

Rather than attempt to find a satisfactory discriminant function for the situation depicted in Figure VI-4-5, the approach is to specify new A and B vectors and to start over. In fact, starting with A=(1,1,1,1) and B=(1,-1,1,-1), separation appeared within twenty iterations. Figures VI-4-6 through VI-4-10 depict the situation after the first, fifth, tenth, fifteenth, and twentieth iterations respectively. The A and B vectors for the figures are:

Figure VI-4-6	A(1)= 1.0000	B(1)= 1.0000
	A(2)= 1.0000	B(2)=-1.0000
	A(3)= 1.0000	B(3)= 1.0000
	A(4)= 1.0000	B(4)=-1.0000

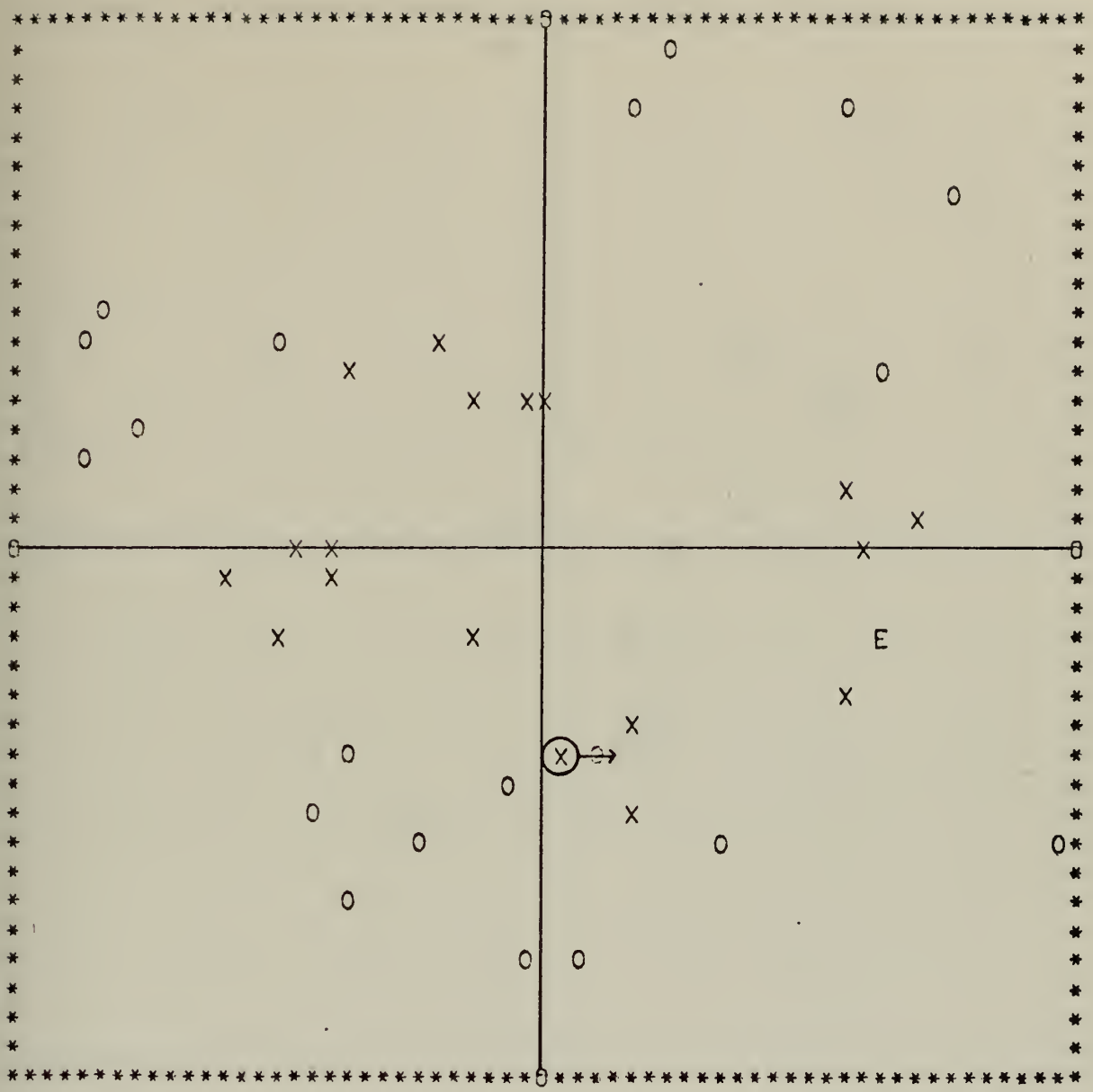


Figure VI-4-1. CASE 4, FIRST STEP

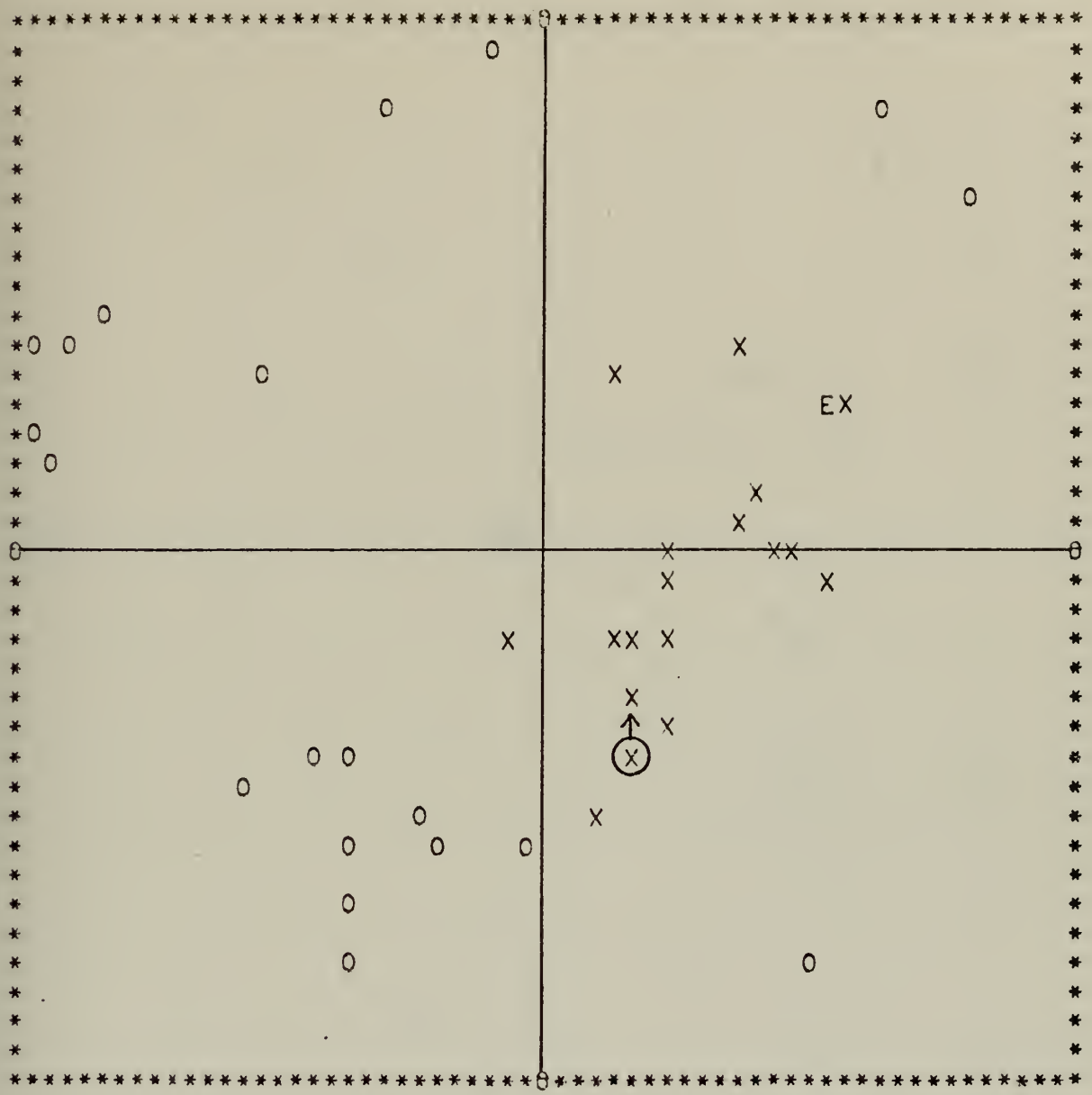


Figure VI-4-2. CASE 4, SECOND STEP

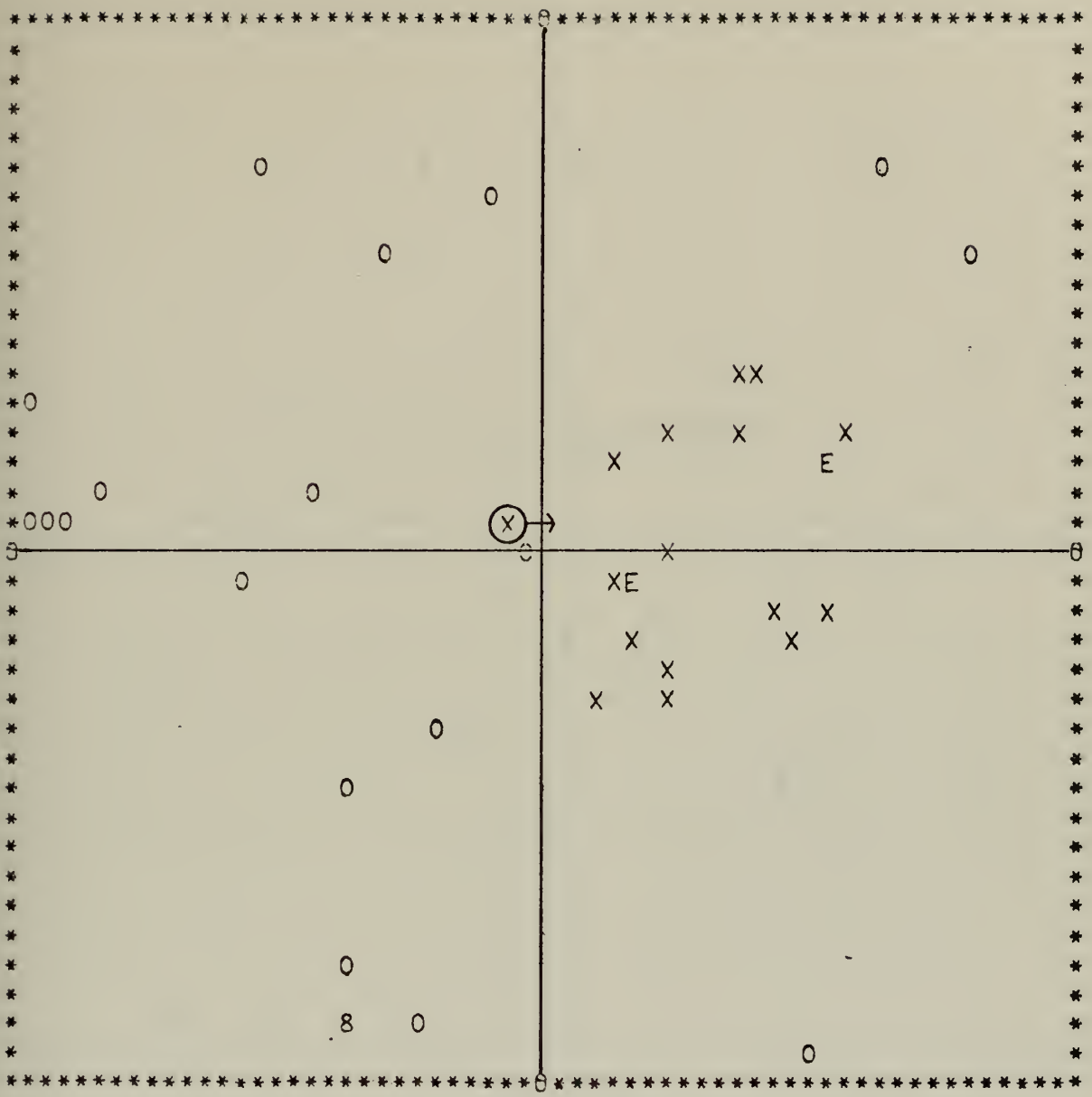


Figure VI-4-3. CASE 4, THIRD STEP

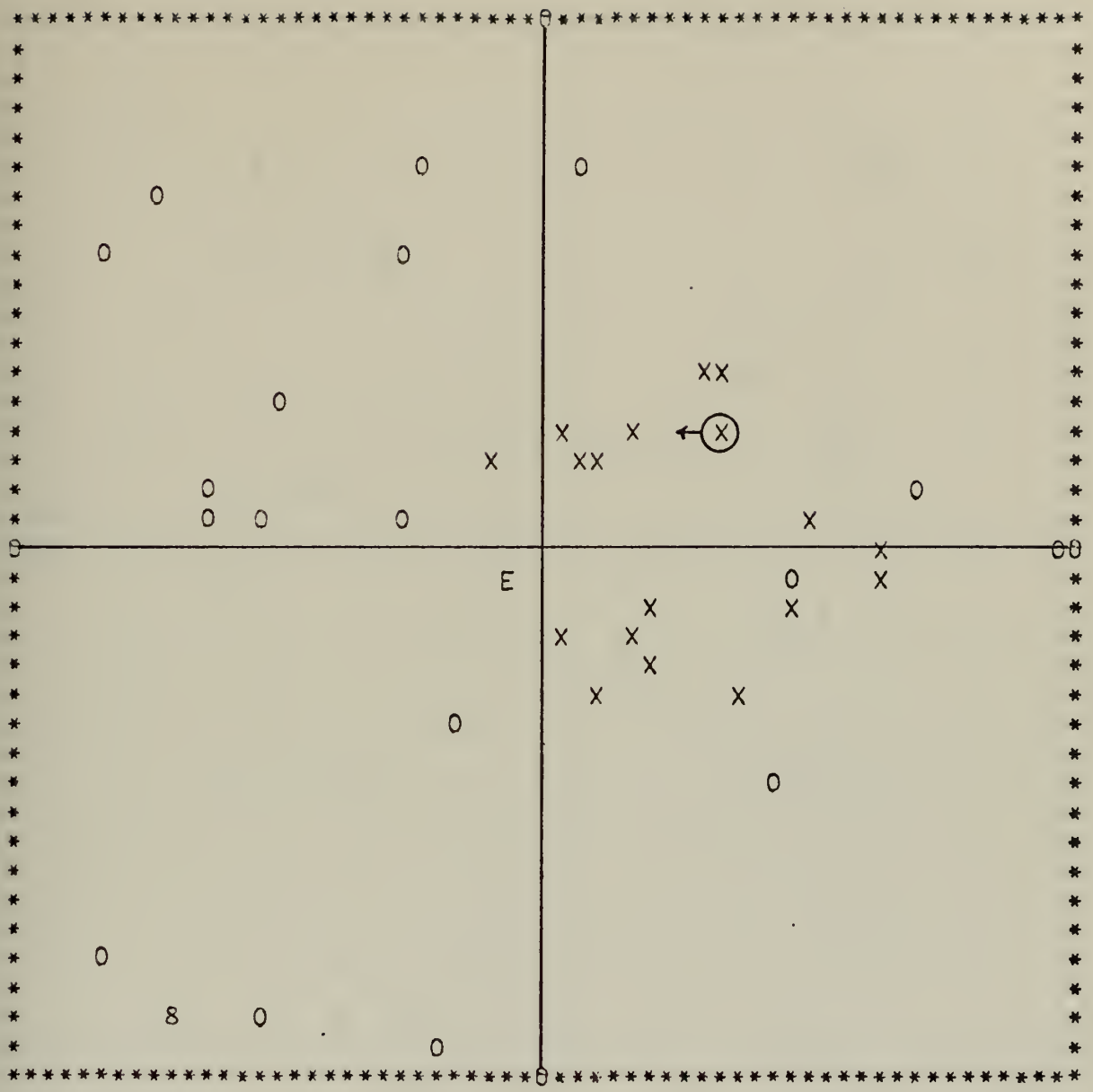


Figure VI-4-4. CASE 4, FOURTH STEP

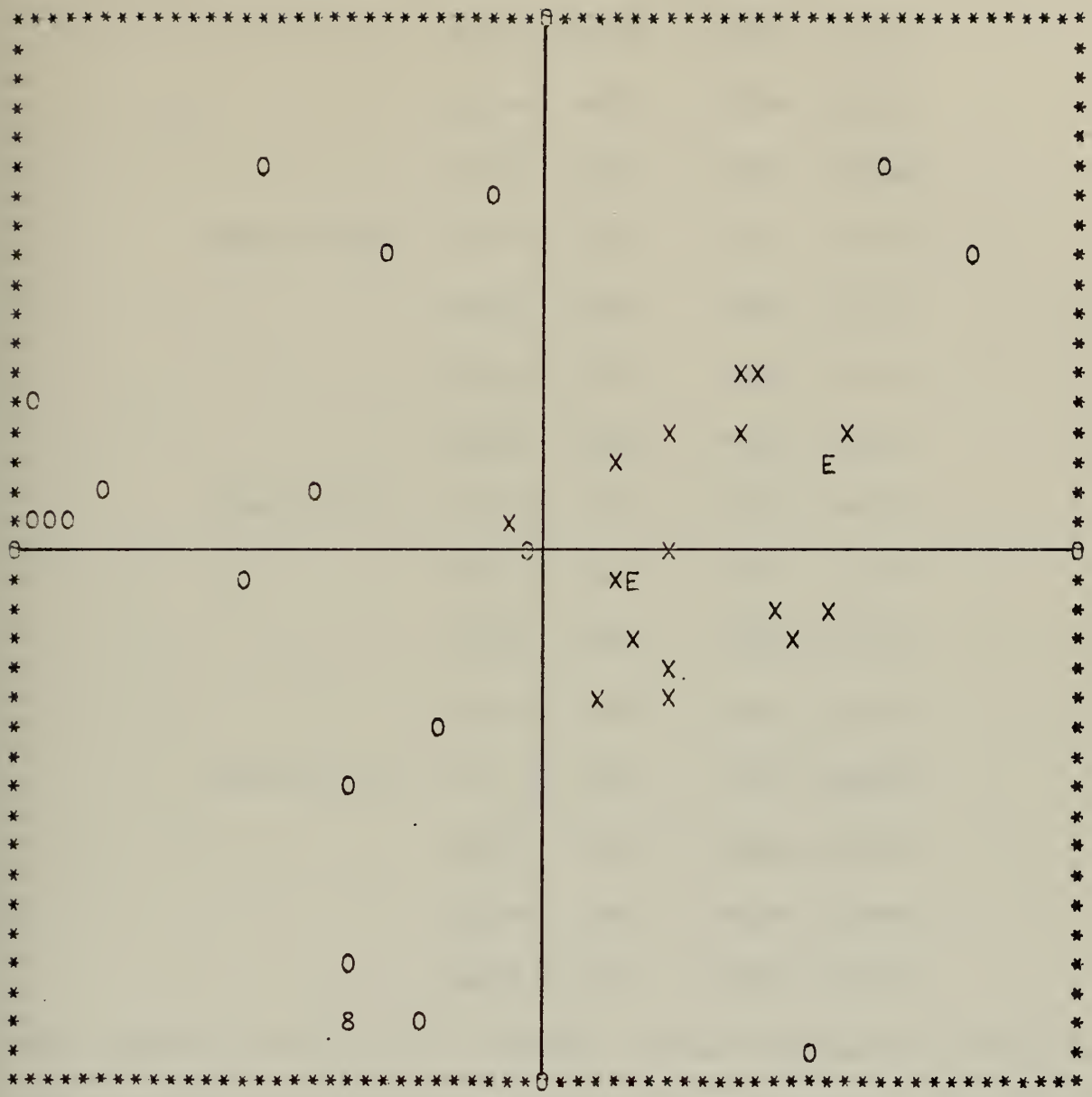


Figure VI-4-5. CASE 4, FIFTH STEP

Figure VI-4-7	A(1)= 2.1930	B(1)= .9800
	A(2)= 2.1246	B(2)=- .5413
	A(3)=- .8528	B(3)= .3012
	A(4)=- .2470	B(4)=- .4790
Figure VI-4-8	A(1)= 1.8910	B(1)= 1.5190
	A(2)= .4019	B(2)=- .4473
	A(3)=- .5144	B(3)= 1.7293
	A(4)= .5510	B(4)=- .4620
Figure VI-4-9	A(1)= 1.5890	B(1)= .8490
	A(2)=-1.3207	B(2)=- .9287
	A(3)=- .1760	B(3)= 3.3314
	A(4)= 1.3490	B(4)=- .1040
Figure VI-4-10	A(1)= 2.1580	B(1)= 1.6860
	A(2)=- .3294	B(2)=-1.0144
	A(3)=- .6366	B(3)= 2.3048
	A(4)= 1.4610	B(4)=-1.2420

The situation in Figure VI-4-10 provides another opportunity for displaying the flexibility possible in choosing a discriminant function. The function in this case is circular with the equation:

$$f(Z)= (A\cdot (Z-C)/TMAX-1/3)^2 + (B\cdot (Z-C)/SMAX)^2-1/4$$

where A and B have the values specified for Figure VI-4-10, TMAX=3.3577, and SMAX=5.8823. Clearly if $f(Z) < 0$, Z is classified as an X; if $f(Z) > 0$, Z is classified as a Y.

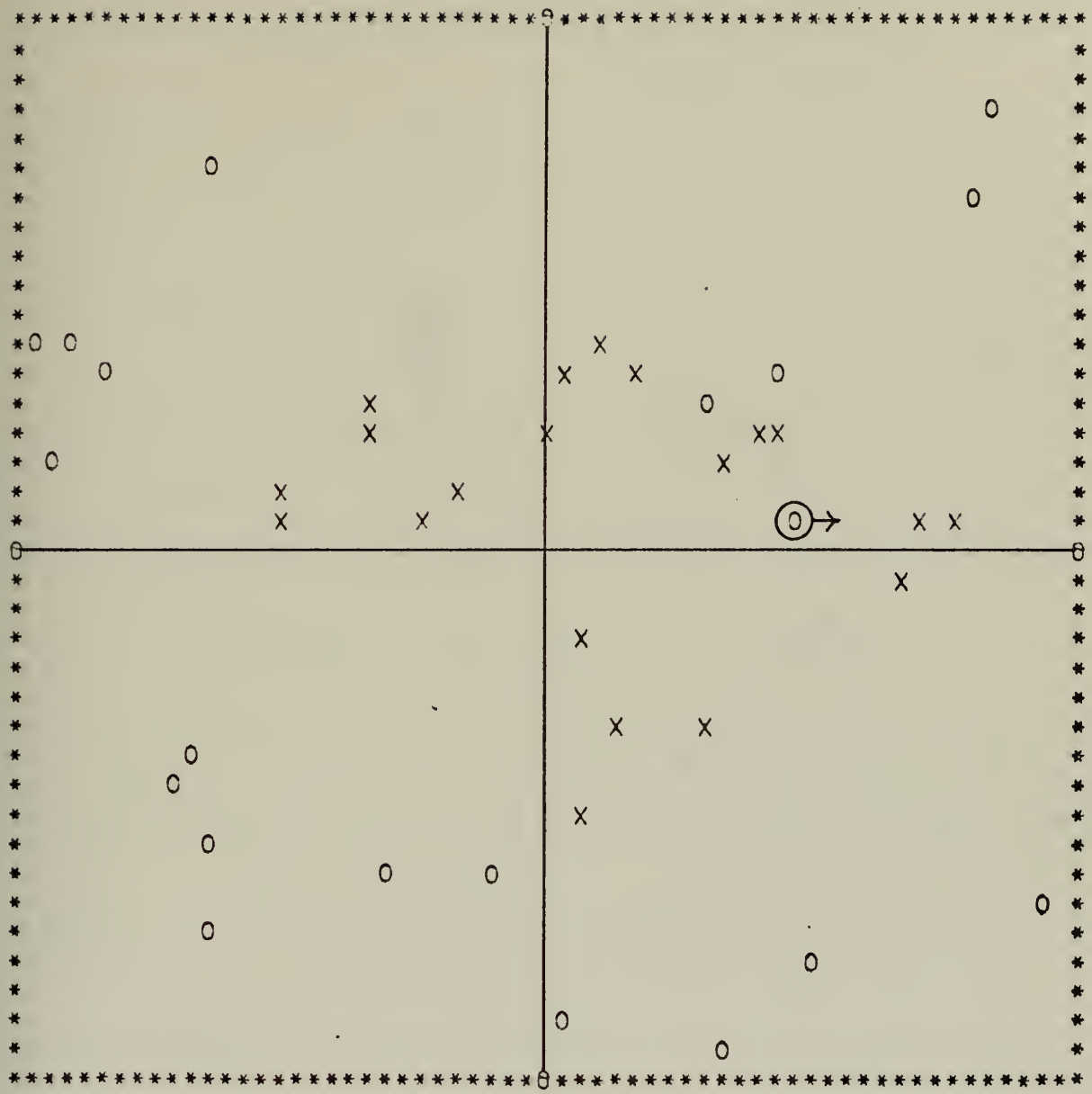


Figure VI-4-6. CASE 4, SIXTH STEP

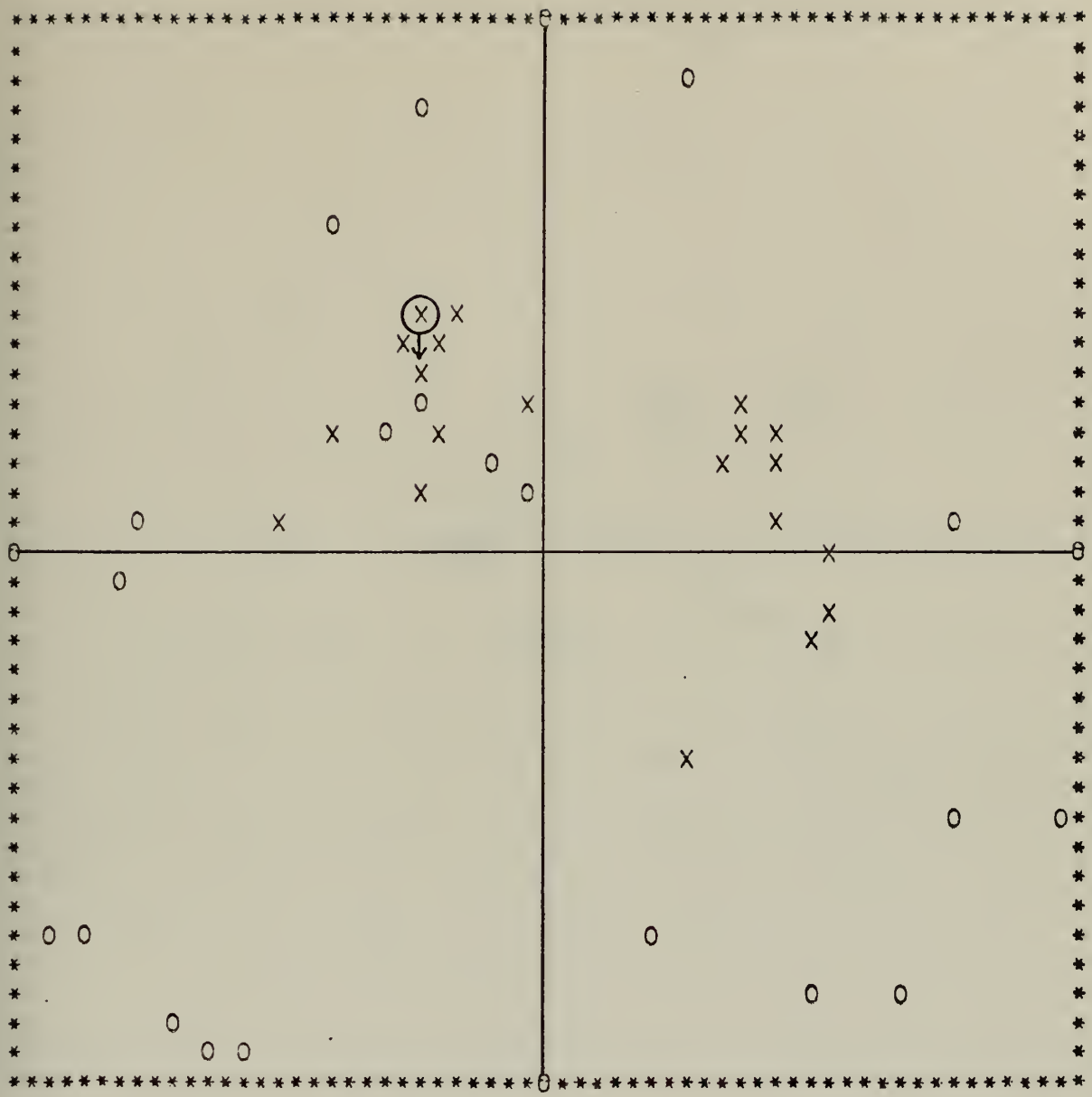


Figure VI-4-7. CASE 4, SEVENTH STEP

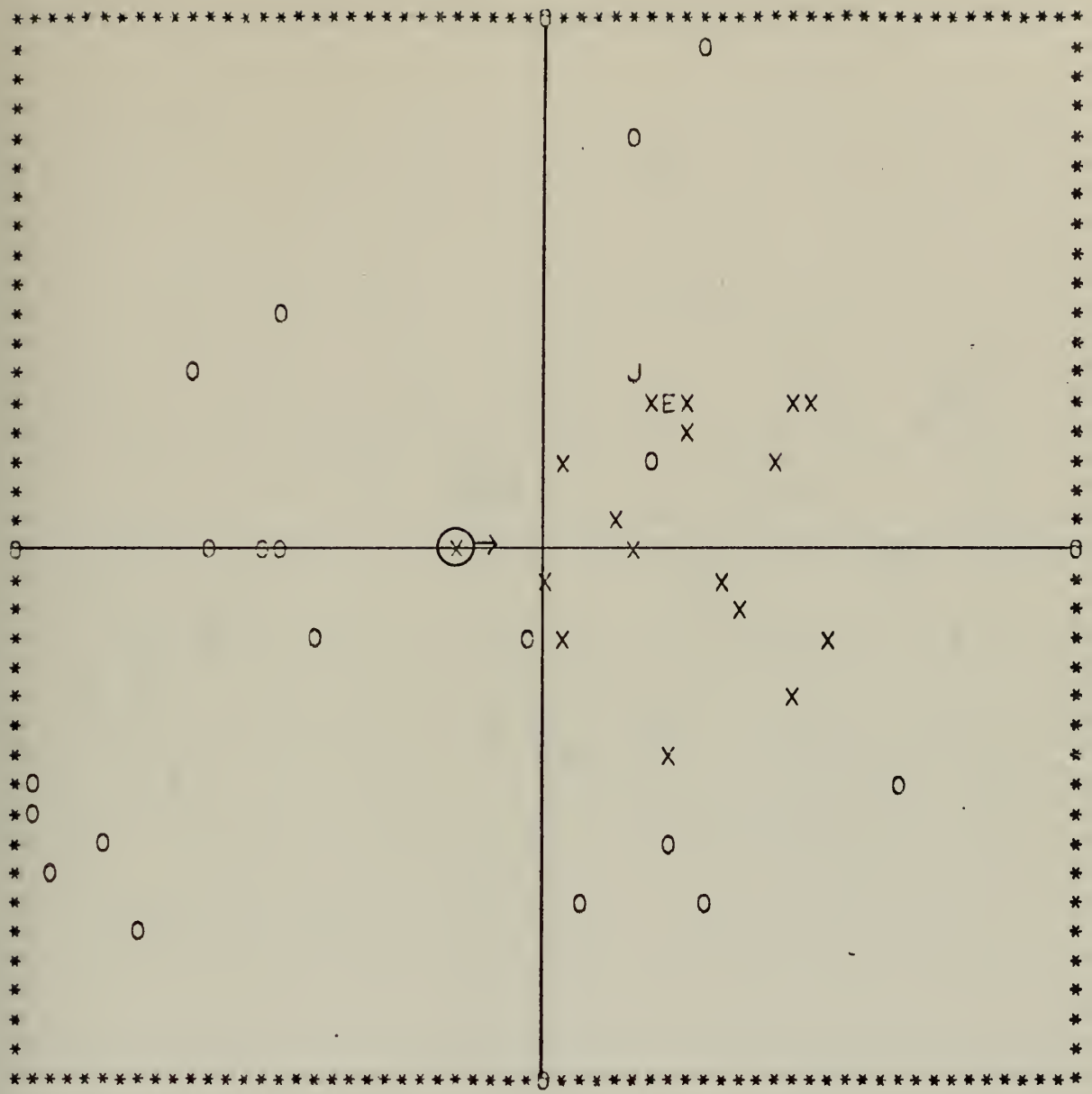


Figure VI-4-8. CASE 4, EIGHTH STEP

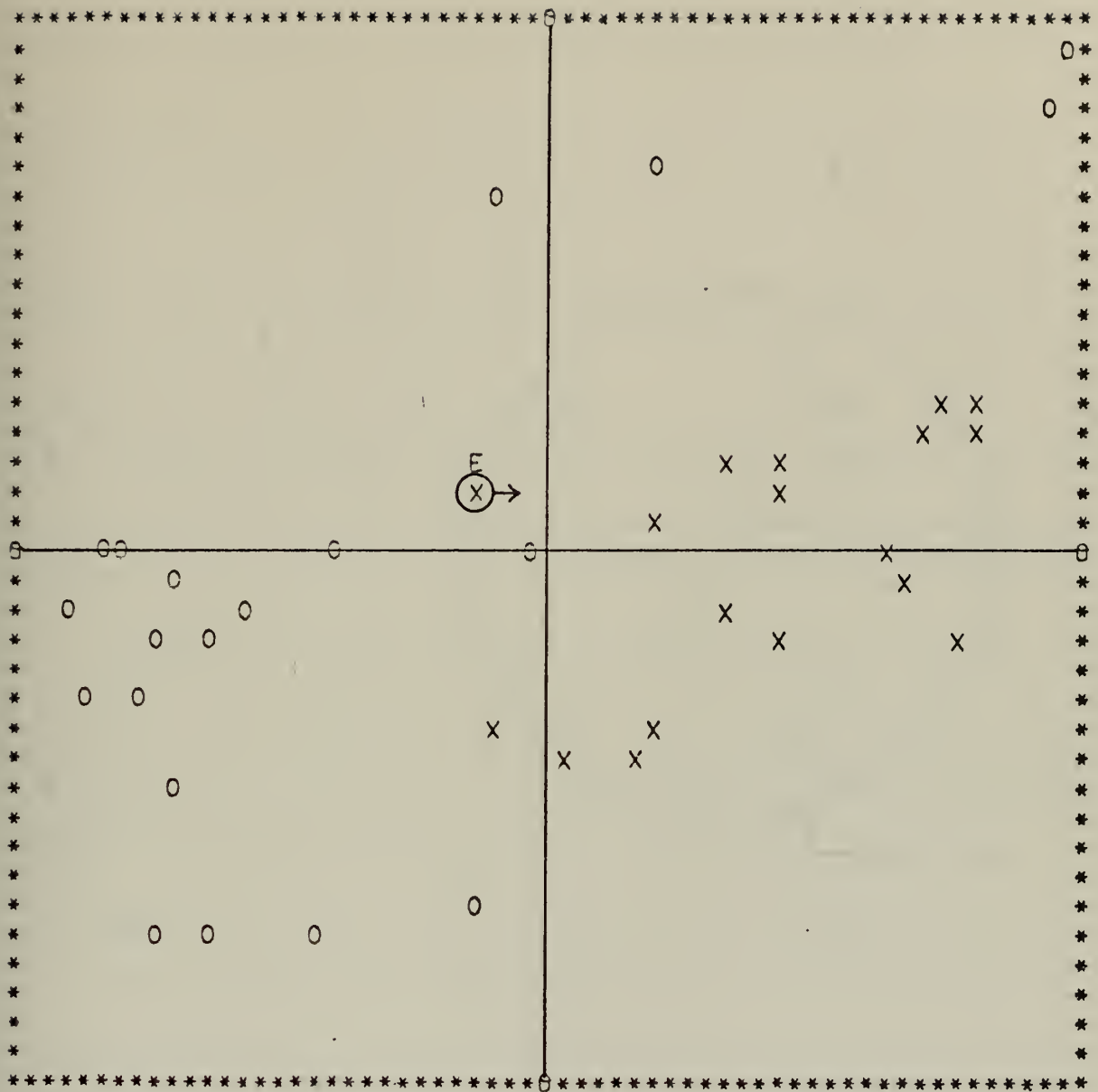


Figure VI-4-9. CASE 4, NINTH STEP

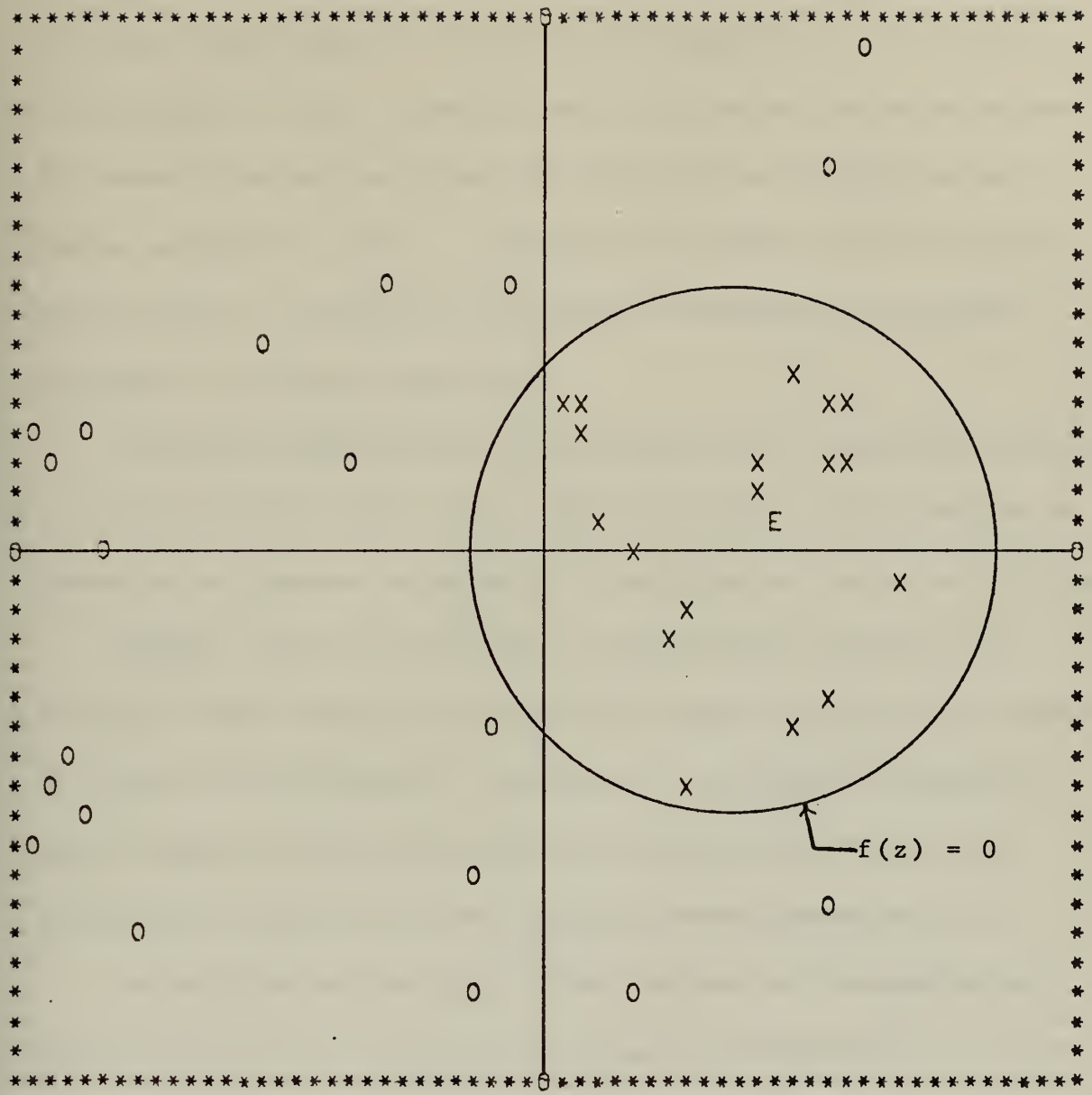


Figure VI-4-10. CASE 4, TENTH STEP

It should be noted that the separation found was a result of the particular samples used. In reality the Y elements must have formed a ring instead of a shell. Clearly there is no possible orientation for the A-B plane into which the sphere and shell could be projected so as to display separation. This is a case where the samples were in fact too small to allow a discriminant function to be derived which would be applicable to the parent populations.

Cases five and six represent applications of the separation program to "real world" data sets. The complexities found in actual data are reflected by the apparent reduction in the performance level of the algorithm.

CASE 5. The data set for this cluster analysis problem is the Jamaican "Fossil" data mentioned in the discussion of Chernoff's method in section two of this paper. The data set, in the form of eighty-two eight-tuples, was drawn from Wright & Switzer's paper "Numerical Classification Applied to Certain Jamaican Eocene Nummulitids" 6 .

As indicated earlier in this paper, the data was represented as a single set of points Ÿ (points on the output are indicated only as 0's or 8's), and iterations were made based on apparent separation; the aim is to emphasize this separation. Twelve iterations were made starting with A=(1,1,1,1,1,1,1,1) and B=(1,-1,1,-1,1,-1,1,-1). Figures VI-5-1 through VI-5-6 were output on the first, second, third, fourth, ninth, and twelfth iterations respectively. Since no discriminant function is generated in a cluster analysis, the values of the A and B vectors have no final importance. On the other hand, the precise identity of each

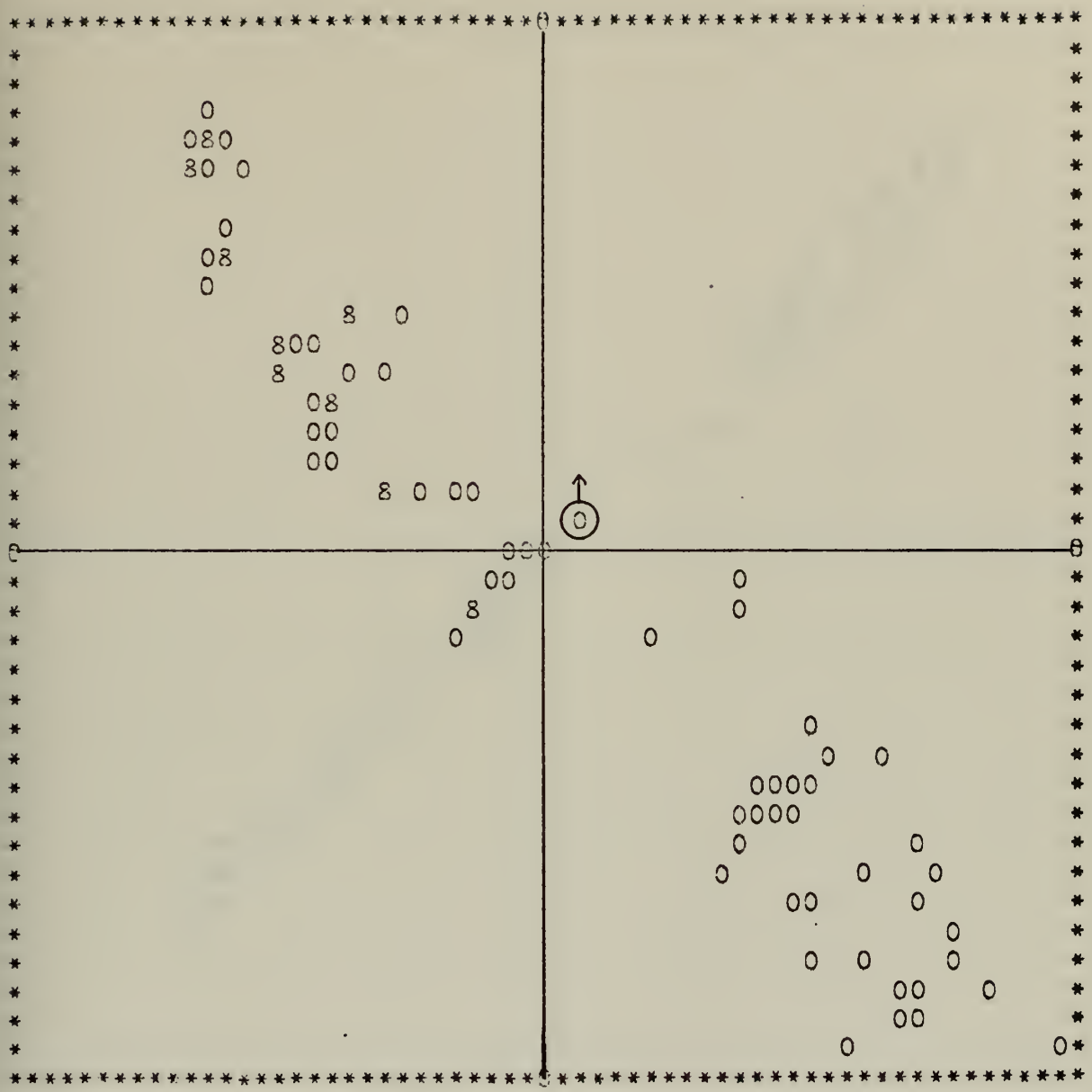


Figure VI-5-1. CASE 5, FIRST STEP

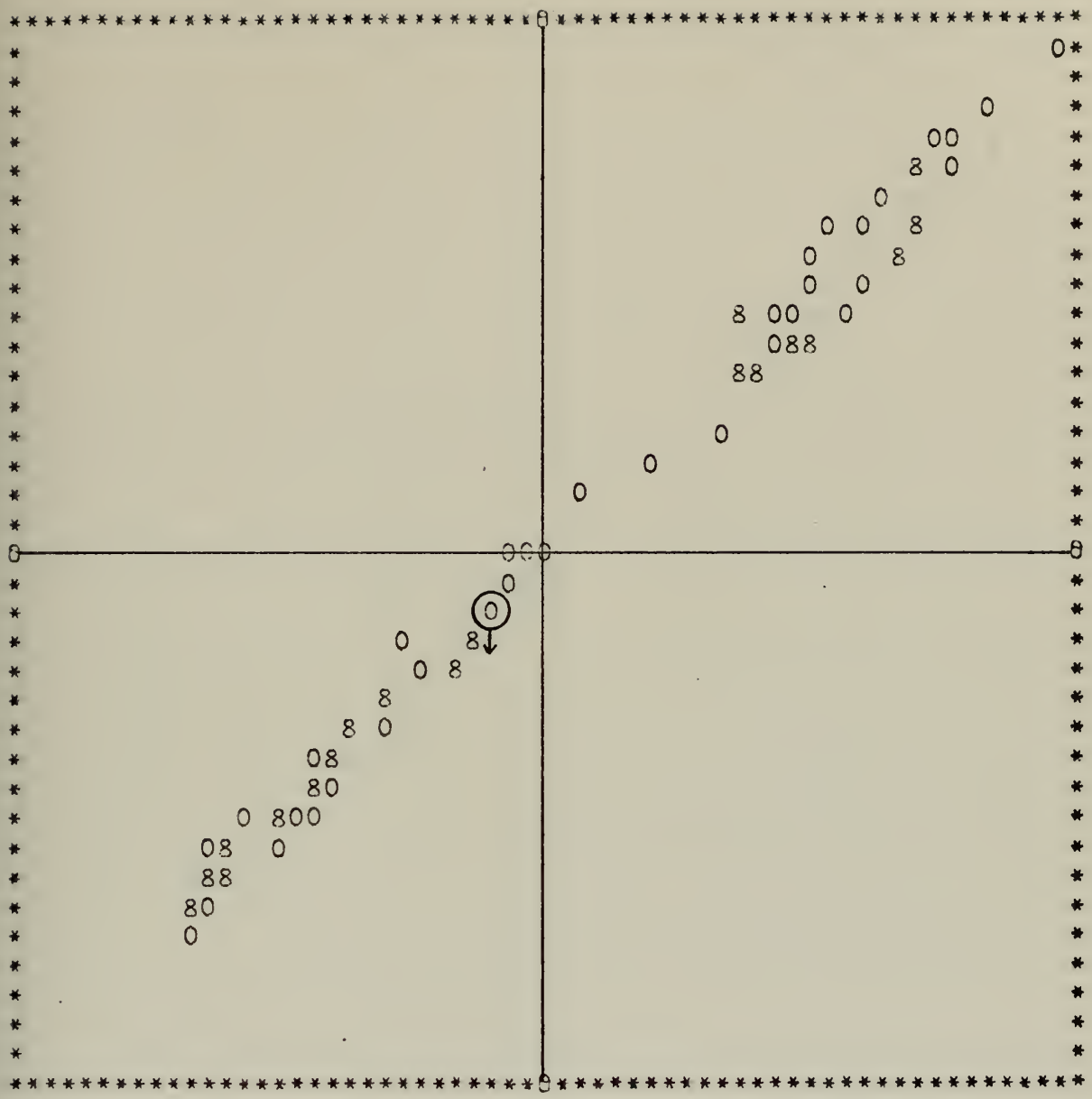


Figure VI-5-2. CASE 5, SECOND STEP

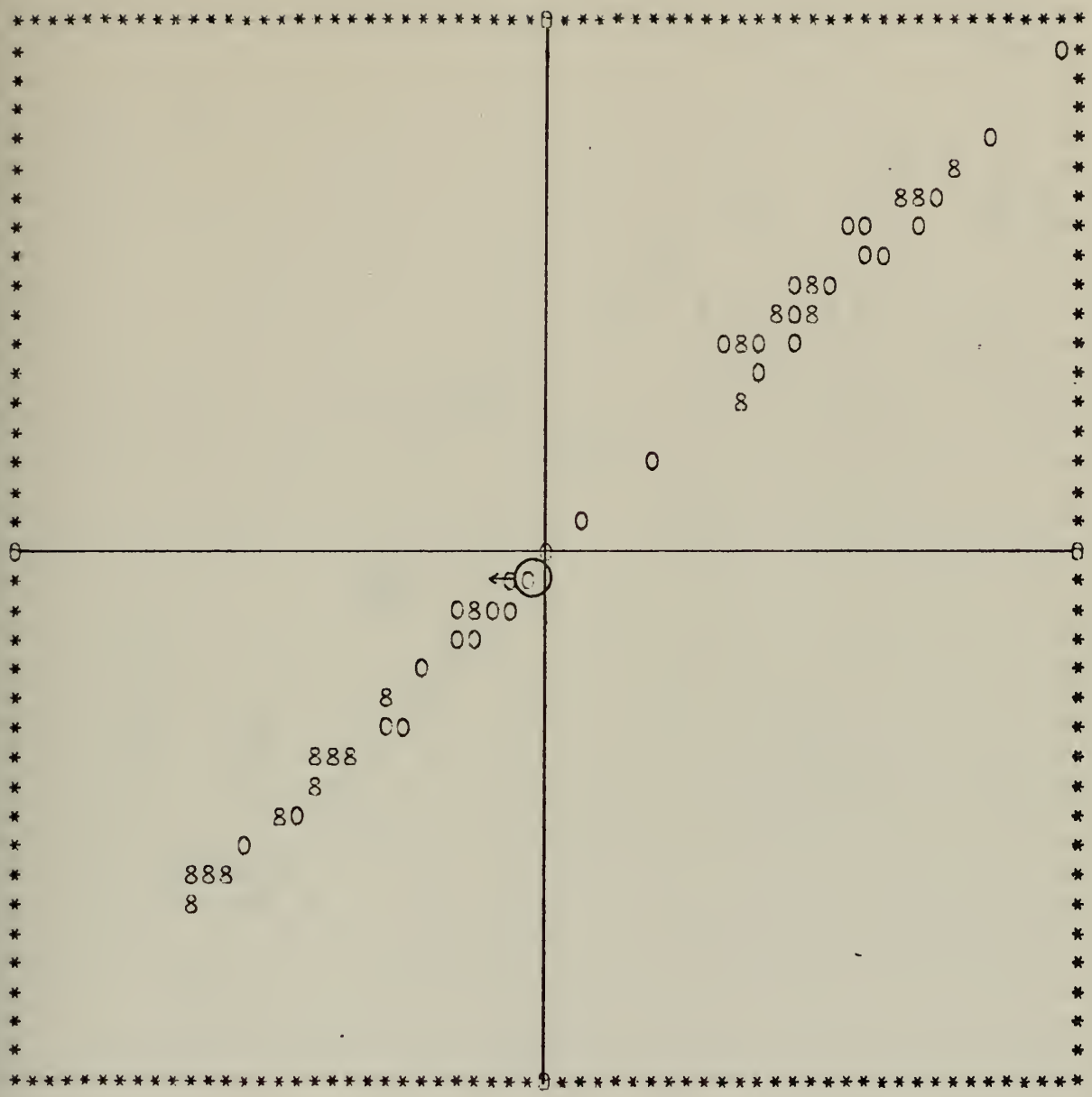


Figure VI-5-3. CASE 5, THIRD STEP

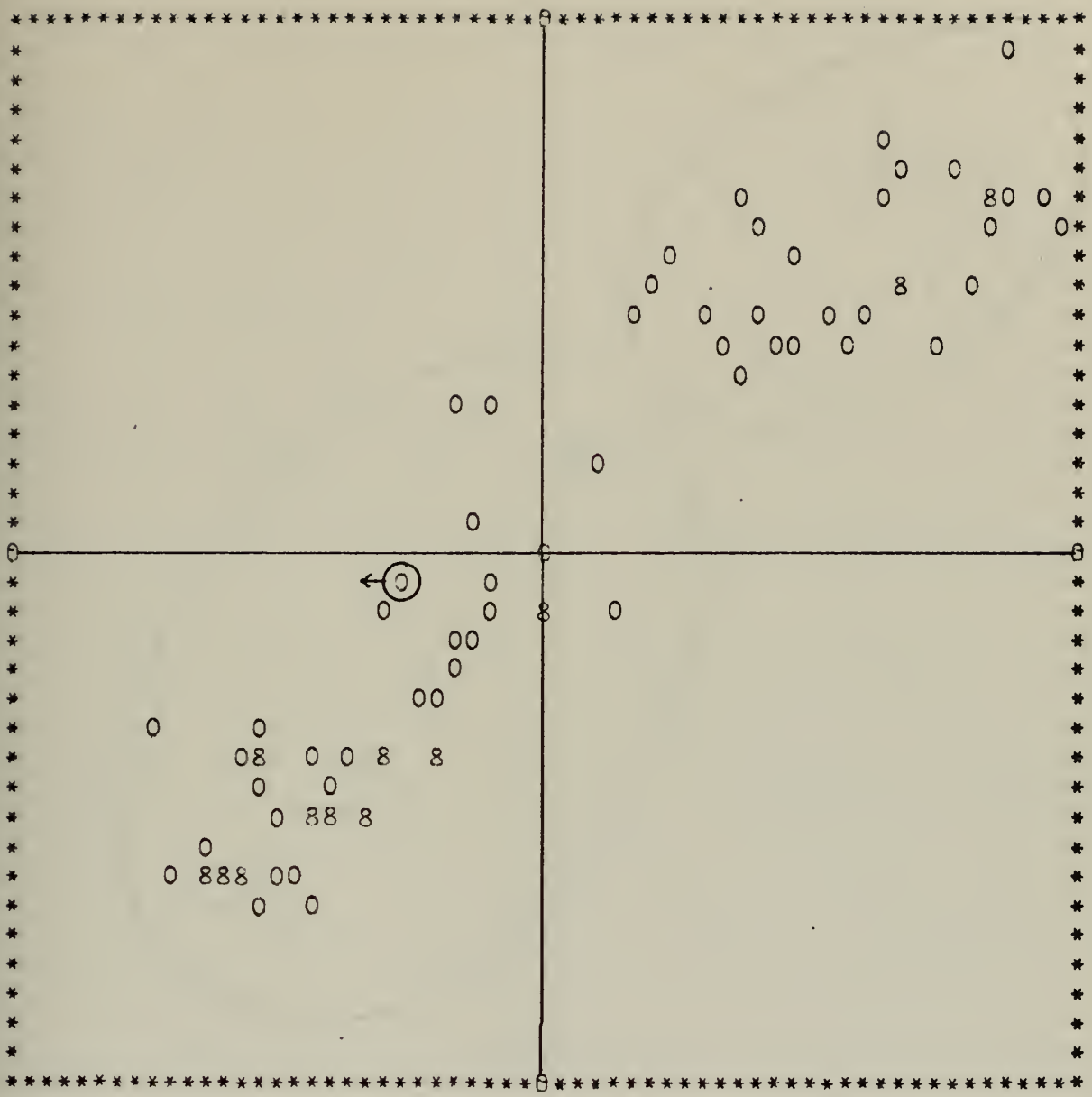


Figure VI-5-4. CASE 5, FOURTH STEP

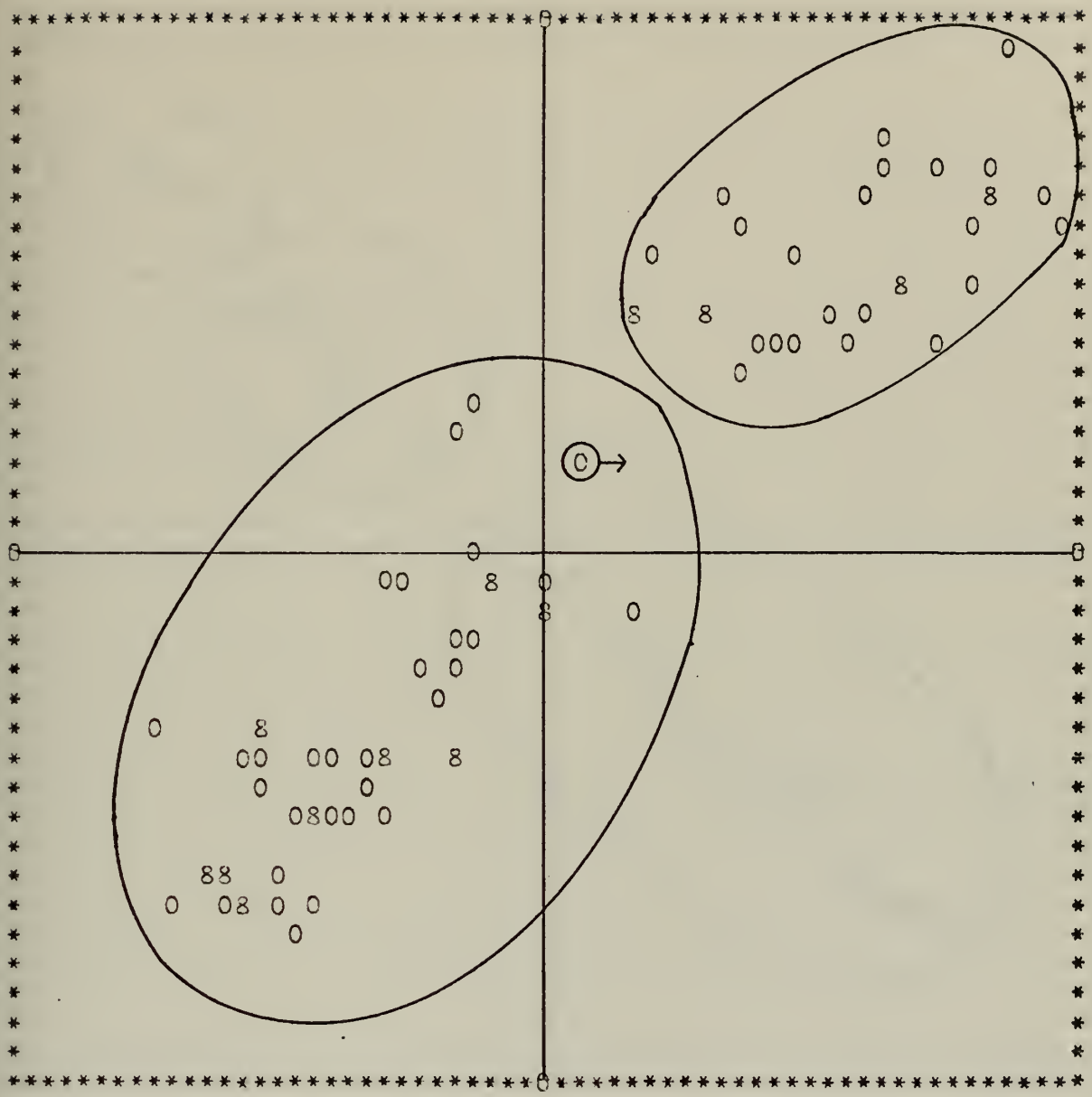


Figure VI-5-5. CASE 5, FIFTH STEP

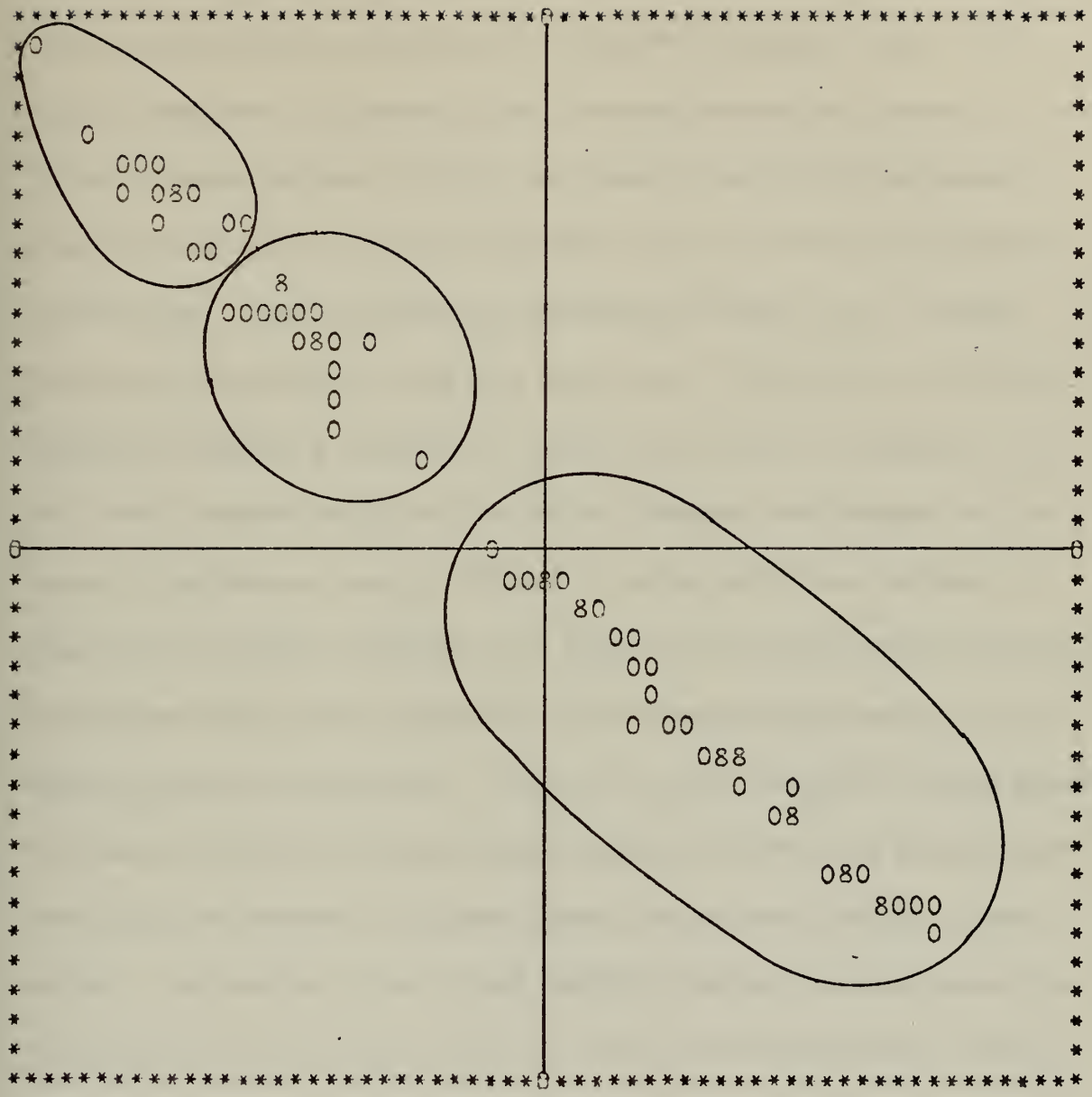


Figure VI-5-6. CASE 5, SIXTH STEP

Y point in the output is now important; this information is deducible from the point coordinates which are also part of the program output. The author found that the graphic output provided indications generally of two clusters; one consisted of points one through thirty-eight and seventy-three through eighty-two, and the other, of points thirty-nine through seventy-two. These clusters are indicated in Figure VI-5-5, which represents the situation after nine iterations. Comparing these results to those of Wright & Switzer who found three clusters, it appears that the present method failed to differentiate between their groups one and three. The clusters found by Wright & Switzer have been indicated in Figure VI-5-6 (which represents the situation after three more iterations), identifying them by the coordinates of their respective members rather than by operator ovservation. Inspection of the positions of these groupings indicates that the clustering scheme proposed by Wright & Switzer might eventually be attained or at least approached by the algorithm's application. An important point is that Wright & Switzer required ninety-six iterations to arrive at their solution; all results of the present method were found in less than thirteen iterations.

CASE 6. The second "real world" problem to be approached with this algorithm is a thirteen dimensional pattern classification problem. Each thirteen-tuple represents a set of medical readings taken from coronary patients at Letterman Hospital in the case of the Y's, and taken from personnel receiving physical examinations at Fort Ord Hospital for the X's. The Letterman patients were known to have been suffering

from coronary disease. No such concrete statement can be made for the Fort Ord examinees. For the latter it was not absolutely known that they did not have heart disease. This separation along with the appearance of the data indicated that if separation existed, it would probably not be well defined. Certain of the features in each thirteen-tuple represented measurable quantities such as age or weight. Other features such as race, medical history, and smoking habits were classifiers rather than measurements.

The application of the algorithm failed to achieve a clear cut separation between the two sets; however as the sequence of Figures VI-6-1 through VI-6-6 indicates, relatively high concentrations of Y's and X's were attained. The A and B vectors for Figure VI-6-6 are as follows:

Figure VI-6-6	A(1)= .0000	B(1)=- .9273
	A(2)=37.0000	B(2)=-12.4091
	A(3)=-20.0000	B(3)= 7.0818
	A(4)= .0000	B(4)=38.3364
	A(5)=-1.0000	B(5)= .6455
	A(6)=-2.0000	B(6)=- .6909
	A(7)=-1.0000	B(7)= .1636
	A(8)=-2.0000	B(8)=-1.4636
	A(9)= .0000	B(9)= .6636

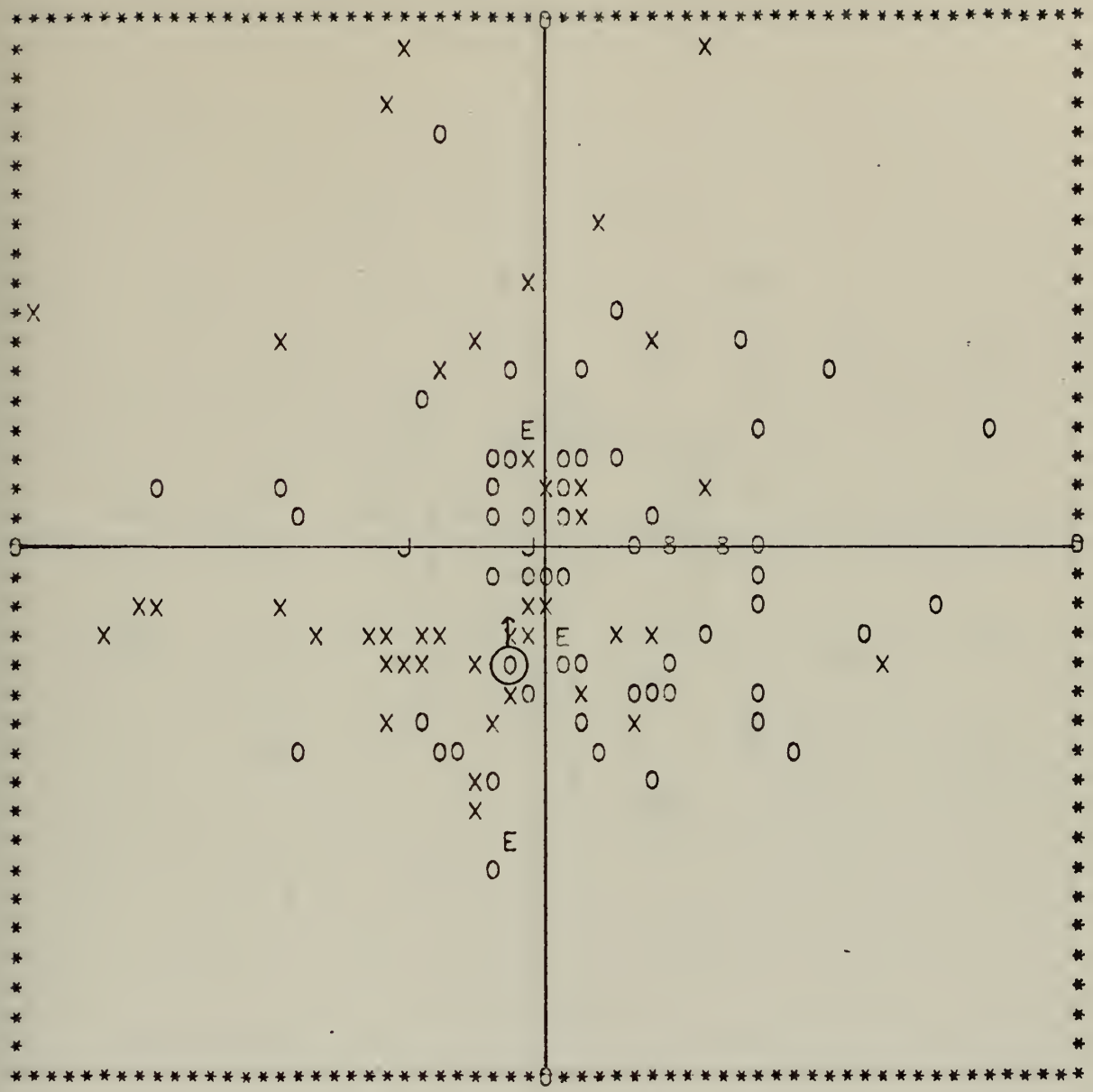


Figure VI-6-1. CASE 6, FIRST STEP

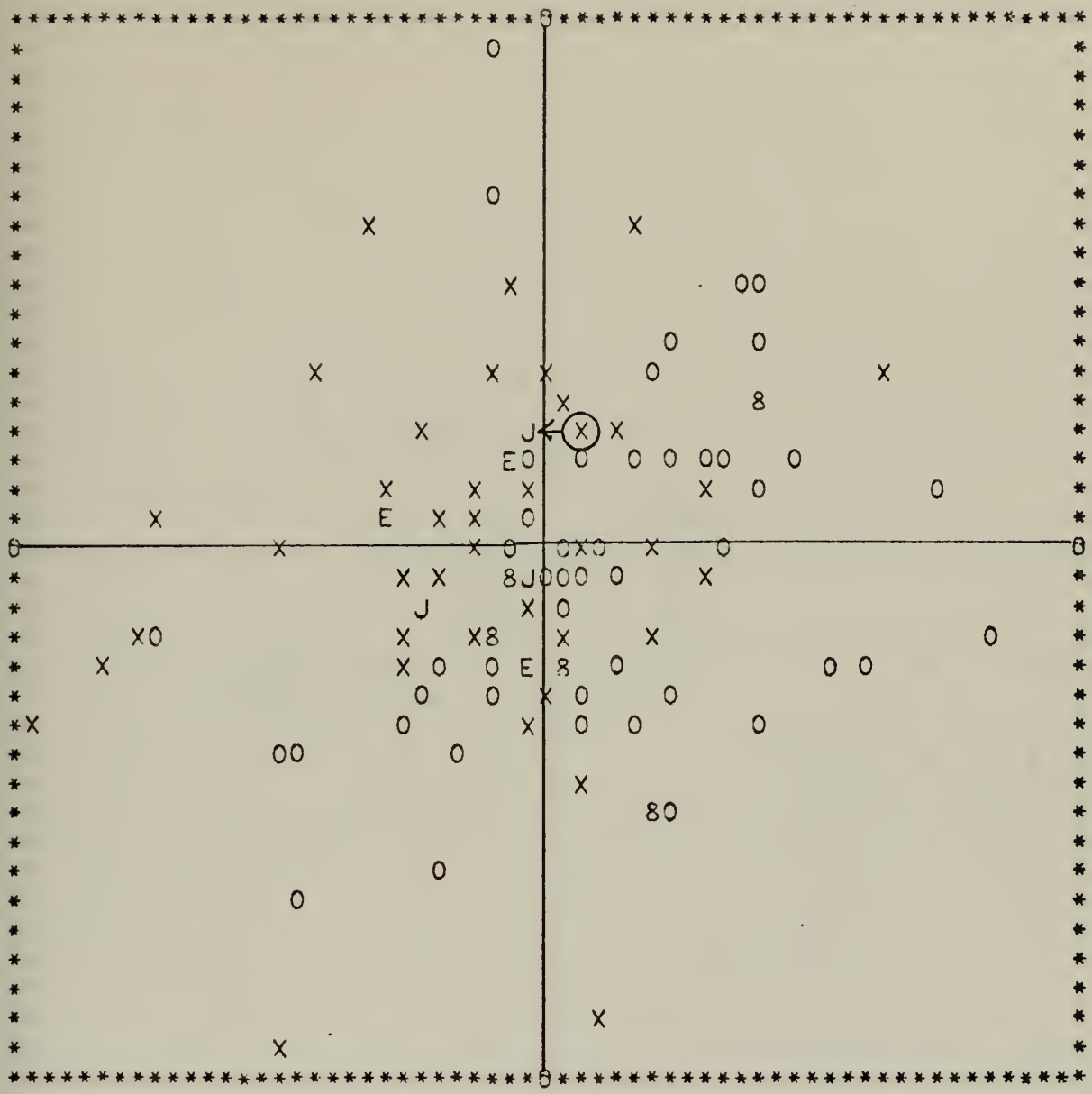


Figure VI-6-2. CASE 6, SECOND STEP

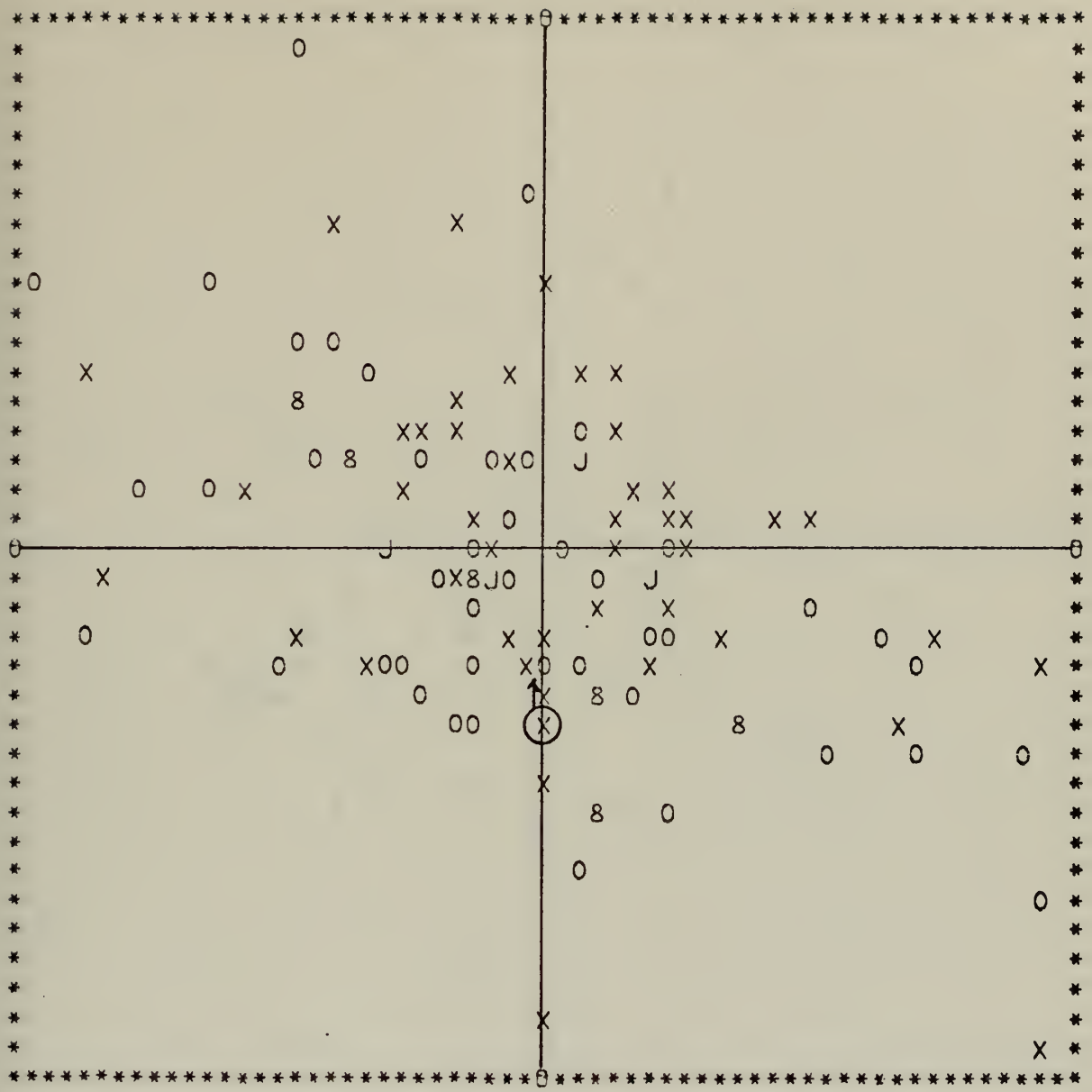


Figure VI-6-3. CASE 6, THIRD STEP

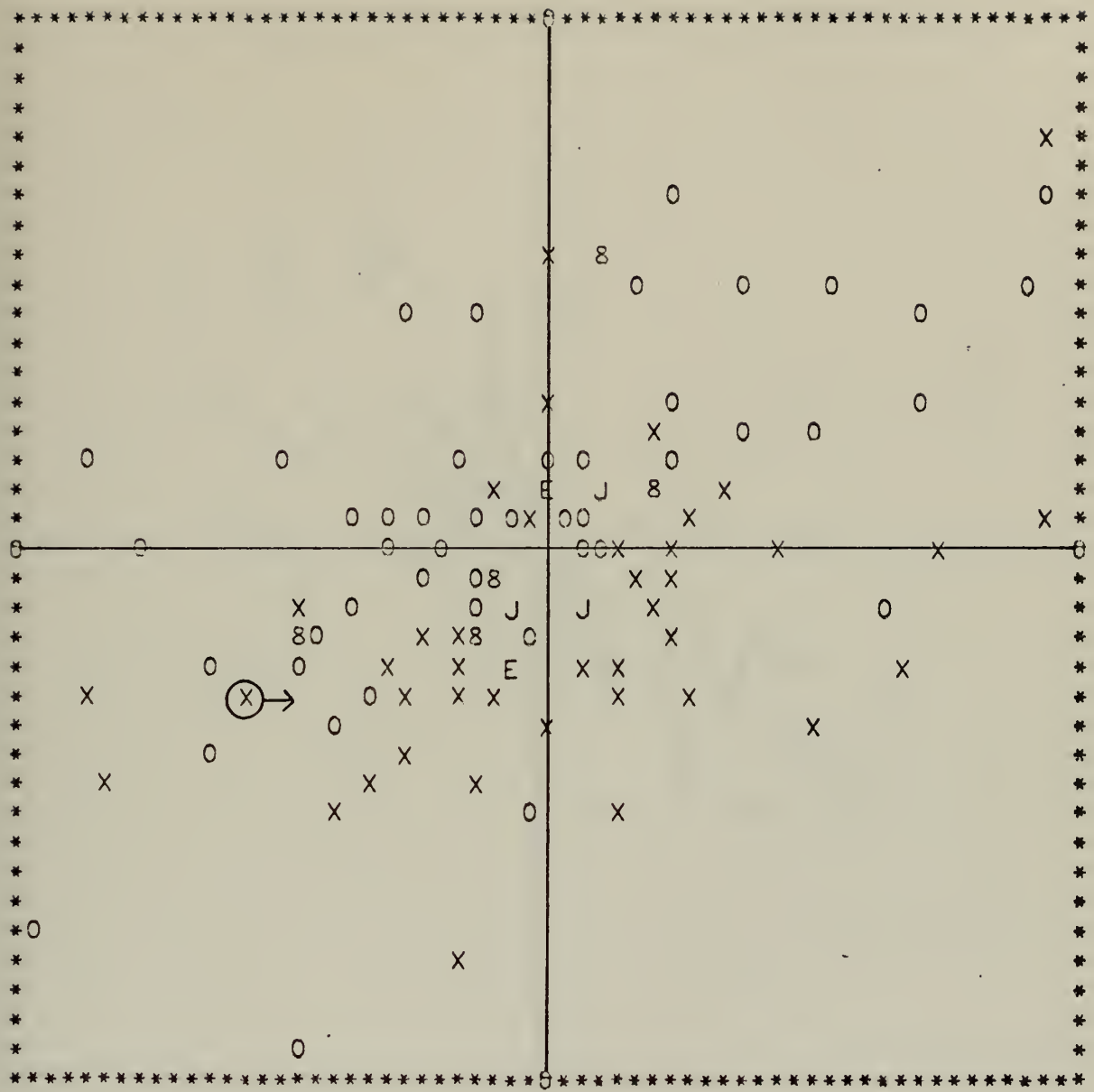


Figure VI-6-4. CASE 6, FOURTH STEP

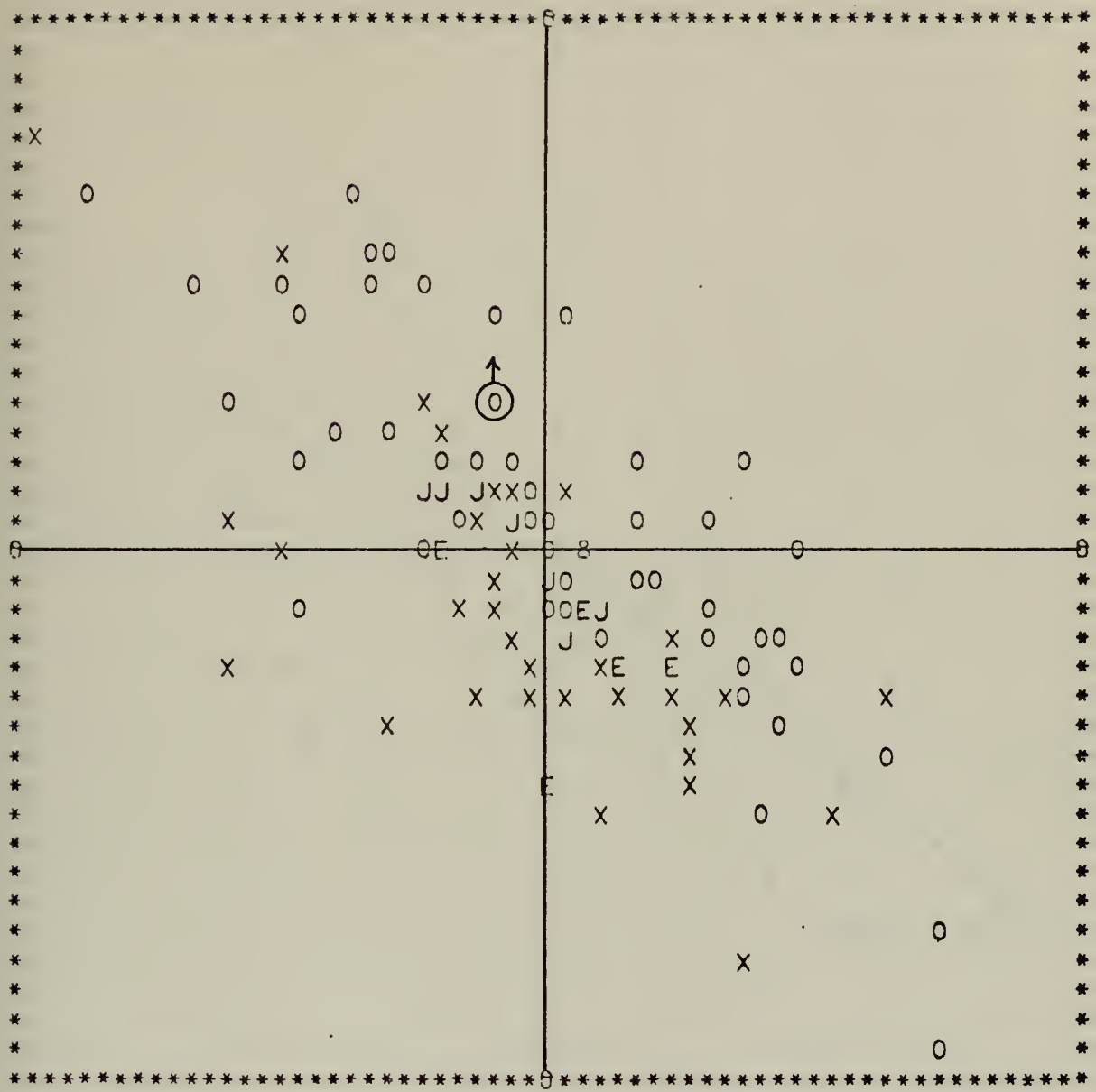


Figure VI-6-5. CASE 6, FIFTH STEP

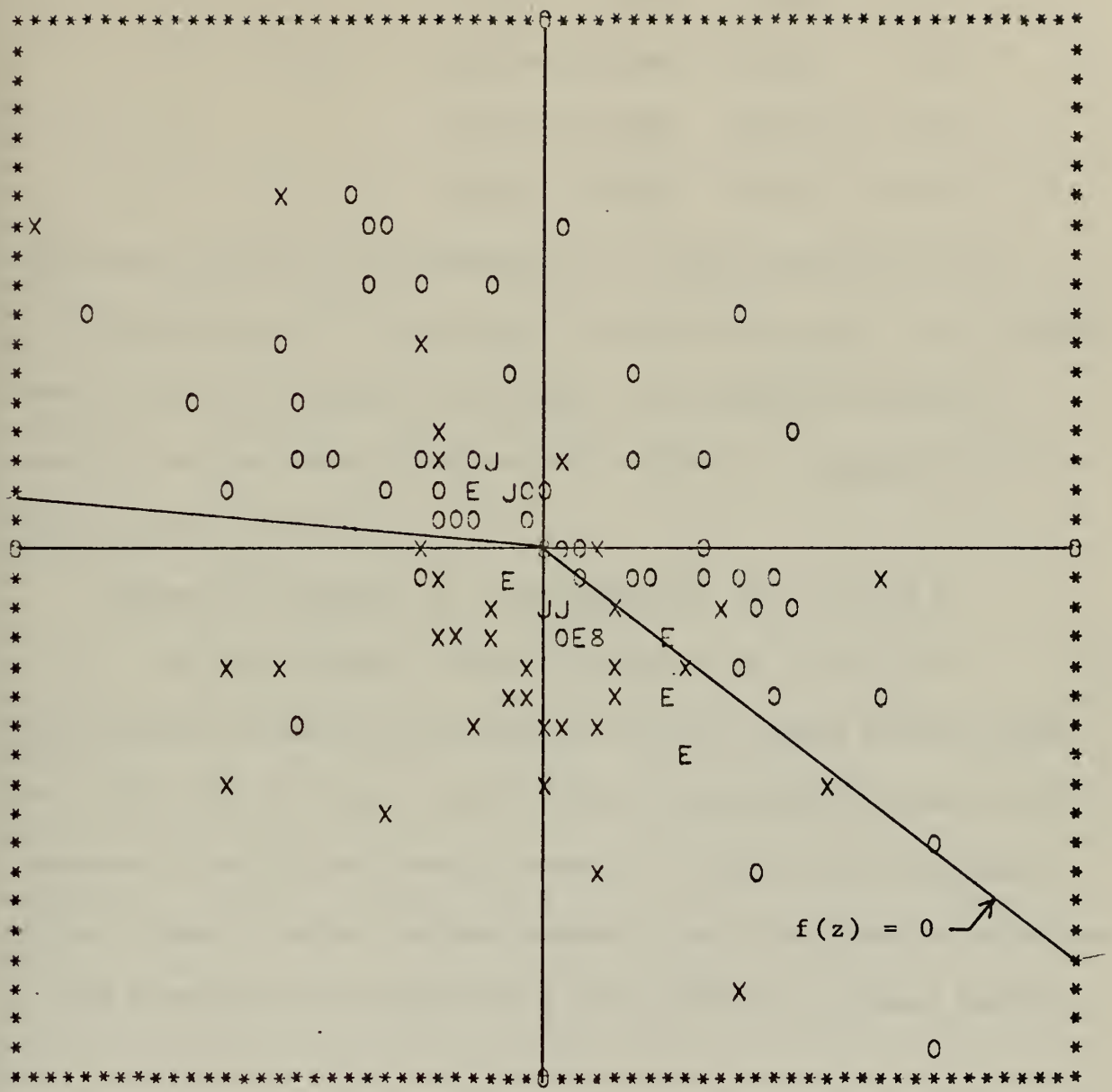


Figure VI-6-6. CASE 6, SIXTH STEP

A(10)= .0000	B(10)= .2545
A(11)=- 1.0000	B(11)= .5818
A(12)=-10.0000	B(12)= 69.0818
A(13)= 3.0000	B(13)=- 8.4455

Obviously no clear cut recommendation for a discriminant function is justified by Figure VI-6-6; however, the X data can be seen to be concentrated in the third quadrant of the figure. This information should be useful in making tentative discriminant judgements. A possible discriminant function would be:

$$f(Z)=B\cdot(Z-C)/SMAX + (A\cdot(Z-C)/TMAX)/10 \quad \text{for } A\cdot(Z-C) < 0$$

$$=B\cdot(Z-C)/SMAX + 3(A\cdot(Z-C)/TMAX)/4 \quad \text{for } A\cdot(Z-C) \geq 0$$

This function classifies thirty-seven out of fifty X points and fifty-three out of sixty Y's correctly. This result is not satisfactory as the results obtained in the first four cases; however it is probably more reasonable to use it than to conduct the great number of iterations likely to be required to find separation (if it exists) and a more suitable discriminant function.

VII. CONCLUSIONS AND RECOMMENDATIONS

Based on the results noted using the generated data sets, it can be concluded that the separation algorithm proposed by Professor Cover is satisfactorily implemented by the program written for this paper. Provided that there exists an orientation for the A-B plane in which the projections of the data points are separated, the algorithm furnishes an avenue for describing this orientation. In both real world applications such an orientation was approached, but not actually found. It is apparent that dimensionality per se is not a problem for the separation algorithm. The display for the thirteen-dimensional hospital problem was no more complex than that for the generated set of four six-dimensional cubes. Dimensionality can complicate implementation of the algorithm when the range of satisfactory A-B plane orientations is small relative to the sample space. A small range requires in general a greater number of iterations. Once separation is achieved, definition of the discriminant function is quite simple. Using the printed output of the program, the discriminant function can be drawn in and its equation found as a polynomial curve.

An important feature of this separation algorithm is that it makes no distributional assumptions or requirements on the data. This is valuable indeed when working with "real world" situations. Additionally the chain of operations on a data point are simple and easy to follow. Once a discriminant function has been defined, it is easily applied to new data points.

As a final comment, the importance of the man-machine approach of this algorithm cannot be overestimated. Increased familiarity with the data seems to reduce the number of iterations required for the achievement of separation. As the same data set was used repeatedly, it was obvious that the operator got a "feel" for the movement observed. Also the man's importance is emphasized when it is realized that he makes all decisions and judgements during the application. It was found that the choice of X and Y as the identifiers for the two classes in the problem was unfortunate. The similarity in appearance of these two letters caused some confusion in the operator. It was to alleviate some of this confusion that the labelling scheme for the output used X's and O's instead of X's and Y's.

Recommendations for future work with the separation algorithm are aimed at allowing the operator to gain a better grasp of the situation he views on the screen of the AGT. First, the video display should be changed from the present X-Y scheme to some other less confusing scheme such as an X-O display.

Second, a feature should be available in the program to indicate when specified densities of points are shown on the screen. For example, if when the density of X's in a part of the screen reaches a certain level, the video of these X's either goes out of focus or assumes a differentiable intensity.

Third, the application of the algorithm might be extended by making the projection onto a surface other than a hyperplane. This modification requires significant alteration to the work presented here.

Fourth, it might be helpful to the operator to have video in the shape of an arrow appear by the point to be moved, indicating direction of movement. Also since each member of the data set has an identifying number, it might be helpful when choosing a point to have the number of that point appear on the screen.

```
C
CC                  APPENDIX I  PROGRAM LISTING
CCC
CCCC
CCCC      SET UP OF ARRAYS, INITIALIZING
CCCC      CONSTANTS, AND DATA INPUT
C
      INTEGER ALPH(8),PIC(61,37)
      DIMENSION ITDIR(3)
      DIMENSION IGDIR(3)
      DIMENSION TOT(13),DAV(13)
      DIMENSION IMAG(301),IFRAM(301),A(13),B(13)
      DIMENSION JMAG(301),JFRAM(301)
      DIMENSION X(50,13),Y(60,13),XX1(50),XX2(50)
      DIMENSION YY1(60),YY2(60)
      DATA (ALPH(I),I=1,8)/1HO,1H8,1HX,1HE,1HJ,1H ,1H*,1HO/
      NAMELIST A,B,IDEV,IER
      IDEV=1
      CLTPUT(101) 'TYPE IDEV=2* AND C/R IF USING AGT 2'
      INPUT(101)
      CALL DTINIT(IDEV,ITDIR,3,IER)
      IF(IER.NE.0) OUTPUT(101) IER,'DTINIT'
      TT=0.01
      M=50
      N=60
      K=13
      READ(5,2187)((Y(I,J),J=1,K),I=1,N)
      IF(M.EQ.0) GO TO 167
      READ(5,2187)((X(I,J),J=1,K),I=1,M)
 2187 FORMAT(F4.1,12F5.0)
C
CC    CENTRALIZING DATA
C
  167 DO 170 J=1,K
      TOT(J)=0.0
  170 CONTINUE
      DO 171 II=1,N
      DO 172 JJ=1,K
      TOT(JJ)=TOT(JJ)+Y(II,JJ)
  172 CONTINUE
  171 CONTINUE
      DO 173 II=1,M
      DO 174 JJ=1,K
      TOT(JJ)=TOT(JJ)+X(II,JJ)
  174 CONTINUE
  173 CONTINUE
      DEN=FLOAT(M+N)
      DO 175 JA=1,K
      DAV(JA)=TOT(JA)/DEN
      WRITE(6,181) JA,DAV(JA)
  181 FORMAT(5X,'DATA COMP. AVE NO.',I3,'=',F10.3)
  175 CONTINUE
      DO 176 IJ=1,N
      DO 177 IK=1,K
      Y(IJ,IK)=Y(IJ,IK)-DAV(IK)
  177 CONTINUE
  176 CONTINUE
      DO 178 IJ=1,M
      DO 179 IK=1,K
      X(IJ,IK)=X(IJ,IK)-DAV(IK)
  179 CONTINUE
  178 CONTINUE
      WRITE(6,190)
  190 FORMAT(5X,'CENTRALIZED DATA',/)
      DO 191 IJ=1,N
      DO 192 IK=1,K
      WRITE(6,193) IJ,IK,Y(IJ,IK)
  193 FORMAT(5X,'Y(',I3,',',I3,')=',F10.3)
  192 CONTINUE
      WRITE(6,194)
  194 FORMAT(5X,/)
  191 CONTINUE
```

```
      DC 195 IJ=1,M
      DO 196 IK=1,K
      WRITE(6,197) IJ,IK,X(IJ,IK)
 197  FCRMAT(5X,'X(',I3,',',I3,')=',F10.3)
 196  CONTINUE
      WRITE(6,194)
 195  CONTINUE
      WRITE(6,200)
 200  FCRMAT(5X,///)
C
C     INPUT COMPONENT VALUES FOR THE A AND B VECTCRS
C
  7   CALL GINPUT(IDEV,ITDIR,1)
      DC 30 JJ=1,K
      WRITE(6,450) JJ,A(JJ),JJ,B(JJ)
 450  FORMAT(5X,'A(',I3,')=',F6.3,' ;B(',I3,')=',F6.3)
 30   CONTINUE
      KST=0
      KRT=0
 10   CALL DGINIT (IDEV,IGDIR,3,IER)
      IF(IER.NE.0) OUTPUT (101) IER,'DGINIT'
C
C     CALCULATION OF DATA POINT PROJECTIONS
C
      SMAX=0.0
      TMAX=0.0
      SUMY1=0.0
      SUMY2=0.0
      DC 12 I=1,N
      YY1(I)=C.0
      YY2(I)=0.0
      DC 15 J=1,K
      YY1(I)=YY1(I)+A(J)*Y(I,J)
      YY2(I)=YY2(I)+B(J)*Y(I,J)
 15   CONTINUE
      SUMY1=SUMY1+YY1(I)
      SUMY2=SUMY2+YY2(I)
 12   CONTINUE
      SUMX1=0.0
      SUMX2=0.0
      DC 14 L=1,M
      XX1(L)=0.0
      XX2(L)=0.0
      DC 17 J=1,K
      XX1(L)=XX1(L)+A(J)*X(L,J)
      XX2(L)=XX2(L)+B(J)*X(L,J)
 17   CONTINUE
      SUMX1=SUMX1+XX1(L)
      SUMX2=SUMX2+XX2(L)
 14   CONTINUE
      AV1=(SUMY1+SUMX1)/FLOAT(M+N)
      AV2=(SUMY2+SUMX2)/FLOAT(M+N)
      WRITE(6,182) AV1,AV2
 182  FORMAT(5X,'SCAL. AVE 1=',F10.6,'  SCAL. AVE 2=',F10.6)

C     SCALING THE PROJECTIONS
C
      DC 11 I=1,N
      YY1(I)=YY1(I)-AV1
      YY2(I)=YY2(I)-AV2
      GA=ABS(YY1(I))
      GB=ABS(YY2(I))
      IF(GA.GT.TMAX) TMAX=GA
      IF(GB.GT.SMAX) SMAX=GB
 11   CONTINUE
      DC 13 L=1,M
      XX1(L)=XX1(L)-AV1
      XX2(L)=XX2(L)-AV2
      ZA=ABS(XX1(L))
      ZB=ABS(XX2(L))
      IF(ZA.GT.TMAX) TMAX=ZA
      IF(ZB.GT.SMAX) SMAX=ZB
```

```
 13   CONTINUE
      KN=5*N
      KK=KN+1
      JL=0
      IFRAM(1)=IHEAD(0,10)
      WRITE(6,6667) TMAX,SMAX
 6667 FORMAT(5X,'TMAX=',F10.6,'   SMAX=',F10.6)
C
C     DRAWING THE Y POINTS ON THE AGT SCREEN
C
      DO 47 I=2,KN,5
      JL=JL+1
      FA=YY1(JL)/TMAX
      FB=YY2(JL)/SMAX
      WRITE(6,5001) JL,FA,JL,FB
      FC=FA+TT
      FD=FA-TT
      FE=FB+TT
      FF=FB-TT
      IFRAM(I)=IPACK(FA,FB,0)
      IFRAM(I+1)=IPACK(FC,FE,1)
      IFRAM(I+2)=IPACK(FA,FF,0)
      IFRAM(I+3)=IPACK(FA,FB,1)
      IFRAM(I+4)=IPACK(FD,FE,1)
 47   CONTINUE
      CALL GRAPHO (IDEV,IFRAM,KK,1,IER)
      IF(IER.NE.0) OUTPUT (101) IER,'GBLK1'
      IN=6*M
      IF(IN.EQ.0) GO TO 55
      JJ=6*M+1
      JN=0
      IMAG(1)=IHEAD(0,10)
C
C     DRAWING THE X POINTS ON THE AGT SCREEN
C
      DO 45 I=2,IN,6
      JN=JN+1
      TA=XX1(JN)/TMAX
      TB=XX2(JN)/SMAX
      WRITE(6,5000) JN,TA,JN,TB
      TC=TA+TT
      TD=TA-TT
      TE=TB+TT
      TF=TB-TT
      IMAG(I)=IPACK(TA,TB,0)
      IMAG(I+1)=IPACK(TC,TE,1)
      IMAG(I+2)=IPACK(TC,TF,0)
      IMAG(I+3)=IPACK(TD,TE,1)
      IMAG(I+4)=IPACK(TD,TF,0)
      IMAG(I+5)=IPACK(TA,TB,1)
 45   CONTINUE
      CALL GRAPHO (IDEV,IMAG,JJ,2,IER)
      IF(IER.NE.0) OUTPUT(101) IER,'GBLK2'
C
C     CHECKING FOR DEPRESSED FUNCTION SWITCHES
C
 55   CALL FNS(IDEV,ISW,IER)
      IF(IER.NE.0) OUTPUT(101) IER,'GBLK3'
      IF(ISW.GE.0) GO TO 55
      IF(LLS(ISW,11).LT.0) GO TO 7
      IF(LLS(ISW,3).LT.0) GO TO 57
      IF(LLS(ISW,13).LT.0) GO TO 58
      IF(LLS(ISW,14).LT.0) GO TO 67
      IF(LLS(ISW,15).LT.0) GO TO 68
      IF(KRT.EQ.1) GO TO 90
      IF(KST.EQ.1) GO TO 91
      IF(LLS(ISW,16).LT.0) GO TO 55
      IF(LLS(ISW,17).LT.0) GO TO 55
      IF(LLS(ISW,18).LT.0) GO TO 55
      GO TO 55
C
C     CHOICE OF DIRECTION OF MOVE WITH Y PICKED
```

```
C
 90   CALL FNS(IDEV,ISW,IER)
      IF(IER.NE.0) OUTPUT(101) IER,'GBLK7'
      IF(ISW.GE.0) GO TO 55
      IF(LLS(ISW,5).LT.0) GO TO 59
      IF(LLS(ISW,6).LT.0) GO TO 60
      IF(LLS(ISW,7).LT.0) GO TO 61
      IF(LLS(ISW,8).LT.0) GO TO 62
      GO TO 55
C
C     CHOICE OF DIRECTION OF MOVE WITH X PICKED
C
 91   CALL FNS(IDEV,ISW,IER)
      IF(IER.NE.0) OUTPUT(101) IER,'GBLK8'
      IF(ISW.GE.0) GO TO 55
      IF(LLS(ISW,5).LT.0) GO TO 63
      IF(LLS(ISW,6).LT.0) GO TO 64
      IF(LLS(ISW,7).LT.0) GO TO 65
      IF(LLS(ISW,8).LT.0) GO TO 66
      GO TO 55
C
C     PICKING A Y
C
 57   CALL GRAPHR(IDEV,IFRAM,KK,1,IER)
      IF(IER.NE.0) OUTPUT(101) IER,'GBLK3'
 151  IF(MOD(IGDIR(1),8).EQ.0) GO TO 151
      CALL GRAPHI(IDEV,JFRAM,1,IER)
      IF(IER.NE.0) OUTPUT(101) IER,'IGBLK'
      KEDS=0
      KST=0
      KRT=1
      KKK=0
      DO 600 I=2,KN,5
      KKK=KKK+1
      IF(JFRAM(I).NE.IFRAM(I)) GO TO 601
      IF(JFRAM(I+1).NE.IFRAM(I+1)) GO TO 601
      IF(JFRAM(I+2).NE.IFRAM(I+2)) GO TO 601
      IF(JFRAM(I+3).NE.IFRAM(I+3)) GO TO 601
      IF(JFRAM(I+4).NE.IFRAM(I+4)) GO TO 601
      GO TO 600
 601  JED=KKK
      GO TO 55
 600  CONTINUE
C
C     PICKING AN X
C
 58   CALL GRAPHR(IDEV,IMAG,JJ,2,IER)
      IF(IER.NE.0) OUTPUT(101) IER,'GBLK4'
 152  IF(MOD(IGDIR(2),8).EQ.0) GO TO 152
      CALL GRAPHI(IDEV,JMAG,2,IER)
      IF(IER.NE.0) OUTPUT(101) IER,'JGBLK'
      KEDS=0
      KST=1
      KRT=0
      LLL=0
      DO 610 I=2,IN,6
      LLL=LLL+1
      IF(JMAG(I).NE.IMAG(I)) GO TO 611
      IF(JMAG(I+1).NE.IMAG(I+1)) GO TO 611
      IF(JMAG(I+2).NE.IMAG(I+2)) GO TO 611
      IF(JMAG(I+3).NE.IMAG(I+3)) GO TO 611
      IF(JMAG(I+4).NE.IMAG(I+4)) GO TO 611
      IF(JMAG(I+5).NE.IMAG(I+5)) GO TO 611
      GO TO 610
 611  LED=LLL
      GO TO 55
 610  CONTINUE
C
C     ITERATION BY MOVING Y TO THE RIGHT
C
 59   IF(JED.EQ.0) GO TO 55
      WRITE(6,2000) JED
```

```
 2000 FORMAT(5X,'MOVING RIGHT USING Y(',I3,')')
      DO 110 KJ=1,K
      A(KJ)=A(KJ)+Y(JED,KJ)
      WRITE(6,1000) KJ,A(KJ)
 110  CONTINUE
      KEDS=1
      GO TO 10
C
C     ITERATION BY MOVING Y TO THE LEFT
C
 60   IF(JED.EQ.0) GO TO 55
      WRITE(6,2001) JED
 2001 FORMAT(5X,'MOVING LEFT USING Y(',I3,')')
      DO 111 KJ=1,K
      A(KJ)=A(KJ)-Y(JED,KJ)
      WRITE(6,1000) KJ,A(KJ)
 111  CONTINUE
      KEDS=2
      GO TO 10
C
C     ITERATION BY MOVING Y UP
C
 61   IF(JED.EQ.0) GO TO 55
      WRITE(6,2002) JED
 2002 FORMAT(5X,'MOVING UP USING Y(',I3,')')
      DO 112 KJ=1,K
      B(KJ)=B(KJ)+Y(JED,KJ)
      WRITE(6,1001) KJ,B(KJ)
 112  CONTINUE
      KEDS=3
      GO TO 10
C
C     ITERATION BY MOVING Y DOWN
C
 62   IF(JED.EQ.0) GO TO 55
      WRITE(6,2003) JED
 2003 FORMAT(5X,'MOVING DOWN USING Y(',I3,')')
      DO 113 KJ=1,K
      B(KJ)=B(KJ)-Y(JED,KJ)
      WRITE(6,1001) KJ,B(KJ)
 113  CONTINUE
      KEDS=4
      GO TO 10
C
C     ITERATION BY MOVING X TO THE RIGHT
C
 63   IF(LED.EQ.0) GO TO 55
      WRITE(6,3000) LED
 3000 FORMAT(5X,'MOVING RIGHT USING X(',I3,')')
      DO 120 KD=1,K
      A(KD)=A(KD)+X(LED,KD)
      WRITE(6,1000) KD,A(KD)
 120  CONTINUE
      KEDS=5
      GO TO 10
C
C     ITERATION BY MOVING X TO THE LEFT
C
 64   IF(LED.EQ.0) GO TO 55
      WRITE(6,3001) LED
 3001 FORMAT(5X,'MOVING LEFT USING X(',I3,')')
      DO 121 KD=1,K
      A(KD)=A(KD)-X(LED,KD)
      WRITE(6,1000) KD,A(KD)
 121  CONTINUE
      KEDS=6
      GO TO 10
C
C     ITERATION BY MOVING X UP
C
 65   IF(LED.EQ.0) GO TO 55
      WRITE(6,3002) LED
```

```
 3002 FORMAT(5X,'MOVING UP USING X(',I3,')')
      DO 122 KD=1,K
      B(KD)=B(KD)+X(LED,KD)
      WRITE(6,1001) KD,B(KD)
 122  CONTINUE
      KEDS=7
      GO TO 10
C
C     ITERATION BY MOVING X DOWN
C
 66   IF(LED.EQ.0) GO TO 55
      WRITE(6,3003) LED
 3003 FORMAT(5X,'MOVING DOWN USING X(',I3,')')
      DO 123 KD=1,K
      B(KD)=B(KD)-X(LED,KD)
      WRITE(6,1001) KD,B(KD)
 123  CONTINUE
      KEDS=8
      GO TO 10
C
C     PLOT OUTPUT ON LINE PRINTER
C
 67   DO 825 I=2,60
      DO 827 J=2,36
      PIC(I,J)=ALPH(6)
 827  CONTINUE
 825  CONTINUE
      DO 704 J=1,18
      PIC(1,J)=ALPH(7)
      PIC(61,J)=ALPH(7)
 704  CONTINUE
      PIC(1,19)=ALPH(8)
      PIC(61,19)=ALPH(8)
      DO 705 J=20,37
      PIC(1,J)=ALPH(7)
      PIC(61,J)=ALPH(7)
 705  CONTINUE
      DO 714 I=1,30
      PIC(I,1)=ALPH(7)
      PIC(I,37)=ALPH(7)
 714  CONTINUE
      PIC(31,1)=ALPH(8)
      PIC(31,37)=ALPH(8)
      DO 715 I=32,61
      PIC(I,1)=ALPH(7)
      PIC(I,37)=ALPH(7)
 715  CONTINUE
      DO 777 JL=1,N
      JH=(YY2(JL)/SMAX)*17.0+19.0
      KH=(YY1(JL)/TMAX)*29.0+31.0
      IF(KH.GT.60) KH=60
      IF(JH.GT.36) JH=36
      JH=38-JH
      IF(PIC(KH,JH).NE.ALPH(6)) PIC(KH,JH)=ALPH(2)
      IF(PIC(KH,JH).EQ.ALPH(6)) PIC(KH,JH)=ALPH(1)
 777  CONTINUE
      DO 888 JN=1,M
      LC=(XX1(JN)/TMAX)*29.0+31.0
      LD=(XX2(JN)/SMAX)*17.0+19.0
      IF(LC.GT.60) LC=60
      IF(LD.GT.36) LD=36
      LD=38-LD
      IF(PIC(LC,LD).EQ.ALPH(6)) GO TO 890
      IF(PIC(LC,LD).EQ.ALPH(1)) PIC(LC,LD)=ALPH(5)
      IF(PIC(LC,LD).EQ.ALPH(2)) PIC(LC,LD)=ALPH(5)
      IF(PIC(LC,LD).EQ.ALPH(3)) PIC(LC,LD)=ALPH(4)
      GO TO 888
 890  PIC(LC,LD)=ALPH(3)
 888  CONTINUE
      WRITE(6,751)
 751  FORMAT('1')
      DO 895 J=1,37
```

```
      WRITE(6,896) (PIC(I,J),I=1,61)
  896 FORMAT(20X,61A1)
  895 CONTINUE
      WRITE(6,751)
      GO TO 55
C
C     REGRESS TO THE PREVIOUS ITERATION
C
   68 WRITE(6,897)
  897 FORMAT(5X,'REGRESSING TO LAST STEP')
      IF(KEDS.EQ.1) GO TO 60
      IF(KEDS.EQ.2) GO TO 59
      IF(KEDS.EQ.3) GO TO 62
      IF(KEDS.EQ.4) GO TO 61
      IF(KEDS.EQ.5) GO TO 64
      IF(KEDS.EQ.6) GO TO 63
      IF(KEDS.EQ.7) GO TO 66
      IF(KEDS.EQ.8) GO TO 65
      GO TO 55
 1000 FORMAT(5X,'NEW VALUE OF A(',I3,')=',F10.4)
 1001 FORMAT(5X,'NEW VALUE OF B(',I3,')=',F10.4)
 5000 FORMAT(5X,'XX1(',I3,')=',F10.4,'XX2(',I3,')=',F10.4)
 5001 FORMAT(5X,'YY1(',I3,')=',F10.4,'YY2(',I3,')=',F10.4)
  183 STOP
      END
C
C     PREPREPARED SUBROUTINE FOR DATA INPUT AT THE AGT
C
$


      SUBROUTINE GINP%IDEV,ITDIR,IBLK,IBUF<
      DIMENSION IBUF%1<,ITDIR%1<
      IB#IBLK&1
      NULL#-1
      IF%IBUF%1<<1,50,100
1     IF%IBUF%1<.NE.-1<GO TO 100
      ENCODE%16,10,IBUF<
10    FORMAT%@NAMELIST INPUT@<
      CALL TEXTO%IDEV,IBUF,4,38,1,1,3,IER<
      IF%IER.NE.0<OUTPUT%101<IER,@GINP1@
      RETURN
50    CALL TEXTO%IDEV,NULL,1,38,1,1,3,IER<
      IF%IER.NE.0<OUTPUT%101<IER,@NULL1@
      CALL TEXTO%IDEV,NULL,1,40,4,1,3,IER<
      IF%IER.NE.0<OUTPUT%101<IER,@NULL2@
      RETURN
100   CALL TEXTR%IDEV,NULL,1,40,4,1,3,IER<
      IF%IER.NE.0<OUTPUT%101<IER,@GINP2@
110   IF%MOD%ITDIR%IB<,8<.EQ.0<GO TO 110
      CALL TEXTI%IDEV,IBUF,24,0,IB,IER<
      IF%IER.NE.0<OUTPUT%101<IER,@GINP3@
      RETURN
      END
$
C
C     PREPREPARED PROGRAMS  IN MACHINE LANGUAGE FOR
C     THE USE OF THE LIGHTPEN AND THE FUNCTION SWITCHES
C
¬META9300 SI,LO,GO
$
$FNS    PZE     0
        BRM     9SETUPN
        PZE     3
IDEV    PZE     0
ISW     PZE     0
IER     PZE     0
        LDA     *IDEV
        BRM     DVNOCK
        BRU     FNSDER
        STA     FNSCL&2
        STA     FIOBFL
```

```
        LDA     #C1000000
        STA     FNSSWO
        ECM     032004
FNSCL   BRM     D>EXEC
        PZE     2
        PZE     0
        PZE     FNSSWO
        ECM     032001
        ECM     032002
        LDA     *IDEV
        SKA     IOBFL
        BRU     $-1
        SKE     #1
        BRU     $&3
        LDA     077766
        BRU     $&2
        LDA     077771
        STA     *ISW
        LDA     #0
        BRU     $&2
FNSDER  LDA     #1
FNSER   STA     *IER
        BRR     FNS
FNSSWO  PZE     0
        PZE     0
        PZE     0
        END
$
¬ECF
¬META93CC SI,LO,GO
$
$GINPUT PZE     0
        BRM     9SETUPN
        PZE     3
IDEV    PZE     0
ITDIR   PZE     0
IBLK    PZE     0
        LDP     IDEV
        STD     TEXTO&3
        LDA     *IBLK
        STA     BLOCK
        LDA     INPADR
        ADD     BUF
        STA     IBUF
        LDA     R>ONES
        STA     *IBUF
        BRM     TEXTO
        PZE     0
        LDA     INPADR
        ADD     READ
        STA     PATCH
        LDA     BRM
        XMA     *PATCH
        STA     BRM
        LDA     #101
        STA     *IBUF
INPADR  BRM     9INPUT
        STZ     *IBUF
        BRM     TEXTO
        PZE     0
        LDA     BRM
        XMA     *PATCH
        STA     BRM
        BRR     GINPUT
*
*
*
TEXTC   PZE     0
        BRM     GINP
        PZE     4
        PZE     0
        PZE     0
```

```
         PZE      BLOCK
IBUF     PZE      0
         MPO      TEXTO
         BRR      TEXTO
*
*
*
BLOCK    PZE      0
PATCH    PZE      0
BRM      BRM      TEXTO
READ     DATA     0563
BUF      DATA     0773
         END
$
¬EOF
¬LOAD XR
¬DATA
```

APPENDIX II. PROGRAM OPERATING INSTRUCTIONS

To ready the program for a set of data a total of eight cards must be prepared and inserted into the deck. For the purposes of this appendix, assume that there are GG Y points and HH X points. Furthermore, assume all points are of dimension II. Then the required cards are as listed below.

Four dimension cards are placed immediately after the card which reads "DIMENSION IGDIR(3)"; this is the third card in the program. These cards should read:

```
DIMENSION DAV(II), TOT(II)
DIMENSION IMAG(6*HH+1),IFRAM(5*GG+1),A(II),B(II)
DIMENSION X(HH,II),Y(GG,II),XX1(HH),XX2(HH),
     YY1(GG),YY2(GG)
DIMENSION JMAG(6*HH+1),JFRAM(5*GG+1)
```

Next following card number fifteen of the program which reads "TT=0.01", the cards to be entered are:

```
M=HH
N=GG
K=II
```

Finally a format statement must be inserted into the program. The number of the statement is 2187; further guidance is not practical since the format statement is used in connection with the formatted input of the

data. As an example, this statement might read "2187 FORMAT(4F10.6). The statement must correspond to the format of the data cards. The data cards are placed behind the program, and the deck is ready to run.

Instructions for operating the XDS9300 computer and the ADAGE AGT console are contained in the publication entitled "ELECTRICAL ENGINEERING COMPUTER LABORATORY" by R. D. Delaura [2] . Familiarity with this document is a must before operating these computers. The following instructions are a guide for this particular program.

1. Energize XDS9300
   a. Depress "RESET" and "POWER" simultaneously, and then "SENSE 2" on the 9300 console.
   b. Turn the teletypewriter on.
   c. Load the program deck into the card reader and depress "POWER" and "START" on the card reader.
   d. Depress "READY" on the line printer.
2. Energize the AGT Graphics Console
   a. Depress "ON" and "RESET" at the AGT cabinet.
   b. Turn on the disk drive.
   c. When the ready light on the disk cabinet appears, energize the "THIS IS IT" circuit breaker on the back of the AGT cabinet.
   d. On the AGT cabinet depress "HALT", "RESET", "RUN", and "PULSE 1".

e. The typewriter at the graphics console should type the date request: "MO/DA/YR=". Type this information in. If this request does not appear, the AGT must be "Bootstrapped". Information for Bootstrapping is available in the laboratory.

f. Type "RESET("MAD",pvv)! where the p and v numbers indicate the location of MAD on the disk. PVV is normally either 101 or 104. When the carriage of the typewriter returns type "GATED!".

g. Place the function switch overlay shown in Figure A-2-1 over the function switches and energize the lightpen. The AGT console is now ready.

3. Compile and execute the program

a. At the XDS9300 console depress "CLEAR" and "CLEAR FLAGS" simultaneously, and then "IDLE", "RESET", "RUN", and "CARDS" The card reader should start readying cards, and the line printer should produce a listing.

b. When execution commences, the teletype should print the statement "TYPE IDEV=2* AND A CARRIAGE RETURN IF USING AGT 2". The input light will then come on. Follow the instructions if using AGT 2; if using AGT 1, simply type an asterisk and a carriage return. The program then reads the data.

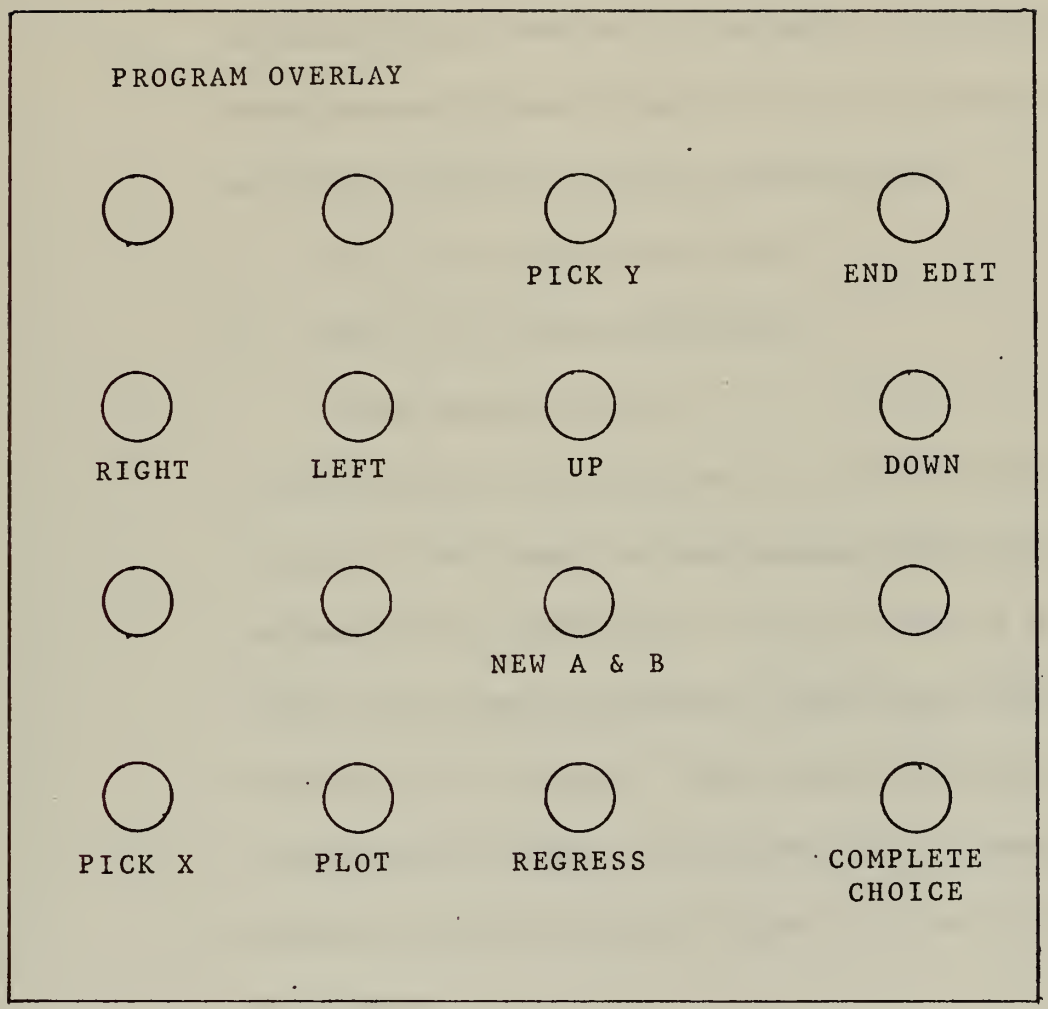


Figure A-2-1. FUNCTION SWITCH OVERLAY

c. At the AGT console the word "NAMELIST" and a blinking cursor will appear at the bottom of the screen. Type in the initial values of the A and B vectors. Any accepted vector notation is acceptable; the following when possible (space limitations may require separate entry of component values) is recommended:

A=1,1,1,1 and carriage return

B=1,1,1,1 and carriage return

* and carriage return

If any mistakes in inputting the A or B vectors, the line printer will output an error message when the carriage return is depressed. Until the asterisk is typed, errors can be corrected, without harm to the execution of the program. When the asterisk and carriage return is typed, the program starts computing projections and the graphics display of video X's and Y's appears.

4. Iterations and output

a. To cause an iteration, the user pushes the switch "PICK X" or "PICK Y" as appropriate. Video appears at the bottom of the screen indicating that the program is in the graphics edit mode. The user depresses the button on the side of the lightpen and places the tip of the pen over the desired point; he then releases the

button. The point selection is indicated when a portion of the picked letter vanishes. When the desired point is so indicated the operator presses the function switches "COMPLETE CHOICE" and "END EDIT". Finally he causes the iteration to occur by depressing one of the switches: "RIGHT", "LEFT", "UP", or "DOWN".

b. By pressing the "REGRESS" button, the user can negate the effects of the last iteration.

c. If the operator desires to initiate a new series of iterations, he depresses the "NEW A&B" button. The word "NAMELIST" and the flashing cursor on the screen and the program is effectively at step 3c.

d. Finally by depressing the switch labelled "PLOT", the user causes the line printer to output a six by six inch representation of the situation he views on the AGT screen.

5. Securing the computer

a. There is no feature in the program which causes execution to terminate. Instead when the operator is finished he secures the computer. First in securing the AGT, he depresses switch "PULSE 1" and types "HOME!". He depresses "HALT" and "RESET" at the AGT cabinet and stops the disk drive. He then flips the "THIS IS IT" circuit breaker to the off position.

b. The card reader and the teletype are turned off.

c. The 9300 is secured by depressing "IDLE", followed by "RESET" and "POWER ON" (simultaneously).

APPENDIX III. DATA LISTING

This appendix provides a listing of the data sets used in the illustrative examples presented in section six of the paper. The data is formatted so that for each case, a row of numbers in the listing represents a single tuple for that problem. Thus, for example, the data set for case one is presented in three columns, while that for case six is in thirteen columns. As an aid for reference, the following index to this appendix is provided:

CASE 1 DATA

Y1	Y2	Y3
1.4760	1.0910	1.6880
1.5520	1.8980	1.9890
1.4770	1.8920	1.5140
1.6330	1.0480	1.5200
1.1660	1.0040	1.0620
1.8390	1.4130	1.5910
1.5960	1.1130	1.6650
1.4980	1.3760	1.0410
1.3660	1.2080	1.8670
1.4940	1.5920	1.7380
1.5280	1.2360	1.8400
1.9900	1.7610	1.7300
1.0630	1.8430	1.7990
1.2730	1.8040	1.8820
1.2510	1.7840	1.9240
1.4090	1.4700	1.0110
1.4660	1.5380	1.4990
1.7840	1.7510	1.4130
1.2930	1.3150	1.9900
1.5360	1.2590	1.0840
1.0230	1.1780	1.3750
1.9470	1.0900	1.2740
1.2390	1.1410	1.5050

1.9490	1.5000	1.7620
1.4020	1.2780	1.6590
1.6640	1.1390	1.6760
1.3350	1.3180	1.4630
1.5510	1.0790	1.4690
1.5730	1.1890	1.9350
1.7510	1.1550	1.9190

X1	X2	X3
.1820	.5940	.7590
.7510	.4200	.0270
.3400	.0070	.0580
.4540	.0790	.5710
.3660	.9420	.7470
.0100	.3710	.2370
.0670	.2880	.3260
.1290	.1430	.3530
.3040	.4830	.2190
.1950	.7130	.8210
1.0000	.2570	.6930
.1160	.1950	.2770
.7240	.5050	.3230
.1800	.8060	.7180
.7020	.3840	.3070
.3450	.3510	.6360
.5060	.9670	.2980
.0290	.1020	.6560
.7670	.7870	.5840
.1310	.0920	.1190
.0380	.8070	.8260
.0440	.9880	.2590
.6430	.5340	.7950

.1000	.9630	.9820
.2960	.4110	.1090
.6920	.8360	.1050
.3370	.6720	.0700
.8820	.4480	.3050
.3980	.6590	.5150
.7680	.3780	.6270

CASE 2 DATA

Y1	Y2	Y3	Y4	Y5	Y6
1.1800	1.7180	1.1000	1.8920	1.6170	1.7030
1.5970	1.6210	1.5260	1.3580	1.5520	1.7460
1.1970	1.5970	1.5400	1.5900	1.7330	1.4840
1.5060	1.5610	1.0350	1.1760	1.8920	1.8990
1.5880	1.3280	1.6390	1.6250	1.7540	1.6840
1.0180	1.0940	1.2890	1.6460	1.2840	1.5200
1.5170	1.6270	1.3170	1.8120	1.1970	1.1370
1.7010	1.4520	1.3420	1.4520	1.8700	1.9170
1.1240	1.2390	1.7110	1.2150	1.5460	1.2090
1.8830	1.1200	2.0000	1.0720	1.3440	1.3970
1.6490	1.4200	1.9980	1.8610	1.4980	1.8100
1.0320	1.3650	1.9130	1.2830	1.4600	1.8250
1.9290	1.2240	1.8800	1.7470	1.4050	1.0800
1.0640	1.9960	1.5410	1.8370	1.0460	1.0390
1.0690	1.2910	1.9790	1.6380	1.1700	1.1950
1.0920	1.2260	1.1740	1.8060	1.2440	1.8020
1.5240	1.3370	1.9190	1.1660	1.5320	1.9480
1.1760	1.6900	1.4870	1.6870	1.4220	1.4130
1.5850	1.6480	1.9620	1.2250	1.3780	1.6270
1.6320	1.1750	1.6150	1.5710	1.7240	1.2160
1.1710	1.6590	1.6430	1.2520	1.8420	1.9580
1.6840	1.9320	1.4370	1.1620	1.3780	1.8270
1.2080	1.2060	1.5440	1.2670	1.4970	1.6530

1.9680	1.3420	1.5310	1.4870	1.1530	1.0740
1.4900	1.1330	1.5750	1.9290	1.0840	1.7150
1.6920	1.3280	1.8990	1.8300	1.1060	1.2220
1.5300	1.9090	1.5220	1.2270	1.8050	1.2390
1.0030	1.1580	1.0800	1.0960	1.7830	1.4620
1.3440	1.0590	1.7920	1.2040	1.5150	1.2900
1.6490	1.4840	1.6560	1.2440	1.8740	1.4870
1.4610	1.1810	1.6360	1.3970	1.9240	1.1520
1.7010	1.3170	1.7840	1.4090	1.7470	1.1710
1.7450	1.6550	1.5510	1.3890	1.3680	1.8820
1.3860	1.0390	1.6430	1.2360	1.1580	1.4720
1.7570	1.3710	1.3980	1.5500	1.8610	1.2130
1.1220	1.3960	1.0450	1.6470	1.2940	1.4500
1.2890	1.6390	1.7970	1.6410	1.8250	1.4830
1.8480	1.1710	1.9780	1.7740	1.5450	1.6930
1.5190	1.0470	1.6790	1.4180	1.9150	1.3880
1.3120	1.7410	1.7030	1.5280	1.0540	1.7750
1.7770	1.0190	1.4150	1.2320	1.4420	1.8390
1.5040	1.0720	1.0800	1.2360	1.5070	1.2360
1.4890	1.2300	1.2440	1.3930	1.2870	1.4500
1.5990	1.1760	1.4850	1.6160	1.4220	1.6330
1.9970	1.1190	1.4000	1.3640	1.1440	1.1940
1.2390	1.3550	1.8640	1.2330	1.0520	1.6570
1.8420	1.1950	1.9410	1.1760	1.0520	1.9500
1.7240	1.8400	1.6020	1.8330	1.9070	1.7560

1.0070	1.5280	1.5400	1.6480	1.5320	1.3720
1.9520	1.9020	1.7750	1.2180	1.9300	1.3440
1.6680	1.3920	1.6360	1.5750	1.2920	1.9370
1.3950	1.2630	1.8380	1.0160	1.9050	1.5380
1.4920	1.9890	1.5140	1.5000	1.6070	1.8030
1.6290	1.3330	1.0270	1.0400	1.7410	1.2820
1.9770	1.6460	1.1450	1.9480	1.3720	1.5850
1.4740	1.9880	1.0880	1.5780	1.8820	1.5570
1.9040	1.3340	1.1040	1.9740	1.0880	1.0210
1.1840	1.7700	1.2590	1.1160	1.5510	1.6540
1.0290	1.2620	1.1120	1.5010	1.3440	1.3500
1.5820	1.7010	1.5980	1.6560	1.0250	1.6470
1.5920	1.4960	1.9630	1.0200	1.8050	1.7030
1.2960	1.1010	1.3690	1.0920	1.7890	1.2970
1.0590	1.1460	1.5310	1.0580	1.9920	1.6490
1.1650	1.1740	1.4850	1.7240	1.1400	1.0050
1.4300	1.9070	1.0890	1.3600	1.1800	1.0280
1.7460	1.9390	1.3440	1.2960	1.6260	1.2600
1.2030	1.6010	1.5250	1.6100	1.5750	1.1510
1.4310	1.0500	1.2420	1.2700	1.1460	1.8740
1.2860	1.1700	1.9390	1.0110	1.4860	1.7680
1.0890	1.4970	1.6230	1.9720	1.5320	1.8760

X1	X2	X3	X4	X5	X6
.5280	.2360	.8400	.9900	.7610	.7300
.0630	.8430	.7990	.2730	.8040	.8820
.2510	.7840	.9240	.4090	.4710	.0110
.4660	.5380	.4990	.7840	.7510	.4130
.2930	.3150	.9900	.5360	.2590	.0840
.0230	.1780	.3750	.9470	.0900	.2740
.2390	.1410	.5050	.9490	.5000	.7620
.4020	.2770	.6590	.6640	.1390	.6760
.3350	.3180	.4630	.5510	.0790	.4690
.5730	.1890	.9350	.7510	.1550	.9200
.9710	.9050	.6570	.1750	.2750	.6280
.9490	.4280	.6740	.4000	.4000	.1710
.9670	.2840	.3050	.4500	.6880	.0550
.9280	.4290	.5430	.4770	.7090	.6420
.1120	.1490	.8840	.5820	.2520	.8250
.6940	.0800	.4200	.7150	.0680	.3000
.1710	.2650	.3710	.5280	.4620	.2820
.5430	.9090	.6120	.0640	.5720	.1500
.2330	.9870	.3460	.3050	.7480	.8770
.4390	.3270	.2310	.7630	.1980	.8160
.1820	.5940	.7590	.7510	.4200	.0270
.3340	.0070	.0580	.4550	.0790	.5710
.3660	.9420	.7470	.0100	.3710	.2370

.0670	.2880	.3260	.1290	.1430	.3530
.3040	.4830	.2190	.1950	.7130	.8210
1.0000	.2570	.6930	.1160	.1940	.2780
.7240	.5050	.3230	.1800	.8060	.7180
.7020	.3840	.3070	.3450	.3510	.6360
.5060	.9670	.2980	.0290	.1020	.6560
.7660	.7870	.5840	.1310	.0920	.1190
.0380	.8070	.8260	.0440	.9880	.2590
.6430	.5340	.7960	.0980	.9630	.9820
.2960	.4120	.1090	.6920	.8360	.1050
.3370	.6720	.0700	.8820	.4480	.3050
.3980	.6590	.5150	.7680	.3780	.6270
.4760	.0910	.6880	.5520	.8970	.9890
.4770	.8920	.5140	.6330	.0480	.5200
.1660	.0040	.0620	.8390	.4120	.5910
.5960	.1120	.6650	.4980	.3760	.0410
.3660	.2080	.8670	.4930	.5920	.7380
.0600	.1300	.5420	.2300	.9750	.7960
.4330	.3730	.6580	.4860	.8540	.4660
.9550	.5410	.4120	.8610	.5130	.5250
.1120	.3640	.7990	.8070	.3800	.0970
.3250	.3210	.1200	.0100	.3380	.9710
.1340	.6920	.9610	.1270	.7700	.1880
.7670	.7340	.1320	.4360	.1450	.4760
.2340	.6790	.3150	.6830	.0800	.9800

.3180	.3060	.5830	.4550	.1370	.6580
.8900	.4230	.5380	.1260	.1380	.6260
.6210	.7120	.4490	.1480	.1080	.2780
.4210	.1050	.2450	.2840	.9340	.7280
.7180	.1460	.6220	.7020	.5230	.5750
.8260	.6350	.2660	.8080	.0580	.5990
.9400	.1890	.9330	.1780	.5400	.4290
.6360	.9780	.6290	.3800	.2920	.8740
.7800	.2020	.7030	.0720	.1500	.8240
.9810	.1380	.6370	.6800	.4160	.8180
.8840	.2890	.6940	.3400	.8760	.1090
.7420	.6210	.9010	.1040	.6380	.5320
.1650	.0940	.4250	.4770	.6980	.2480
.5580	.5280	.6880	.2780	.9530	.2640
.5700	.1750	.7170	.9160	.3200	.2530
.5110	.0660	.8640	.4980	.8720	.1860
.5590	.0440	.1750	.7050	.0160	.5620
.4030	.9310	.2990	.6540	.6000	.7330
.4630	.4810	.1160	.0740	.8700	.2130
.7390	.0940	.6510	.9670	.3490	.1730
.5040	.6630	.6380	.7120	.5130	.8220
.2120	.0560	.5920	.6580	.3910	.5660

CASE 3 DATA

Y1	Y2	Y3	Y4	Y5	Y6
3.1800	3.7180	.1000	.8920	.6170	.7030
3.5970	3.6210	.5260	.3580	.5520	.7460
3.1970	3.5970	.5400	.5900	.7330	.4840
3.5060	3.5610	.0350	.1760	.8920	.8990
3.5880	3.3280	.6390	.6250	.7540	.6840
3.0180	3.0940	.2890	.6460	.2840	.5200
3.5170	3.6270	.3170	.8120	.1970	.1370
3.7010	3.4520	.3420	.4520	.8700	.9170
3.1240	3.2390	.7110	.2150	.5460	.2090
3.8830	3.1200	1.0000	.0720	.3440	.3970
3.6490	3.4200	.9980	.8610	.4980	.8100
3.0320	3.3650	.9130	.2830	.4600	.8250
3.9290	3.2240	.8800	.7470	.4050	.0800
3.0640	3.9960	.5410	.8370	.0460	.0390
3.0690	3.2910	.9790	.6380	.1700	.1950
3.0920	3.2260	.1740	.8060	.2440	.8020
3.5240	3.3370	.9190	.1660	.5320	.9480
3.1760	3.6900	.4870	.6870	.4220	.4130
3.5850	3.6480	.9620	.2250	.3780	.6270
3.6320	3.1750	.6150	.5710	.7240	.2160
3.1710	3.9590	.6430	.2520	.8420	.9580
3.6840	3.9320	.4370	.1620	.3780	.8270
3.2080	3.2060	.5440	.2670	.4970	.6530

3.9680	3.3420	.5310	.4870	.1530	.0740
3.4900	3.1330	.5750	.9290	.0840	.7150
3.6920	3.3280	.8990	.8300	.1060	.2220
3.5300	3.9090	.5220	.2270	.8050	.2390
3.0030	3.1580	.0800	.0960	.7830	.4620
3.3440	3.0590	.7920	.2040	.5150	.2900
3.6490	3.4840	.6560	.2440	.8740	.4870
3.4610	3.1810	.6360	.3970	.9240	.1520
3.7010	3.3170	.7840	.4090	.7470	.1710
3.7450	3.6550	.5510	.3890	.3680	.8820
3.3860	3.0390	.6430	.2360	.1580	.4720
3.7570	3.3710	.3980	.5500	.8610	.2130
3.1220	3.3960	.0450	.6470	.2940	.4500
3.2890	3.6390	.7970	.6410	.8250	.4830
3.8480	3.1710	.9780	.7740	.5450	.6930
3.5190	3.0470	.6790	.4180	.9150	.3880
3.3120	3.7410	.7030	.5280	.0540	.7750
3.7770	3.0190	.4150	.2320	.4420	.8390
3.5040	3.0720	.0800	.2360	.5070	.2360
3.4890	3.2300	.2440	.3930	.2870	.4500
3.5990	3.1760	.4850	.6160	.4220	.6330
3.9970	3.1190	.4000	.3640	.1440	.1940
3.2390	3.3550	.8640	.2330	.0520	.6570
3.8420	3.1950	.9410	.1760	.0520	.9500
3.7240	3.8400	.6020	.8330	.9070	.7560

3.0070	3.5280	.5400	.6480	.5320	.3720
3.9520	3.9020	.7750	.2180	.9300	.3440
3.6680	3.3920	.6360	.5750	.2920	.9370
3.3950	3.2630	.8380	.0160	.9050	.5380
3.4920	3.9890	.5140	.5000	.6070	.8030
3.6290	3.3330	.0270	.0400	.7410	.2820
3.9770	3.6460	.1450	.9480	.3720	.5850
3.4740	3.9880	.0880	.5780	.8820	.5570
3.9040	3.3340	.1040	.9740	.0880	.0210
3.1840	3.7700	.2590	.1160	.5510	.6540
3.0290	3.2620	.1120	.5010	.3440	.3500
3.5820	3.7010	.5980	.6560	.0250	.6470
3.5920	3.4960	.9630	.0200	.8050	.7030
3.2960	3.1010	.3690	.0920	.7890	.2970
3.0590	3.1460	.5310	.0580	.9920	.6490
3.1650	3.1740	.4850	.7240	.1400	.0050
3.4300	3.9070	.0890	.3600	.1800	.0280
3.7460	3.9390	.3440	.2960	.6260	.2600
3.2030	3.6010	.5250	.6100	.5750	.1510
3.4310	3.0500	.2420	.2700	.1460	.8740
3.2860	3.1700	.9390	.0110	.4860	.7680
3.0890	3.4970	.6230	.9720	.5320	.8760
.5280	.2360	.8400	.9900	.7610	.7300
.0630	.8430	.7990	.2730	.8040	.8820
.2510	.7840	.9240	.4090	.4710	.0110

.4660	.5380	.4990	.7840	.7510	.4130
.2930	.3150	.9900	.5360	.2590	.0840
.0230	.1780	.3750	.9470	.0900	.2740
.2390	.1410	.5050	.9490	.5000	.7620
.4020	.2770	.6590	.6640	.1390	.6760
.3350	.3180	.4630	.5510	.0790	.4690
.5730	.1890	.9350	.7510	.1550	.9200
.9710	.9050	.6570	.1750	.2750	.6280
.9490	.4280	.6740	.4000	.4000	.1710
.9670	.2840	.3050	.4500	.6880	.0550
.9280	.4290	.5430	.4770	.7090	.6420
.1120	.1490	.8840	.5820	.2520	.8250
.6940	.0800	.4200	.7150	.0680	.3000
.1710	.2650	.3710	.5280	.4620	.2820
.5430	.9090	.6120	.0640	.5720	.1500
.2330	.9870	.3460	.3050	.7480	.8770
.4390	.3270	.2310	.7630	.1980	.8160
.1820	.5940	.7590	.7510	.4200	.0270
.3340	.0070	.0580	.4550	.0790	.5710
.3660	.9420	.7470	.0100	.3710	.2370
.0670	.2880	.3260	.1290	.1430	.3530
.3040	.4830	.2190	.1950	.7130	.8210
1.0000	.2570	.6930	.1160	.1940	.2780
.7240	.5050	.3230	.1800	.8060	.7180
.7020	.3840	.3070	.3450	.3510	.6360

.5060	.9670	.2980	.0290	.1020	.6560
.7660	.7870	.5840	.1310	.0920	.1190
.0380	.8070	.8260	.0440	.9880	.2590
.6430	.5340	.7960	.0980	.9630	.9820
.2960	.4120	.1090	.6920	.8360	.1050
.3370	.6720	.0700	.8820	.4480	.3050
.3980	.6590	.5150	.7680	.3780	.6270
.4760	.0910	.6880	.5520	.8970	.9890
.4770	.8920	.5140	.6330	.0480	.5200
.1660	.0040	.0620	.8390	.4120	.5910
.5960	.1120	.6650	.4980	.3760	.0410
.3660	.2080	.8670	.4930	.5920	.7380
.0600	.1300	.5420	.2300	.9750	.7960
.4330	.3730	.6580	.4860	.8540	.4660
.9550	.5410	.4120	.8610	.5130	.5250
.1120	.3640	.7990	.8070	.3800	.0970
.3250	.3210	.1200	.0100	.3380	.9710
.1340	.6920	.9610	.1270	.7700	.1880
.7670	.7340	.1320	.4360	.1450	.4760
.2340	.6790	.3150	.6830	.0800	.9800
.3180	.3060	.5830	.4550	.1370	.6580
.8900	.4230	.5380	.1260	.1380	.6260
.6210	.7120	.4490	.1480	.1080	.2780
.4210	.1050	.2450	.2840	.9340	.7280
.7180	.1460	.6220	.7020	.5230	.5750

.8260	.6350	.2660	.8080	.0580	.5990
.9400	.1890	.9330	.1780	.5400	.4290
.6360	.9780	.6290	.3800	.2920	.8740
.7800	.2020	.7030	.0720	.1500	.8240
.9810	.1880	.6370	.6800	.4160	.8180
.8840	.2890	.6940	.3400	.8760	.1090
.7420	.6210	.9010	.1040	.6380	.5320
.1650	.0940	.4250	.4770	.6980	.2480
.5580	.5280	.6880	.2780	.9530	.2640
.5700	.1750	.7170	.9160	.3200	.2530
.5110	.0660	.8640	.4980	.8720	.1860
.5590	.0440	.1750	.7050	.0160	.5620
.4030	.9310	.2990	.6540	.6000	.7330
.4630	.4810	.1160	.0740	.8700	.2130
.7390	.0940	.6510	.9670	.3490	.1730
.5040	.6630	.6380	.7120	.5130	.8220
.2120	.0510	.5920	.0580	.3910	.5200

X1	X2	X3	X4	X5	X6
3.4640	.9010	.2180	.9150	.0230	.1260
3.3570	.5030	.1080	.4400	.6590	.1430
3.0480	.5760	.9700	.1080	.8830	.6020
3.5320	.6140	.3460	.5500	.9720	.9800
3.6970	.7270	.7140	.8950	.1700	.7180
3.8590	.2520	.8740	.2630	.2250	.5570
3.6910	.9750	.4000	.3330	.6170	.0050
3.1560	.7950	.5210	.2180	.1500	.6340
3.2460	.8700	.1050	.7040	.1150	.5490
3.0840	.0420	.8060	.6870	.8600	.2520
3.6890	.7660	.8250	.8010	.7040	.2390
3.8290	.6400	.7410	.2790	.4400	.6230
3.5530	.2540	.9070	.6750	.3440	.6310
3.6950	.1420	.3280	.7960	.1610	.3490
3.3420	.1090	.6850	.3620	.8450	.8840
3.4880	.8670	.7670	.8030	.7200	.7040
3.7470	.5210	.0150	.3010	.1110	.5830
3.4270	.1510	.5930	.5620	.0910	.5170
3.8230	.3510	.3130	.4820	.0320	.3120
3.6860	.8950	.2910	.9310	.5920	.0320
3.9100	.6470	.9150	.9760	.1130	.5300
3.5020	.8300	.4790	.2220	.0770	.5570
3.8230	.6880	.8280	.1530	.1400	.6250

3.8000	.6290	.1120	.1640	.6960	.3140
3.7750	.1790	.7560	.8350	.3740	.2640
3.0430	.4360	.2650	.1990	.6560	.8420
3.6780	.8130	.9790	.6690	.2380	.6000
3.3910	.3870	.0170	.9600	.8410	.6940
3.2830	.8970	.1850	.4620	.0940	.4830
3.9930	.3540	.5910	.7420	.1640	.6410
3.3260	.2800	.9300	.4040	.0820	.0270
3.6620	.8840	.6990	.1790	.9820	.7610
3.1540	.0240	.3470	.1360	.4580	.6300
3.9990	.2760	.5250	.9830	.5610	.2110
3.3580	.2120	.3330	.6630	.9770	.2710
3.7940	.7420	.0130	.5240	.6400	.8940
3.1250	.7390	.3950	.4280	.1640	.5780
3.0610	.8440	.2180	.6860	.0480	.3790
3.1910	.3060	.0990	.6780	.2060	.0730
3.4580	.0580	.8480	.9880	.0370	.9960
3.6540	.4420	.7120	.8260	.6200	.8240
3.6060	.7420	.1510	.5890	.2940	.1360
3.8280	.4200	.3800	.4470	.7800	.1730
3.6980	.5720	.2080	.1550	.9730	.1030
3.4200	.7940	.2140	.1010	.9590	.2730
3.4460	.7400	.3860	.9340	.6190	.9700
3.2420	.3970	.3590	.5120	.7930	.1990
3.6470	.6650	.5900	.4140	.5520	.0970

3.7120	.3050	.2030	.5100	.5070	.4440
3.4690	.4180	.9040	.9960	.2810	.1820
3.2500	.1550	.2780	.6280	.6650	.0230
3.5090	.6720	.2800	.5370	.3570	.9500
3.1440	.3860	.5020	.4110	.4440	.1830
3.0920	.5670	.2290	.6010	.1060	.3430
3.9600	.7830	.8540	.1100	.0920	.8750
3.7400	.2310	.0530	.9600	.8330	.4330
3.7090	.0260	.5080	.8920	.5330	.6400
3.1080	.9480	.9020	.1240	.3410	.8950
3.6940	.7320	.4400	.9790	.9150	.3080
3.4330	.3130	.5460	.8510	.4550	.2940
3.6350	.7630	.2020	.0070	.0000	.3020
3.6360	.7510	.4830	.7110	.6260	.9940
3.7310	.8500	.5000	.3220	.3010	.3360
3.1440	.1810	.7990	.8180	.6800	.8610
3.1360	.3130	.9250	.1100	.8270	.5030
3.6050	.7800	.7160	.1450	.4830	.5110
3.0600	.6190	.1390	.2810	.1420	.0280
3.4300	.8880	.6170	.7770	.3840	.0790
3.7460	.0500	.4300	.5280	.7050	.7900
3.4980	.7510	.5660	.3270	.5980	.2720
.3920	3.0080	.6440	.0770	.6360	.4280
.1010	3.1880	.3410	.6280	.9080	.9800
.0310	3.9910	.9650	.3060	.3530	.1620

.6470	3.7120	.5470	.8590	.2310	.5910
.8130	3.2510	.6430	.2020	.7690	.6270
.2590	3.4530	.9120	.2920	.2910	.3220
.8290	3.9090	.8560	.6340	.3170	.9870
.1420	3.4940	.0190	.6520	.1080	.0060
.1780	3.7690	.4450	.7180	.5860	.5610
.7990	3.7760	.7100	.6160	.7660	.9200
.3040	3.0830	.7580	.9820	.1320	.6390
.1490	3.4600	.3850	.4620	.8510	.3610
.6110	3.6420	.3100	.0730	.9100	.7540
.6150	3.9640	.0140	.7480	.6330	.9440
.0770	3.6600	.0540	.7080	.5880	.8210
.8580	3.1960	.0800	.5640	.3570	.3430
.9740	3.3720	.7680	.7750	.5550	.8190
.3260	3.4760	.2230	.4900	.1430	.3050
.4380	3.8080	.4090	.0230	.2650	.9330
.2220	3.2930	.7920	.8420	.6760	.0130
.7680	3.8280	.4000	.5260	.2160	.8480
.5860	3.3070	.0690	.8310	.1270	.7100
.3290	3.9850	.1040	.2660	.6410	.7090
.5680	3.2460	.0630	.0940	.2720	.9180
.0350	3.5120	.1320	.9950	.7420	.2430
.5620	3.0220	.6390	.2920	.6800	.9090
.0390	3.7500	.0320	.5090	.3260	.6890
.3600	3.1360	.6270	.2640	.5870	.7420

.7300	3.7550	.3840	.7730	.1540	.3580
.8410	3.1170	.5200	.7360	.0290	.7750
.2940	3.7700	.5790	.9270	.4730	.4320
.6620	3.4380	.2400	.4220	.7720	.3380
.0430	3.6550	.8690	.4270	.0340	.8290
.1760	3.3640	.6100	.9370	.7680	.0610
.4730	3.5920	.9960	.8710	.1360	.3120
.2480	3.1010	.2700	.6050	.1920	.3120
.5200	3.8710	.2380	.4800	.5650	.0450
.9160	3.6580	.7960	.2220	.0440	.8730
.5850	3.6290	.7660	.0630	.2680	.7400
.5820	3.9750	.9990	.9100	.2840	.8930
.3700	3.0850	.0540	.7780	.3100	.1320
.4900	3.6900	.8750	.9610	.2420	.1100
.7370	3.3540	.8240	.2860	.7840	.9030
.4110	3.3370	.3610	.3620	.1640	.0440
.5760	3.2860	.6530	.9330	.9120	.4140
.8290	3.2700	.3430	.0310	.3430	.5470
.5800	3.6950	.0160	.1950	.6830	.0340
.3690	3.2790	.7160	.6110	.1270	.4210
.3820	3.6450	.5420	.1500	.2560	.4850
.1270	3.1980	.0120	.6590	.8560	.8010
.5720	3.4330	.3760	.5190	.0270	.9090
.2860	3.5050	.3600	.9250	.9240	.7610
.2860	3.7920	.9530	.9280	.6610	.4420

.5830	3.1080	.2180	.9280	.2830	.4910
.4470	3.9900	.3380	.2960	.6000	.2820
.6280	3.1680	.0160	.9830	.0900	.5360
.0120	3.5190	.4070	.6850	.6030	.3120
.1260	3.3290	.7640	.8650	.7680	.7830
.0200	3.2700	.1700	.1980	.7710	.8130
.8310	3.4950	.5830	.5500	.2900	.7670
.7950	3.1570	.7200	.5480	.5000	.0200
.1740	3.6320	.6630	.2370	.8280	.2300
.4840	3.7240	.8260	.9400	.7660	.4380
.2480	3.6520	.2240	.1180	.3720	.4540
.3020	3.5440	.9520	.9510	.0800	.5970
.7180	3.3850	.0400	.9460	.8180	.6950
.0330	3.2690	.7260	.0820	.9670	.6790
.8710	3.4570	.1220	.4420	.4670	.0570
.1810	3.2470	.1890	.0560	.9360	.5260
.3070	3.0370	.5560	.7400	.3160	.3080

CASE 4 DATA

Y1	Y2	Y3	Y4
-.6820	-.3590	-.1320	-1.8380
-.7650	1.4370	.6250	.8080
.5220	-.4680	-.2980	-1.8480
-.7580	-.5810	1.7140	.3730
-1.9940	.0860	-.1110	-.0550
.7020	-.7940	1.6710	.2390
.3690	-.8250	1.6220	.5890
-.8440	-.2490	.0820	-1.7860
-.7320	.0710	-.4260	-1.7740
-.7350	.0330	-.0700	-1.8580
-1.9130	.3420	.1820	.2230
-.8180	1.4470	-.9770	.0940
-.3800	1.4660	-1.1350	-.6260
-.5510	1.5800	-.9870	.4580
-1.9410	-.3660	.1630	.0220
.1130	1.5680	-1.0680	.3450
.0980	1.5600	.3780	-1.1680
-1.9060	.2180	-.4960	.1410
-1.0050	-.4050	1.6320	.1690
-1.9770	-.2900	.0450	-.0510

X1	X2	X3	X4
.1370	.6680	-.5700	-.4500
-.2450	-.7640	-.5690	-.0990
-.1140	-.8740	.2970	.3620
.4360	.5580	.4460	.2870
.1970	.5910	.4740	-.5380
.0770	-.9320	-.2750	-.1710
.2170	.6450	-.6260	.1710
.0970	.5710	-.7950	.1120
.2930	.6520	.2490	.2060
-.2080	-.7060	-.4710	-.3580
.1990	-.8330	.3580	.1330
.2670	-.7540	.3010	.2510
.0280	-.9320	-.3580	-.0480
-.4860	-.7850	.1990	.2100
.2540	.6090	.2680	-.5980
.1630	-.8880	.2430	.2220
.4100	.3490	.3970	.3460
.2020	-.7230	-.5960	.1500
-.3550	.5350	-.6690	.3320
.2280	.5960	.4750	-.5370

CASE 5 DATA

Y1	Y2	Y3	Y4	Y5	Y6	Y7	Y8
160.0	5.1	0.0	28.0	70.0	450.0	4.10	1.0
155.0	5.2	8.0	27.0	85.0	400.0	3.12	7.0
130.0	5.0	0.0	26.0	75.0	560.0	2.96	9.0
161.0	5.0	0.0	27.0	70.0	665.0	4.04	1.0
135.0	5.0	2.0	27.0	88.0	570.0	3.05	3.0
165.0	5.0	1.0	23.0	95.0	675.0	3.72	4.0
150.0	5.0	9.0	29.0	90.0	580.0	3.83	5.0
148.0	4.8	8.0	26.0	85.0	390.0	3.75	8.0
150.0	4.5	7.0	31.0	60.0	435.0	3.05	0.0
120.0	4.0	6.0	33.0	55.0	440.0	2.78	8.0
120.0	5.1	8.0	32.0	56.0	650.0	2.89	3.0
190.0	4.2	8.0	30.0	55.0	640.0	3.46	7.0
100.0	4.4	9.0	35.0	48.0	430.0	2.98	6.0
150.0	4.0	7.0	29.0	65.0	650.0	2.84	0.0
90.0	4.6	8.0	30.0	70.0	655.0	2.96	7.0
120.0	4.7	7.0	35.0	67.0	645.0	2.78	0.0
200.0	4.3	9.0	30.0	62.0	660.0	3.20	3.0
120.0	4.1	8.0	28.0	63.0	530.0	3.05	9.0
105.0	5.0	7.0	27.0	64.0	435.0	3.55	7.0
210.0	5.2	8.0	26.0	67.0	440.0	3.10	0.0
90.0	4.0	0.0	25.0	68.0	430.0	3.41	1.0
110.0	5.2	1.0	25.0	60.0	530.0	2.97	5.0
100.0	4.3	9.0	25.0	70.0	440.0	3.62	7.0

70.0	4.5	8.0	23.0	64.0	450.0	3.40	9.0
100.0	4.8	9.0	27.0	65.0	355.0	3.10	4.0
130.0	5.2	9.0	25.0	70.0	380.0	2.80	9.0
90.0	4.5	1.0	37.0	74.0	350.0	2.95	1.0
80.0	4.6	0.0	32.0	78.0	450.0	2.12	2.0
95.0	4.9	0.0	25.0	82.0	260.0	3.11	3.0
70.0	4.4	2.0	30.0	85.0	262.0	3.22	4.0
80.0	4.2	3.0	36.0	69.0	260.0	3.35	6.0
95.0	5.1	5.0	31.0	70.0	265.0	3.40	9.0
100.0	4.6	1.0	24.0	76.0	270.0	3.54	0.0
95.0	4.8	0.0	27.0	74.0	355.0	3.63	1.0
85.0	4.7	2.0	25.0	73.0	360.0	3.17	9.0
70.0	4.8	1.0	26.0	78.0	365.0	3.18	0.0
80.0	5.4	0.0	21.0	80.0	370.0	2.90	2.0
85.0	5.5	3.0	33.0	81.0	355.0	3.12	5.0
200.0	3.4	0.0	24.0	98.0	1210.0	4.10	0.0
260.0	3.1	8.0	21.0	110.0	1220.0	3.89	6.0
195.0	3.0	9.0	20.0	105.0	1130.0	2.98	4.0
195.0	3.2	9.0	19.0	110.0	1010.0	3.10	6.0
220.0	3.3	0.0	24.0	95.0	1205.0	3.54	9.0
220.0	3.0	8.0	25.0	90.0	1210.0	3.78	4.0
190.0	3.4	9.0	26.0	96.0	1070.0	3.36	0.0
285.0	3.0	1.0	19.0	100.0	990.0	4.23	1.0
300.0	3.0	9.0	20.0	102.0	1120.0	3.75	0.0
225.0	3.0	0.0	22.0	105.0	985.0	3.80	5.0

260.0	3.4	8.0	22.0	97.0	1090.0	2.94	8.0
280.0	3.0	8.0	20.0	112.0	1200.0	2.70	1.0
300.0	3.4	0.0	20.0	108.0	835.0	3.20	5.0
310.0	3.0	1.0	19.0	106.0	1055.0	3.45	7.0
290.0	3.1	2.0	26.0	94.0	1240.0	3.50	1.0
260.0	3.0	8.0	22.0	98.0	1015.0	4.60	9.0
290.0	3.3	9.0	25.0	100.0	1010.0	3.56	2.0
160.0	3.1	1.0	20.0	79.0	1170.0	4.26	4.0
240.0	3.5	1.0	20.0	88.0	990.0	4.40	2.0
195.0	3.1	8.0	21.0	81.0	975.0	3.58	4.0
290.0	3.4	0.0	19.0	94.0	860.0	3.40	0.0
210.0	3.5	9.0	22.0	96.0	950.0	2.70	7.0
180.0	3.0	1.0	22.0	97.0	990.0	3.83	8.0
205.0	2.9	1.0	23.0	90.0	805.0	3.25	8.0
215.0	3.4	8.0	21.0	100.0	700.0	4.05	2.0
270.0	3.1	8.0	20.0	111.0	1170.0	4.60	1.0
290.0	3.0	9.0	23.0	102.0	1350.0	4.42	3.0
320.0	3.2	0.0	19.0	87.0	1160.0	3.64	8.0
210.0	3.0	9.0	18.0	112.0	1010.0	3.81	5.0
210.0	3.0	8.0	21.0	95.0	1190.0	4.00	0.0
185.0	3.4	9.0	25.0	96.0	1055.0	4.18	1.0
200.0	3.2	8.0	26.0	98.0	980.0	3.46	8.0
170.0	2.9	9.0	20.0	95.0	1095.0	4.15	0.0
140.0	3.0	9.0	20.0	98.0	990.0	4.25	2.0
110.0	4.9	9.0	22.0	130.0	220.0	4.51	9.0

100.0	5.6	8.0	19.0	128.0	216.0	2.60	4.0
95.0	4.9	8.0	24.0	124.0	218.0	2.45	0.0
65.0	6.2	9.0	30.0	134.0	200.0	2.61	2.0
55.0	5.0	0.0	27.0	128.0	205.0	2.62	1.0
70.0	5.3	7.0	28.0	118.0	204.0	2.35	2.0
85.0	4.9	1.0	19.0	117.0	206.0	2.90	4.0
115.0	5.0	0.0	21.0	122.0	198.0	3.25	7.0
95.0	4.8	8.0	27.0	114.0	228.0	2.65	1.0
120.0	6.1	9.0	24.0	120.0	244.0	2.82	8.0

CASE 6 DATA

Y1	Y2	Y3	Y4	Y5	Y6	Y7	Y8	Y9	Y10	Y11	Y12	Y13
1.	365.	152.	90.	4.	2.	4.	3.	1.	1.	5.	250.	39.
1.	517.	126.	70.	1.	2.	3.	2.	2.	2.	2.	326.	38.
1.	343.	180.	86.	1.	1.	3.	2.	2.	3.	2.	278.	42.
1.	432.	110.	65.	4.	1.	3.	2.	4.	4.	2.	278.	42.
1.	376.	115.	80.	1.	1.	3.	2.	1.	3.	2.	278.	41.
1.	410.	110.	70.	1.	1.	3.	2.	2.	3.	2.	255.	43.
1.	408.	105.	65.	1.	1.	3.	2.	2.	3.	2.	278.	50.
1.	410.	150.	100.	1.	2.	4.	2.	2.	4.	2.	278.	44.
1.	315.	118.	76.	1.	2.	3.	2.	2.	4.	5.	315.	28.
1.	546.	150.	70.	4.	1.	3.	2.	2.	3.	1.	212.	47.
1.	437.	120.	70.	4.	2.	3.	2.	2.	4.	2.	372.	39.
1.	506.	106.	72.	4.	2.	3.	2.	2.	3.	1.	145.	37.
1.	338.	110.	70.	2.	2.	3.	2.	1.	3.	1.	278.	39.
1.	446.	120.	70.	1.	1.	2.	2.	1.	4.	1.	278.	50.
1.	426.	135.	80.	1.	1.	3.	2.	1.	4.	1.	278.	48.
1.	410.	140.	85.	1.	1.	3.	2.	2.	4.	1.	250.	35.
1.	483.	120.	70.	1.	2.	3.	2.	2.	4.	1.	278.	48.
3.	377.	114.	70.	2.	2.	2.	2.	2.	4.	5.	350.	38.
3.	410.	130.	85.	2.	1.	3.	2.	2.	4.	2.	264.	39.
3.	414.	120.	80.	4.	2.	1.	2.	1.	3.	5.	246.	49.
1.	369.	140.	90.	4.	1.	3.	2.	1.	4.	1.	278.	38.
1.	335.	132.	82.	3.	1.	3.	1.	1.	4.	1.	278.	38.
1.	410.	140.	80.	3.	1.	3.	1.	2.	4.	2.	268.	40.

1.	422.	152.	92.	1.	2.	2.	2.	2.	3.	2.	278.	49.
1.	517.	110.	80.	1.	2.	3.	2.	2.	4.	5.	250.	53.
1.	250.	130.	70.	1.	2.	3.	2.	2.	4.	2.	315.	45.
1.	466.	114.	74.	1.	1.	2.	2.	2.	4.	3.	250.	38.
1.	366.	120.	90.	1.	1.	2.	2.	2.	4.	3.	278.	47.
1.	271.	110.	70.	1.	1.	3.	2.	2.	4.	5.	295.	36.
1.	467.	130.	70.	4.	1.	3.	2.	2.	4.	5.	278.	41.
1.	372.	128.	92.	4.	1.	3.	2.	2.	4.	5.	353.	43.
1.	300.	140.	75.	4.	2.	3.	2.	2.	4.	5.	200.	38.
1.	348.	110.	80.	4.	1.	2.	1.	2.	4.	2.	335.	40.
1.	288.	130.	80.	1.	1.	4.	2.	2.	4.	2.	264.	47.
1.	477.	115.	80.	1.	1.	2.	2.	2.	4.	2.	245.	43.
1.	406.	140.	110.	2.	2.	2.	2.	2.	4.	5.	300.	33.
1.	489.	110.	64.	1.	2.	2.	2.	2.	3.	5.	278.	42.
1.	406.	130.	80.	4.	2.	4.	2.	2.	3.	5.	278.	51.
1.	367.	135.	85.	1.	2.	3.	2.	1.	3.	3.	350.	40.
1.	444.	125.	80.	1.	2.	3.	2.	1.	3.	1.	302.	45.
1.	402.	122.	70.	1.	1.	1.	2.	2.	3.	1.	278.	46.
1.	462.	120.	78.	1.	2.	3.	1.	2.	4.	1.	278.	41.
1.	511.	140.	95.	4.	1.	2.	2.	1.	3.	1.	350.	45.
1.	410.	140.	90.	4.	1.	2.	2.	1.	3.	2.	245.	52.
1.	446.	90.	60.	1.	1.	4.	1.	1.	3.	2.	202.	40.
1.	468.	110.	70.	1.	2.	3.	2.	1.	4.	1.	314.	37.
1.	410.	130.	80.	1.	2.	3.	2.	2.	4.	5.	351.	40.
1.	410.	140.	88.	1.	2.	3.	2.	2.	4.	1.	242.	42.

1.	355.	120.	70.	1.	1.	3.	2.	2.	4.	3.	366.	23.
2.	394.	116.	80.	1.	2.	1.	2.	2.	4.	2.	283.	42.
2.	312.	100.	60.	1.	2.	1.	2.	2.	4.	2.	310.	47.
1.	383.	100.	60.	1.	2.	1.	2.	2.	4.	2.	310.	47.
1.	489.	140.	80.	1.	2.	2.	2.	1.	4.	2.	278.	46.
1.	410.	130.	82.	1.	2.	2.	2.	1.	4.	2.	250.	41.
1.	404.	140.	98.	1.	2.	2.	2.	1.	4.	2.	278.	43.
1.	378.	140.	90.	1.	2.	3.	1.	2.	3.	1.	267.	43.
1.	369.	130.	70.	2.	1.	4.	2.	1.	3.	1.	278.	44.
1.	388.	133.	90.	3.	1.	3.	2.	3.	4.	5.	288.	43.
1.	483.	110.	80.	4.	1.	2.	2.	2.	4.	2.	277.	39.
3.	461.	140.	90.	1.	2.	4.	1.	1.	2.	1.	330.	52.

X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	X11	X12	X13
1.	404.	160.	78.	4.	2.	1.	2.	1.	4.	1.	200.	67.
1.	418.	118.	76.	1.	1.	3.	3.	1.	1.	2.	298.	43.
1.	288.	128.	86.	1.	2.	4.	3.	1.	1.	2.	209.	37.
2.	349.	170.	110.	3.	2.	1.	1.	4.	1.	5.	343.	40.
1.	396.	118.	72.	3.	1.	2.	3.	1.	1.	2.	221.	37.
3.	411.	180.	120.	1.	2.	4.	1.	1.	1.	2.	196.	52.
1.	443.	160.	90.	4.	2.	3.	3.	1.	1.	2.	271.	37.
1.	479.	140.	90.	1.	2.	2.	1.	1.	1.	1.	260.	40.
3.	388.	140.	90.	1.	2.	3.	1.	1.	1.	5.	281.	40.
2.	445.	130.	88.	4.	2.	4.	3.	1.	1.	1.	215.	42.
1.	389.	110.	78.	3.	2.	4.	3.	1.	1.	2.	249.	32.
1.	406.	114.	70.	2.	2.	3.	1.	1.	1.	5.	248.	40.
1.	431.	100.	60.	4.	2.	3.	1.	1.	1.	1.	224.	34.
2.	413.	144.	86.	4.	2.	4.	3.	1.	1.	1.	247.	41.
1.	411.	110.	50.	4.	1.	4.	1.	1.	1.	2.	240.	40.
1.	406.	110.	76.	1.	1.	3.	3.	1.	4.	2.	248.	23.
1.	444.	120.	80.	1.	2.	1.	3.	1.	1.	1.	151.	42.
2.	375.	120.	70.	1.	2.	3.	3.	1.	1.	3.	248.	41.
1.	383.	120.	80.	2.	1.	3.	3.	1.	1.	1.	256.	36.
2.	439.	114.	80.	1.	2.	4.	3.	1.	1.	2.	248.	40.
2.	489.	104.	70.	1.	2.	3.	3.	1.	3.	1.	221.	44.
1.	387.	118.	90.	1.	1.	4.	3.	1.	1.	2.	279.	43.
3.	351.	120.	90.	4.	2.	3.	3.	2.	5.	5.	258.	36.

1.	350.	110.	66.	1.	1.	3.	3.	1.	1.	3.	248.	41.
3.	536.	124.	84.	4.	2.	2.	3.	1.	1.	5.	282.	33.
2.	432.	110.	70.	1.	1.	4.	1.	1.	2.	1.	240.	55.
3.	393.	150.	90.	3.	1.	3.	1.	1.	1.	5.	245.	46.
1.	369.	132.	80.	1.	2.	2.	3.	1.	1.	1.	237.	40.
2.	418.	120.	80.	3.	2.	3.	3.	1.	3.	2.	236.	40.
1.	383.	126.	84.	1.	1.	4.	3.	1.	0.	1.	221.	36.
1.	325.	135.	80.	1.	2.	4.	3.	1.	5.	5.	175.	30.
1.	395.	120.	78.	4.	1.	4.	1.	1.	1.	1.	178.	40.
1.	375.	117.	75.	1.	1.	4.	3.	1.	1.	1.	229.	39.
1.	432.	142.	80.	4.	2.	4.	3.	1.	1.	1.	235.	43.
2.	267.	120.	84.	4.	2.	4.	3.	1.	1.	1.	213.	40.
1.	276.	158.	100.	1.	2.	4.	3.	1.	1.	2.	167.	41.
3.	489.	188.	70.	1.	2.	3.	3.	1.	1.	3.	248.	44.
1.	369.	120.	84.	1.	1.	4.	3.	1.	1.	1.	294.	43.
1.	398.	150.	90.	4.	2.	4.	3.	1.	1.	1.	223.	42.
1.	442.	118.	70.	3.	2.	2.	3.	1.	1.	5.	240.	44.
1.	394.	130.	80.	1.	2.	3.	3.	1.	1.	5.	288.	31.
1.	444.	120.	78.	3.	2.	2.	3.	1.	2.	1.	256.	42.
1.	389.	140.	95.	1.	1.	4.	3.	2.	2.	5.	273.	44.
1.	359.	130.	100.	1.	1.	2.	3.	1.	1.	5.	195.	35.
2.	232.	142.	100.	1.	1.	4.	1.	1.	1.	5.	325.	55.
1.	379.	106.	72.	4.	2.	4.	3.	1.	1.	1.	264.	34.
1.	378.	140.	100.	4.	1.	4.	1.	1.	1.	1.	309.	39.
1.	381.	150.	90.	4.	1.	4.	1.	1.	1.	1.	282.	54.

2.	364.	128.	82.	4.	1.	4.	3.	1.	1.	5.	250.	33.
2.	437.	140.	90.	4.	2.	4.	3.	1.	1.	5.	248.	36.

BIBLIOGRAPHY

1. Chernoff, Herman, The Use of Faces to Represent Points in n-Dimensional Space Graphically, Technical Report No. 71, Department of Statistics, Stanford Univ., December 27, 1971.

2. Delaura, R.D., Electrical Engineering Computer Laboratory, Manual for the Fortran User, Naval Postgraduate School, Monterey, Calif., 10 February 1972.

3. Duda, Richard O., and Hard, Peter E., Pattern Classification and Scene Analysis, Wiley, 1973.

4. Meisel, William S., Computer Oriented Approaches to Pattern Recognition, Academic Press, 1972.

5. Minsky, Marvin, and Papert, Seymour, Perceptrons, The MIT Press, 1969.

6. Wright, R.M., and Switzer, P., "Numerical Classification Applied to Certain Jamaican Eocene Nummulitids," Mathematical Geology, Vol. 3, No. 3.

Initial Distribution List

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3.	Asst Professor B. O. Shubert, Code 55 Sy Department of Operations Research and Administrative Sciences Naval Postgraduate School Monterey, California 93940	2
4.	Professor T. M. Cover Department of Electrical Engineering Stanford University Stanford, California 93305	1
5.	Asst Professor M. U. Thomas, Code 55To Department of Operations Research and Administrative Sciences Naval Postgraduate School Monterey, California 93940	1
6.	LT G. P. Lauzon, USN 86 Quinn Road Lynn, Massachusetts 01904	1
7.	Naval Postgraduate School Department of Operations Research and Administrative Sciences Monterey, California 93940	1
8.	Chief of Naval Personnel Pers 11b Department of the Navy Washington, D. C. 20370	1

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

ORIGINATING ACTIVITY (Corporate author) Naval Postgraduate School Monterey, California 93940	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP

REPORT TITLE

A Computer-Graphics Separation Algorithm for Pattern Classification and Cluster Analysis

DESCRIPTIVE NOTES (Type of report and, inclusive dates)

Master's Thesis; September 1973

AUTHOR(S) (First name, middle initial, last name)

Gilbert Paul Lauzon

REPORT DATE September 1973	7a. TOTAL NO. OF PAGES 135	7b. NO. OF REFS 6
8a. CONTRACT OR GRANT NO. b. PROJECT NO. c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) 9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)	

10. DISTRIBUTION STATEMENT

Approved for public release; distribution unlimited

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY Naval Postgraduate School Monterey, California 93940

13. ABSTRACT

A separation algorithm applicable to the pattern classification and cluster analysis of n-dimensional ($n > 2$) data is presented. The algorithm reduces the dimensionality of the problem by projecting each point into a plane. This plane is presented to the user on a computer graphics console screen. The operator picks a point on the screen with a lightpen and chooses a "direction of movement" to achieve or increase separation, thereby causing an iteration of the algorithm. Each iteration is in fact a reorientation of the plane into which the data points are projected. Iterations continue until satisfactory separation is achieved. The algorithm is not restricted by the dimensionality of the data, nor are any distributional assumptions required. Results from six case studies indicate that the algorithm is a useful tool for the analysis of multi-dimensional data.

DD FORM 1 NOV 68 1473 (PAGE 1)

S/N 0101-807-6811

Security Classification

A-31408

KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
ttern Classification						
luster Analysis						
iscriminant Function						
omputer Graphics						
an-Machine						