SPACE RESEARCH COORDINATION CENTER

LEARNING ALGORITHMS FOR MULTI-CLASS PATTERN CLASSIFICATION AND PROBLEMS ASSOCIATED WITH ON-LINE HANDWRITTEN CHARACTER RECOGNITION

BY

T. L. TENG AND C. C. LI

DEPARTMENT OF ELECTRICAL ENGINEERING

SRCC REPORT NO. 119

NGL-39-011-002

UNIVERSITY OF PITTSBURGH

PITTSBURGH, PENNSYLVANIA

13 FEBRUARY 1970

The Space Research Coordination Center, established in May, 1963, has the following functions: (1) it administers predoctoral and postdoctoral fellowships in space-related science and engineering programs; (2) it makes available, on application and after review, allocations to assist new faculty members in the Division of the Natural Sciences and the School of Engineering to initiate research programs or to permit established faculty members to do preliminary work on research ideas of a novel character; (3) in the Division of the Natural Sciences it makes an annual allocation of funds to the interdisciplinary Laboratory for Atmospheric and Space Sciences; (4) in the School of Engineering it makes a similar allocation of funds to the Department of Metallurgical and Materials Engineering and to the program in Engineering Systems Management of the Department of Industrial Engineering; and (5) in concert with the University's Knowledge Availability Systems Center, it seeks to assist in the orderly transfer of new space-generated knowledge in industrial application. The Center also issues periodic reports of space-oriented research and a comprehensive annual report.

The Center is supported by an Institutional Grant (~~NSG-416~~) from the National Aeronautics and Space Administration, strongly supplemented by grants from the A. W. Mellon Educational and Charitable Trust, the Maurice Falk Medical Fund, the Richard King Mellon Foundation and the Sarah Mellon Scaife Foundation. Much of the work described in SRCC reports is financed by other grants, made to individual faculty members.

SPACE RESEARCH COORDINATION CENTER

LEARNING ALGORITHMS FOR MULTI-CLASS PATTERN CLASSIFICATION AND PROBLEMS ASSOCIATED WITH ON-LINE HANDWRITTEN CHARACTER RECOGNITION

by

T. L. Teng

and

C. C. Li, Principal Investigator

Department of Electrical Engineering
University of Pittsburgh
Pittsburgh, Pennsylvania 15213

December 1969

ACKNOWLEDGEMENT

This work was supported by the Grant NsG-416 from the National Aeronautics and Space Administration. The authors would like to gratefully acknowledge the support which they received during the past three years. The authors would like also to thank their colleagues, Professor T. W. Sze, Mr. M. J. Zobrak, Mr. T. E. Brand, and Dr. L. C. Geary for their helpful discussions.

ABSTRACT

LEARNING ALGORITHMS FOR MULTI-CLASS PATTERN CLASSIFICATION AND PROBLEMS ASSOCIATED WITH ON-LINE HANDWRITTEN CHARACTER RECOGNITION

There are two contributions made in this dissertation: (1) a study of an on-line handwritten alphanumeric character recognition system, and (2) the proposition of a new learning algorithm for multi-class linear pattern classification.

In the study of an on-line handwritten character recognition system, each character was written in a standard stroke sequence. The character was encoded by the modified, quantized direction-time features where, in addition to the directions of the stroke segments, three kinds of additional feature components were

incorporated: jumping component, reference component, and end component, to enhance the pattern separability and the feasibility of reconstruction. Fifty sets of characters were collected and the corresponding 20-dimensional pattern vectors were extracted with eight quantized directions; one set was considered as the prototype patterns, $\underline{P}_n$, (n=1,2,...,R). A pattern $\underline{x}$ was classified according to the minimum metric $\underline{w}_n^t \underline{\phi}(\underline{P}_n,\underline{x})$ where $\underline{w}_n$'s were trained weight vectors and each component of $\underline{\phi}(\underline{P}_n,\underline{x})$ was interpreted as a metric giving the corresponding component of the deviation of $\underline{x}$ from $\underline{P}_n$ in a special, finite and discrete space where each coordinate was an integer of modulus 8. The weight vectors $\underline{w}_r$'s, (r=1,2,...,R), were trained by a modification of the Zobrak-Sze algorithm. The modified learning algorithm has been proved to be convergent, if the sample patterns are ϕ-separable. By the use of thirty-five training sets of ten characters, a set of solution weight vectors was obtained after 14 training cycles. When being tested by all 49 sets of characters, it gave a recognition error rate of 2.04%. The recognition system was, however, very sensitive to the stroke sequence of the handwritten characters.

For the multi-class linear pattern classification, $\underline{X}^t\underline{w}_j > \underline{X}^t\underline{w}_i$ for all $i \neq j$ and for all $\underline{X}$ belonging to the class C_j, (j=1,2,...,R), where $\underline{X}$ is an augmented pattern vector of dimension D, and $\underline{w}_j$'s are R augmented weight vectors of dimension D. The proposed multi-class learning algorithm was based upon the mapping of pattern vectors of R classes into the neighborhood of R vertices

of a (R-1)-dimensional equilateral simplex with its centroid at the origin. From the (R-1)-dimensional vertex vector, $\underline{e}_j$, (j=1,2,...,R), matrices $\underline{E}_j$'s were defined where $\underline{E}_j = [\underline{e}_j - \underline{e}_1, \ldots, \underline{e}_j - \underline{e}_{j-1}, \underline{e}_j - \underline{e}_{j+1}, \ldots, \underline{e}_j - \underline{e}_R]$. Let the N x D pattern matrix $\underline{A}$ be formed by R submatrices $\underline{A}_j$, (j=1,2,...,R), each of which is composed of n_j transposed, augmented training patterns of class C_j as its row vectors, $(n_1 + n_2 + \ldots + n_R = N)$. Let $\underline{U}$ be a D x (R-1) weight matrix, and $\underline{B}$ and $\underline{Y}$ be two N x (R-1) matrices each of which has submatrices $\underline{B}_j$ and $\underline{Y}_j$ respectively corresponding to the class grouping $\underline{A}_j$ in $\underline{A}$. The new learning algorithm is a generalization of the Ho-Kashyap two-class algorithm to find a solution $\underline{U}$ from which the desired weight vectors $\underline{w}_j$ can be obtained by $\underline{w}_j = \underline{U}\,\underline{e}_j$, (j=1,2,...,R). This algorithm is given as follows: $\underline{U}(0) = \underline{A}^{\#}\underline{B}(0)$, $\underline{Y}(k) = \underline{A}\,\underline{U}(k) - \underline{B}(k)$, $\underline{Z}_j(k) = \underline{Y}_j(k)\,\underline{E}_j$, $\underline{B}(k+1) = \underline{B}(k) + p\underline{H}(k)$, $\underline{H}_j(k) = [\underline{Z}_j(k) + \underline{\Lambda}_j(k)]\,\underline{E}_j^{-1}$, and $\underline{U}(k+1) = \underline{U}(k) + p\underline{A}^{\#}\underline{H}(k)$, where $0. < p. < 1$, the elements of $\underline{\Lambda}_j(k)$ are given by ${}_{\ell}\Lambda_{jq} = {}_{\ell}Z_{jq}\mathrm{Sgn}({}_{\ell}Z_{jq})$, and all row vectors in $\underline{B}_j(0)$ are equal to $\beta\underline{e}_j^{\,t}$ with $\beta > 0$. The convergence of the algorithm has been proved. This algorithm, as compared to the Wee-Fu algorithm of complete generalization, involves a pattern matrix of much smaller dimension and hence requires less storage and less computing time for $\underline{A}^{\#}$. It was demonstrated experimentally that the computing time and cost per iteration were considerably smaller, although it might take more iterations to converge. Hence, for cases where only a relatively small number of iterations are required, this new multi-class

learning algorithm will offer some advantages in saving the overall computing time and storage.

DESCRIPTORS

Character Recognition
Learning Machines
Recognition

TABLE OF CONTENTS

Page

LIST OF FIGURES

LIST OF TABLES

1.0. INTRODUCTION

1.1 General Background on Handwritten Character Recognition

A great deal of work has been done during the past decade on theory and implementation of pattern recognition.[1,9,16,24,33-35,43,44]* Three recent survey papers together reviewed very well the development and advance in this field and gave a summary of the current state of the art[31,42,51]. Among the most important problems in the general field of pattern recognition is the recognition of handwritten characters. There are two types of handwritten character recognition problems. One deals with the situation where already formed characters are to be recognized, for example, the recognition of zip code on the mail and the recognition of other handwritten texts. The optical scanning is used to obtain coordinate information of a character, and an adequate set of features are to be extracted for proper classification[10-13,25,27,29,30,40,41,45,52]. The other deals with the recognition of a character while it is being written, for example, in the graphical-data processing for man-machine communication[2,3,5,28,46]. It can also be used in a teaching machine where school children are taught to write an alphanumeric character on a special writing pad and the machine recognition is fed back to the children via audio-visual displays for learning reinforcement[53,54]. This second type of problem pertains to the on-

*Parenthetical references placed superior to the line of text refer to the bibliography.

line handwritten character recognition in real time. The requirement that measurement and feature extraction take place when the handwriting is being produced is a very distinctive aspect of the problem.

One of the important works on automatic recognition of handwriting was started by Frishhopf and Harmon[23,37]. Their approach relied on the study of the motion of the writing instrument as a function of time. The continuous horizontal and vertical displacement information was derived through the use of a captive stylus of a telewriter. With several constraints of the system, the features of handwritten letters were composed of vertical extrema, retrograde strokes, cusps, and special marks. Based upon these features, the handwritten characters were recognized according to a decision tree. At the same time, Eden undertook the study of handwriting generation and recognition under the stimulation of what is going on in the speech recognition[14,38,39,48]. He developed a formal description of cursive writing based on a set of four primitive stroke symbols and defined rules for how these strokes are transformed by rotation and reflection and translated up and down so that each letter can be defined as a unique finite sequence of strokes. A somewhat qualitative sinusoidal description of actual writing dynamics was also developed in which the strokes were described by segments of periodic functions. Freeman developed an encoding method of geometric curves by a chain of quantized slopes[15, 18-22]. He also proposed three different transformations - directionality, asymmetry, and curvature - from which a variety of pattern properties can be derived. Important characteristics of encoded curves such

as length, width, height, symmetry about various axes, area moments, and center of gravity can be calculated. These features have been used in a contour correlation algorithm for digital curve matching or recognition. Kuhl derived from Freeman's peripheral code three invariants of handprinted characters, namely, number of free ends, mapping of angular directions, and the absence or presence of high rates of curvature in the peripheral code segments for a character recognition system[36]. This set of features has the advantage of being independent of variations in character size, proportion, thickness, and translation of the images.

Grenaias, et al., studied a recognition system for handwritten numerals by contour analysis, where the time derivatives of the x- and y-axis motions of a character following scan were used[27]. Spinrad studied a scheme of machine recognition of hand-printed block letters, where individual pen strokes were abstracted into straight line segments and then the set of normalized line patterns were to be identified with one member of the possible character set[48]. Teitelman[49] and Brown[7] divided the writing surface of a pad into several regions. As a character was drawn on the pad with a light pen, the information on each region which the pen passed through was transmitted to the recognition system. The list of regions then became a space-time characteristic list for classification purpose. Bernstein[2] and Groner[28] used the RAND tablet as the input device. In Bernstein's scheme of processing the stroke information, corners were detected and marked, the

center of the minimum rectangle defined by the minima and maxima of x and y was defined, the rectangle was divided into five regions and a descriptive string was generated for each stroke. The information on the legitimate successor strokes were linked to that of a previous stroke according to a character dictionary for classification of the complete character. Groner developed a program following a tree structure to test in succession the direction of each stroke in one of the four quantized directions, corners, ending points, the ratio of height to width, etc., and to group sets of strokes into characters.

In conjunction with a teaching and learning project, Zobrak and Sze[53,54] applied the "direction-time" encoding method to provide for each handwritten character a feature vector whose successive components represent the quantized direction of the pen motion during sequential increments of time. They essentially used the discriminant function approach for character recognition and proposed a training algorithm which was experimentally tested on ten sets of thirty-five handwritten alphanumeric characters where eight quantized direction numbers and ten feature vector components were used. Their preliminary result gave an indication that this approach is simple, fast, and able to recognize reasonably well-drawn characters with a low error rate. Following their development, one part of this dissertation is to extend and modify, in theory and experiment, this approach of on-line handwritten character recognition.

1.2. Review of Deterministic Multi-Class Pattern Classification Algorithms

The character recognition problem involves, of course, multi-class pattern classification. In general, if the R-class patterns are linearly separable, there exist R augmented weight vectors $\underline{w}_j$ to construct R discriminant functions $g_j(x) = \underline{x}^t\underline{w}_j$, $(j = 1,2,\ldots,R)$, such that $g_j(\underline{x}) > g_i(x)$ for all $i \neq j$ and all $\underline{x}$ belonging to the class C_j. The classical, deterministic training algorithms for multi-class pattern classification of perceptron type have been well presented in the book by Nilsson[43]. Chaplin and Levadi formulated another set of inequalities, based upon the concept of equilateral simplex, to represent the linear separation of R-class patterns; thus, they presented a generalized linear threshold decision algorithm for multiple classes[8].

It is conceivable that any 2-class algorithm can be applied to R-class pattern classification via the following two procedures: (a) separate pairwise each class from every other class in a certain sequence, and (b) separate in a sequence each class from all the rest if the patterns are so constrained that such a linear separation exists. Among the currently available 2-class training algorithms of either iterative type or else[17,32,43,47], it is well recognized that the Ho-Kashyap dichotomization algorithm has relatively high rate of convergence for linearly separable patterns and, in addition, also provides for a test of linear separability of the given training patterns. The extension of the Ho-Kashyap algorithm to multi-class pattern classification has been made by Blaydon[4], Wee and Fu[50]. Wee and Fu

developed two R-class training algorithms: the complete generalization and the generalization with constraint. Their complete generalization algorithm involves a $(R-1)N \times RD$ pattern matrix; their generalization with constraint involves an $N \times D$ pattern matrix. Blaydon proposed a generalization based upon Chaplin and Levadi's concept of mapping R pattern classes into R vertices of a $(R - 1)$-dimensional equilateral simplex with its centroid at the origin; but there is no convergence proof for his algorithm. In the second part of this dissertation, a new multi-class training algorithm is developed and its convergence proved following the above-mentioned mapping concept. It is a complete generalization involving only an $N \times D$ pattern matrix. It can also be used to test the nonlinear separability of the direction-time feature vectors of a set of handwritten alphanumeric characters, as described in the first part of the thesis, and thus to confirm the requirement of using a nonlinear algorithm.

1.3. Objectives of this Research

Following the above discussion, the main objectives of this dissertation are twofold:

(a) to prove a convergence theorem for the Zobrak-Sze algorithm, and to make an extensive simulation study of on-line handwritten character recognition of children's handwriting with some modification of encoding scheme for feature vectors;

(b) to propose a new and efficient training algorithm for multi-class pattern classification which is a generalization of the Ho-Kashyap algorithm.

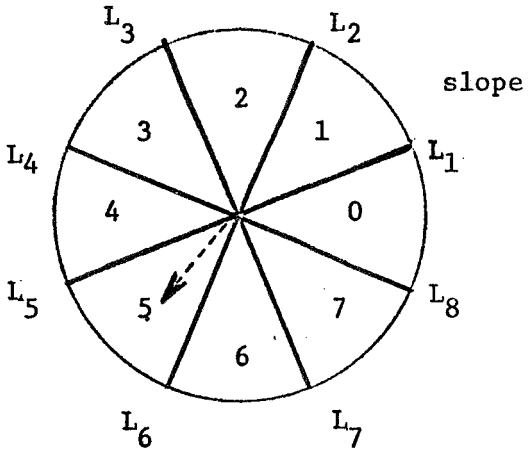
The feature extraction of the children's handwritten characters and the property of feature components are discussed in Section 2.0. The convergence proof of the Zobrak-Sze algorithm and experimental results on the simulated, on-line handwritten character recognition are given in Section 3.0. A new multi-class training algorithm and its convergence proof are presented in Section 4.0; some experimental work demonstrating its advantages and disadvantages is discussed in Section 5.0.

2.0 FEATURE EXTRACTION FOR ON-LINE HANDWRITTEN CHARACTER RECOGNITION

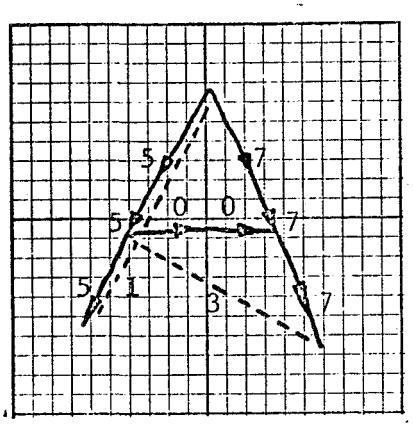
2.1 Review of Direction-Time Encoding Method

In this section, the quantized direction sequences of handwritten alphanumeric characters will be studied. As mentioned in Section 1.1, this encoding method was originated by Freeman and succeedingly developed by Zobrak and Sze for extracting some sequential information of handwritten characters for on-line recognition. This method is briefly described in the following.

Let one write an alphanumeric character on a uniformly masked square. The horizontal and vertical coordinates of the strokes in a character are sequentially stored as the character is being written. The "direction-time" encoding method provides a d-dimensional pattern vector $\underline{x}$ for each handwritten character. The components x_1, x_2, . . ., and x_d, each of which is an integer ranging from 0 to m, represent the directions of the strokes or the segments of strokes in succession. Let the starting point of a segment be considered as an origin and its circular neighborhood be divided into m + 1 equal sectors, each sector is coded by an integer, 0, 1, 2, ..., or m (where m + 1 is an even number), in the counterclockwise sequence as illustrated in Fig. 2.1(a). Thus, the jth sector is bounded by radial lines L_j and L_{j+1} except j=0 where the 0th sector is bounded by L_{m+1} and L_0. If the stroke segment lies within the jth sector or coincides with L_{j+1}, the direction of this segment is encoded by the number j (j=0, 1, 2, ..., m). For the i-th segment in a given character, $x_i = j$. Therefore, if d

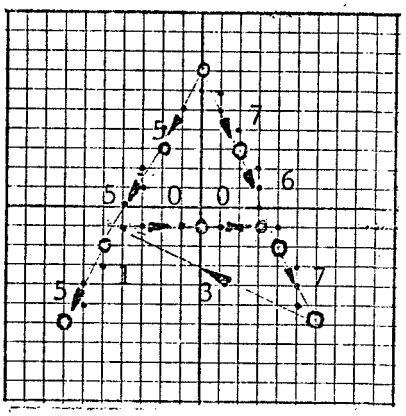


(a) Direction number of stroke segment m = 7



Pattern vector $\underline{X}$ = (5,5,5,1,7,7,7,3,0,0,)

(b) An example with d = 10 and m = 7



Coordinate sequence $[(X_i, Y_i)]$
= [(10,17), (10,16), (09,15), (08,14), (08,13), (07,12), (07,11), (06,10), (06,09), (05,08), (05,07), (04,06), (04,05), (03,04), (00,00), (10,17), (11,16), (11,15), (12,14), (12,13), (13,12), (13,11), (13,10), (14,09), (14,08), (15,07), (15,06), (15,05), (16,04), (00,00), (06,09), (07,09), (08,09), (09,09), (10,09), (11,09), (12,09), (13,09), (00,00), (00,00)]

Pattern vector $\underline{x}_A$ = [5,5,5,1,7,6,7,3,0,0]

(c) The quantized direction-time features.

Fig. 2.1 Direction-time Encoding of a Handwritten Character "A".

is chosen appropriately, every character can be broken down into a sequence of s strokes or segments of strokes, where $s \leq d$. When the pen point is lifted up and moved from the end of one stroke to the starting point of the next stroke, a special component x_q defined as a jumping component is assigned which denotes the direction of the virtual line segment from the end point of the first stroke to the starting point of the next stroke. This concept is illustrated in Fig. 2.1(b) for a handwritten "A" with d = 10, m = 7, and s = 3.

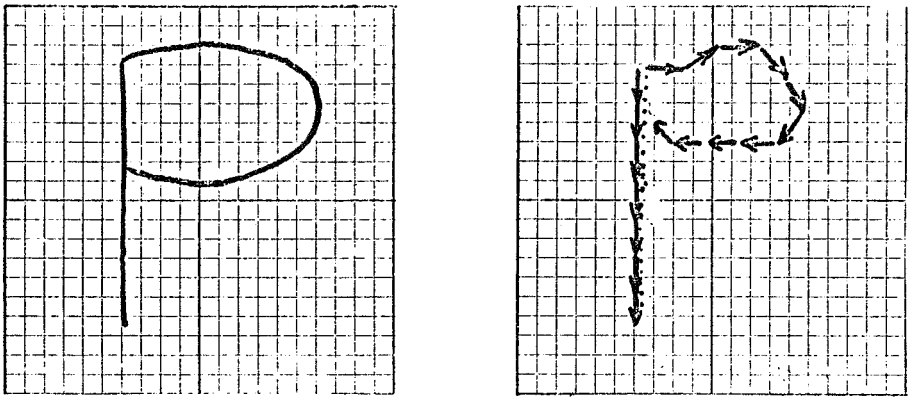
On a practical writing pad, the strokes of the character interset with the mask lines at various points. For each intersecting point, the mask node or grid which lies closest to it is selected. The (X, Y) coordinates of these grid-interesections along every stroke are sequentially stored as the character is being written. This is illustrated in Fig. 2.1(c). In the illustrated coordinate sequence, (00,00) is placed to indicate the separation between two successive strokes, while [(00,00), (00,00)] indicate the end of the character. Note that the origin as well as three other corners on the writing pad is never used as a grid intersection. The quantized direction-time features calculated from this sequence of grid intersection coordinates are also indicated in Fig. 2.1(c), where $\underline{x}_A$ represents the pattern vector of A composed of these features.

2.2 Modification of Direction-Time Features

The above mentioned encoding technique was used by Zobrak and Sze to obtain 10-dimensional pattern vectors for 10 sets of thirty-five handwritten characters, A, B, C, . . ., . . ., X, Y, Z, 1, 2, . . . 9. They also proposed a multi-class training algorithm for on-line handwritten character recognition based upon such pattern vectors. Using all ten sets as training samples they were able to reduce the error rate to 6% after 15 training cycles[52], yet there was no indication of convergence. It was observed that possible ambiquities occurred in the following pattern groups: A, H, 5; D, P; R, K; V, X; and Q, G, C, O, S. This led one to consider the modification of the direction-time encoding method.

The ambiguity between some R and K patterns was introduced by small dimensionality of pattern vectors. This difficulty can be overcome by increasing the dimension by two-fold. For example, for two specific samples R and K, $\underline{x}_R$ = [6, 6, 6, 6, 2, 7, 5, 0, 7, 6] and $\underline{x}_K$ = [6, 6, 6, 6, 2, 5, 5, 1, 6, 6] if d = 10; there are only three distinct pattern components between $\underline{x}_R$ amd $\underline{x}_K$. However, if d = 20, $\underline{x}_R$ = [6, 6, 6, 6, 6, 6, 6, 6, 2, 0,0, 6, 4, 4, 0, 7, 7, 6, 7, 5] and $\underline{x}_K$ = [6, 6, 6, 6, 6, 6, 6, 6, 2, 5, 5, 5, 5, 1, 6, 6, 6, 6, 6, 6]; in this case, there are ten distinct pattern components to enhance the separation between these two classes. Hence, in what follows, a 20-dimensional pattern space will be used for the feature extraction.

Indeed, the quantized direction-time feature vector provides the advantage of being relatively insensitive to the variations in character size, proportion, and translation. Its simplicity may, at times, cause fuzzy distinction between some simple character pairs whose writing dynamics appear similar, for example, Q and G, V and X, and C and O. In this connection, two types of special direction components can be incorporated into the pattern vectors to improve the pattern separability. One is termed reference component and the other is end component. A reference component is defined as the quantized direction from the starting point of the first stroke to the beginning of another stroke. Hence, there is one reference component after every jumping component. The end component is defined as the quantized direction from the starting point to the end point of the character. With the introduction of the end component, the difference between pattern vectors $\underline{x}_O$ and $\underline{x}_C$, $\underline{x}_V$ and $\underline{x}_X$, or $\underline{x}_H$ and $\underline{x}_5$ is increased. With the aid of the reference component, the distinction between pattern vectors $\underline{x}_Q$ and $\underline{x}_G$ is also improved as demonstrated by sample patterns $\underline{x}_Q$ = [3, 5,5, 6, 5, 6, 6, 7, 0, 1, 1, 2, 2, 2, 3, 2, $\underline{6}$, $\underline{6}$, 6, $\underline{6}$,] and $\underline{x}_G$ = [3, 4, 5, 5, 5, 6, 6, 7, 7, 7, 0, 0, 1, 2, 3, $\underline{4}$, $\underline{5}$, 0, 0, $\underline{6}$],where eleven out of twenty pattern components are different. Furthermore, with the inclusion of reference components, the character may be reconstructed from the direction-time pattern vector provided that the number of direction components assigned to each stroke is also known; this is illustrated in Fig. 2.2. With these modifications on the direction-time encoding method, there are [d-2(s-1)-1] pattern components which are actually derived from the direction of stroke



The original character The reconstructed character

$\underline{x}_p$ = [6,6,6,6,6,6,6,$\underline{2}$,$\underline{0}$,0,1,0,7,7,5,4,4,4,3,$\underline{6}$]

The direction components $\underline{2}$, $\underline{0}$, $\underline{6}$ marked by underlines, represent the jumping component, reference component, and end component respectively.

Fig. 2.2. Reconstruction of a Character From Its Modified Direction-time Pattern Vector and Supplementary Information.

Let a_i be the number of pattern components which are derived from the i-th stroke. a_i is initially determined by

$$a_i = \text{Integer part of } \{[d - 2(s-1)-1] \frac{p_i}{p} + 0.5\} \tag{2.2}$$

A proper way of dividing strokes into segments requires that

$$\sum_{i=1}^{s} a_i = d - 2(s-1) - 1 \tag{2.3}$$

In many instances, this does not hold initially because of the integer quantization involved in (2.2); instead,

$$\sum_{i=1}^{s} a_i - [d - 2(s-1) - 1] = \pm Q \neq 0 \tag{2.4}$$

where Q is a positive integer which should be reduced to zero by changing Q of the a_i's which contribute larger quantization errors in (2.2). If two or more a_i's contribute equal amount of quantization errors and if only one remains to be adjusted, the one corresponding to the smallest i will be selected. Each of the Q a_i's must be decreased by one if it is + Q in (2.4) and increased by one if it is -Q. With these adjusted values of a_i's, the condition (2.3) will be satisfied. In all cases, however, $a_i \geq 1$ must be maintained lest to neglect any short stroke, for example, in G and Q. The average segment length for the i-th stroke is then given by

$$b_i = \frac{1}{a_i} (p_i - 1) \tag{2.5}$$

Beginning from the first grid intersection of the i-th stroke, the end point of the j-th segment in the i-th stroke is selected by rounding off $[1 + j\ b_i]$, $(j = 1, 2, \ldots, a_i)$, which indicates its order in the sequence of grid intersections of the i-th stroke. In this way, the last grid intersection of each stroke is naturally the end point of the last segment on the i-th stroke. The quantized direction of each segment can be easily determined according to the rule specified in Section 2.1.

The following remarks should be noted with regard to the computation of the pattern components. In some instances, the starting point of the first stroke and that of another actually coincide and hence the corresponding reference component becomes undefined. For this reason, it is suggested to always use the second grid intersection of each stroke other than the first one to determine the corresponding reference component along with the starting point of the character. In cases of characters with a single closed stroke, like 0 and 8, where the starting point and the end point coincide, the direction number of the end component would be derived from the ratio $\frac{0}{0}$ which is indeterminate. For the sake of convenience, it is arbitrarily set at infinity which results in a direction number 2. The last remark deals with the situation where the total number p of grid interesections of a character is less than or equal to the dimension d. It is recalled that the above discussed encoding procedure presumes the condition $p > d$. In case of $p \leq d$, an enlargement and interpolation process is necessary to facilitate the segmentation and calculation of d pattern components. Assume that both X-scale and Y-scale of the character which has s strokes are increased by two-fold. Let the

original grid intersections (X_i, Y_i) be mapped into the new coordinates (X'_j, Y'_i) and let one interpolation be taken between every two successive grid intersections of each stroke. For the j-th stroke with p_j grid intersections, there will be $2p_j-1$ points after enlargement and interpolation. In total, there will be 2p-s points available for pattern component determination if (2p-s) > d. Let ℓ_j denote the total number of grid intersections of the first j strokes, (j = 1, 2, . . ., s),

$$\ell_j = \sum_{n=1}^{j} p_n \tag{2.6}$$

The original coordinates of the k-th grid intersection of the j-th stroke, i.e., (X_i, Y_i) with

$$i = \sum_{n=1}^{j-1} p_n + k$$

$$= \ell_{j-1} + k, \quad \text{for } k = 1, 2, \ldots, p_j,$$

are transformed into the new coordinates, (X'_r, Y'_r), where

$$r = \sum_{n=1}^{j-1} (2p_n - 1) + (2k-1)$$

$$= 2(\ell_{j-1} + k) - j$$

$$= 2i - j,$$

as the (2k-1)-th points of the j-th stroke after enlargement and interpolation; $X'_r = 2X_i$ and $Y'_r = 2Y_i$. The coordinates of the interpolated points, (X'_t, Y'_t), are determined from $X'_t = X_i + X_{i-1}$ and $Y'_t = Y_i + Y_{i-1}$ where

$$t = \sum_{n=1}^{j-1} (2p_n - 1) + (2k-2)$$

$$= 2\,(\ell_{j-1} + k) - j - 1$$

$$= 2i - j - 1 \qquad \text{for } k = 2, \ldots, p_j.$$

Hence, there will be in total 2p-s points available for pattern components determination if (2p-s) > d. If (2p-s) is still less than or equal to d, the same procedure can be repeated until the required condition is satisfied.

2.4 Data on Fifty Sets of Handwritten Characters

A digital computer program has been written in MAD language for processing the feature extraction based upon the modified direction-time encoding method. The flow chart and the complete program listing are given in Appendix A.

Fifty sets of handwritten alphanumeric characters were collected. These sets were written by fifty individuals; twenty-seven of them are school children with ages ranging from seven to eleven years old, and the rest are students at college level. Each character was written, in a prescribed standard sequence of strokes, on a 2" x 2" square with 20 x 20 divisions. Guided by the discussion in Section 2.2, the following ten characters were chosen for the present study: A, C, D, G, H, O, P, Q, S, and 5. They are respectively designated as class 1 through class 10. These fifty sets of characters have been processed by the use of the above mentioned computer program to obtain the corresponding sets of quantized

direction-time pattern vectors with the choice of d = 20 and m = 7. These 500 characters together with their corresponding pattern vectors are shown in appendix B. These data have been used for the simulation study of the on-line handwritten character recognition which is to be discussed in Section 3.0.

3.0 THE CONVERGENCE PROOF OF A LEARNING ALGORITHM AND EXPERIMENTAL RESULTS OF AN ON-LINE HANDWRITTEN CHARACTER RECOGNITION SYSTEM

3.1 An On-line Handwritten Character Recognition System

An on-line handwritten character recognition system will be discussed in the following. The recognition system consists of two parts: feature extractor and classifer, as shown in Fig. 3.1 . The extraction of quantized direction-time features of on-line handwritten characters has been investigated in Section 2.0 . The pattern components x_i's are a set of integers of modulo $m + 1 = 8$ and form a special cyclic group. It has been demonstrated that the handwritten alphanumeric characters as represented by their direction-time features are not linearly separable, as will be shown later in Section 5.2. Therefore, a nonlinear processor is used to facilitate the classification. One set of characters is stored as the set of prototype patterns. The pattern vector for any unknown character is compared with every prototype pattern to form a set of metrices,which are nonlinear function of the pattern components, within the classifier. The discriminant functions are used to classify the unknown character based upon the concept of minimum metric. A training algorithm originally proposed by Zobrak and Sze[53] is adopted here with only a slight modification following the Duda-Fossum multi-class training rule. This learning algorithm will be reviewed in Section 3.2 . The convergence of the algorithm, which was not discussed by Zobrak and Sze, will be proven in Section 3.3 . Experimental results will be described

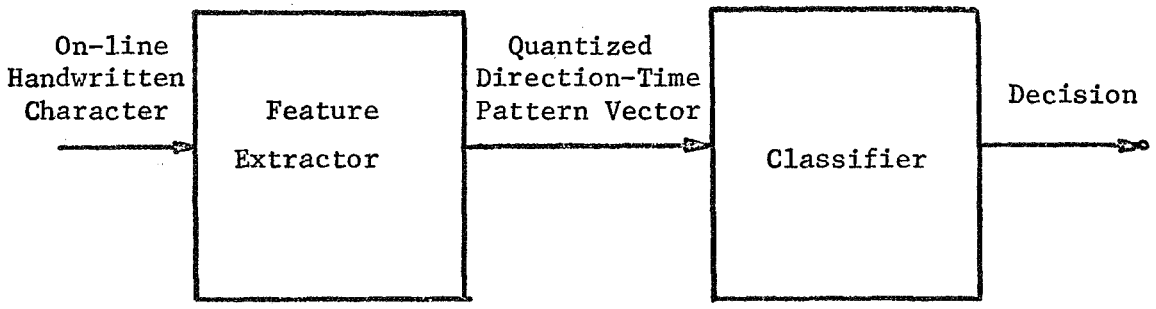


Fig. 3.1 The Block Diagram of an On-line Handwritten Character Recognition System.

in Section 3.4 .

3.2 A Learning Algorithm

The learning algorithm of Zobrak and Sze in its modified form is described in the following. Let a certain set of R handwritten characters be chosen as the set of R prototype patterns, which are encoded by quantized direction-time features and represented by R pattern vectors $\underline{P}_n$'s (n=1,2, . . . , R) . $\underline{P}_n$ is of dimension d and p_{ni} (i = 1, 2, . . . , d) is its i-th component. Let $\phi_i(\underline{P}_n, \underline{x})$ be a measure of the difference between the i-th component of the prototype pattern $\underline{P}_n$ and that of the pattern $\underline{x}$ under consideration; $\phi_i(\underline{P}_n, \underline{x})$ is a metric and is defined by

$$\phi_i(\underline{P}_n, \underline{x}) = \begin{cases} | p_{ni} - x_i | & \text{if } | p_{ni} - x_i | \leq \frac{m+1}{2} \\ (m+1) - | p_{ni} - x_i | & \text{if } | p_{ni} - x_i | > \frac{m+1}{2} \end{cases} \tag{3.1}$$

$$(i = 1, 2, . . . , d ; \quad n = 1, 2, . . . , R)$$

where m + 1 is an even integer representing m + 1 quantized direction numbers. Each $\phi_i(\underline{P}_n, \underline{x})$ is chosen as such because, as mentioned before , x_i's are a set of integers of modulo m + 1 and form a cyclic group. Let $\underline{\phi}(\underline{P}_n , \underline{x})$ be a d-dimensional vector consisting of $\phi_i(\underline{P}_n , \underline{x})$, (i = 1, 2, . . . , d), as its components,

$$\underline{\phi}(\underline{P}_n, \underline{x}) = [\phi_1(\underline{P}_n, \underline{x}), \phi_2(\underline{P}_n, \underline{x}), . . . , \phi_d(\underline{P}_n, \underline{x})]^t,$$

and $\underline{w}_n$ be a d-dimensional weight vector for the prototype pattern $\underline{P}_n$,

$$\underline{w}_n = (w_{n1} , w_{n2} , \ldots , w_{nd})^t$$

The R discriminant functions for R classes are defined by

$$g_n (\underline{P}_n , \underline{x}) = \underline{w}_n^t \cdot \underline{\phi}(\underline{P}_n , \underline{x}) \qquad (3.2)$$
$$(n = 1, 2, \ldots , R)$$

$g_n (\underline{P}_n, \underline{x})$ is also a metric defined in the finite and discrete pattern space. The decision is made according to the minimum metric. For a fixed set of weight vectors $\underline{w}_n$, (n = 1, 2, . . . ,R), the handwritten character represented by $\underline{x}$ is classified as the character represented by the prototype pattern $\underline{P}_r$, that is, in the class C_r, if

$$g_r (\underline{P}_r , \underline{x}) < g_j (\underline{P}_j , \underline{x}) \quad \text{for all } j = 1,2,\ldots,R \text{ but } j \neq r \qquad (3.3)$$

During the k-th step in the learning phase , if the pattern $\underline{x}(k)$ belongs to the class C_r and $g_r(\underline{P}_r,\underline{x}) \geq g_j(\underline{P}_j,\underline{x})$ for some $j \neq r$ in a non-empty set of integer J, the correction of the set of weight vector $\underline{w}_h$, (h =1, 2, . . . , R), is governed by the following training rule :

$$\underline{w}_h (k + 1) = \underline{w}_h (k) + \delta\underline{w}_h (k)$$

$$= \begin{cases} \underline{w}_r (k) - a_k \underline{\phi} (\underline{P}_r, x(k)) & \text{for } h=r \\ \underline{w}_j (k) + \alpha_{jk} a_k \underline{\phi}(\underline{P}_j, \underline{x}(k)) & \text{for } h=j\neq r \text{ and } j\in J \\ \underline{w}_h (k) & \text{otherwise} \end{cases} \qquad (3.4)$$

where

$$0 < a_{min} \leq a_k \leq a_{max}$$
$$\alpha_{jk} > 0 \quad \text{and} \quad \sum_{j \in J} \alpha_{jk} = 1. \tag{3.5}$$

and w_{hi} (k+1) will be set to an arbitrary small positive number $\delta>0$ for any correction which will cause $w_{hi}(k+1)$ to be negative.

3.3 Convergence Proof of The Learning Algorithm

Assume that the patterns are separable by the discriminant function (3.2) . Let the learning algorithm (3.4) begin with any set of initial weight vectors $[\ \underline{w}_1(1), \underline{w}_2(1), \ldots, \underline{w}_R(1)\ ]$. Then, for some finite iterations k^* ,

$$\underline{w}_i\ (k^*) = \underline{w}_i(k^* + 1) = \underline{w}_i\ (k^* + 2) = \ldots\ ,$$
$$(i = 1, 2, \ldots, R)$$

which gives a set of solution weight vectors, that is ,

$$\underline{w}_r{}^t\ (k^*).\ \underline{\phi}\ (\underline{P}_r, \underline{x}) < \underline{w}_j{}^t\ (k^*).\ \underline{\phi}\ (\underline{P}_j, \underline{x})$$
$$(j = 1, 2, \ldots, R\ ;\ j \neq r)$$

for all $\underline{x}$ belonging to the Class C_r, (r = 1, 2, . . . , R).

The convergence proof of the learning algorithm is primarily based on the concept of minimum metric. It is recalled that $\phi_i(\underline{P}_n,\underline{x})$ defined in (3.1) gives the ith component of the deviation of $\underline{x}$ from $\underline{P}_n$ and the discriminant function $g_n(\underline{P}_n,\underline{x}) = \underline{w}_n{}^t.\ \phi(\underline{P}_n,\underline{x})$, (n = 1, 2, . . ., R), are metrics defined in the finite and discret pattern space.

Since it is assumed that the patterns $\underline{x}$'s are separable by these discriminant functions $g_n(\underline{P}_n, \underline{x})$, there exists a set of solution weight vectors, $\underline{w}_n$'s , (n = 1, 2, . . . , R), such that for $\underline{x}$ in the class C_r, (r = 1, 2, . . . , R),

$$\underline{w}_r^t(k) \cdot \underline{\phi}(\underline{P}_r, \underline{x}) < \underline{w}_j^t(k) \cdot \underline{\phi}(\underline{P}_j, \underline{x}) \qquad (3.6)$$
$$(\text{for all } j = 1, 2, \ldots, R \; ; \; j \neq r)$$

Therefore, $\underline{x}$'s are ϕ-separable.

For $\underline{x}(k)$, the k-th pattern in the training sequence leading to a correction, the error-correction scheme in (3.4) may be viewed as a set of nonlinear transformations, $\underline{\phi}(\underline{P}_j, \underline{x}(k))$, (j= 1, 2, . . . , R), followed by a linear classifier with a fixed increment error-correction rule given by

$$\underline{V}(k+1) = \underline{V}(k) + \underline{Z}(k) \qquad (3.7)$$

where

$$\underline{V}(k) = \text{a Rd-dimensional weight vector}$$
$$= [\underline{w}_1^t(k), \underline{w}_2^t(k), \ldots, \underline{w}_R^t(k)]^t \qquad (3.8)$$

$$\underline{Z}(k) = \text{a 2-class Rd-dimensional pattern vector}$$
$$= [\underline{0}^t, \ldots, \underbrace{-a_k \underline{\phi}^t(\underline{P}_r, \underline{x}(k))}_{r\text{th block}}, \underline{0}^t, \ldots, \underbrace{\alpha_{jk} a_k \underline{\phi}^t(\underline{P}_j, \underline{x}(k))}_{j\text{th block}},$$
$$\ldots, \underbrace{\alpha_{qk} a_k \underline{\phi}^t(\underline{P}_q, \underline{x}(k))}_{q\text{th block}}, \ldots, \underline{0}^t, \ldots]^t \qquad (3.9)$$

$a_k > 0$, $\alpha_{jk} > 0$, and $\sum_{j \in J} \alpha_{jk} = 1$, and j,q, etc. belong to a set

J for which

$$\underline{w}_r^t(k)\cdot\underline{\phi}(\underline{P}_r,\underline{x}(k)) \geq \underline{w}_i^t(k)\cdot\underline{\phi}(\underline{P}_i,\underline{x}(k))$$
$$(i = j, q, \ldots, \varepsilon J)$$

This procedure can be illustrated in Fig. 3.2 . Note that

$$\underline{V}^t(k)\,Z(k) = [\,\underline{w}_1^t(k),\ \underline{w}_2^t(k),\ \ldots,\ \underline{w}_R^t(k)\,]\begin{bmatrix} \\ \\ \\ -a_k\,\underline{\phi}(\underline{P}_r,x(k)) \\ \\ \\ \alpha_{jk}a_k\underline{\phi}(\underline{P}_j,\underline{x}(k)) \\ \\ \\ \alpha_{qk}a_k\underline{\phi}(\underline{P}_q,\underline{x}(k)) \\ \\ \end{bmatrix}$$

$$= a_k\,[\,-\underline{w}_r^t(k)\underline{\phi}(\underline{P}_r,\underline{x}(k)) + \sum_{j\varepsilon J}\alpha_{jk}\underline{w}_j^t(k)\underline{\phi}(\underline{P}_j,\underline{x}(k))\,] \qquad (3.10)$$

For $\underline{x}(k) \varepsilon C_r$, the correct classification is given by

$$\underline{V}^t(k)\,\underline{Z}(k) > 0 \qquad (3.11)$$

which implies , from (3.10) and the fact $a_k > 0$, that

$$\underline{w}_r^t(k)\cdot\phi(\underline{P}_r,\underline{x}(k)) < \sum_{j\varepsilon J}\alpha_{jk}\underline{w}_j^t(k)\cdot\underline{\phi}(\underline{P}_j,\underline{x}(k)) \qquad (3.12)$$

This follows from the assumption of ϕ-separability of $\underline{x}$, since

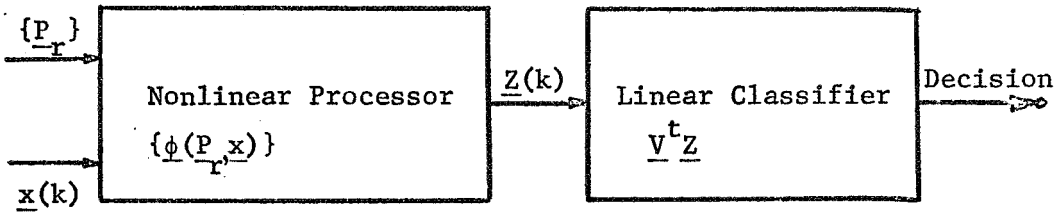


Fig. 3.2 The Interpretation of the Learning Algorithm as a Nonlinear Processor Followed by a Linear Classifier.

$$\alpha_{jk}\underline{w}_r(k).\ \underline{\phi}(\ \underline{P}_r,\underline{x}(k)\ ) < \alpha_{jk}\underline{w}_j(k).\ \underline{\phi}(\ \underline{P}_j,\underline{x}(k))$$

for all $j \in J$

and $\sum_{j\in J} \alpha_{jk} = 1$. On the other hand,

$$\underline{V}^t(\ k\ )\ Z(\ k\ ) = a_k[-\underline{w}_r{}^t(k).\underline{\phi}(\ \underline{P}_r,\underline{x}(k)) + \sum_{j\in J} \alpha_{jk}\underline{w}_j{}^t(k).\underline{\phi}(\ \underline{P}_j,\underline{x}(k)]$$

$$\leq\ 0$$

or

$$\underline{w}_r{}^t(k).\ \underline{\phi}(\ \underline{P}_r,\underline{x}(k)) \geq \sum_{j\in J} \alpha_{jk}\underline{w}_j{}^t(k).\underline{\phi}(\ \underline{P}_j,\underline{x}(k)) \quad (3.13)$$

requires

$$\underline{w}_r{}^t(k).\underline{\phi}(\ \underline{P}_r,\underline{x}(k)) \geq \text{ at least one of } \underline{w}_j{}^t(k)\underline{\phi}(\ \underline{P}_j,\underline{x}(k)) \quad (3.14)$$

which implies that $\underline{x}(\ k\ )$ is incorrectly classified by the set of weight vectors $\underline{w}_i(\ k\ )$, $(\ i = 1, 2, \ldots, R\ )$. Inequality (3.14) can be verified by the argument of contradiction. Suppose that (3.13) is true but

$$\underline{w}_r{}^t(k).\underline{\phi}(\ \underline{P}_r,\underline{x}(k)) \not\geq \text{ at least one of } \underline{w}_j{}^t(k)\underline{\phi}(\ \underline{P}_j,\underline{x}(k))$$

or, in other words,

$$\underline{w}_r{}^t(k).\ \underline{\phi}(\ \underline{P}_r,\underline{x}(k)) < \underline{w}_j{}^t(k).\underline{\phi}(\ \underline{P}_j,\underline{x}(k)) \quad \text{for all } j \in J.$$

Multiply both sides of the above inequality by α_{jk},

$$\alpha_{jk}\underline{w}_r{}^t(k).\ \underline{\phi}(\ \underline{P}_r,\underline{x}(k)) < \alpha_{jk}\underline{w}_j{}^t(k).\ \underline{\phi}(\ \underline{P}_j,\underline{x}(k)).$$

Taking summation on both sides over J, and noting that $\sum_{j\in J} \alpha_{jk} = 1$, one has

$$\underline{w}_r^t(k).\phi\ (\underline{P}_r,\underline{x}(k)) < \sum_{j\epsilon J} \alpha_{jk}\ \underline{w}_j^t(k).\phi\ (\underline{P}_j,\underline{x}(k))$$

which contradicts to the assumption (3.13). Therefore, $\underline{V}^t(k)\underline{Z}(k) < 0$ requires inequality (3.14) to be satisfied which implies that $\underline{x}(k)$ is misclassified and a correction in weight vectors is needed.

Following the above discussion, the ϕ-separability of $\underline{x}$ is equivalent to the linear separability of $\underline{Z}$. Since the learning algorithm (3.4) is equivalent to the algorithm (3.7)

$$\underline{V}(k+1) = \underline{V}(k)+\underline{Z}(k)$$

which is a convergent error correction rule for the linearly separable $\underline{Z}$ [43], it follows that the algorithm (3.4) converges in a finite number of iterations k^* if the training samples $\underline{x}$ are ϕ-separable.

3.4 EXPERIMENTAL STUDIES

Computer simulation studies have been carried out for the proposed on-line handwritten character recognition system. Fifth sets of pattern vectors of ten characters, A, C, D, G, H, O, P, Q, S, and 5, as mentioned in Section 2.4 and listed in Appendix B were used as the input data. The set No.0 was used as the set of prototype patterns and the rest as training patterns or test patterns. These pattern vectors are formed by modified direction-time features with twenty dimensions and eight direction sectors. The learning algorithm (3.4) has been simulated in MAD language on the IBM 7090 digital computer. A description of this computer program is given in Appendix C.

In four experiments described below, the initial weight vectors $\underline{w}_i(1)$, (i=1,2,...,k), were all set to $(1,1,...,1)^t$, and $\alpha_{jk} = \frac{1}{L}$ where L is the number of elements in the set J during each iteration. Example 3.1 is given to illustrate the ϕ-separability of the on-line handwritten characters. Example 3.2 is to show the effects of a_k on the convergence ra rate of the learning algorithm and the effect of the training sample size on the recognition accuracy. In Example 3.3, the modified direction-time feature extraction is shown to be more advantageous than the original direction-time feature extraction as indicated by the faster convergence rate in the training phase. Finally, when the writing sequence of some characters is varied from the prototype, the recognition error is discussed in Example 3.4.

3.41 Example 3.1.

The objective of this experiment was to test the learning algorithm discussed in Section 3.2 and the ϕ-separability of the sample pattern vectors of the on-line handwritten characters provided in Appendix B. The data set No. 0 was considered as the set of prototype patterns. Twenty-five training sets (set No. 1 to No. 25) were used. a_k was chosen to be 0.125 and $\alpha_{jk} = 1/L$ where L is the number of elements in the non-empty set J. As mentioned earlier, the initial weight components were all set to be unity; the initial recognition error rate was 16.8%. After seven traning cycles, the error rate was reduced to zero. The solution weight vectors obtained as shown in Table 3.1, where $\underline{w}_i$'s (i=A,C,D,G,H,O, P,Q,S, and 5) are listed row-wise as 20-dimensional vectors. The convergence rate during the training phase is depicted in Figure 3.3. This set of

Table 3.1 Solution Weight Vectors Obtained for $a_k = 0.125$ and with 25 Training Sets

	Weight Components																			
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
A	1.2	1.5	1.0	0.9	1.7	0.8	0.7	1.1	1.1	0.8	1.1	0.6	1.5	0.5	1.6	0.2	0.8	1.3	0.7	0.6
C	1.1	1.5	0.9	1.3	1.0	1.0	0.9	1.0	1.3	1.1	1.4	0.8	1.3	1.3	1.3	1.6	1.9	1.4	1.1	1.0
D	1.7	1.6	1.8	1.5	1.8	0.4	1.9	1.2	2.3	1.1	1.3	2.1	0.8	0.6	1.1	0.7	0.3	0.3	0.2	1.1
G	0.8	0.6	0.3	0.6	1.4	0.8	1.4	1.1	1.3	1.8	1.6	0.6	0.8	0.9	0.8	0.2	0.1	1.8	2.1	1.7
H	1.2	1.3	0.9	1.5	1.2	1.1	1.4	0.0	0.9	1.2	0.8	1.6	1.2	0.9	0.9	1.5	1.8	1.0	0.8	1.9
O	0.6	0.8	0.3	0.0	0.1	0.1	0.6	0.5	0.9	1.0	0.1	0.7	0.4	0.8	0.5	0.8	0.9	1.6	1.4	2.2
P	1.3	1.2	1.2	1.0	0.7	0.5	0.6	0.0	0.0	1.0	1.7	2.5	0.0	1.4	1.3	1.3	2.1	1.8	2.6	1.0
Q	0.6	0.9	0.8	0.7	1.0	0.4	0.4	0.3	0.3	0.7	1.2	1.2	0.6	0.9	0.0	0.3	0.6	0.6	1.3	1.5
S	0.9	1.0	1.2	1.6	1.4	1.3	0.0	0.9	1.0	0.5	0.4	0.7	1.0	0.9	0.5	1.5	1.0	1.4	1.3	1.2
5	0.9	1.0	0.4	0.9	1.0	0.9	0.8	0.8	0.5	0.6	0.9	1.0	0.6	1.3	1.0	1.1	1.0	1.0	1.0	1.3

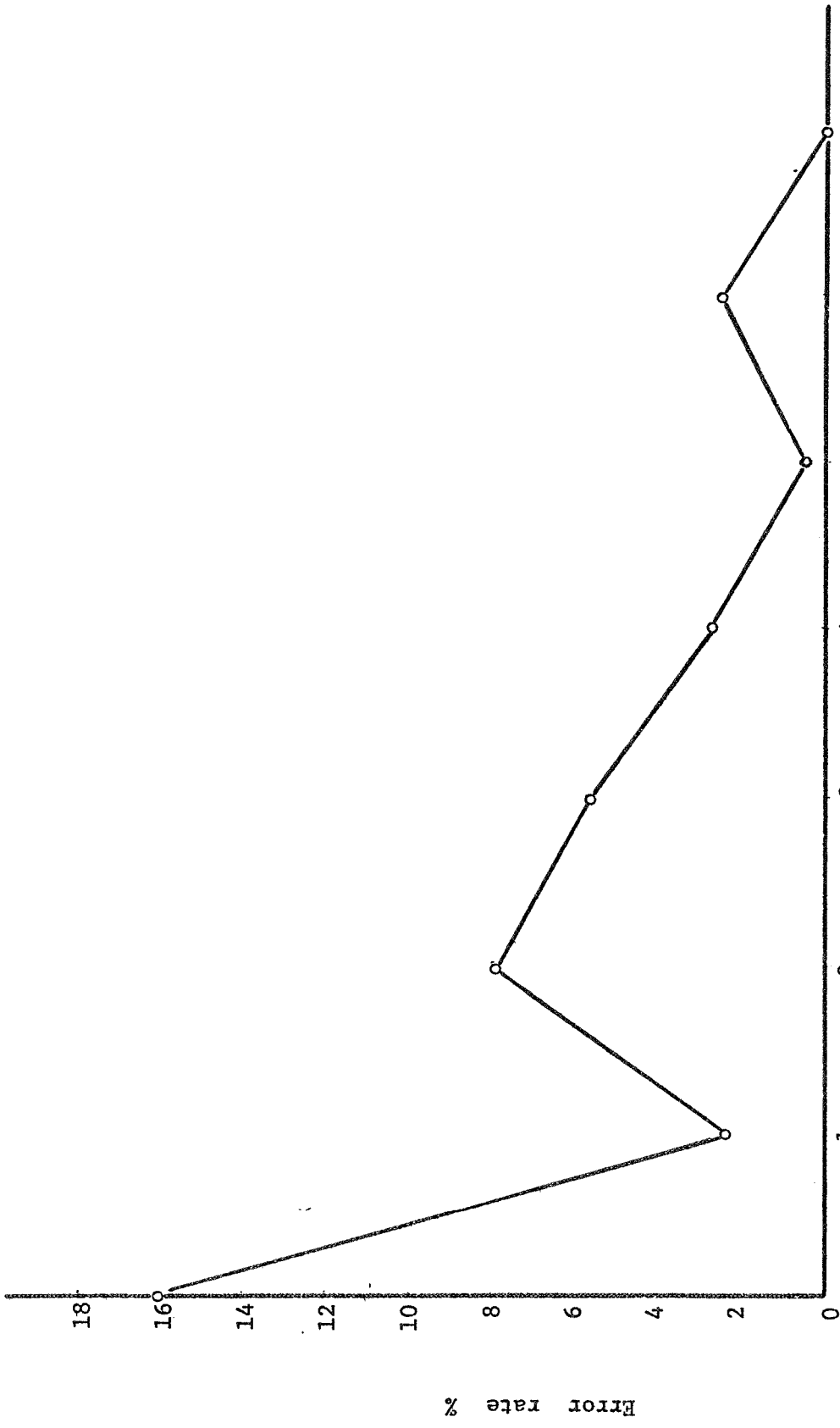


Fig. 3.3 Error Rate During the Training Phase (a_k = 0.125, No.of Training sets=25).

solution weight vectors was used to recognized all 49 sets of characters listed in Appendix B, the recognition error-rate was found to be 2.86%.

3.42 Example 3.2.

In this example, two experiments were performed to investigate: (i) the effect of the coefficient a_k on the convergence rate of the learning algorithm, and (ii) the effect of the training sample size on the recognition accuracy.

In the first experiment, twenty-five training sets (set No.1 to set No.25) were used, and four different values of a_k were selected: they were 1.000, 0.500, 0.250, and 0.125. The results on the convergence rate in the training phase are shown in Fig. 3.4. For a_k= 1.000, 0.500, 0.250, and 0.125, the number of training cycles required are 11, 11,8, and 7 respectively. This seems to indicate that the faster convergence may be obtained by using smaller a_k. This observation may be caused by the fact that the initial weight vectors are fairly close to the solution weight vectors, so that the smaller incremental correction at each step may easily bring the weight vectors to their respective solution region.

In the second experiment, a_k was chosen to be 0.125, but the number of training sets was varied at 10, 15, 20, 25, 30, and 35. In each of these cases, the training was completed and a set of weight vectors was obtained. By using each set of these weight vectors, the

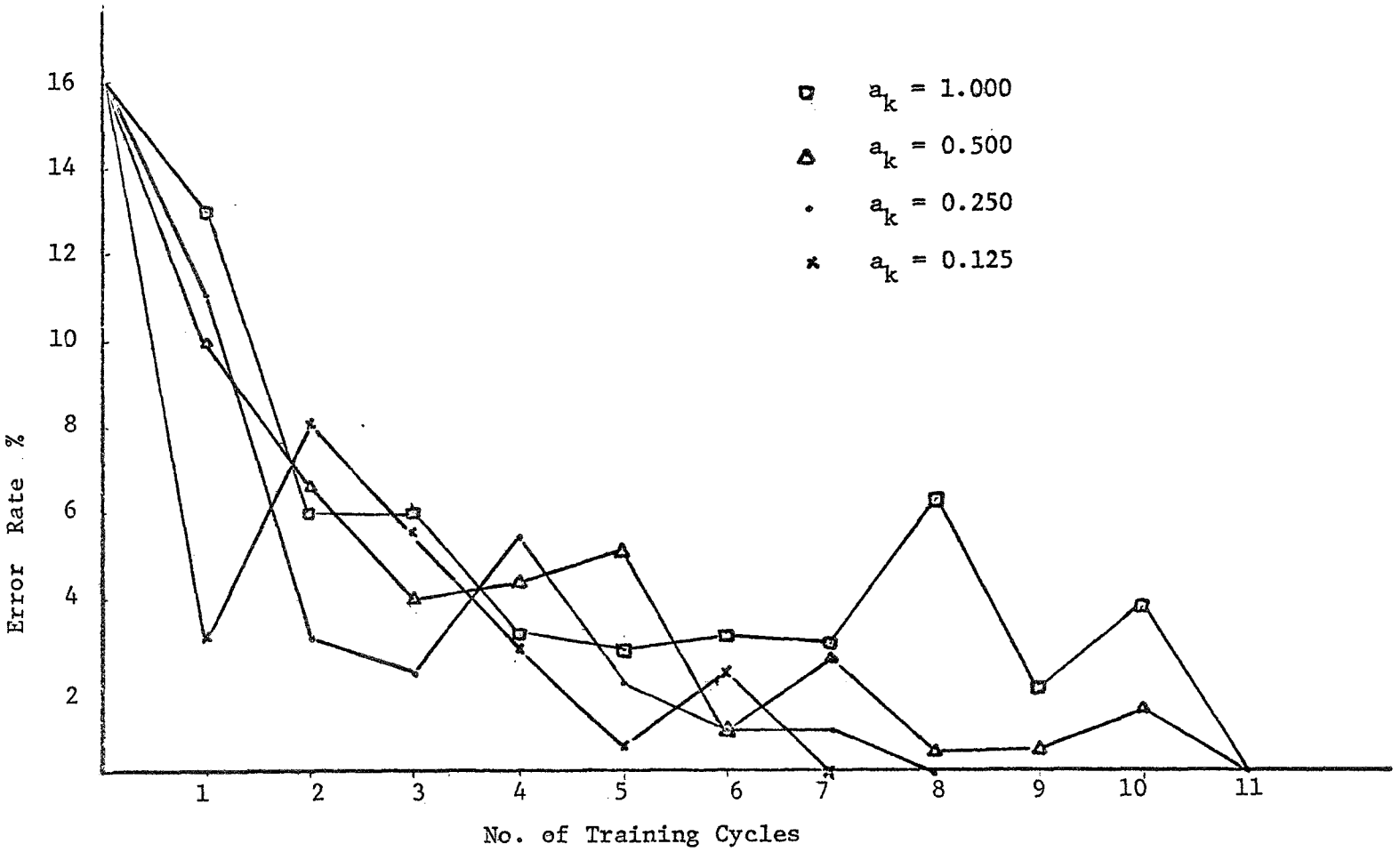


Fig. 3.4 Effect of a_k On the Convergence Rate of the Learning Algorithm.

(No. of training sets = 25)

entire 490 characters (set No.1 to set No.49) were recognized. The recognition error rate versus the corresponding training sample size is shown in Fig. 3.5 . It is obvious, naturally, that the larger the training sample size is, the smaller the recognition error rate will be. The solution weight vectors obtained by using 35 training sets are shown in Table 3.2; the corresponding recognition error rate is reduced to 2.04%. Of course, the computation cost during the learning phase increases in general as the training sample size increases. This is illustrated in Table 3.3.

3.43 Example 3.3.

In this experiment, 11 sets of characters (set No.0 to set No. 10) were encoded by two different feature extraction methods: (i) the modified direction-time features and (ii) the Zobrak-Sze direction-time features. Hence, two different groups of pattern vectors were trained according to the learning algorithm (3.4). Each group had its own prototype set and ten training sets. The training was completed in each case. For the training sets encoded by the Zobrak-Sze direction-time features, it required 10 training cycles to reduce the error to zero, in contrast to 3 training cycles for the sets encoded by the modified direction-time features. The convergence rates for both cases during the training phase are shown in Fig. 3.6. This illustrates that the modified direction-time feature of on-line handwritten characters are indeed better than the original direction-time features.

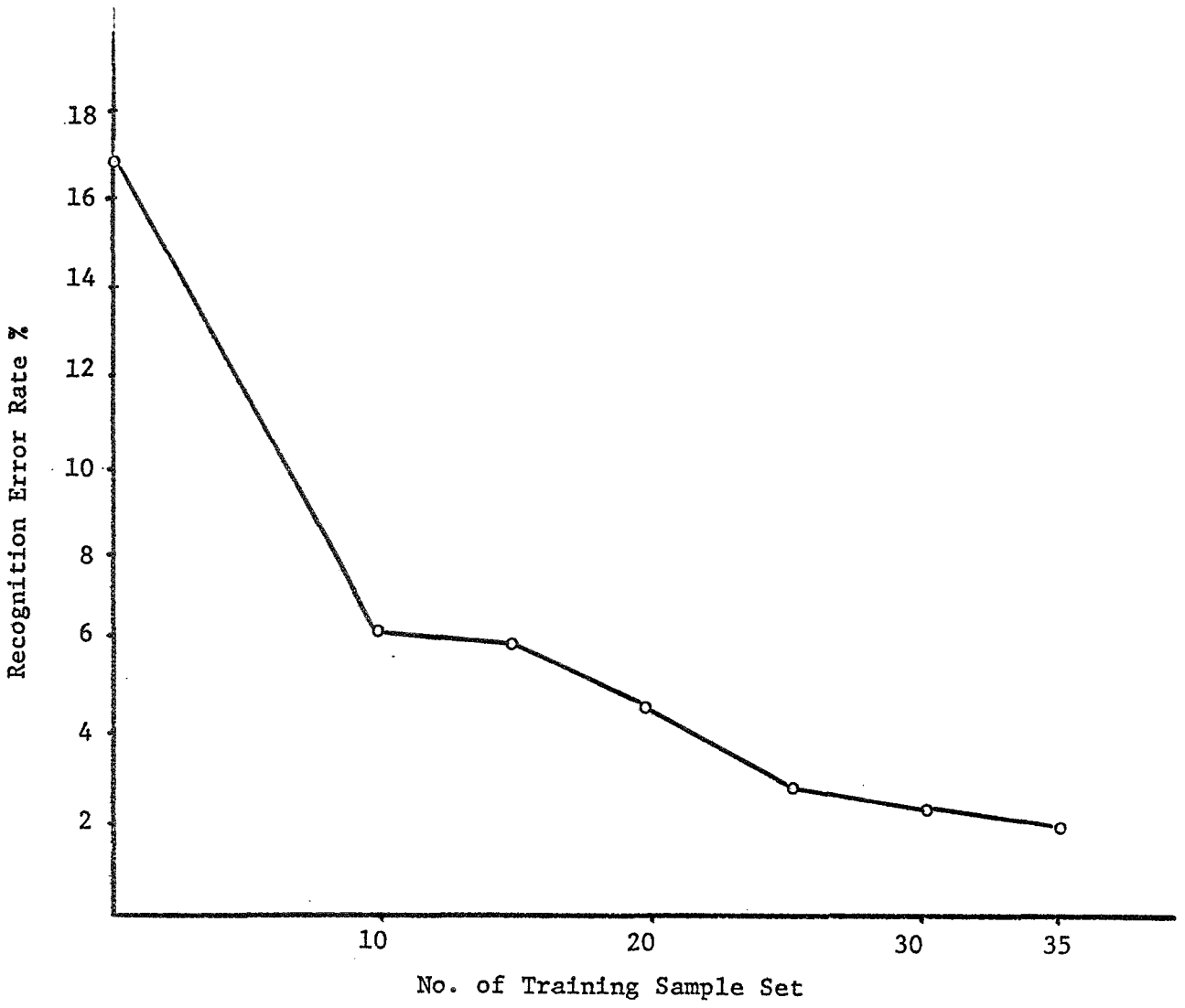


Fig. 3.5 Recognition Error Rate Versus Training Sample Size.

Table 3.2 Solution Weight Vectors Obtained for $a_k = 0.125$ and with 35 Training Sets

	Weight Components																			
	1 *	2 *	3 *	4 *	5 *	6 *	7 *	8 *	9 *	10 *	11 *	12 *	13 *	14 *	15 *	16 *	17 *	18 *	19 *	20 *
A	2.2	2.1	1.0	0.9	1.5	0.9	0.8	1.0	1.8	0.4	1.4	0.5	2.3	1.0	0.5	0.1	0.7	1.8	0.5	0.2
C	1.0	2.3	0.8	1.6	1.0	1.0	0.4	1.0	1.3	1.1	1.4	0.2	1.3	1.2	1.2	1.9	2.4	1.3	1.0	1.0
D	1.7	1.3	1.8	1.3	0.9	0.6	1.9	0.8	2.3	1.4	1.4	2.5	0.8	0.7	0.6	0.8	0.1	0.7	0.6	1.0
G	0.8	0.7	0.0	0.1	1.8	0.8	1.4	1.3	1.2	1.7	1.3	0.5	0.9	1.0	0.3	0.0	0.0	2.0	2.2	1.7
H	1.5	1.7	1.2	0.9	0.4	0.5	2.5	0.0	0.6	1.2	0.6	1.7	2.3	1.1	0.3	1.8	2.0	0.9	0.3	3.3
O	0.3	0.8	0.3	0.0	0.1	0.1	0.5	0.3	0.8	0.9	0.1	0.7	0.7	0.8	0.5	0.9	0.7	1.7	1.4	2.5
P	1.4	1.4	1.1	0.9	1.3	0.2	0.1	0.0	0.1	1.0	1.9	2.8	0.2	1.4	1.7	1.5	2.2	1.4	2.1	1.0
Q	0.6	0.9	0.7	0.7	1.0	0.3	0.4	0.3	0.3	0.9	1.4	1.6	0.7	1.0	0.0	0.1	0.5	0.6	1.3	1.5
S	1.0	1.0	1.2	1.9	1.8	1.3	0.1	1.2	0.7	0.4	0.1	0.2	1.0	0.6	0.1	1.2	1.1	1.5	1.0	1.2
5	0.9	1.3	0.7	0.6	1.3	0.8	0.8	1.0	0.7	0.4	1.0	1.0	0.9	1.4	1.0	1.0	1.0	1.0	1.0	1.5

Table 3.3 Computation Cost During Learning Versus Training Sample Size

Training Sample Size (sets)	10	15	20	25	30	35
Computer Time (IBM 7090, MAD language)	2'23"	4'51.3"	11'10.3"	10'18.4"	13'38.8"	27'42"
Computing Cost	$4.76	$9.70	$22.34	$20.60	$27.28	$55.38
No. of Training Cycles Required	3	5	10	7	8	14

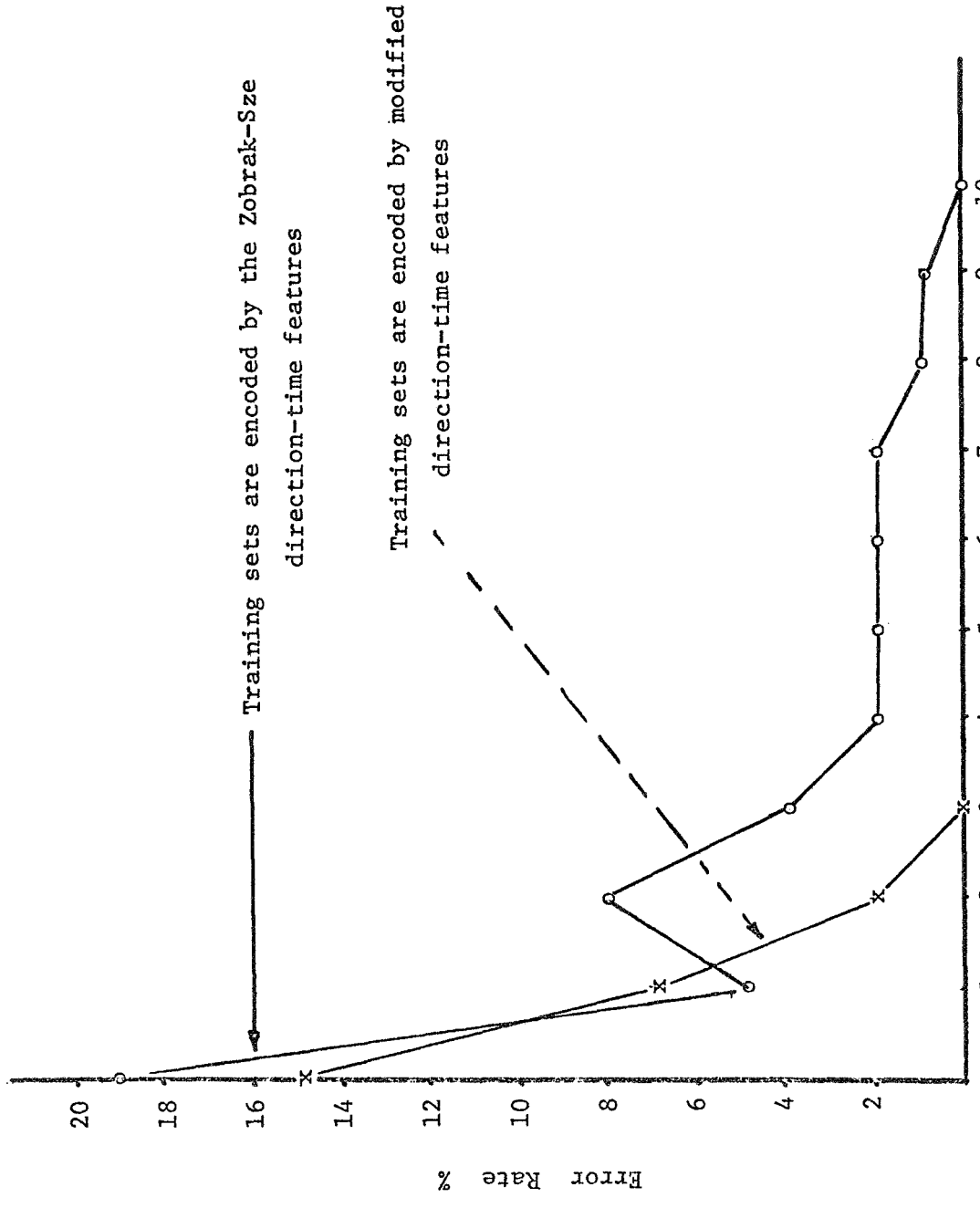
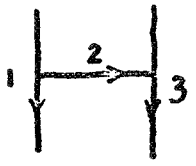


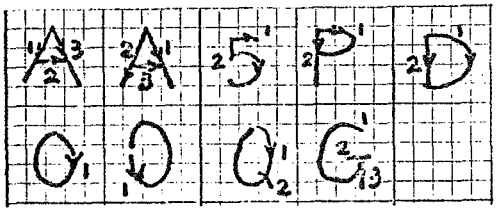
Fig. 3.6 Error Rate During the Training Phase (a_k= 0.125, No. of training sets= 10)

3.44 Example 3.4.

In previous three examples, all the experiments were performed with the pattern vectors listed in Appendix B where each character was written according to a standard stroke sequence. If the writing sequence is deviated from the standard, the quantized direction-time feature vector for the character is changed accordingly and becomes considerably different from the prototype pattern. For example, the character H was written by some subject in the following non-standard sequence: the left vertical bar was written first, followed by the horizontal bar, and then the right vertical bar; let this be designated by the symbol

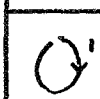
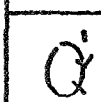
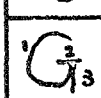


some non-standard writing sequences for characters A, 5, P, D, O, Q, and G are represented by



In this experiment, thirty-one characters among the sets No.26 through No.49 were traced in such non-standarad writing sequences as listed in Table 3.4 and encoded accordingly. These pattern vectors were

Table 3.4 —— Character Sets Traced in Non-standard Writing Sequences for Example 3.4.

	Set No.
	29
	39
	27,28,30,32,42,49
	26,31,34,36,37,38,40,42,44,45,46,47,48,49
	46
	47
	36,37
	35
	47
	41,46,47

classified by the set of weight vectors listed in Table 3.2 which was obtained with 35 training sets. Only one character, G in set No.47, could be recognized while thirty others were either misclassified or ambiguous. This result indicates that the on-line handwritten character recognition scheme discussed here is very sensitive to the writing sequence of the character, as expected.

3.5 Summary of Results

The on-line handwritten character recognition system shown in Fig. 3.1 has been extensively studied. The convergence of the learning algorithm (3.4) has been formally proved. It terminates in a finite number of training cycles if the training patterns are ϕ-separable, where ϕ-functions are defined in (3.1). The quantized direction-time pattern vectors for on-line handwritten characters, A, C, D, G, H, O, P, Q, S, and 5, are shown to be ϕ-separable by experiments. For initial weight vectors having unity components, it has been demonstrated that the smaller value of a_k, for example $a_k = 0.125$, gives the faster convergence rate during the training phase. With thirty-five training sets, the set of weight vectors $\underline{w}_i$, (i= 1, 2,..., R), has been obtained as shown in Table 3.2. By using this set of weight vectors,all forty-nine sets of characters were classified with a recognition error rate of 2.04%. The method, however, is very sensitive to the stroke sequence in writing a character because of the sequential nature of the feature extraction.

4.0 A NEW LEARNING ALGORITHM FOR MULTI-CLASS PATTERN CLASSIFICATION

4.1. General Remarks

In this section, a new learning algorithm is proposed for multi-class pattern classification. It is a generalization of the Ho-Kashyap two-class algorithm, based upon the mapping of pattern vectors of R classes into the neighborhood of R vertices of a (R - 1)-dimensional equilateral simplex with its centroid at the origin. Before the detail derivation is presented, a brief review is in order.

4.11. The Ho-Kashyap Two-class Algorithm

It is well recognized that the Ho-Kashyap two-class algorithm has relatively high rate of convergence for linearly separable patterns and also provides for a test of linear separability of a given set of sample patterns [32]. The algorithm is to find a solution weight vector (augmented weight vector) $\underline{\alpha}$ of the inequality $\underline{A\alpha} > \underline{\beta} > \underline{0}$ which describes the requirement of the linear pattern classification. It is derived from the viewpoint of minimizing a criterion function $J = ||\underline{A\alpha} - \underline{\beta}||^2$ according to the following two steps: (1) for a fixed $\underline{\beta}>\underline{0}$,determine a least square fit solution $\underline{\alpha}$ of the equation $\underline{A\alpha}-\underline{\beta}=\underline{0}$, and then (2) for a fixed $\underline{\alpha}$, adjust $\underline{\beta}$ in the direction of the steepest descent of J, subject to the constraint $\underline{\beta}>\underline{0}$. The Ho-Kashyap algorithm is given as follows:

$$
\begin{cases}
\underline{\alpha}(0) = \underline{A}^{\#}\underline{\beta}(0), \ \underline{\beta}(0) > \underline{0} \ \text{ otherwise arbitrary,} \\
\underline{y}(k) = \underline{A}\underline{\alpha}(k) - \underline{\beta}(k), \\
\underline{\alpha}(k+1) = \underline{\alpha}(k) + p\underline{A}^{\#}(\underline{y}(k) + |\underline{y}(k)|), \\
\qquad\qquad 0 < p \leq 1 \\
\underline{\beta}(k+1) = \underline{\beta}(k) + p(\underline{y}(k) + |\underline{y}(k)|)
\end{cases}
$$

where

$$
\underline{A} = \begin{bmatrix}
1, {}_1\underline{x}_1^t \\
1, {}_2\underline{x}_1^t \\
\vdots \\
1, {}_{n_1}\underline{x}_1^t \\
-1, -{}_1\underline{x}_2^t \\
-1, -{}_2\underline{x}_2^t \\
\vdots \\
-1, -{}_{n_2}\underline{x}_2^t
\end{bmatrix}
$$

= an N x D pattern matrix formed by n_1 training patterns of class 1 and n_2 training patterns of class 2,

${}_\ell\underline{x}_j^t$ = the transpose of the d x 1 pattern vector ${}_\ell\underline{x}_j$ where j denotes the pattern class C_j, and ℓ denotes the ℓth training

pattern of class C_j,

$\underline{A}^{\#}$ = the generalized inverse of $\underline{A}$,

$N = n_1 + n_2$,

and

$D = d + 1$.

A solution $\underline{\alpha}$ can be obtained in a finite number of iteration steps if the training patterns are linearly separable; the non-linear separability is indicated by $\underline{y}(k^*) \leq \underline{0}$ at some k^* which means that all components of $\underline{y}(k^*)$ are non-positive but with at least one negative component.

4.12. The Wee-Fu Generalization to R-class Algorithms

The generalization of the Ho-Kashyap two-class algorithm to multi-class algorithm has been made by C. C. Blaydon[4] ,W. G. Wee and K. S. Fu[50]. Wee and Fu have developed two R-class algorithms: the complete generalization and the generalization with constraint.

Wee and Fu's complete generalization is applicable to any type of linearly separable patterns as shown in Fig. 4.1(a) and (b). It is obtained by mapping the original N augmented sample pattern vectors in E^D into a new pattern space over which a (R - 1)N x RD pattern matrix $\underline{A}$ in E^{RD} is defined and the dichotomization algorithm can be directly applied to obtain a vector $\underline{\alpha}$ consisting of R weight vectors $\underline{w}_j$'s. Thus, in this case,

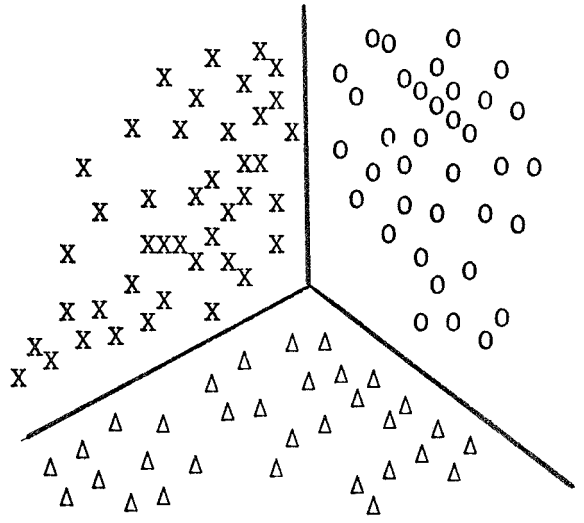
$$\underline{A} = \begin{bmatrix} \underline{A}_1(1) \\ \underline{A}_2(1) \\ \vdots \\ \underline{A}_{n_1}(1) \\ \underline{A}_1(2) \\ \vdots \\ \underline{A}_{n_2}(2) \\ \vdots \\ \underline{A}_{n_R}(R) \end{bmatrix} = \text{a } (R-1)N \text{ x } RD \text{ matrix}$$

$$\underline{A}_\ell(j) = \begin{bmatrix} -\ell\underline{x}_j^t, & 0, \ldots, & \ell\underline{x}_j^t, & 0, \ldots, & 0 \\ 0, & -\ell\underline{x}_j^t, & 0, \ldots, & \ell\underline{x}_j^t, & 0, \ldots, 0 \\ \vdots & & \vdots & & \vdots \\ 0, \ldots & & \ell\underline{x}_j^t, & 0, \ldots, & -\ell\underline{x}_j^t \end{bmatrix}$$

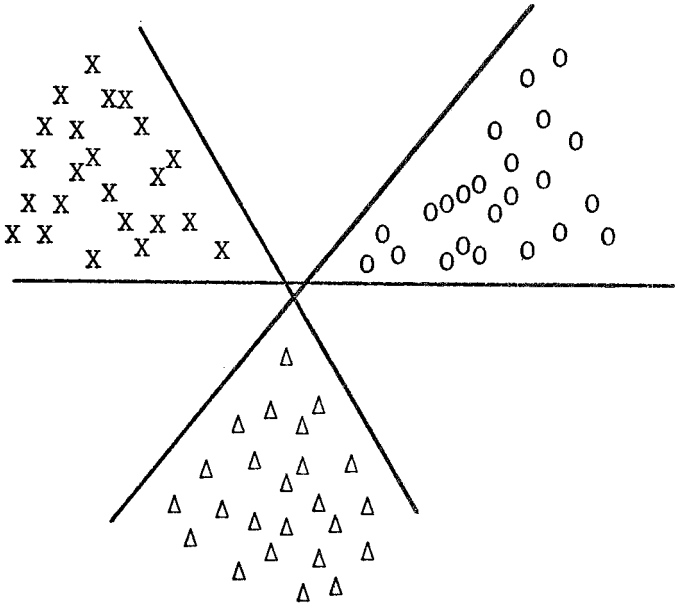
= a $(R-1) \times RD$ submatrix

$(j=1,2, \ldots, k;\ \ell=1,2, \ldots, n_j)$

$N = n_1 + n_2 + \ldots + n_R$



(a)



(b)

Fig. 4.1 Two Types of Linearly Separable Patterns of Multiple Classes.

$$\underline{\alpha} = \begin{bmatrix} \underline{w}_1 \\ \underline{w}_2 \\ \cdot \\ \cdot \\ \cdot \\ \underline{w}_R \end{bmatrix} = \text{a RD-dimensional vector}$$

and

$$\underline{\beta} = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \cdot \\ \cdot \\ \cdot \\ \beta_{(R-1)N} \end{bmatrix}$$

= a (R-1)N-dimensional vector with all $\beta_i > 0$,

($i = 1,2,\ldots,(R-1)N$).

Their generalization with constraint gives the following algorithm to determine a D x R weight matrix $\underline{\alpha}$:

$$\begin{cases} \underline{\alpha}(0) = \underline{A}^{\#}\underline{\beta}(0) \\ Y(k) = \underline{A}\,\underline{\alpha}(k) - \underline{\beta}(k) \\ \underline{\alpha}(k+1) = \underline{\alpha}(k) + p\underline{A}^{\#}\,[\underline{Y}(k) + \underline{Y}_c(k)] \\ \underline{\beta}(k+1) = \underline{\beta}(k) + p[\underline{Y}(k) + \underline{Y}_c(k)] \end{cases}$$

where

$$\underline{A} = \begin{bmatrix} 1 , {}_1\underline{x}_1^t \\ 1 , {}_2\underline{x}_1^t \\ \vdots \\ 1 , {}_{n_1}\underline{x}^t \\ 1 , {}_1\underline{x}_2^t \\ \vdots \\ 1 , {}_{n_2}\underline{x}_2^t \\ \vdots \\ 1 , {}_{n_R}\underline{x}_R^t \end{bmatrix}$$

= an N x D pattern matrix

$$\underline{\alpha}(k) = [\underline{w}_1(k), \underline{w}_2(k), \ldots, \underline{w}_R(k)]$$

= a D x R weight matrix,

$$\underline{\beta}(k) = [\underline{b}_1(k), \underline{b}_2(k), \ldots, \underline{b}_R(k)]$$

$$= \begin{bmatrix} b_{11}(k), & b_{12}(k), & \cdots, & b_{1R}(k) \\ & \vdots & & \\ b_{N1}(k), & b_{N2}(k), & \cdots, & b_{NR}(k) \end{bmatrix}$$

= an N x R matrix,

$$\underline{Y}(k) = [\underline{y}_1(k), \underline{y}_2(k), \cdots, y_R(k)]$$

= an N x R matrix,

$$\underline{Y}_c(k) = [{}_c\underline{y}_1(k), {}_c\underline{y}_2(k), \cdots, {}_c\underline{y}_R(k)]$$

$$= \begin{bmatrix} {}_cy_{11}(k), & {}_cy_{12}(k), & \cdots, & {}_cy_{1R}(k) \\ \vdots & \vdots & & \vdots \\ {}_cy_{N1}(k), & {}_cy_{N2}(k), & \cdots, & {}_cy_{NR}(k) \end{bmatrix}$$

= an N x R matrix,

$${}_cy_{ij}(k) = \left| {}_cy_{ij}^{(k)} \right| \operatorname{sgn}(b_{ij}(0)),$$

and

$$\begin{cases} b_{ij}(0) > 0 & \text{if } \underline{x}_i \in C_j \\ b_{ij}(0) \leq 0 & \text{if } \underline{x}_i \notin C_j \end{cases}$$

A solution of the weight matrix $\underline{\alpha}$ can be obtained in a finite number of steps; however, the constraint restricts this algorithm to be applicable only to those problems where each pattern class can be linearly separated from all the rest as shown in Fig. 4.1(b). The condition that at some k^*

$$y_{ij}(k^*) \leq 0 \text{ for all } \underline{x}_i \in C_j$$

$$y_{ij}(k^*) > 0 \text{ for all } \underline{x}_i \notin C_j$$

indicates only that the patterns are not linearly separable of the type shown in Fig. 4.1(b); but it does not lead to the conclusion of non-linear separability.

4.13. Blaydon's Generalization

Blaydon attempted to develop a R-class algorithm for the determination of a D x (R-1) weight matrix $\underline{\alpha}$ based upon the concept of mapping R pattern classes into R vertices of a (R-1)-dimensional equilateral simplex with its centroid at the origin.[4] He used an NxD pattern matrix $\underline{A}$ and suggested a correction rule for $\underline{\beta}(k)$, which apparently lacks of proper increment and hence there is no convergence proof for his algorithm.

The new R-class algorithm which is proposed in this section utilizes the similar mapping concept. With the aid of a newly discovered property of the simplex vertex vectors, the convergence proof of this algorithm can be obtained.

4.2. Basic Definitions

Let $\underline{x}$ be a dx1 pattern vector, ${}_{\ell}\underline{x}_j$ designate the ℓth sample pattern vector in class C_j, (j=1, 2, . . ., R), $\underline{X}$ be a Dx1 augmented pattern vector where $\underline{X}^t = (1, \underline{x}^t)$, and $\underline{w}_j$ be a Dx1 weight vector in the jth discriminant function $g_j(\underline{X}) = \underline{X}^t \underline{w}_j$. The pattern matrix $\underline{A}$ is defined as

$$\underline{A} = \begin{bmatrix} \underline{A}_1 \\ \underline{A}_2 \\ \cdot \\ \cdot \\ \cdot \\ \underline{A}_R \end{bmatrix} \tag{4.1}$$

where the submatrix $\underline{A}_j$ is, in turn, defined by

$$\underline{A}_j = \begin{bmatrix} 1, & {}_1\underline{x}_j^t \\ 1, & {}_2\underline{x}_j^t \\ . & . \quad . \\ 1, & {}_{n_j}\underline{x}_j^t \end{bmatrix} = \begin{bmatrix} {}_1\underline{X}_j^t \\ {}_2\underline{X}_j^t \\ \cdot \\ \cdot \\ \cdot \\ {}_{n_j}\underline{X}_j^t \end{bmatrix} \tag{4.2}$$

$\underline{A}_j$ is n_j x D, $\underline{A}$ is N x D, $N = n_1 + n_2 + \ldots + n_R$, and $D = d + 1$. Let $\underline{U}$ be a D x (R-1) weight matrix whose solution is to be given by the proposed algorithm, $\underline{B}$ and $\underline{Y}$ be two N x (R-1) matrices each of which has submatrices $\underline{B}_j$ or $\underline{Y}_j$ respectively corresponding to the class

grouping $\underline{A}_j$ in the matrix $\underline{A}$,

$$\underline{B} = \begin{bmatrix} \underline{B}_1 \\ \underline{B}_2 \\ \cdot \\ \cdot \\ \cdot \\ \underline{B}_R \end{bmatrix} \tag{4.3}$$

$$\underline{B}_j = \begin{bmatrix} {}_1\underline{B}_j \\ {}_2\underline{B}_j \\ \cdot \\ \cdot \\ \cdot \\ {}_{n_j}\underline{B}_j \end{bmatrix} = \text{an } n_j \text{ x } (R - 1) \text{ matrix} \tag{4.4}$$

$$\underline{Y} = \begin{bmatrix} \underline{Y}_1 \\ \underline{Y}_2 \\ \cdot \\ \cdot \\ \cdot \\ \underline{Y}_R \end{bmatrix} \tag{4.5}$$

$$= [\, \underline{Y}_1, \underline{Y}_2, \ldots, \underline{Y}_q, \ldots, \underline{Y}_{R-1}]$$

$$\underline{Y}_j = \begin{bmatrix} {}_1\underline{Y}_j \\ {}_2\underline{Y}_j \\ \cdot \\ \cdot \\ \cdot \\ {}_{n_j}\underline{Y}_j \end{bmatrix} = \text{an } n_j \text{ x (R - 1) matrix} \qquad (4.6)$$

$$\underline{y}_{iq} = {}_{\ell}\underline{Y}_{jq} \qquad (4.7)$$

$$(i = n_1 + n_2 + \ldots + n_j + \ell\ )$$

${}_{\ell}\underline{B}_j$, ${}_{\ell}\underline{Y}_j$ are 1 x (R-1) row vectors and $\underline{y}_q$ is an Nx1 column vector. $\underline{A}$, $\underline{U}$, $\underline{B}$, and $\underline{Y}$ are related by the following equation

$$\underline{Y} = \underline{A}\,\underline{U} - \underline{B}$$

or

$$\underline{Y}_j = \underline{A}_j\underline{U} - \underline{B}_j \qquad (4.8)$$

$$(j = 1, 2, \ldots, R\ )$$

Consider the (R-1)-dimensional equilateral simplex with its centroid at the origin. There are R vertices of this simplex, associated with each is a (R - 1) x 1 vertex vector $\underline{e}_j$, (j= 1, 2,...,R), whose components $\underline{e}_{ji}$'s are specified by the following equation

$$\underline{e}_{ji} = \begin{cases} \left[\dfrac{R}{R-1}\left(\dfrac{R-i}{R-i+1}\right)\right]^{1/2} & \text{for } i = j \\ \left[\dfrac{1}{R-i}\left[\dfrac{R}{R-1}\left(\dfrac{R-i}{R-i+1}\right)\right]^{1/2}\right. & \text{for } i < j \\ 0 & \text{for } i > j \end{cases} \tag{4.9}$$

Chaplin and Levadi formulated a set of inequalities

$$|| \underline{X}^t \underline{U} - \underline{e}_j || < || \underline{X}^t \underline{U} - \underline{e}_i || \qquad \text{for } \underline{X} \in C_j \tag{4.10}$$

$$\text{for all } i \neq j,\ j = 1, 2, \ldots, R$$

as a representation of linear separation of R-class patterns[8]. Each vertex vector $\underline{e}_j$ is associated with one pattern class, C_j. The pattern $\underline{X}$ is classified according to the nearest neighborhood of the mapping $\underline{X}^t \underline{U}$ to the vertices as illustrated in Fig. 4.2 They proved that (4.10) is equivalent to the set of inequalities of discriminant functions

$$\underline{X}^t \underline{w}_j > \underline{X}^t \underline{w}_i$$

or

$$g_j(X) > g_i(\underline{X}) \qquad \text{for } \underline{X} \in C_j \tag{4.11}$$

$$\text{for all } i \neq j,\ j=1,2,\ldots,R$$

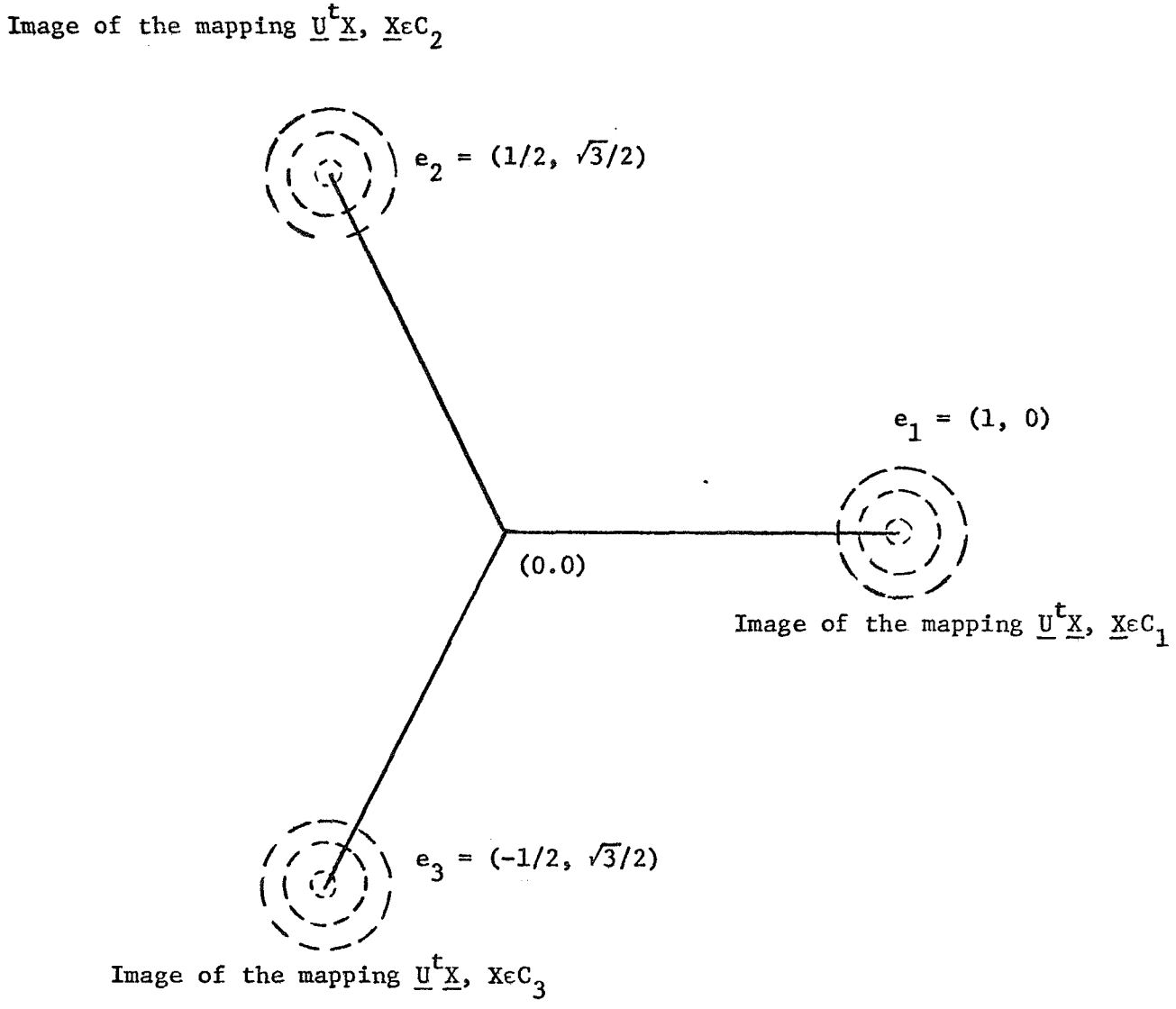


Fig. 4.2 Vertices of Equilateral Simplex for R = 3 and Pattern Classification According to the Mapping of $\underline{U}^t\underline{X}$ to the Nearest Neighborhood of Vertices.

where

$$\underline{w}_j = \underline{U}\,\underline{e}_j, \qquad (j = 1, 2, \cdots, R) \tag{4.12}$$

Thus, (4.11) can be written as

$$\underline{X}^t\underline{U}\,(\underline{e}_j - \underline{e}_i) > 0 \quad \text{for } \underline{X} \in C_j \tag{4.13}$$

$$\text{for all } i \neq j,\ j = 1, 2, \cdots, R$$

Now, let E_j be a (R-1) x (R-1) non-singular matrix defined as

$$E_j \triangleq [\underline{e}_j - \underline{e}_1, \cdots, \underline{e}_j - \underline{e}_{j-1}, \underline{e}_j - \underline{e}_{j+1}, \cdots, \underline{e}_j - \underline{e}_R] \tag{4.14}$$

It follows from (4.13) that

$$\underline{X}^t\underline{U}\,\underline{E}_j > \underline{0}^t \qquad \text{for } \underline{X} \in C_j,\ j = 1, 2, \cdots, R \tag{4.15}$$

Therefore, the set of linear inequalities which will be studied is

$$\underline{A}_j\underline{U}\,\underline{E}_j > \underline{0} \qquad \text{for all } j,\ j=1,2,\cdots,R \tag{4.16}$$

Let $\underline{Z}_j$ be an n_j x (R-1) matrix defined by

$$\underline{Z}_j \triangleq \underline{Y}_j\,\underline{E}_j \qquad j=1, 2, \cdots, R \tag{4.17}$$

From (4.8) and (4.17), one has

$$\underline{Z}_j = \underline{Y}_j \underline{E}_j = \underline{A}_j \underline{U} \underline{E}_j - \underline{B}_j \underline{E}_j, \quad j=1, 2, \cdots, R \tag{4.18}$$

4.3. Properties of Vertex Vectors

The vertex vectors of an equilateral simplex with its centroid at the origin have the following properties:

(i) $||\underline{e}_j|| = 1$ for all $j=1, 2, \cdots, R$

(ii) $||\underline{e}_j - \underline{e}_i|| = ||\underline{e}_j - \underline{e}_k||$ for all $i \neq k \neq j$

(iii) $\underline{e}_j^t(\underline{e}_j - \underline{e}_i) > 0$ for all $i \neq j$, $j = 1, 2, \cdots, R$

(iv) $(\underline{e}_j - \underline{e}_i)^t (\underline{e}_j - \underline{e}_k) = (\underline{e}_j - \underline{e}_m)^t (\underline{e}_j - \underline{e}_n)$ for $i, k, m, n \neq j$

(v)

$$\underline{F}_R \triangleq \underline{E}_j^t \underline{E}_j = C_R \begin{bmatrix} 1 & 1/2 & \cdots & 1/2 \\ 1/2 & 1 & & \vdots \\ \vdots & & & \vdots \\ \vdots & & & 1/2 \\ 1/2 & & \cdots & 1 \end{bmatrix} \quad \text{for all } j \tag{4.19}$$

where C_R is a constant depending on R

(vi) $\underline{G}_R = [\, g_{ij} \,] \triangleq \underline{F}_R^{-1} = (\underline{E}_j^t \underline{E}_j)^{-1}$

where

$$g_{ii} = g > 0$$

$$g_{ij} = g_{ji} = g' < 0 \quad \text{for all } i \neq j \tag{4.20}$$

Note that properties (v) and (vi) are obtained by extensive enumeration. The matrices $\underline{E}_j$'s, $\underline{F}_R$'s, and $\underline{G}_R$'s for R=2,...,10 are listed in Appendix D. Property (vi) is important in the convergence proof of the proposed R-class classification algorithm.

4.4. The Proposed Multi-Class Algorithm

4.41 Development of the Algorithm

Let $\underline{B}$ satisfy the following condition

$$\underline{B}_j \underline{E}_j > \underline{0}$$

or

$$\ell\underline{B}_j \underline{E}_j > \underline{0} \qquad \text{for all } j = 1, 2, \ldots, R. \qquad (4.21)$$
$$\text{for all } \ell = 1, 2, \ldots, n_j.$$

It can be seen from

$$\underline{Z}_j = \underline{Y}_j\underline{E}_j = \underline{A}_j \underline{U} \underline{E}_j - \underline{B}_j \underline{E}_j \qquad \text{for } j = 1, 2, \ldots, R$$

that, if (4.21) is satisfied and $\underline{Z}_j = \underline{Y}_j\underline{E}_j > \underline{0}$ for all j, the inequality (4.16)

$$\underline{A}_j \underline{U} \underline{E}_j > \underline{0} \qquad \text{for all } j, j=1, 2, \ldots, R$$

follows immediately.

Similar to the Ho-Kashyap and Blaydon's procedures, consider first the minimization of a criterion function

$$J(\underline{Y}) = || \underline{Y} ||^2 = \text{Tr } (\underline{Y}^t \underline{Y}) \tag{4.22}$$

Let the elements of $\underline{A}$, $\underline{U}$, $\underline{B}$, and $\underline{Y}$ be represented by a_{ij}, u_{ij}, b_{ij}, and y_{ij} respectively;

$\underline{A} = [a_{ij}]$ for $i = 1, 2, \ldots, N$, $j = 1, 2, \ldots, D$,

$\underline{U} = [u_{ij}]$ for $i = 1, 2, \ldots, D$, $j = 1, 2, \ldots, R-1$,

$\underline{B} = [b_{ij}]$ for $i = 1, 2, \ldots, N$, $j = 1, 2, \ldots, R-1$,

and $\underline{Y} = [y_{ij}]$ for $i = 1, 2, \ldots, N$, $j = 1, 2, \ldots, R-1$.

From (4.8),

$$y_{ij} = \sum_{k=1}^{D} a_{ik} u_{kj} - b_{ij}$$

J(Y) in (4.22) can be rewritten as

$$J(\underline{Y}) = \sum_{i=1}^{N} \sum_{j=1}^{R-1} J_{ij}(\underline{Y}) = \sum_{i=1}^{N} \sum_{j=1}^{R-1} y_{ij}^2$$

$$= \sum_{i=1}^{N} \sum_{j=1}^{R-1} \left(\sum_{k=1}^{D} a_{ij} u_{kj} - b_{ij} \right)^2 \tag{4.23}$$

where

$$J_{ij}(\underline{Y}) = y_{ij}^2 = \left(\sum_{k=1}^{D} a_{ik} u_{kj} - b_{ij} \right)^2$$

Take gradients of $J(\underline{Y})$ with respect to $\underline{U}$ and $\underline{B}$,

$$\frac{\partial J_{ij}(\underline{Y})}{\partial \underline{U}} = 2y_{ij} \frac{\partial y_{ij}}{\partial \underline{U}}$$

$$= 2y_{ij} \begin{bmatrix} & & & \text{jth column} \downarrow & & \\ 0 & \cdots & 0, & a_{il}, & 0 & \cdots \\ \vdots & & & & & \\ 0 & \cdots & 0, & a_{iN}, & 0 & \cdots \end{bmatrix}$$

$$\frac{\partial J(\underline{Y})}{\partial \underline{U}} = 2 \begin{bmatrix} \sum_{i=1}^{N} a_{il}\, y_{il} & \cdots & \sum_{i=1}^{N} a_{il}\, y_{ij} & \cdots & \sum_{i=1}^{N} a_{il}\, y_{1,R-1} \\ \cdots & & & & \cdots \\ \sum_{i=1}^{N} a_{iN}\, y_{iN} & \cdots & \sum_{i=1}^{N} a_{iN}\, y_{ij} & \cdots & \sum_{i=1}^{N} a_{iN}\, y_{1,R-1} \end{bmatrix}$$

$$= 2\, \underline{A}^{t} \underline{Y} \qquad (4.24)$$

$$\frac{\partial J_{ij}(\underline{Y})}{\partial \underline{B}} = 2y_{ij} \frac{\partial y_{ij}}{\partial \underline{B}}$$

$$= 2y_{ij} \begin{bmatrix} 0 & \cdots & 0 & \cdots & 0 \\ \vdots & & \vdots & & \vdots \\ 0 & \cdots & -1 & \cdots & 0 \\ \vdots & & \vdots & & \vdots \\ 0 & \cdots & 0 & \cdots & 0 \end{bmatrix} \leftarrow \text{ith row}$$

$$\uparrow$$

$$\text{jth column}$$

$$\frac{\partial J}{\partial \underline{B}} = -2 \begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1,R-1} \\ \cdot & \cdot & \cdot & \cdot \\ y_{N1} & y_{N2} & \cdots & y_{N,R-1} \end{bmatrix}$$

$$= -2\,\underline{Y} \tag{4.25}$$

For unconstraint $\underline{U}$, $\frac{\partial J}{\partial \underline{U}} = 0$ implies

$$\underline{Y} = \underline{A}\,\underline{U} - \underline{B} = \underline{0}.$$

Therefore for a fixed $\underline{B}$, one has a least square fit solution of

$$\underline{U} = \underline{A}^{\#}\,\underline{B} \tag{4.26}$$

On the other hand, for a fixed $\underline{U}$, if $\underline{B}$ is incremented according to the gradient descent procedure to reduce $J(\underline{Y})$ at each step,

$$\ell\underline{B}_j(k+1) = \ell\underline{B}_j(k) + \delta_\ell\underline{B}_j(k)$$

$$\delta_\ell\underline{B}_j(k) = [\, \delta_\ell B_{j1}(k),\ \delta_\ell B_{j2}(k),\ \cdots,\ \delta_\ell B_{j,R-1}(k)\,],$$

$$\delta_\ell B_{jq}(k) = \begin{cases} -p_\ell \left[\dfrac{\partial J(\underline{Y})}{\partial \underline{B}}\right]_{jq} = 2p\ {}_\ell Y_{jq} & \text{if } \ell\underline{Y}_j\,(\underline{e}_j - \underline{e}_q) > 0 \quad \text{for } q \neq j, \\ 0 & \text{if } \ell\underline{Y}_j\,(\underline{e}_j - \underline{e}_q) \leq 0 \quad \text{for } q \neq j, \end{cases}$$

it follows that the constraint (4.21) cannot be satisfied. Therefore, a different adjustment procedure should be considered.

A solution $\underline{U}$ of inequality (4.16) can be obtained by the following iterative adjustment on ${}_{\ell}\underline{B}_j$,

$$
\begin{aligned}
{}_{\ell}\underline{B}_j(k+1) &= {}_{\ell}\underline{B}_j(k) + p\,{}_{\ell}\underline{H}_j(k) \\
&= {}_{\ell}\underline{B}_j(k) + p\,[{}_{\ell}\underline{Z}_j(k) + {}_{\ell}\underline{\Lambda}_j(k)]\;\underline{E}_j^{-1}
\end{aligned}
\tag{4.27}
$$

where k = the iteration number

p = a constant, $0<p\leq 1$

$$
{}_{\ell}\underline{H}_j \triangleq [{}_{\ell}\underline{Z}_j + {}_{\ell}\underline{\Lambda}_j]\;\underline{E}_j^{-1} = {}_{\ell}\underline{Y}_j + {}_{\ell}\underline{\Lambda}_j\;E_j^{-1} \tag{4.28}
$$

= the ℓth row vector in an $n_j \times (R-1)$ matrix $\underline{H}_j$,

$$
\underline{H}_j = \begin{bmatrix} {}_1\underline{H}_j \\ \cdot \\ \cdot \\ \cdot \\ {}_{\ell}\underline{H}_j \\ \cdot \\ \cdot \\ \cdot \\ {}_{n_j}\underline{H}_j \end{bmatrix} = \underline{Y}_j + \underline{\Lambda}_j\;E_j^{-1} = [\underline{Z}_j + \underline{\Lambda}_j]\;E_j^{-1}
$$

$$
\underline{H} = \begin{bmatrix} \underline{H}_1 \\ \underline{H}_2 \\ \cdot \\ \cdot \\ \cdot \\ \underline{H}_R \end{bmatrix}
$$

$$
= [\underline{h}_1, \cdots, \underline{h}_q, \cdots, \underline{h}_{R-1}]
$$

$$h_{iq} = {}_{\ell}H_{jq}\ , \quad i = n_1 + \cdots + n_{j-1} + \ell$$

$${}_{\ell}\underline{Z}_j = ({}_{\ell}Z_{j1}, \ldots, {}_{\ell}Z_{jq}, \ldots, {}_{\ell}Z_{j(R-1)})$$

$$= {}_{\ell}\underline{Y}_j \, \underline{E}_j$$

$$= \text{the } \ell^{\text{th}} \text{ row vector in } \underline{Z}_j$$

$${}_{\ell}\underline{\Lambda}_j = ({}_{\ell}\Lambda_{j1}, \ldots, {}_{\ell}\Lambda_{jq}, \ldots, {}_{\ell}\Lambda_{j(R-1)}) \tag{4.29}$$

$$= \text{the } \ell^{\text{th}} \text{ row vector in an } n_j x(R-1) \text{ matrix } \underline{\Lambda}_j$$

and

$${}_{\ell}\Lambda_{jq} \triangleq {}_{\ell}Z_{jq} \ \text{sgn}\,({}_{\ell}Z_{jq}) \tag{4.30}$$

From (4.30), it is obvious that

$${}_{\ell}Z_{jq}(k) + {}_{\ell}\Lambda_{jq}(k) = \begin{cases} 2\,{}_{\ell}Z_{jq}(k) & \text{if } {}_{\ell}Z_{jq} > 0 \\ 0 & \text{if } {}_{\ell}Z_{jq} \leq 0 \end{cases} \tag{4.31}$$

$${}_{\ell}Z_{jq}(k) - {}_{\ell}\Lambda_{jq}(k) = \begin{cases} 0 & \text{if } {}_{\ell}Z_{jq}(k) \geq 0 \\ 2\,{}_{\ell}Z_{jq}(k) & \text{if } {}_{\ell}Z_{jq}(k) < 0 \end{cases} \tag{4.32}$$

If the choice of ${}_{\ell}\underline{B}_j(0)$ is such that

$${}_{\ell}\underline{B}_j(0)\,\underline{E}_j > \underline{0} \qquad \text{for all } j, \tag{4.33}$$

it follows from (4.27), (4.30), and (4.33)

$$\ell \underline{B}_j(k+1)\ \underline{E}_j \geq \ell \underline{B}_j(k)\ \underline{E}_j \geq \cdots \geq \ell \underline{B}_j(0)\ \underline{E}_j > \underline{0} \qquad \text{for all } j \tag{4.34}$$

which satisfies the condition given in (4.21). Hence, the complete algorithm for the multiclass classification under consideration can be proposed as follows:

$$\begin{cases} \underline{U}\ (0) = \underline{A}^{\#}\ \underline{B}\ (0) \\ \underline{Y}\ (k) = \underline{A}\ \underline{U}\ (k) - \underline{B}\ (k), \quad \underline{Z}_j(k) = \underline{Y}_j(k)\ \underline{E}_j \\ \underline{B}\ (k+1) = \underline{B}(k) + p\underline{H}(k), \quad H_j(k) = [\underline{Z}_j(k) + \underline{\Lambda}_j(k)]\ \underline{E}_j^{-1} \\ \underline{U}(k+1) = \underline{U}(k) + p\ \underline{A}^{\#}\ \underline{H}(k) \end{cases} \tag{4.35}$$

where $\underline{B}(0)$ may be chosen as

$$\underline{B}(0) = \begin{bmatrix} \underline{B}_1(0) \\ \vdots \\ \underline{B}_j(0) \\ \vdots \\ \underline{B}_R(0) \end{bmatrix} = \beta \begin{bmatrix} \underline{e}_1^t \\ \vdots \\ \underline{e}_1^t \\ - - - \\ \vdots \\ - - - \\ \underline{e}_j^t \\ \vdots \\ \underline{e}_j^t \\ - - - \\ \vdots \\ - - - \\ \underline{e}_R^t \\ \vdots \\ \underline{e}_R^t \end{bmatrix}, \quad \beta > 0 \text{ otherwise arbitrary} \tag{4.36}$$

A recursive relation in $\underline{Y}(k)$ is then obtained as follows

$$\underline{Y}\ (k+1) = \underline{Y}(k) + p\ (\underline{A}\ \underline{A}^{\#} - \underline{I})\ \underline{H}(k) \qquad (4.37)$$

4.42. Lemma

Consider the set of inequalities (4.16)

$$\underline{A}_j\ \underline{U}\ \underline{E}_j > \underline{0} \quad \text{for all } j,\ j=1, 2, \cdots, R$$

and the algorithm (4.35) to solve for it. Then

1) $\underline{Z}_j(k) = \underline{Y}_j(k)\ \underline{E}_j \not> \underline{0}$ for all j and for any k.

2) If (4.16) is consistent, then

$\underline{Z}_j(k) = \underline{Y}_j(k)\ \underline{E}_j \not< \underline{0}$ for all j and for all k.

where the expression $\underline{Y}\ \underline{E} \lesssim \underline{0}$ means that the components of every column vector in $\underline{Y}\ \underline{E}$ are non-positive with at least one negative component in each column.

Proof:

1) Let $\underline{Y}_j(k)\ \underline{E}_j \geq \underline{0}$ for all $j = 1, 2, \cdots, R$, for some k.

Since from (4.33),

$$\underline{B}_j(k)\ \underline{E}_j > \underline{0} \quad \text{for all } j$$

then

$$(\underline{Y}_j(k)\ \underline{E}_j)^t\ \underline{B}_j(k)\ \underline{E}_j > \underline{E}_j{}^t\ \underline{Y}_j^t(k)\ \underline{B}_j(k)\ \underline{E}_j > \underline{0}$$

$$\underline{Y}_j{}^t(k)\ \underline{B}_j(k) > \underline{0} \quad \text{for all } j$$

It follows that

$$\underline{Y}^t(k)\ \underline{B}(k) > \underline{0}$$

But

$$\underline{Y}(k) = (\underline{A}\underline{A}^{\#} - I)\ \underline{B}(k)$$

$$\underline{Y}^t(k)\ \underline{B}(k) = \underline{B}^t(k)\ (\underline{A}\underline{A}^{\#} - \underline{I})\ \underline{B}(k) \leq \underline{0}$$

since

$$(\underline{A}\ \underline{A}^{\#} - I) \leq \underline{0}\ .$$

This is a contradiction. Hence

$$\underline{Y}_j(k)\ \underline{E}_j \not\geq \underline{0} \quad \text{for all } j \text{ and for any } k.$$

2) Assume that (4.16) is consistent but

$$\underline{Y}_j(k)\ \underline{E}_j \leq \underline{0} \quad \text{for all } j \text{ and for some } k$$

The consistency of (4.16) implies the existence of a $\underline{U}^*$ and a $\underline{B}^*$ such that

$$\underline{A}\ \underline{U}^* = \underline{B}^* \quad \text{or} \quad \underline{A}_j\ \underline{U}^* = \underline{B}_j^* \qquad \text{for all } j$$

and

$$\underline{A}_j\ \underline{U}^*\ \underline{E}_j = \underline{B}_j^*\ \underline{E}_j > \underline{0} \qquad \text{for all } j.$$

Therefore,

$$(\underline{Y}_j(k)\ \underline{E}_j)^t\ \underline{B}_j^*\ \underline{E}_j = \underline{E}_j^t\ \underline{Y}_j^t(k)\ \underline{B}_j^*\ \underline{E}_j < 0 \text{ for all } j$$

$$\underline{Y}_j^t(k)\ \underline{B}_j^* < \underline{0} \qquad \text{for all } j$$

and

$$\underline{Y}^t(k)\ \underline{B}^* < \underline{0}\ .$$

But for any $\underline{Y}(k)$,

$$\underline{A}^t\underline{Y}(k) = \underline{A}^t\,[\underline{A}\,\underline{U}\,(k) - \underline{B}(k)]$$

$$= A^t(\underline{A}\,\underline{A}^{\#} - \underline{I})\,B(k)$$

$$= (\underline{A}^t\underline{A}\,A^{\#} - \underline{A}^t)\,\underline{B}(k) = \underline{0}\,\underline{B}(k) = \underline{0}$$

thus

$$\underline{U}^{*t}\,\underline{A}^t\,\underline{Y}(k) = \underline{0}$$

or

$$\underline{Y}^t(k)\,\underline{A}\,\underline{U}^* = \underline{Y}^t(k)\,\underline{B}^* = \underline{0}$$

which is a contradiction. Hence, if (4.16) is consistent,

$$\underline{Y}_j(k)\,\underline{E}_j \neq 0 \text{ for all } j \text{ and for any } k$$

The convergence proof of the multi-class algorithm (4.35) for a solution weight $\underline{U}$ matrix for the algorithm is given by the following theorem.

4.43. Theorem:

Consider the set of linear inequalities

$$\underline{A}_j\,\underline{U}\,E_j > \underline{0} \quad \text{for all } j,\ j = 1, 2, \cdots, R$$

and the algorithm (4.35) to solve for it, and let

$$V(\underline{Y}(k)) \triangleq ||\,\underline{Y}(k)\,||^2$$

$$\triangleq \text{Tr}\,[\underline{Y}^t(k)\,\underline{Y}(k)]$$

$$= \sum_{q=1}^{R-1} ||y_q(k)||^2$$

or

$$V(\underline{Y}(k)) = \mathrm{Tr}\,[\underline{Y}(k)\,\underline{Y}^{t}(k)]$$

$$= \sum_{j=1}^{R} \sum_{\ell=1}^{n_j} ||\,{}_{\ell}\underline{Y}_j(k)||^2 \qquad (4.38)$$

1) If the set of linear inequalities is consistent, then

a) $\Delta V\,(\underline{Y}(k)) \triangleq V(\underline{Y}(k+1)) - V(\underline{Y}(k)) < 0$ and

$\lim_{k\to\infty} V\,(Y(k)) = 0$ implying convergence to a solution

in an infinite number of iterations; and

b) actually, a solution of weight matrix is obtained in a finite number of steps.

2) If the set of linear inequalities is inconsistent, then there exists a positive integer k^* such that

$\Delta V\,[\underline{Y}(k)] < 0$ for $k < k^*$

$\Delta V\,[\underline{Y}(k)] = 0$ for $k \geq k^*$

$\underline{Z}_j(k) = \underline{Y}_j(k)\,\underline{E}_j \not< \underline{0}$ for $k < k^*$ and for all j

$\underline{Z}_j(k) = \underline{Y}_j(k)\,\underline{E}_j \leq 0$ for $k \geq k^*$ and for all j

and

$\underline{U}(k) = \underline{U}\,(k^*)$ for $k \geq k^*$

$\underline{B}\,(k) = \underline{B}\,(k^*)$ for $k \geq k^*$

In other words, the occurrence of a matrix $\underline{Y}(k)$ such that ${}_{\ell}\underline{Y}_j(k)\,\underline{E}_j$ has all nonpositive elements for all ℓ and all j at any at any step k terminates the algorithm and indicates the nonlinear separability of the R-class patterns.

Proof:

Part 1) : With reference to the recurrence relation in $\underline{Y}(k)$ given by (4.37). $V(Y(k))$ defined in (4.38) can be considered as a Liapunov function which is positive definite for all $\underline{Y}(k)$.

$$\Delta V[\underline{Y}(k)] \triangleq V[\underline{Y}(k+1)] - V[\underline{Y}(k)]$$
$$= \text{Tr}\ [\underline{Y}^t(k+1)\ Y(k) \ - \ \underline{Y}^t(k)\ Y(k)]$$

since

$$\underline{Y}^t(k+1)\ \underline{Y}(k+1) - \underline{Y}^t(k)\ \underline{Y}(k)$$
$$= [\underline{Y}(k) + p(\underline{A}\ \underline{A}^{\#} - \underline{I})\ H(k)]^t\ [\underline{Y}(k)+p(\underline{A}\ \underline{A}^{\#}-\underline{I})\ \underline{H}(k)]-\underline{Y}^t(k)\underline{Y}(k)$$
$$= p\ \underline{H}^t(k)\ (\underline{A}\ \underline{A}^{\#}-\underline{I})\ \underline{Y}(k) + p\ \underline{Y}^t(k)\ (\underline{A}\ \underline{A}^{\#}-\underline{I})\ H(k)$$
$$+p^2\underline{H}^t(k)\ (I - \underline{A}\ \underline{A}^{\#})\ \underline{H}(k)$$

and

$$\underline{A}\ \underline{A}^{\#}\ \underline{Y}(k) = \underline{A}\ \underline{A}^{\#}\ [\underline{A}\ \underline{U}(k) - \underline{B}\ (k)] = \underline{A}\ \underline{A}^{\#}[\underline{A}\ \underline{A}^{\#}\ \underline{B}(k) - \underline{B}(k)]$$
$$= \underline{A}\ \underline{A}^{\#}\ \underline{B}(k) - \underline{A}\ \underline{A}^{\#}\ \underline{B}(k) = \underline{0}$$

then

$$\Delta\ V(\underline{Y}(k)) = -\ p\ \text{Tr}\ [\underline{H}^t(k)\ \underline{Y}(k) + \underline{Y}^t(k)\ \underline{H}(k)]$$
$$+\ p^2\text{Tr}\ [\underline{H}^t(k)\ (\underline{I} - \underline{A}\ \underline{A}^{\#})\ \underline{H}(k)]$$
$$= -2p\ \text{Tr}\ [\underline{H}^t(k)\ \underline{Y}(k)] + p^2\ \text{Tr}\ [\underline{H}^t(k)(\underline{I}-\underline{A}\ \underline{A}^{\#})\ \underline{H}(k)]$$
$$= -2p \sum_{q=1}^{R-1} \underline{h}_q{}^t(k)\underline{y}_q(k)+p^2 \sum_{q=1}^{R-1} \underline{h}_q{}^t(k)\ (\underline{I} - \underline{A}\ \underline{A}^{\#})\underline{h}_q(k)$$
$$= -2p \sum_{j=1}^{R} \sum_{\ell=1}^{n_j} \underline{\ell H}_j(k)\underline{\ell Y}_j{}^t(k)$$
$$+\ p^2 \sum_{q=1}^{R-1} \underline{h}_q^t(k)\ (\ \underline{I} - \underline{A}\ \underline{A}^{\#}\ )\ \underline{h}_q(k) \qquad (4.39)$$

Note that

$$-2p\,{}_{\ell}\underline{H}_j(k)\,{}_{\ell}\underline{Y}_j^{\,t}(k) = -2p\{[{}_{\ell}\underline{Z}_j+{}_{\ell}\underline{\Lambda}_j]\underline{E}_j^{-1}\}\ ({}_{\ell}\underline{Z}_j\underline{E}_j^{-1})^t$$

$$=-\,p\{[{}_{\ell}\underline{Z}_j(k) + {}_{\ell}\underline{\Lambda}_j(k)]\underline{E}_j^{-1}\}\ \{[{}_{\ell}\underline{Z}_j(k) + {}_{\ell}\underline{\Lambda}_j(k)]\underline{E}_j^{-1}\}^t$$

$$-p\{[{}_{\ell}\underline{Z}_j(k) + {}_{\ell}\underline{\Lambda}_j(k)]\underline{E}_j^{-1}\}\ \{[{}_{\ell}\underline{Z}_j(k) - {}_{\ell}\underline{\Lambda}_j(k)]\underline{E}_j^{-1}\}^t$$

$$=-p[{}_{\ell}\underline{H}_j(k)\,{}_{\ell}\underline{H}_j^{\,t}(k)] -p\ [{}_{\ell}\underline{Z}_j(k) + {}_{\ell}\underline{\Lambda}_j(k)]\underline{E}_j^{-1}(\underline{E}_j^t)^{-1}[{}_{\ell}\underline{Z}_j - {}_{\ell}\underline{\Lambda}_j(k)]^t$$

$$=-p[{}_{\ell}\underline{H}_j(k)\,{}_{\ell}\underline{H}_j^{\,t}(k)]-p[{}_{\ell}\underline{Z}_j(k)+{}_{\ell}\underline{\Lambda}_j(k)]\underline{G}_R[{}_{\ell}\underline{Z}_j(k)-{}_{\ell}\underline{\Lambda}_j(k)]^t \tag{4.40}$$

The components of $[{}_{\ell}\underline{Z}_j(k) + {}_{\ell}\underline{\Lambda}_j(k)]$ are either positive or zero and the corresponding components of $[{}_{\ell}\underline{Z}_j(k) - {}_{\ell}\underline{\Lambda}_j(k)]$ are either zero or negative. Since $g_{qm}<0$ for $q\neq m$,

$$[{}_{\ell}\underline{Z}_j(k) + {}_{\ell}\underline{\Lambda}_j(k)]\ \underline{G}_R\ [{}_{\ell}Z_j(k)-{}_{\ell}\Lambda_j(k)]^t$$

$$= \sum_{m=1}^{R-1}\ \sum_{\substack{q=1\\ m\neq q}}^{R-1} (2\,{}_{\ell}Z_{jq})\ g_{qm}\ (2\,{}_{\ell}Z_{jm})$$

$$= 4\sum_{m=1}^{R-1}\ \sum_{q=1}^{R-1} {}_{\ell}Z_{jq}g_{qm}{}_{\ell}Z_{jm} = {}_{\ell}\varepsilon_j \geq 0 \quad \text{for all } j \text{ and } \ell \tag{4.41}$$

where

$${}_{\ell}Z_{jq} \geq 0 \text{ and } {}_{\ell}Z_{jm} \leq 0$$

Substitute (4.41) into (4.40) which, in turn, is substituted into (4.38),

$$\Delta V[\underline{Y}(k)] = -p \sum_{j=1}^{R} \sum_{\ell=1}^{n_j} {}_{\ell}\underline{H}_j(k)\ {}_{\ell}\underline{H}_j{}^t(k) - p \sum_{j=1}^{R} \sum_{\ell=1}^{n_j} {}_{\ell}\varepsilon_j$$

$$+p^2 \sum_{q=1}^{R-1} \underline{h}_q{}^t(k)\ (\underline{I} - \underline{A}\ \underline{A}^{\#})\ \underline{h}_q(k)$$

$$= -p\ \mathrm{Tr}\ [\underline{H}(k)\ \underline{H}^t(k)] - p\varepsilon + p^2 \sum_{q=1}^{R-1} \underline{h}_q{}^t(k)\ (\underline{I} - \underline{A}\ \underline{A}^{\#})\underline{h}_q(k)$$

$$= -p\ \mathrm{Tr}\ [\underline{H}^t(k)\ \underline{H}(k)] - p\ \varepsilon + p^2 \sum_{q=1}^{R-1} \underline{h}_q{}^t(k)(\underline{I} - \underline{A}\ \underline{A}^{\#})\underline{h}_q(k)$$

$$= -p \sum_{q=1}^{R-1} \underline{h}_q{}^t(k)\underline{h}_q(k) - p\ \varepsilon + p^2 \sum_{q=1}^{R-1} \underline{h}_q{}^t(k)(\underline{I} - \underline{A}\ \underline{A}^{\#})\underline{h}_q(k)$$

$$= -p\ \varepsilon - \sum_{q=1}^{R-1} \underline{h}_q{}^t(k)[p\ \underline{I} - p^2\ (\underline{I} - \underline{A}\ \underline{A}^{\#})]\underline{h}_q(k)$$

$$= -p\ \varepsilon - \mathrm{Tr}\ \{\underline{H}^t(k)[p\ \underline{I} - p^2(\underline{I} - \underline{A}\ \underline{A}^{\#})]\ \underline{H}(k)\} \qquad (4.42)$$

where

$$\varepsilon \triangleq \sum_{j=1}^{R} \sum_{\ell=1}^{n_j} {}_{\ell}\varepsilon_j \geq 0 \quad \text{for all } \underline{Z}_j \text{ or } \underline{Y}_j \qquad (4.43)$$

$$= 0 \quad \text{if } \underline{Z}_j = 0 \text{ or } \underline{Y}_j = 0 \text{ for all } j$$

If

$$0 < p < 1$$

$(p\ \underline{I} - p^2(\underline{I} - \underline{A}\ \underline{A}^{\#}))$ is positive semidefinite, hence $\Delta V[\underline{Y}(k)]$ is negative definite in $\underline{H}(k)$ or in $\underline{Z}_j(k) + \underline{\Lambda}_j(k)$ for all j. Note that $\underline{H}_j(k) = 0$ only if $\underline{Z}_j(k) = 0$ or $\underline{Z}_j(k) \leq \underline{0}$. Since it is assumed that the set of inequality (4.16) is consistent, from the lemma $\underline{Z}_j(k) = \underline{Y}_j(k)\ \underline{E}_j \not< 0$ for all j, therefore

$$\Delta V\ [\underline{Y}(k)] < 0 \qquad \text{for all } \underline{Z}(k)\neq\underline{0} \text{ or } \underline{Y}(k)\neq\underline{0}$$

$$= 0 \qquad \text{if } \underline{Z}(k) = \underline{0} \text{ or } \underline{Y}(k) = \underline{0}$$

$\Delta V\ [\ Y(k)\ ]$ is negative definite in $\underline{Y}(k)$, the solution $\underline{Y}(k) = \underline{0}$ of (4.37) can be reached asymptotically, that is,

$$\lim_{k\to\infty} V[\underline{Y}(k)] = \lim_{k\to\infty} \mathrm{Tr}\ [\underline{Y}^t(k)\ \underline{Y}(k)] = 0$$

which assures a solution $\underline{U}^{**}$ from $\underline{Y}^{**} = \underline{A}\ \underline{U}^{**} - \underline{B} = \underline{0}$ such that $\underline{A}_j\ \underline{U}^{**}\underline{E}_j = \underline{B}_j\ \underline{E}_j > \underline{0}$ for all j. This completes the proof of part (a).

Next, from (4.34) and (4.36)

$$_{\ell}\underline{B}_j(k)\underline{E}_j \geq\ _{\ell}\underline{B}_j(0)\underline{E}_j \geq \beta\, \underline{e}_j^{\,t}\underline{E}_j > \underline{0} \text{ for all } j$$

Since $_{\ell}\underline{H}_j(k)\ \underline{E}_j = {}_{\ell}\underline{Z}_j + {}_{\ell}\underline{\Lambda}_j$ has only positive or zero elements,

$$_{\ell}\underline{B}_j\ (k+1)\ \underline{E}_j = {}_{\ell}\underline{B}_j(k)\ \underline{E}_j + p\,{}_{\ell}\underline{H}_j(k)\ \underline{E}_j$$

$$= \beta(1+\delta)\ \underline{e}_j^{\,t}\ \underline{E}_j > \underline{0} \text{ for all } j \text{ and } \ell \qquad (4.44)$$

where $\delta>0$. $V[Y(k)] < 1$ at a certain finite k implied that

$$||\,{}_{\ell}Y_j(k)\,||^2 < 1$$

and $$_{\ell}\underline{Y}_j(k)\underline{E}_j > -\ \underline{e}_j^{\,t}\underline{E}_j \qquad \text{for all } j \text{ and } \ell$$

when k is sufficiently large. It follows that

$$\underline{Y}_j(k)\underline{E}_j > -\underline{B}_j(0)\underline{E}_j$$

and

$$\underline{A}_j\underline{U}(k)\underline{E}_j = \underline{B}_j(k)\underline{E}_j + \underline{Y}_j(k)\underline{E}_j$$

$$\geq (1+\delta)\underline{B}_j(0)\underline{E}_j - \underline{B}_j(0)\underline{E}_j$$

$$\geq \delta\ \underline{B}_j(0)\underline{E}_j > \underline{0} \text{ for all } j.$$

which indicates that a solution $\underline{U}^* = U(k)$ is obtained in a finite number of steps. This completes the proof of part 1b).

part 2).

If the set of inequalities

$$\underline{A}_j\ \underline{U}\ \underline{E}_j > \underline{0} \qquad \text{for all } j,\ j=1,\ 2,\cdots,\ R$$

is inconsistent, $\underline{Y}(k)$ cannot be $\underline{0}$ and hence $V[\underline{Y}(k)]$ cannot become zero for any $k > 0$. There must exist a value of k, $k=k^*$, such that

$$\Delta V[\underline{Y}(k)] < 0 \qquad \text{for } 0 \leq k < k^*$$

$$= 0 \qquad \text{for } k = k^*$$

But, as shown in part 1, $\Delta V[\underline{Y}(k^*)] = 0$ only if either $\underline{Z}_j(k^*) = \underline{Y}_j(k^*)\underline{E}_j = \underline{0}$ or $\underline{Z}_j(k^*) = \underline{Y}_j(k^*)\underline{E}_j \leq \underline{0}$ for all j. Since $\underline{Y}(k^*) \neq \underline{0}$, this implies that

$$\underline{Z}_j(k^*) = \underline{Y}_j(k^*)\underline{E}_j \leq 0 \text{ for all } j$$

Hence, from (4.35)

$$\underline{H}_j(k) = 0 \quad \& \quad \underline{H}(k) = 0 \qquad \text{for all } k \geq k^*$$

$$\underline{B}(k) = \underline{B}(k^*) \qquad \text{for all } k \geq k^*$$

$$\underline{U}(k) = \underline{U}(k^*) \qquad \text{for all } k \geq k^*$$

$$\underline{Y}(k) = \underline{Y}(k^*) \qquad \text{for all } k \geq k^*$$

and

$$\Delta V[\underline{Y}(k)] = 0 \qquad \text{for all } k \geq k^*$$

Then, it completes the proof of the theorem.

Therefore, the iterative algorithm (4.35) provides for a test of linear separability and has the property of convergence in a finite number of steps if the sample patterns are linearly separable.

4.44 Discussion

The proposed multi-class learning algorithm is a generalization of the Ho-Kashyap algorithm without constraint. Note that the pattern matrix $\underline{A}$, of dimension N x D, is the same as the one used in the Wee - Fu generalization with constraint. Its size is much smaller than that of the pattern matrix used in the Wee - Fu complete generalization, the dimension of which is (R-1)N x RD. Hence, it will take much less time and require less storage to compute $\underline{A}^{\#}$ in the proposed algorithm. However, the proposed algorithm requires computation and storage of R matrices $\underline{E}_j$'s, each of which is of dimension (R - 1)x(R - 1). Also, in each iteration, additional computations on $\underline{Z}_j(k) = \underline{Y}_j(k)\underline{E}_j$ and $\underline{H}_j(k) = [\underline{Z}_j(k) + \underline{\Lambda}_j(k)]\ \underline{E}_j^{-1}$ are needed, although the smaller dimension

of $\underline{A}^{\#}$ saves multiplication time in each increment of $\underline{U}(k)$. Nevertheless, for problems where only a small iterations are involved, this multi-class algorithm offers some advantages in saving computing time and possibly reducing storage requirement.

5.0. EXPERIMENTAL RESULT ON THE PROPOSED MULTI-CLASS LEARNING ALGORITHM

Several experiments have been performed to illustrate the application of the new multi-class learning algorithm proposed in Section 4.0. These experiments are computer simulation studies on the determination of linear decision surfaces for multi-class separation as well as on the test of non-linear separability. The proposed algorithm is programmed in Fortran IV language and all experiments have been run on IBM 360/50 PTSS (University of Pittsburgh Time Sharing System). A description of this program along with the flow chart and program listing is given in Appendix E.

In one example, the set of ten handwritten alphanumeric characters, represented by their direction-time features as given in Section 2.0, were found not linearly separable in the given feature space. This justified the use of the specific metric defined in Section 3.0, for measuring the difference between any pattern and a set of prototype patterns, which led to the non-linear processor discussed in that section. In three other experiments, comparisons were made between the proposed algorithm and the Wee - Fu algorithm to confirm some advantages of the proposed multi-class algorithm.

5.1. Simple Illustrative Examples

5.11 Example 5.1

In the example, the proposed algorithm was used to find three weight vectors for linear discriminant functions to classify a set of 2-dimensional, 3-class sample patterns depicted in Fig. 5.1. The pattern matrix $\underline{A}$ was given by

$$\underline{A} = \begin{bmatrix} 1 & 4.000 & 12.000 \\ 1 & 5.000 & 8.000 \\ 1 & 1.000 & 7.000 \\ 1 & 4.000 & 3.000 \\ 1 & 9.000 & 2.000 \\ 1 & 12.000 & 4.000 \\ 1 & 9.000 & 9.000 \\ 1 & 11.000 & 8.000 \\ 1 & 10.000 & 10.000 \end{bmatrix}$$

where the first three rows are three transposed augmented pattern vectors of class 1, the next three rows of class 2, and the last three rows of class 3. Choose $p = 1$ and $\beta = 1$. For $R = 3$, $\underline{B}(0)$ was given by

x_1	x_2	class
4.000	12.000	1
5.000	8.000	1
1.000	7.000	1
4.000	3.000	2
9.000	2.000	2
12.000	4.000	2
9.000	9.000	3
11.000	8.000	3
10.000	10.000	3

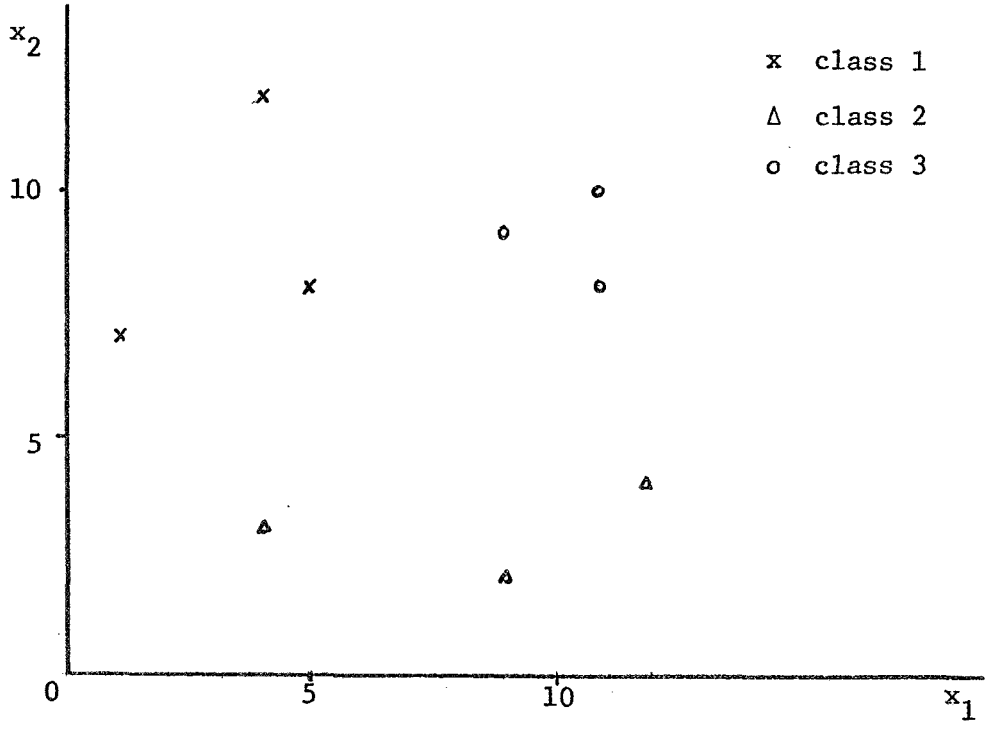


Fig. 5.1 Sample Patterns of Example 5.1.

$$\underline{B}(0) = \begin{bmatrix} \underline{e}_1^t \\ \underline{e}_1^t \\ \underline{e}_1^t \\ \underline{e}_2^t \\ \underline{e}_2^t \\ \underline{e}_2^t \\ \underline{e}_3^t \\ \underline{e}_3^t \\ \underline{e}_3^t \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ -0.5 & 0.866 \\ -0.5 & 0.866 \\ -0.5 & 0.866 \\ -0.5 & -0.866 \\ -0.5 & -0.866 \\ -0.5 & -0.866 \end{bmatrix}$$

In the zeroth iteration, i.e., k = 0, the solution weight matrix $\underline{U}$ was reached since $\underline{A}_j\underline{U}(0)\underline{E}_j > \underline{0}$ for all j = 1, 2, 3, where

$$\underline{U}(0) = \begin{bmatrix} 0.473 & 1.689 \\ -0.143 & -0.058 \\ 0.079 & -0.182 \end{bmatrix}$$

$$\begin{bmatrix} \\ \underline{A}_1\underline{U}(0)\underline{E}_1 \\ \\ \\ \underline{A}_2\underline{U}(0)\underline{E}_2 \\ \\ \\ \underline{A}_3\underline{U}(0)\underline{E}_3 \\ \\ \end{bmatrix} = \begin{bmatrix} 1.909 & 0.660 \\ 0.640 & 0.548 \\ 1.018 & 1.642 \\ 0.578 & 1.581 \\ 1.673 & 1.394 \\ 1.611 & 0.464 \\ 0.545 & 0.807 \\ 1.035 & 0.693 \\ 0.847 & 1.222 \end{bmatrix}$$

Hence, the weight vectors $\underline{w}_1$, $\underline{w}_2$, $\underline{w}_3$ and the three decision hyperplanes were obtained in the following

$$\underline{w}_1 = \underline{U}(0)\,\underline{e}_1 = \begin{bmatrix} 0.473 \\ -0.143 \\ 0.079 \end{bmatrix}$$

$$\underline{w}_2 = \underline{U}(0)\,\underline{e}_2 = \begin{bmatrix} 1.227 \\ -0.428 \\ 0.1185 \end{bmatrix}$$

$$\underline{w}_3 = \underline{U}(0)\,\underline{e}_3 = \begin{bmatrix} 1.699 \\ 0.572 \\ -0.1975 \end{bmatrix}$$

$$g_1(\underline{X}) = \underline{X}^t \underline{w}_1 = 0.473-0.143x_1 + 0.079x_2 = 0$$

$$g_2(\underline{X}) = \underline{X}^t \underline{w}_2 = 1.227-0.48x_1 + 0.1182x_2 = 0$$

$$g_3(\underline{X}) = \underline{X}^t \underline{w}_3 = -1.699+0.572x_1 - 0.1972x_2 = 0$$

5.12 Example 5.2

This example was to illustrate a case of non-linear separability of sample patterns. Consider the 2-dimensional 3-class sample patterns shown in Fig. 5.2, the pattern matrix $\underline{A}$ was given by

$$\underline{A} = \begin{bmatrix} 1 & 4.000 & 12.000 \\ 1 & 11.000 & 9.000 \\ 1 & 1.000 & 7.000 \\ 1 & 4.000 & 3.000 \\ 1 & 2.000 & 10.000 \\ 1 & 12.000 & 4.000 \\ 1 & 9.000 & 2.000 \\ 1 & 11.000 & 8.000 \\ 1 & 10.000 & 10.000 \end{bmatrix}$$

The $\underline{B}(0)$ matrix and the value of p were chosen the same as those in Example 5.1. The non-linear separability of the sample patterns was indicated at the zeroth iteration (k = 0) of the proposed algorithm by noticing that

$$\underline{Z}_j(0) = \underline{Y}_j(0)\underline{E}_j$$

$$= \underline{A}_j\underline{U}(0)\underline{E}_j - \underline{B}_j(0)\underline{E}_j \leq \underline{0} \text{ for all } j$$

x_1	x_2	class
4.000	12.000	1
11.000	9.000	1
1.000	7.000	1
4.000	3.000	2
2.000	10.000	2
12.000	4.000	2
9.000	2.000	3
11.000	8.000	3
10.000	10.000	3

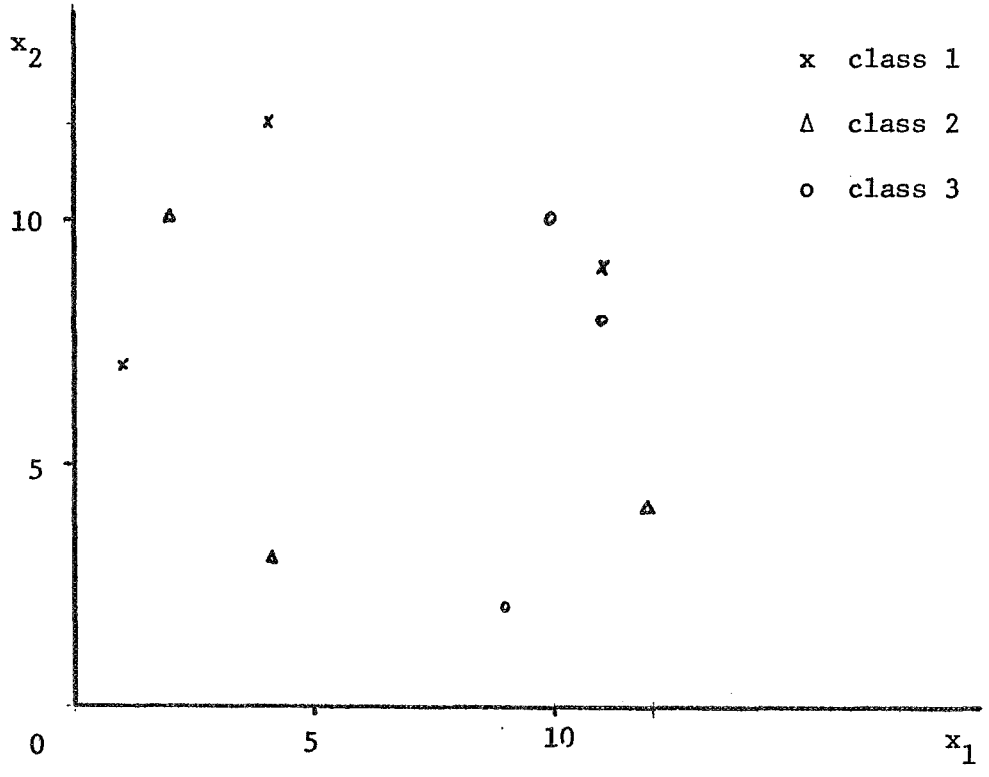


Fig. 5.2 Sample Patterns of Example 5.2.

The value of $\underline{U}(0)$ and $\underline{Z}_j(0)$ are listed below

$$\underline{U}(0) = \begin{bmatrix} -0.348 & 0.861 \\ -0.041 & -0.076 \\ 0.088 & -0.044 \end{bmatrix}$$

$$\underline{Z}_1(0) = \begin{bmatrix} -0.703 & -0.655 \\ -1.177 & -1.827 \\ -1.570 & -0.745 \end{bmatrix}$$

$$\underline{Z}_2(0) = \begin{bmatrix} -0.764 & -0.766 \\ -1.946 & -1.036 \\ -0.977 & -1.901 \end{bmatrix}$$

$$\underline{Z}_3(0) = \begin{bmatrix} -0.769 & -1.648 \\ -1.079 & -0.926 \\ -1.394 & -0.906 \end{bmatrix}$$

5.2. Test of Non-linear Separability of Direction-Time Features of Handwritten Alphanumeric Characters

5.21 Example 5.3

Consider the direction-time features of the ten handwritten characters discussed in Section 2.0 and listed in Appendix B. The example was to show that this set of characters, as represented by this direction-time pattern vectors, are not linearly separable. Let the characters A, C, D, G, H, O, P, Q, S, and 5 be referred to as pattern class 1, 2, 3, . . ., and 10 respectively. Let the prototype set and the

next twenty-four sets listed in Appendix B be taken as twenty-five training sets. In this case, R = 10, d = 20, the dimension of the pattern matrix $\underline{A}$ is 250 x 21, and that of the weight matrix $\underline{U}$ is 21x9. B(0) was chosen as

$$\underline{B}(0) = \begin{bmatrix} \underline{e}_1^t \\ \underline{e}_1^t \\ \vdots \\ \underline{e}_1^t \\ \hline \underline{e}_2^t \\ \vdots \\ \hline \vdots \\ \hline \underline{e}_{10}^t \\ \vdots \\ \underline{e}_{10}^t \end{bmatrix}$$

which is of dimension 250x9, and p=1. After fourth iteration, (k=4), the non-linear separability was indicated. Thus, it was necessary to use a non-linear processor which was described in Section 3.2.

5.3 Comparison of the Proposed Algorithm with the Wee-Fu Algorithm

5.31 Example 5.4

Consider a 3-class training problem of two-dimensional patterns. The sample patterns are represented in Fig. 5.3. There are thirty sample patterns in total, ten in each class. Hence, N = 30, R = 3, d = 2, and D = 3; the pattern matrix $\underline{A}$ for the proposed algorithm is of dimension 30 x 3. By applying the proposed algorithm and choosing p = 1, β = 1, B(0) as given in (4.36), a solution weight matrix $\underline{U}$ was obtained at the eight iteration (k = 8)

$$\underline{U}(8) = \begin{bmatrix} -0.183 & 3.100 \\ -0.297 & -0.175 \\ 0.640 & -0.787 \end{bmatrix}$$

Hence, the weight vectors $\underline{w}_j$, (j = 1,2,3), were given by

$$\underline{w}_1 = \begin{bmatrix} -0.133 \\ -0.297 \\ 0.640 \end{bmatrix}, \quad \underline{w}_2 = \begin{bmatrix} 2.777 \\ -0.003 \\ -1.002 \end{bmatrix}, \quad \underline{w}_3 = \begin{bmatrix} -2.592 \\ -0.200 \\ 0.3615 \end{bmatrix}.$$

The pattern vectors and $\underline{X}^t\ \underline{U}(8)\ \underline{E}_j$ are listed in Table 5.1. This example was computed on the IBM 350/50 PTSS (University of Pittsburgh Time Sharing System). It took 6.6 seconds (cost \$0.62) for obtaining the generalized inverse $\underline{A}^{\#}$ of dimension 3 x 30, 7.8 seconds (\$0.79) for the iterative computation, and a total of 14.2 seconds (\$1,42) for running the complete job.

This example was also experimented by the Wee-Fu algorithms.

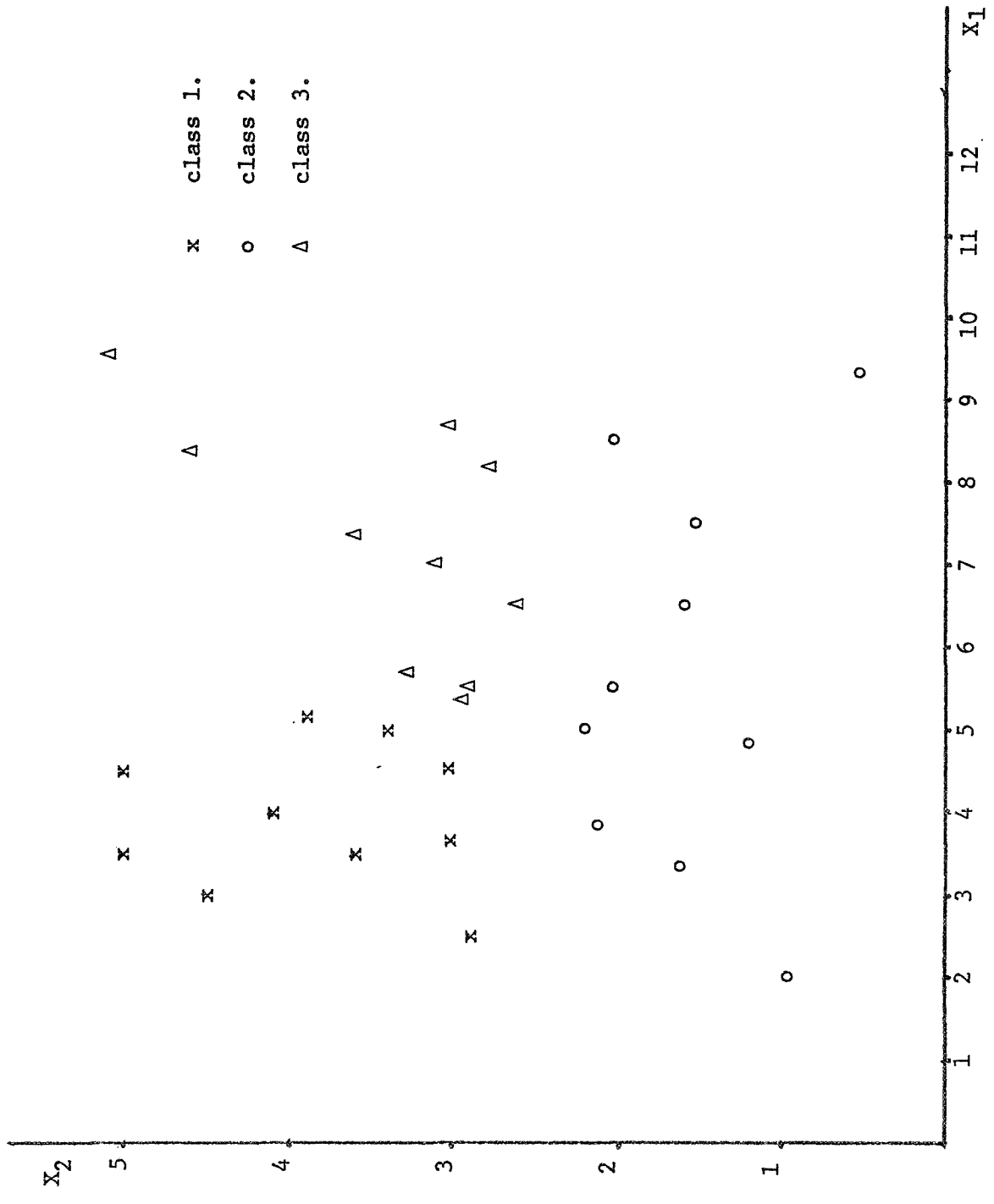


Fig. 5.3 Sample Patterns of Example 5.4.

Table 5.1- Tabulation of $\underline{x}$ and $\underline{X}^t\underline{U}\underline{E}_j$ for Eample 5.4

2-dim. Training Patterns $\underline{x}^t$		Pattern Class j	$\underline{X}^t\underline{U}(8)\underline{E}_j$	
2.5	2.9	1	1.068	1.728
3.0	4.5	1	3.548	1.876
4.5	3.0	1	0.645	0.562
3.5	5.0	1	4.222	1.717
3.5	3.6	1	1.924	1.326
3.7	3.0	1	0.880	1.040
4.0	4.1	1	2.598	1.167
4.5	5.0	1	3.928	1.120
4.9	3.4	1	1.184	0.435
5.1	3.9	1	1.946	0.455
2.0	1.0	2	1.904	3.401
3.3	1.5	2	1.465	2.325
3.9	2.1	2	0.656	1.326
4.8	1.2	2	2.398	2.279
5.0	2.2	2	0.815	0.855
5.5	2.0	2	1.290	0.977
6.5	1.6	2	2.241	1.219
7.5	1.5	2	2.699	1.052
8.5	2.0	2	2.172	0.067
9.3	0.5	2	4.869	1.869
6.5	2.7	3	0.715	0.280
7.0	3.1	3	0.902	0.977
7.4	3.4	3	1.057	1.507
8.1	2.8	3	1.642	0.902
8.4	4.5	3	1.347	3.309
8.5	3.0	3	1.524	1.295
9.5	5.0	3	1.864	4.324
5.7	3.3	3	0.070	0.855
5.4	2.9	3	0.003	0.219
5.6	2.8	3	0.150	0.144

Since the sample patterns are distributed in a general, pairwise,linearly separable manner, their algorithm with constraint failed to separate them; with p=1, $b_{ij}(0) = 1$ for $\underline{x}_i \in C_j$, and $b_{ij}(0) = -1$ for $\underline{x}_i \notin C_j$, the algorithm indicated non-separability after 150 iterations. Apply their algorithm of complete generalization , with p = 1 and $\beta_i(0) = 1$ for all i, solution weight vectors were obtained after three iterations (k = 3),

$$\underline{\alpha}(3) = \begin{bmatrix} \underline{w}_1 \\ \underline{w}_2 \\ \underline{w}_3 \end{bmatrix} = \begin{bmatrix} -0.14639 \\ -0.26379 \\ 0.53966 \\ 1.94767 \\ 0.03506 \\ -0.78923 \\ -1.80129 \\ 0.22844 \\ 0.24944 \end{bmatrix}$$

In this case , however , the pattern matrix is of dimension 60 x 9. It took 23.0 seconds (cost \$2.18) for obtaining the generalized inverse $\underline{A}^{\#}$, 5.8 seconds (\$0.60) for the iterative computations, and a total of 27.8 seconds (\$2.78) for running the complete job.

Even though the proposed algorithm took a few more iterations, and hence required slightly more computer time for iterative computations, than the Wee-Fu algorithm of complete generalization , this disadvantage was obviously overcome by the saving in computing a generalized inverse of a much smaller dimension. Consequently, the proposed algorithm worked better in so far as the total computer time and the cost were concerned.

5.32 Example 5.5

The sample patterns considered in this example consist of sixty patterns of 3-classes and 4 dimensions, twenty in each class. Hence, $R = 3$, $N = 60$, $d = 4$ and $D = 5$. They are distributed in a specially, linearly separable manner so that the Wee-Fu algorithm with constraint can separate them. This is the same example as given by Wee and Fu in their Table 1.[26] The training was performed by the proposed algorithm and also by the Wee-Fu algorithm with constraint on IBM 360/50 in batch processes. In both cases, solution matrices were obtained at the zeroth iteration ($k = 0$). With $p = 1$, $\beta = 1$, $\underline{B}(0)$ as given in (4.36), the proposed algorithm gave its solution weight matrix

$$\underline{U}(0) = \begin{bmatrix} -0.643 & 0.538 \\ 0.143 & -0.165 \\ 0.086 & 0.088 \\ -0.143 & -0.037 \\ -0.050 & 0.024 \end{bmatrix}$$

The weight vectors used in the discriminant functions were then given by

$$\underline{w}_1 = \begin{bmatrix} -0.643 \\ 0.143 \\ 0.086 \\ -0.143 \\ -0.050 \end{bmatrix}, \quad \underline{w}_2 = \begin{bmatrix} 0.144 \\ -0.071 \\ 0.119 \\ -0.104 \\ -0.004 \end{bmatrix}, \quad \underline{w}_3 = \begin{bmatrix} -0.787 \\ 0.214 \\ -0.033 \\ -0.039 \\ -0.046 \end{bmatrix}.$$

It took 17.5 seconds computer time and $1.42 for running the complete job. The sample patterns and $\underline{\chi}^t\ \underline{U}(0)\ \underline{E}_j$ are listed in Table 5.2. On the other hand, with $p = 1$ and $b_{ij}(0) = 1$ for $\underline{\chi}_i \in C_j$, and $b_{ij}(0) = -1$ for $\underline{\chi}_i \notin C_j$, the Wee-Fu algorithm with constraint gave its solution weight matrix

$$\underline{\alpha}(0) = \left[\underline{w}_1, \underline{w}_2, \underline{w}_3 \right]$$

$$= \begin{bmatrix} -1.18140 & 0.73957 & -0.55806 \\ 0.20159 & -0.27150 & 0.06991 \\ 0.09221 & 0.06711 & -0.15933 \\ -0.20730 & 0.02936 & 0.17793 \\ 0.08011 & -0.01589 & -0.06422 \end{bmatrix}$$

It took 17.4 seconds computer time and also $1.42 for the complete job. The result was expected since both algorithms dealt with the same generalized inverse $\underline{A}^{\#}$ of dimension 5 x 60, and both terminated at the zeroth iteration. Since R is small, R = 3 in this case, the additional computation $\underline{U}(0)\ \underline{E}_j$ required in the proposed algorithm did not take appreciably more computer time.

5.33 Example 5.6

This example is the same as the second example given by Wee and Fu in their Table 2.[26] Again, there are sixty sample patterns of 3 classes and 4 dimensions, twenty in each class; hence R = 3, N = 60, d = 4 and D = 5. A comparison will be made between the training by the proposed algorithm and that by the Wee-Fu algorithm of complete generalization.

Table 5.2 - Tabulation of $\underline{x}$ and $\underline{X}^t\underline{U}\underline{E}_j$ for Example 5.5

4-dim. Training Patterns $\underline{x}^t$				Pattern Class j	$\underline{X}^t\underline{U}(0)\underline{E}_j$	
8.560	7.519	2.629	2.843	1	1.204	0.300
7.404	8.209	5.046	1.920	1	1.127	1.247
8.694	5.552	5.141	3.110	1	1.366	1.025
8.403	8.718	4.079	1.412	1	1.366	1.025
5.248	6.316	1.110	8.944	1	1.027	1.775
7.578	6.330	4.439	6.260	1	1.148	0.860
7.150	6.441	0.508	2.643	1	1.520	1.469
6.694	4.852	2.265	4.085	1	1.032	0.817
6.005	3.697	3.545	6.266	1	0.833	0.660
6.879	2.559	3.717	9.613	1	1.704	1.288
7.949	9.378	3.454	7.828	1	0.580	0.305
5.500	9.198	0.080	7.331	1	1.018	0.541
5.851	5.862	1.709	2.180	1	1.707	1.908
5.320	5.135	3.440	7.260	1	1.410	2.480
7.696	2.049	1.481	8.635	1	1.412	0.732
7.924	1.944	0.227	9.013	1	0.780	0.916
8.103	9.616	3.230	6.754	1	0.515	0.797
9.467	3.250	0.176	1.439	1	0.708	1.537
9.284	9.378	6.122	7.161	1	1.001	0.526
8.852	8.407	6.123	6.896	1	1.756	1.919
0.756	3.616	3.717	7.163	2	1.085	0.759
0.205	9.578	3.417	2.380	2	1.352	2.223
1.134	6.346	3.236	0.437	2	1.262	1.392
1.656	8.843	6.937	8.712	2	1.163	1.739
1.091	9.127	8.887	4.688	2	1.932	1.650
2.358	5.509	7.635	9.368	2	1.179	1.010
5.243	8.254	5.593	1.146	2	1.179	1.010
2.221	9.665	3.170	1.717	2	0.053	1.039

Table 5.2 (cont'd)

4-dim. Training Patterns $\underline{x}^t$				Pattern Class j	$\underline{X}^t\underline{U}(0)\underline{E}_j$	
0.091	5.508	4.811	9.291	2	0.616	1.647
0.961	6.051	2.530	1.297	2	1.188	1.604
1.531	5.331	6.540	6.152	2	1.162	1.478
2.314	5.344	5.400	7.649	2	1.458	1.045
0.374	5.559	9.273	9.519	2	1.461	1.154
1.164	3.507	2.886	3.543	2	1.256	1.336
0.763	3.141	2.687	0.502	2	0.894	1.063
4.102	7.601	7.145	7.922	2	2.179	1.489
3.733	6.247	5.952	6.337	2	1.163	1.102
2.903	6.390	7.029	9.934	2	1.096	0.603
3.081	9.997	5.615	2.640	2	2.646	0.976
3.081	9.997	4.615	2.640	2	0.435	0.801
8.286	0.661	9.112	6.899	3	0.475	0.123
5.126	0.854	9.239	8.891	3	1.344	1.629
8.644	1.418	6.672	5.388	3	1.368	0.618
5.591	1.733	7.805	8.658	3	1.352	1.714
8.954	0.688	7.911	1.624	3	1.352	1.714
6.519	4.452	8.670	1.811	3	0.824	0.536
7.815	3.921	6.963	3.187	3	1.506	1.962
6.073	1.609	9.487	0.162	3	1.304	1.131
7.372	2.902	8.860	4.521	3	1.079	0.729
6.900	3.223	9.258	2.009	3	0.928	2.085
9.173	1.493	6.953	3.300	3	0.540	1.015
6.315	4.079	7.905	6.475	3	2.057	1.158
9.200	1.591	6.778	4.759	3	1.119	1.109
9.045	0.275	6.996	2.505	3	1.430	1.056
6.921	2.210	8.221	6.815	3	0.928	1.771
6.617	0.928	7.579	4.089	3	0.527	0.482
6.617	0.928	7.579	4.089	3	0.720	1.671
4.807	1.071	3.486	1.824	3	1.418	1.384
4.302	3.516	9.666	6.246	3	0.912	0.948
8.720	1.465	9.224	5.084	3	1.304	1.132

Both algorithms have been run on the IBM 360/50 PTSS, and solutions obtained at the zeroth iteration ($k = 0$). The pattern matrix $\underline{A}$ used by the proposed algorithm is of dimension 60 x 5. With $p = 1$, $\beta = 1$ and $\underline{B}(0)$ as given in (4.36), the proposed algorithm gave its solution weight matrix at the zeroth iteration

$$\underline{U}(0) = \begin{bmatrix} -0.637 & 0.562 \\ 0.151 & -0.148 \\ 0.069 & 0.098 \\ -0.155 & -0.064 \\ 0.060 & 0.021 \end{bmatrix}$$

The weight vectors used in the discriminant functions were then given by

$$\underline{w}_1 = \begin{bmatrix} -0.067 \\ 0.151 \\ 0.069 \\ -0.155 \\ 0.060 \end{bmatrix}, \quad \underline{w}_2 = \begin{bmatrix} 0.805 \\ -0.204 \\ 0.050 \\ 0.022 \\ -0.019 \end{bmatrix}, \quad \underline{w}_3 = \begin{bmatrix} -0.168 \\ 0.043 \\ -0.119 \\ 0.133 \\ -0.048 \end{bmatrix}.$$

The sample patterns and $\underline{X}^t\underline{U}(0)\ \underline{E}_j$ are listed in Table 5.3. It took 9.2 seconds (cost \$0.84) for obtaining the generalized inverse of $\underline{A}^{\#}$ of dimension 5 x 60, 6.3 seconds (\$1.63) for running the complete job.

However, the Wee-Fu algorithm of complete generalization with $p = 1$ and $\beta_i(0) = 1$ for all i, gave the following solution weight vector

Table 5.3 - Tabulation of $\underline{x}$ and $\underline{X}^t\underline{U}\underline{E}_j$ for Example 5.6

4-dim. Training Patterns $\underline{x}^t$				Pattern Class j	$\underline{X}^t\underline{U}(0)\underline{E}_j$	
8.691	5.552	5.141	3.110	1	1.476	1.345
7.399	9.642	3.766	1.688	1	0.583	0.562
8.403	8.718	4.079	1.412	1	1.060	0.290
8.403	8.718	4.079	1.412	1	1.083	0.981
5.248	6.318	1.101	8.944	1	0.989	1.892
7.578	6.330	4.439	6.260	1	1.030	0.870
7.510	6.441	0.508	2.643	1	1.318	1.593
4.494	4.852	2.265	4.085	1	0.918	0.897
7.404	8.209	5.046	1.920	1	0.581	0.477
8.560	7.519	2.629	2.843	1	1.081	0.661
6.005	3.097	3.545	6.266	1	1.507	1.936
6.879	2.559	3.717	9.613	1	1.197	2.581
7.949	9.378	3.454	7.828	1	0.599	0.958
5.500	9.198	0.080	7.331	1	0.455	0.819
8.150	4.422	3.693	6.654	1	1.687	1.186
5.851	5.862	1.709	2.180	1	2.016	1.592
5.320	5.135	3.440	7.260	1	1.528	1.945
5.160	8.889	1.915	3.205	1	2.052	1.185
8.880	8.703	9.492	9.671	1	1.459	1.225
8.103	9.616	3.230	9.754	1	1.268	0.970
3.384	6.590	9.342	9.039	2	1.248	1.239
0.205	9.578	3.417	2.380	2	1.632	2.253
1.134	6.346	3.236	0.437	2	1.462	1.416
1.656	8.843	6.927	8.712	2	1.289	1.595
1.091	9.127	8.887	4.688	2	2.122	1.424
2.358	5.509	7.635	9.368	2	1.181	0.794
2.358	5.509	7.635	9.368	2	0.335	0.451
3.881	8.524	3.460	3.179	2	0.909	1.755
2.221	9.665	3.170	1.717	2	1.490	1.686
0.901	5.508	4.811	9.291	2	1.341	1.521

Table 5.3 (cont'd)

4-dim. Training Patterns $\underline{x}^t$				Pattern Class j	$\underline{X}^t\underline{U}(0)\underline{E}_j$	
0.961	6.057	2.530	1.297	2	1.515	0.981
0.763	3.141	2.687	0.502	2	0.934	0.960
1.531	5.331	6.540	6.152	2	2.164	1.133
0.756	3.616	3.717	7.163	2	1.219	1.078
2.314	5.344	5.400	7.549	2	1.551	1.030
0.374	5.559	9.273	9.519	2	0.540	0.705
1.164	3.507	2.886	3.543	2	0.592	0.648
3.780	6.908	9.830	7.601	2	0.823	0.892
0.377	4.166	9.329	2.549	2	0.956	1.352
4.102	7.601	7.145	7.922	2	0.788	1.464
5.848	6.343	8.885	5.068	3	1.410	1.801
8.826	0.661	9.112	6.899	3	1.507	0.901
5.126	0.854	9.239	8.891	3	0.691	1.547
8.720	1.465	9.224	5.084	3	0.906	0.720
8.720	1.465	9.224	5.084	3	1.561	2.024
5.591	1.733	7.805	8.648	3	1.293	0.840
8.954	0.698	7.911	1.624	3	0.623	1.022
6.617	0.928	7.579	4.089	3	2.288	1.360
6.519	4.452	8.670	1.811	3	1.262	1.245
9.987	0.934	6.499	2.726	3	1.635	1.205
7.815	3.921	6.963	3.187	3	0.932	1.777
6.073	1.609	9.487	0.162	3	0.657	0.597
7.372	2.902	8.860	4.521	3	0.712	1.695
6.900	3.223	9.258	2.009	3	1.470	1.473
9.173	1.493	6.953	3.300	3	1.005	1.093
6.315	4.079	7.905	6.475	3	1.386	1.260
9.200	1.591	6.778	4.759	3	1.377	1.256
7.045	0.275	6.996	2.505	3	0.600	0.398
6.921	2.210	8.221	6.815	3	1.496	0.382
6.617	0.928	7.579	4.089	3	1.445	1.854

at the zeroth iteration

$$\underline{\alpha}^{t}(0) = \left[\underline{w}_1^t \quad \underline{w}_2^t \quad \underline{w}_3^t \right]$$

$$= [-0.36058 \quad 0.10861 \quad 0.05157 \quad -0.10972 \quad 0.03012 \quad 0.55555$$
$$-0.15618 \quad 0.02860 \quad 0.02052 \quad 0.00141 \quad -0.19487 \quad 0.04735$$
$$-0.08101 \quad 0.08861 \quad -0.03218 \;]$$

The pattern matrix $\underline{A}$ used by this algorithm is of a much larger dimension, 120 x 15. Hence, it took considerably more time, 89.7 seconds, (cost \$8.26) for obtaining the generalized inverse $\underline{A}^{\#}$ of dimension 15 x 120. The other computations took 5.5 seconds (\$0.94), and the complete job required 95.2 seconds (\$9.20).

This example illustrated that, for a fairly large number of sample patterns, there was a tremendous difference in computer time requirement for computing the generalized inverse $\underline{A}^{\#}$ of the proposed algorithm and that of the Wee-Fu algorithm of complete generalization. The difference was almost tenfold in this example. For the case of algorithm termination at the zeroth iteration, the difference in other computations involved was between $\underline{A}_j \; \underline{U}(0) \; \underline{E}_j$, (j = 1,2,...,R), for the former and $\underline{A} \; \underline{\alpha}(0)$ for the latter. This difference was at a relatively insignificant level. As far as the total computer time or the total cost was concerned, the proposed algorithm offered about 90% saving.

5.34 Discussion

Based on the results obtained from the last three examples, the following remarks can be made:

(i) The pattern matrix $\underline{A}$ used in the proposed algorithm and the Wee-Fu algorithm with constraint are identical. Thus, the same amount of computing time will be required for the computation of $\underline{A}^{\#}$. Example 5.5 illustrated that both algorithms were equally efficient. The proposed algorithm requires a slightly more computer time in each iterative computation. However, the Wee-Fu algorithm with constraint attempts to make a more restrictive linear separation, if separable, it is more difficult to achieve in general and it can be expected that it may require more iterations for some problems. Furthermore, it is not a general learning algorithm for linear classification. Therefore, it is more meaningful to compare the proposed algorithm with the Wee-Fu algorithm of complete generalization.

(ii) From both Example 5.4 and Example 5.6, it was noted that, for the Wee-Fu algorithm of complete generalization, the size of the pattern matrix in each case was increased by 2 x 3 times. This resulted in the increase of computing time and hence the computing cost for $\underline{A}^{\#}$ by nearly ten times, as indicated in Table 5.4. It is not known, however, the precise relationship between the increase in computing time for $\underline{A}^{\#}$ and the increase in the size of pattern matrix $\underline{A}$.

(iii) In example 5.4, when the proposed algorithm was used, eight iterations required 7.8 seconds computing time and cost of \$0.79; hence it took 0.98 second per iteration and cost \$0.099 per iteration. When the Wee-Fu algorithm of complete generalization was used, three iterations required 5.8 seconds computing time and cost \$0.60, hence it took 1.93

Table 5.4 - Comparison of Computation Requirements of the Proposed Algorithm and the Wee-Fu Algorithm of Complete Generalization

			Proposed Algorithm	Wee- Fu Algorithm
Example 5.4 R= 3 N= 30 d= 2 D= 3	Computation of $\underline{A}^{\#}$	Dim. of $\underline{A}$	30 x 3	60 x 9
		Time	6.6 sec.	23.0 sec.
		Cost	$ 0.62	$ 2.18
	Iterative Computations	Iterations	8	3
		Time	7.8 sec.	5.8 sec.
		Cost	$0.79	$ 0.60
		Time/iteration	0.98	1.93 sec.
		Cost/iteration	$0.099	$ 0.20
	Total Computation	Time	14.2 sec	27.8 sec.
		Cost	$1.41	$ 2.78
Example 5.6 R= 3 N= 60 d= 4 D= 5	Computation of $\underline{A}^{\#}$	Dim. of $\underline{A}$	60 x 5	120 x 15
		Time	9.2 sec.	89.7 sec.
		Cost	$0.84	$ 8.26
	Iterative Computations	Iterations	0	0
		Time	6.3 sec.	5.5 sec.
		Cost	$0.79	$ 0.94
	Toatl Computation	Time	15.5 sec.	95.2 sec.
		Cost	$1.63	$ 9.20

second per iteration and cost $0.20 per iteration. This comparison is also listed in Table 5.4. Therefore, the time required or the computing cost for each iterative computation for the proposed algorithm is actually less. Even though it may take more iterations to converge, the proposed algorithm is more advantageous provided that the increase in the iterative computations can still be compensated for by its saving in computing $\underline{A}^{\#}$ and $\underline{A}\ \underline{U}(k)$.

6.0 CONCLUSIONS AND SUGGESTIONS FOR FURTHER RESEARCH

There are essentially two parts in the dissertation. The first part concerns the study of an on-line handwritten alphanumeric character recognition system. The characters were encoded by a modification of the quantized direction-time features. In this modification, three kinds of feature components: jumping component, reference component, and end component, were included to enhance the pattern separability and the feasibility of reconstruction. Fifth sets of ten (R=10) characters, A, C, D, G, H. O. P. Q, S, and 5, were collected: each character was written in a standard stroke sequence on a square with 20 x 20 grid points. Eight quantized directions were used along with the twenty dimensional pattern space. One set was chosen as the prototype patterns, $\underline{P}_n$, (n=1,2,...,R). According to Zobrak and Sze, an unknown pattern, $\underline{x}$, was classified according to the minimum metric $\underline{w}_n^t \underline{\phi}(\underline{P}_n,\underline{x})$ where each $\phi_i(\underline{P}_n,\underline{x})$- function given in (3.1) could be interpreted as a metric giving the i-th component of the deviation of $\underline{x}$ from $\underline{P}_n$ in the finite and discrete pattern space characterized by a cyclic group. The weight vectors $\underline{w}_r$, (r=1,2,...,R) were trained by a learning algorithm (3.4) which was a modification of the Zobrak-Sze algorithm following the Duda-Fossum error-correction rule. The pattern vectors extracted from the handwritten characters were experimentally proved not linearly separable but ϕ-separable. By using thirty-five training sets, the solution weight vectors were obtained as listed in Table 3.2. The learning phase took 14 training cycles, required 27'42" computer time on IBM 7090 (in MAD language). This set of weight vectors were used to

recognize all forty-nine sets of characters, the recognition error rate was 2.04% which seemed to be better than the currently known result of 15 or 20% by other approaches tested in all thirty-five characters. Although the sample size was limited in the present study, nevertheless, this preliminary result appears to be rather encouraging. It was noted, however, that this recognition system is very sensitive to the stroke sequence of the handwritten characters.

In the second part of this dissertation, a new learning algorithm for multi-class pattern classification has been presented. This is a generalization of the Ho-Kashyap two-class algorithm, different from that due to Wee and Fu. It is based upon the mapping of pattern vectors of R classes into the neighborhood of R vertices of a (R-1)-dimensional equilateral simplex with its centroid at the origin. Associated with each vertex is a (R-1)-dimensional vector $\underline{e}_j$,(j=1,2,..,R). Matrices $\underline{E}_j$, (j=1,2,...,R), of dimensions (R-1)x(R-1), were defined from these vertex vectors as in (4.14). Two properties of $\underline{E}_j$ were discovered in Section 4.3, which were essential in proving the learning algorithm. If the augmented pattern vectors, $\underline{X}$'s, of dimension D are linearly separable, there exist R weight vectors, $\underline{w}_j$'s, (j=1,2,...,R), for R discriminant functions $g_j(\underline{X}) = \underline{X}^t\underline{w}_j$, (j=1,2,...,R), such that $g_j(\underline{X}) > g_i(\underline{X})$ for $\underline{X}$ C_j and for all i≠j. The learning algorithm is an iterative one to find a D x (R-1) weight matrix $\underline{U}$, from which $\underline{w}_j$ can be obtained by $\underline{w}_j = \underline{U}\,\underline{e}_j$, (j=1,2,...,R). The n_j augmented sample patterns of class C_j are arranged to form a matrix $\underline{A}_j$ in (4.2); the NxD pattern matrix $\underline{A}$ is formed by R submatrix $\underline{A}_j$,(j=1,2,...,R; $n_1+n_2+...+n_R = N$) as defined in (4.1). $\underline{B}$ and $\underline{Y}$ are two N x (R-1) matrices

each of which has submatrices $\underline{B}_j$ and $\underline{Y}_j$ respectively corresponding to the class grouping $\underline{A}_j$ in $\underline{A}$. The learning algorithm (4.35) is rewritten in the following:

$$\underline{U}(0) = \underline{A}^{\#}\ \underline{B}(0)\ ,$$

$$\underline{Y}(k) = \underline{AU}(k) - \underline{B}(k)\ ,$$

$$\underline{B}(k+1) = \underline{B}(k) + p\underline{H}(k)\ ,$$

$$\underline{Z}_j(k) = \underline{Y}_j(k)\ \underline{E}_j\ ,$$

$$\underline{H}_j(k) = [\ \underline{Z}_j(k) + \underline{\Lambda}_j(k)\ ]\underline{E}_j^{-1}\ ,$$

$$\underline{U}(k+1) = \underline{U}(k) + p\underline{A}^{\#}\underline{H}(k)\ .$$

where $0 < p \leq 1$, k is the iteration number, the elements of $\underline{\Lambda}_j(k)$ are given by ${}_{\ell}\Lambda_{jq} = {}_{\ell}Z_{jq}\ \mathrm{Sgn}({}_{\ell}Z_{jq})$, and all row vectors in $\underline{B}_j(0)$ are equal to $\beta\underline{e}_j^{\,t}$, $\beta>0$. The convergence of the algorithm was proved. The algorithm also provide for a test of nonlinear separability of sample patterns by noticing the occurrence of a $\underline{Y}(k)$ such that $\underline{Y}_j(k)\underline{E}_j$ has all non-positive elements for all ℓ and for all j. This algorithm was compared with the Wee-Fu algorithm of complete generalization. It was noticed that the pattern matrix used in this learning algorithm is of much amaller dimension, hence, it requires much less storage and takes much less time for computing $\underline{A}^{\#}$. Computer program for the algorithm was written in Fortran IV and experiments were carried out on IBM 360/50 University of Pittsburgh Time Sharing System. It was demonstrated that even in a case where this learning algorithm took more iterations to converge, the computing time and cost per iteration were considerably less than those of the Wee-Fu algorithm. Consequently for problems where only a relatively small number of iterations are required, this

new multi-class learning algorithm will offer some advantages in saving overall computing time as well as storage.

Based upon the conclusions drawn above, the following problems are suggested for further studies:

(A) to continue the study of the on-line handwritten character recognition system - to train and test all the thirty-five charater classes by grouping them according to the number of strokes and by introducing adaptive structure;

(B) to modify the learning and recognition schemes where a character written in different stroke sequence can be correctly classified; and

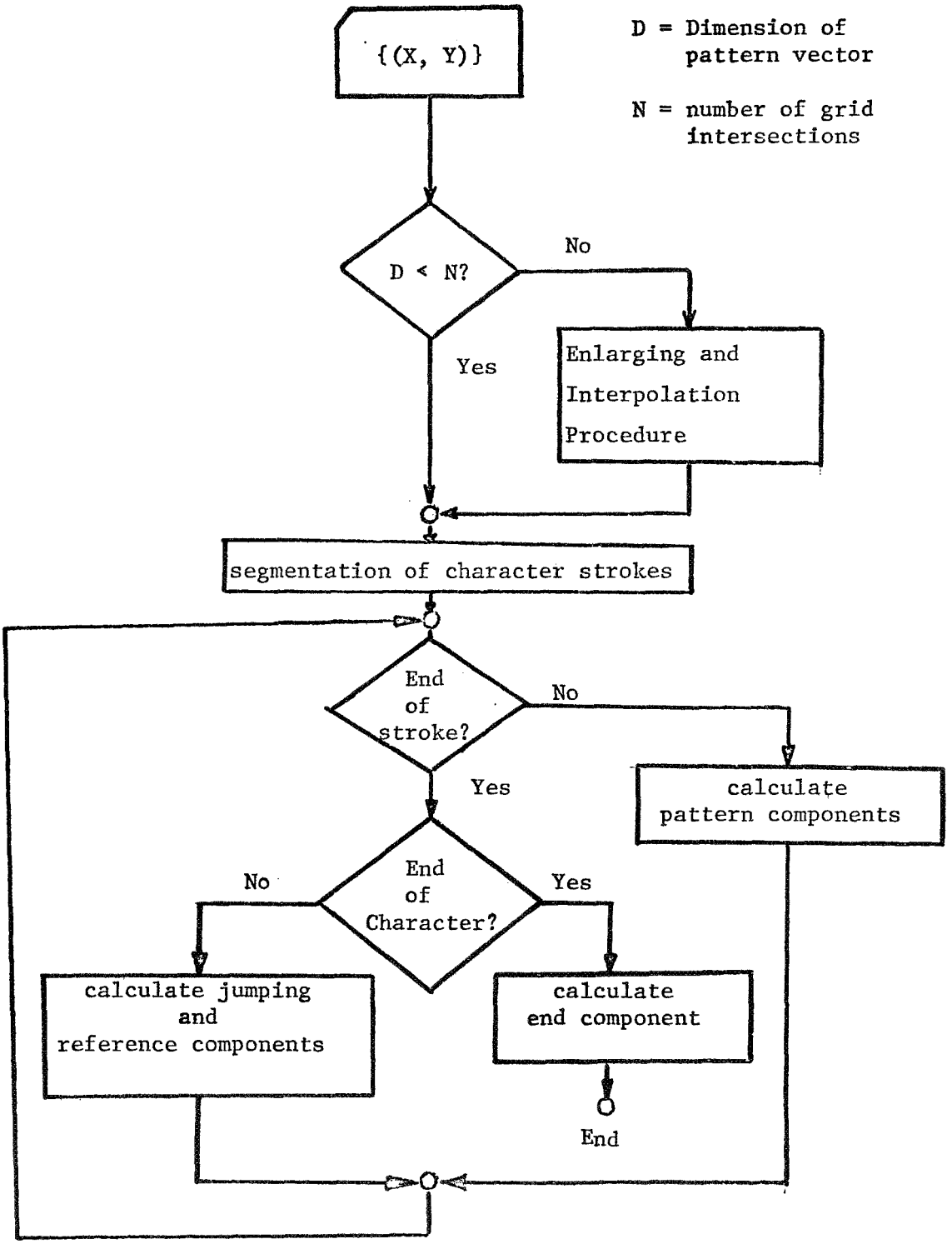
(C) to extend the multi-class learning algorithm (4.35) for linear separation to one for piecewise linear separation.

Appendix A

Digital Computer Program For Generation of Quantized Direction-Time Pattern Vectors

The modified direction-time encoding method discussed in Section 2.2 and 2.3 has been programmed in MAD language and run on IBM 7090 at the computer center of the University of Pittsburgh. The flow chart and the program listing are shown in the following pages.

Flow Chart of Pattern Vector Generator



```
          DIMENSION CHARAC(50),X(200), Y(200), UNKNOW(50), IDCHAR(35),
         1 NP(20), Z7(20), IZ(20), DA(40), PATT(50*50),ES(20),DZ(20)
          INTEGER I,K,X,Y,P1,P2,J,L,SUMIZ,IZ,Q,IM,LL,LLL,DA,ES,SC,NN,
         1 PATT,DD,UNKNOW,CHARAC,II,N,IM,MM
          BOOLEAN S,T,U,V,W,Z
          READ FORMAT ALPHA, IDCHAR(1)...IDCHAR(35)
          VECTOR VALUES ALPHA= $35C1*$
          READ AND PRINT DATA LIM1, LIM2
          READ AND PRINT DATA DD
          FD=DD
START     THROUGH A5, FOR I=1,1,I.G.200
          X(I)=0
A5        Y(I)=0
          LL=1
          LLL=13
A10       READ FORMAT DATA,DA(1)...DA(26)
          VECTOR VALUES DATA=$13(2I2,S2)*$
          I=1
          THROUGH A20, FOR J=LL,1,J.G.LLL
          X(J)=DA(I)
          Y(J)= DA(I+1)
A20       I=I+2
          WHENEVER DA(23).E.0 .AND.DA(24).E.0.AND.DA(25).E.0.AND.DA(26).E.0,
         1 TRANSFER TO A30
          LL=LLL+1
          LLL=LLL+13
          TRANSFER TO A10
A30       PRINT COMMENT $1$
          THROUGH A40, FOR I=1,1,I.G.13
A40       PRINT  FORMAT AAA,X(I),Y(I),X(I+13),Y(I+13),X(I+26),Y(I+26),
         1 X(I+39),Y(I+39),X(I+52),  Y(I+52),X(I+65),Y(I+65),X(I+78),
         2 Y(I+78)
          VECTOR VALUES  AAA=$S6,7(2I5, S3)*$
          ES(0)=-1
          J=1
          I=1
ALP       WHENEVER X(I).NE.0.0, TRANSFER TO BRA
          ES(J)=I-1
          NP(J)= ES(J)-ES(J-1)-1
          WHENEVER X(I+1).E.0.0, TRANSFER TO DEL
          J=J+1
BRA       I=I+1
          TRANSFER TO ALP
DEL       NS=J
          NN=J
          PRINT RESULTS ES(1)...ES(NN), NP(1)...NP(NN)
          SUMNP=0.0
          THROUGH ECH, FOR I=1,1,I.G.NN
ECH       SUMNP= SUMNP+NP(I)
          SUMIZ=0
          THROUGH FOX, FOR I=1,1,I.G.NN
          ZZ(I)=(NP(I)*(FD-2*NS+1.0))/SUMNP
```

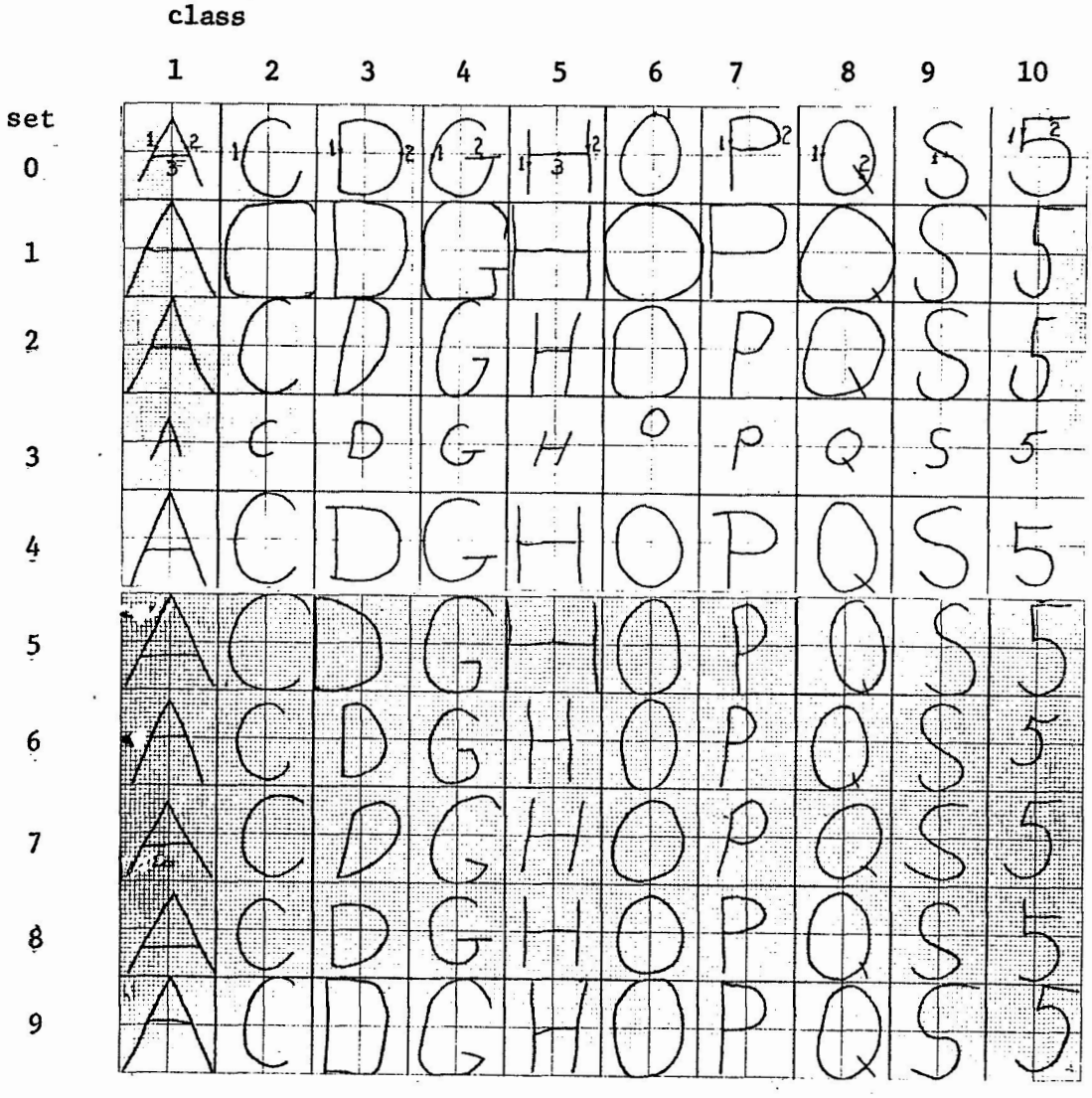
```
            IZ(I)=ZZ(I)+0.5
FOX         SUMIZ= SUMIZ+IZ(I)
            Q=SUMIZ-(FD-2*NS+1.0)
            PRINT RESULTS Q, SUMIZ, IZ(1)...IZ(NN), ZZ(1)...ZZ(NN)
GUL         WHENEVER Q.E.0, TRANSFER TO IND
            MAX=NP(1)
            IM=1
            THROUGH HOT, FOR I=2,1,I.G.NN
            WHENEVER NP(I).G.MAX
            MAX=NP(I)
            IM=I
            OTHERWISE
HOT         END OF CONDITIONAL
            WHENEVER Q.L.0
            IZ(IM)=IZ(IM)+1
            Q=Q+1
            OTHERWISE
            IZ(IM)=IZ(IM)-1
            Q=Q-1
            END OF CONDITIONAL
            PRINT RESULTS Q, MAX, IM, IZ(IM), IZ(1)...IZ(NN)
            TRANSFER TO GUL
IND         THROUGH A70, FOR I=1,1,I.G.50
            THROUGH A70, FOR J=1,1,J.G.50
A70         PATT(I,J) = $ $

            THROUGH D71, FOR J=1,1,J.G.ES(NN)
D71         PATT((Y(J)+1),(X(J)+1)) = $X$
            PRINT FORMAT TITLE
            VECTOR VALUES TITLE = $1H1,S40,37HTHE UNKNOWN PATTERN IS AS SHOWN BELOW
           1 /*$
            PRINT FORMAT PICT, (I=50,-1,I.L.1,(J=1,1,J.G.50,PATT(I,J)))
            VECTOR VALUES PICT = $(S45,50C1)*$

            WHENEVER ES(NN).L.25
            WHENEVER NN.G.1
            THROUGH TED, FOR J=NN,-1,J.L.2
            THROUGH CEC, FOR N= ES(J),-1, N.L.(ES(J-1)+2)
            X(2*(N-NN)+1)=2*X(N)
            Y(2*(N-NN)+1)=2*Y(N)
            WHENEVER X(N-1).E.0.AND. Y(N-1).E.0
            X(2*(N-NN))=X(N-1)
            Y(2*(N-NN))=Y(N-1)
            NN=NN-1
            OTHERWISE
            X(2*(N-NN))= X(N)+X(N-1)
            Y(2*(N-NN))= Y(N)+Y(N-1)
            END OF CONDITIONAL
CEC
TED
            THROUGH TENG,FOR J= ES(1),-1,J.L.2
            Y(2*J-1)= 2*Y(J)
            X(2*J-1)= 2*X(J)
            X(2*(J-1))=X(J)+ X(J-1)
            Y(2*(J-1))=Y(J)+ Y(J-1)
TENG
            X(1)=2*X(1)
            Y(1)=2*Y(1)
            OTHERWISE
            WHENEVER ES(NN).G.20, TRANSFER TO  GTE
```

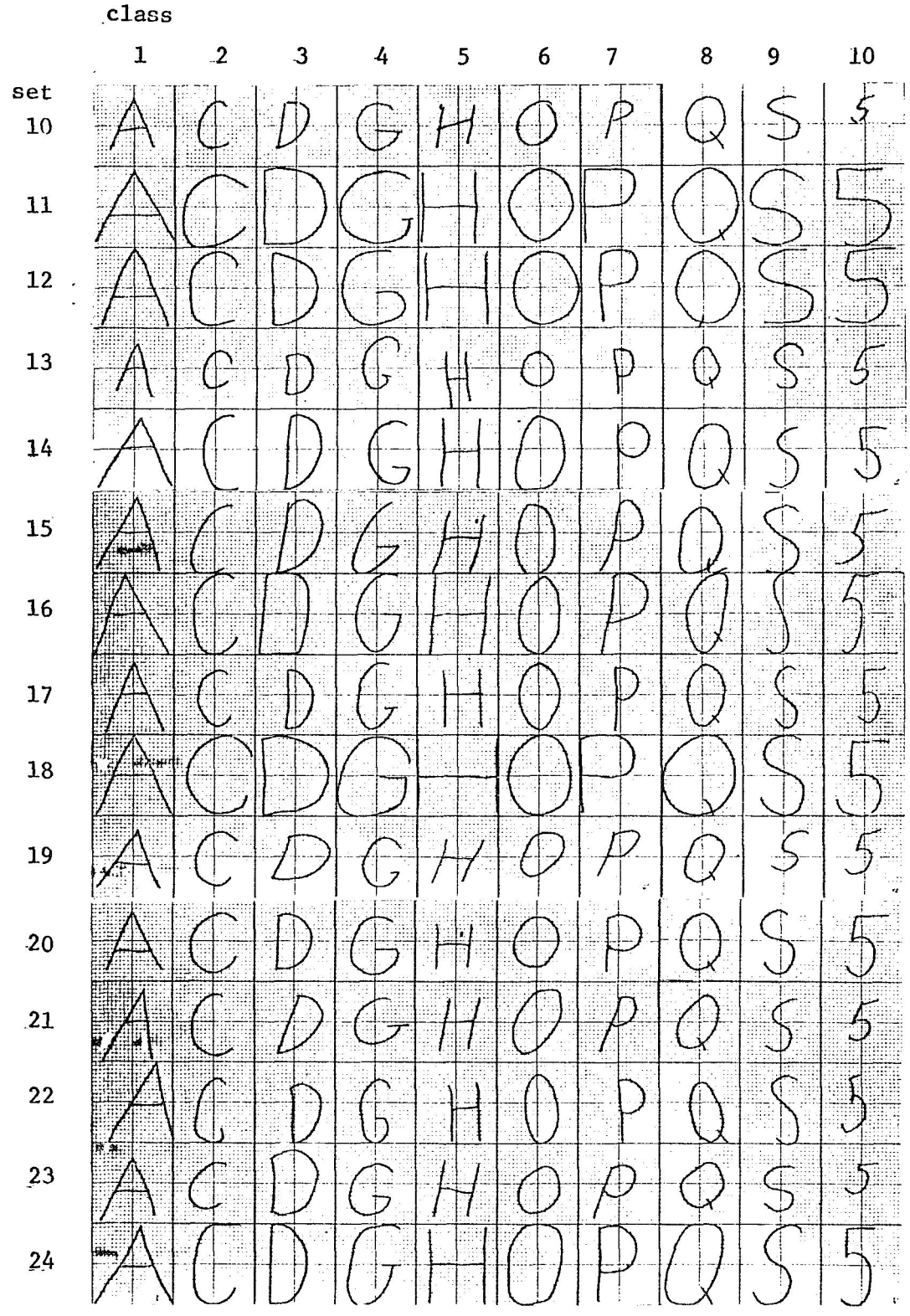
```
            THROUGH CECI ,FOR N=ES(NN),-1,N.L.2
            X(2*N-1)=2*X(N)
            Y(2*N-1)=2*Y(N)
            X(2*N-2)=X(N)+X(N-1)
            Y(2*N-2)=Y(N)+Y(N-1)
CECI
            X(1)=2*X(1)
            Y(1)=2*Y(1)
            END OF CONDITIONAL
            TRANSFER TO A30
            END OF CONDITIONAL
GTE         PRINT COMMENT $1$
            LL=1
            LLL=IZ(1)
            SC=1
            WHENEVER NN.E.1
            MM=1
            SC=3
            OTHERWISE
            MM=NN-1
            END OF CONDITIONAL
            THROUGH LIM, FOR I=1,1,I.G.MM
GTO         NP(I)=ES(I)-ES(I-1)-1
            FLOAT=NP(I)-1.0
            FIZ=IZ(I)
            INCREM= FLOAT/FIZ
            K=IZ(I)



            P1=ES(I-1)+2
            FP1=P1
            FP2=FP1+INCREM
            P2=FP2+0.5
C10         THROUGH C620, FOR J=1,1,J.G.K
            D=X(P1)-X(P2)
            N=Y(P1)-Y(P2)
            WHENEVER .ABS.(D).L.1.0E-10
            WHENEVER N.G.0.0
            UNKNOW(J)=6
            OTHERWISE
            UNKNOW(J)=2
            END OF CONDITIONAL
            TRANSFER TO C20
            OTHERWISE
            END OF CONDITIONAL
            SLOPE=N/D
            U=D.LE.0.0
            V=N.LE.0.0
            W=(SLOPE-LIM1).LE.0.0
            S=(SLOPE+LIM1).LE.0.0
            T=(SLOPE-LIM2).LE.0.0
            Z=(SLOPE+LIM2).LE.0.0
            WHENEVER U.AND.V.AND.W.OR.U.AND.(.NOT.(V)).AND.(.NOT.(S)),
          1 UNKNOW(J) = 0
            WHENEVER U.AND.V.AND.T.AND.(.NOT.(W)), UNKNOW(J) = 1
            WHENEVER U.AND.V.AND.(.NOT.(W)).AND.(.NOT.(T)).OR.(.NOT.(U)).AND.
          1 V.AND.S.AND.Z, UNKNOW(J)=2
            WHENEVER (.NOT.(U)).AND.V.AND.S.AND.(.NOT.(Z)), UNKNOW(J)=3
```

```
          WHENEVER (.NOT.(U)).AND.V.AND.(.NOT.(S)).OR.(.NOT.(U)).AND.(.NOT.(
         1 V)).AND.W, UNKNOW(J) = 4
          WHENEVER (.NOT.(U)).AND.(.NOT.(V)).AND.(.NOT.(W)).AND.T,
         1 UNKNOW(J) = 5
          WHENEVER U.AND.(.NOT.(V)).AND.S.AND.Z.OR.(.NOT.(U.OR.V
         1 .OR.W.OR.T)), UNKNOW(J) = 6
          WHENEVER U.AND.(.NOT.(V)).AND.S.AND.(.NOT.(Z)), UNKNOW(J)=7
C20       P1=P2
          FP2=FP2+INCREM
C620      P2=FP2+0.5
          PRINT RESULTS UNKNOW(1)...UNKNOW(K)

          L=1
          THROUGH KIL , FOR J=LL,1,J.G.LLL
          CHARAC(J)=UNKNOW(L)
KIL       L=L+1

          WHENEVER SC.E.1
          SC=2
          P1=ES(I)
          P2=ES(I)+
          K=1
          LL=LLL+1
          LLL=LL
          TRANSFER TO C10
          OR WHENEVER SC.E.2
          SC=0
          P1=1
          P2=ES(I)+
          K=1
          LL=LLL+1
          LLL=LL
          TRANSFER TO C10
          OR WHENEVER SC.E.3
          SC=4
          P1=1
          P2=ES(NN)
          K=1
          LL=LLL+1
          LLL=LL
          TRANSFER TO C10
          OR WHENEVER SC.E.4
          TRANSFER TO NEAR
          OTHERWISE
          SC=1
          LL=LLL+1
          LLL=LLL+I7(I+1)
LIM       END OF CONDITIONAL
          I=NN
          SC=3
          TRANSFER TO GTO

NEAR      PRINT FORMAT CHAR, CHARAC(1)...CHARAC(DD)
          VECTOR VALUES CHAR= $5011*$
          TRANSFER TO START
          END OF PROGRAM
```

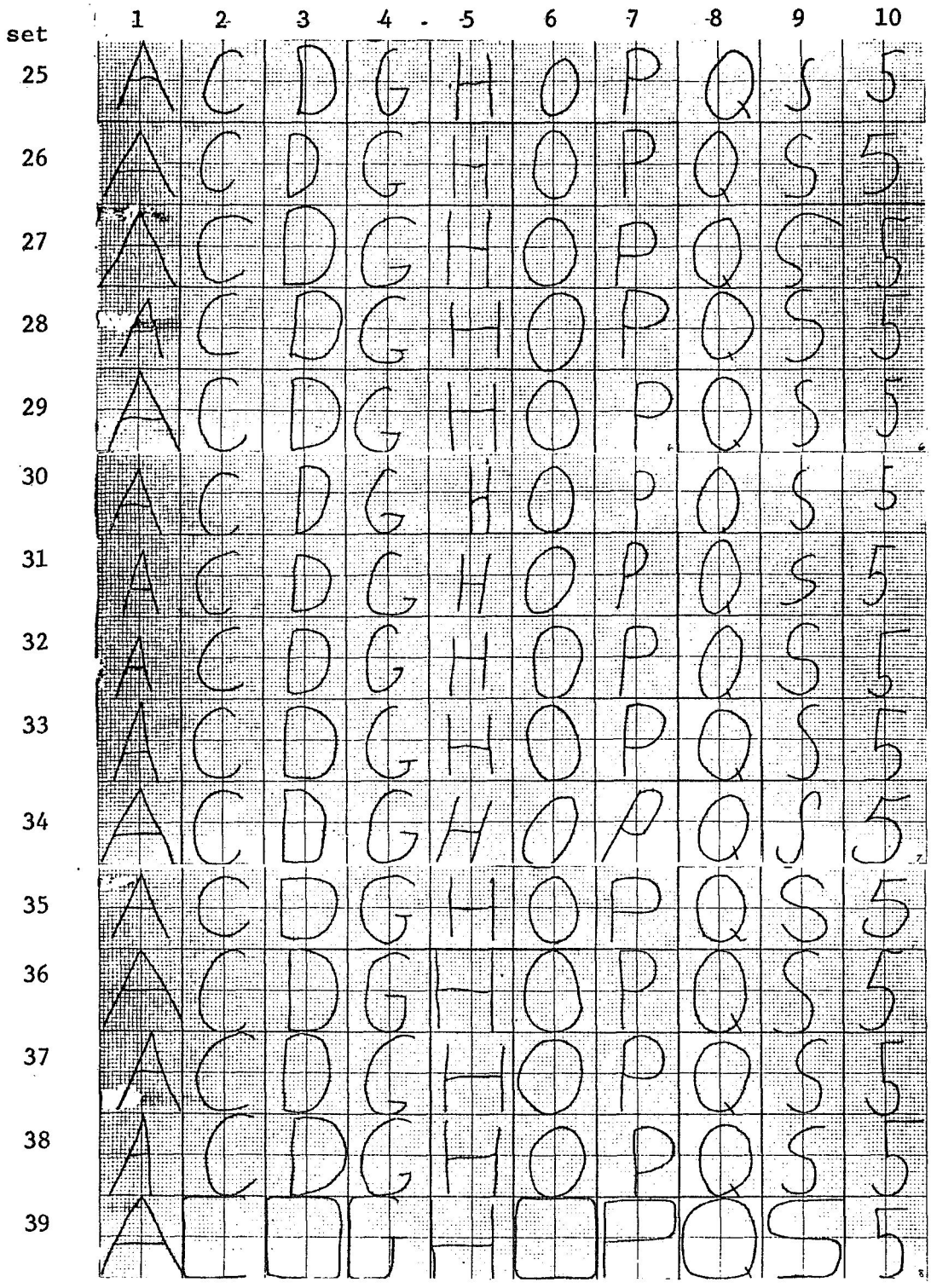
Appendix B

Handwritten Alphanumeric Characters and Their Quantized Direction-Time Pattern Vectors

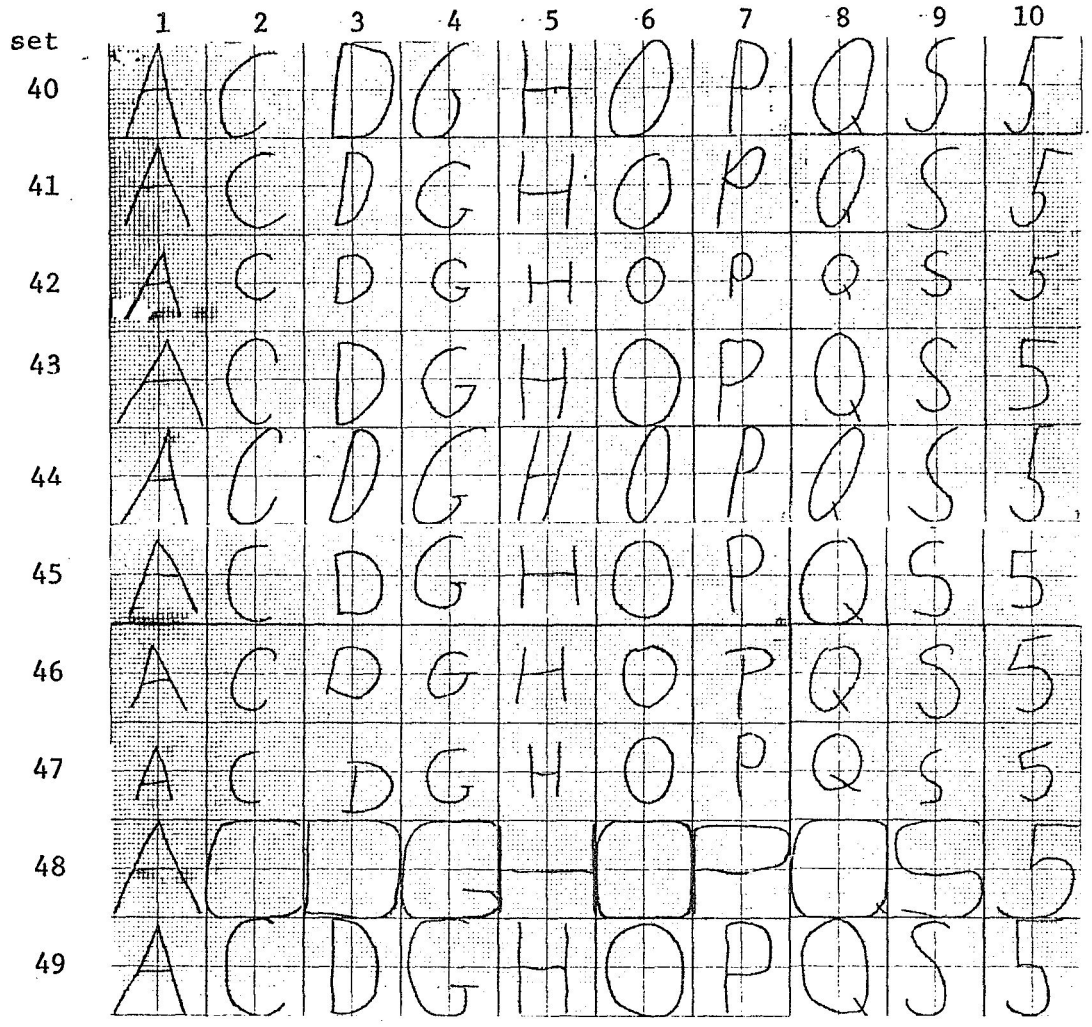



class
1 2 3 4 5 6 7 8 9 10
set
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24

class



class



CHARACTERISTICS OF THE IDEAL SET

A**	5.0	5.0	5.0	5.0	5.0	6.0	1.0	7.0	7.0	6.0	7.0	6.0	7.0	3.0	6.0	.0	.0	.0	1.0	7.0
C**	3.0	4.0	4.0	5.0	5.0	5.0	5.0	6.0	6.0	6.0	6.0	7.0	7.0	7.0	.0	.0	1.0	1.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	.0	7.0	7.0	6.0	6.0	5.0	5.0	4.0	4.0	4.0	6.0
G**	3.0	4.0	5.0	5.0	5.0	6.0	6.0	7.0	7.0	7.0	.0	1.0	2.0	2.0	4.0	5.0	.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	.0	7.0
O**	5.0	5.0	5.0	5.0	6.0	6.0	7.0	7.0	.0	.0	1.0	1.0	1.0	3.0	2.0	2.0	3.0	3.0	4.0	2.0
P**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	1.0	.0	7.0	7.0	5.0	4.0	4.0	4.0	3.0	6.0
Q**	4.0	5.0	6.0	6.0	6.0	7.0	.0	.0	1.0	2.0	2.0	2.0	3.0	3.0	6.0	6.0	7.0	7.0	7.0	6.0
S**	3.0	4.0	4.0	5.0	5.0	7.0	.0	7.0	7.0	7.0	7.0	6.0	6.0	5.0	2.0	4.0	3.0	4.0	2.0	5.0
5**	6.0	6.0	5.0	1.0	.0	7.0	7.0	6.0	5.0	5.0	4.0	4.0	3.0	3.0	1.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 1

A**	5.0	6.0	5.0	5.0	6.0	6.0	1.0	6.0	6.0	7.0	7.0	6.0	6.0	7.0	3.0	6.0	.0	.0	.0	7.0
C**	3.0	4.0	4.0	4.0	4.0	5.0	5.0	5.0	6.0	6.0	6.0	6.0	.0	.0	.0	.0	.0	.0	2.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	.0	7.0	6.0	6.0	5.0	5.0	5.0	4.0	4.0	6.0
G**	2.0	3.0	4.0	4.0	4.0	5.0	6.0	6.0	6.0	6.0	.0	.0	.0	1.0	2.0	4.0	5.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	.0	7.0
O**	4.0	5.0	5.0	5.0	6.0	6.0	6.0	7.0	.0	.0	.0	.0	1.0	2.0	2.0	2.0	3.0	4.0	4.0	2.0
P**	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	.0	.0	7.0	6.0	5.0	4.0	4.0	4.0	4.0	6.0
Q**	5.0	5.0	5.0	6.0	6.0	7.0	.0	.0	.0	1.0	1.0	2.0	2.0	3.0	3.0	3.0	6.0	6.0	6.0	6.0
S**	3.0	3.0	4.0	4.0	4.0	5.0	6.0	5.0	6.0	7.0	7.0	7.0	7.0	6.0	5.0	4.0	3.0	3.0	2.0	5.0
5**	6.0	6.0	6.0	6.0	7.0	6.0	6.0	6.0	5.0	5.0	3.0	3.0	2.0	1.0	.0	.0	7.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 2

A**	6.0	5.0	6.0	6.0	5.0	5.0	1.0	7.0	6.0	7.0	6.0	6.0	7.0	7.0	3.0	6.0	.0	.0	.0	6.0
C**	3.0	4.0	5.0	5.0	5.0	5.0	5.0	6.0	6.0	6.0	6.0	6.0	7.0	6.0	7.0	.0	1.0	.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	7.0	7.0	6.0	6.0	6.0	5.0	5.0	5.0	4.0	6.0
G**	3.0	5.0	5.0	6.0	5.0	6.0	5.0	7.0	6.0	7.0	.0	1.0	1.0	2.0	2.0	4.0	6.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	7.0
O**	3.0	4.0	4.0	5.0	6.0	6.0	6.0	6.0	6.0	7.0	.0	.0	.0	1.0	2.0	2.0	2.0	2.0	3.0	2.0
P**	6.0	6.0	6.0	6.0	6.0	6.0	5.0	6.0	2.0	.0	.0	7.0	.0	7.0	5.0	5.0	5.0	5.0	3.0	6.0
Q**	4.0	5.0	6.0	6.0	6.0	6.0	7.0	.0	.0	1.0	2.0	2.0	2.0	2.0	6.0	6.0	7.0	7.0	7.0	6.0
S**	3.0	4.0	4.0	5.0	6.0	6.0	7.0	7.0	7.0	.0	7.0	5.0	5.0	5.0	4.0	4.0	3.0	3.0	.0	5.0
5**	5.0	6.0	6.0	6.0	.0	7.0	6.0	6.0	5.0	5.0	4.0	3.0	3.0	1.0	2.0	1.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 3

A**	5.0	6.0	5.0	6.0	5.0	5.0	1.0	6.0	6.0	7.0	6.0	6.0	7.0	6.0	3.0	6.0	.0	.0	.0	6.0
C**	4.0	4.0	4.0	4.0	5.0	5.0	5.0	6.0	6.0	6.0	6.0	6.0	6.0	7.0	7.0	.0	.0	1.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	5.0	6.0	2.0	.0	.0	.0	.0	7.0	7.0	6.0	5.0	5.0	4.0	4.0	4.0	6.0
G**	4.0	4.0	5.0	5.0	6.0	6.0	6.0	7.0	.0	.0	.0	1.0	2.0	5.0	6.0	.0	.0	.0	.0	7.0
H**	6.0	5.0	6.0	5.0	6.0	6.0	1.0	.0	6.0	5.0	5.0	6.0	5.0	3.0	6.0	.0	.0	.0	.0	7.0
O**	4.0	4.0	5.0	5.0	5.0	6.0	6.0	7.0	7.0	.0	.0	.0	1.0	1.0	2.0	2.0	2.0	2.0	3.0	2.0
P**	6.0	6.0	6.0	6.0	6.0	5.0	6.0	6.0	2.0	.0	.0	.0	.0	7.0	6.0	5.0	4.0	4.0	4.0	6.0
Q**	4.0	5.0	6.0	5.0	6.0	7.0	.0	.0	1.0	1.0	2.0	2.0	3.0	2.0	6.0	6.0	7.0	7.0	7.0	6.0
S**	3.0	3.0	4.0	4.0	5.0	5.0	6.0	6.0	7.0	7.0	.0	7.0	6.0	6.0	6.0	5.0	4.0	4.0	4.0	5.0
5**	5.0	5.0	6.0	.0	.0	6.0	6.0	6.0	5.0	5.0	5.0	4.0	3.0	1.0	.0	.0	1.0	1.0	.0	.0

CHARACTERISTICS OF SET NO. 4

A**	6.0	5.0	6.0	6.0	5.0	6.0	1.0	6.0	6.0	7.0	6.0	7.0	6.0	7.0	3.0	6.0	.0	.0	.0	6.0
C**	3.0	3.0	4.0	4.0	4.0	5.0	6.0	5.0	6.0	6.0	6.0	6.0	7.0	7.0	7.0	.0	.0	1.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	2.0	4.0	.0	.0	.0	.0	7.0	6.0	6.0	5.0	5.0	4.0	4.0	4.0	6.0
G**	4.0	4.0	5.0	4.0	5.0	6.0	6.0	6.0	7.0	7.0	.0	.0	.0	1.0	3.0	6.0	.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	.0	7.0
O**	5.0	5.0	5.0	6.0	6.0	6.0	7.0	7.0	7.0	.0	1.0	1.0	2.0	2.0	2.0	3.0	3.0	3.0	3.0	2.0
P**	6.0	6.0	6.0	6.0	6.0	6.0	2.0	4.0	1.0	.0	.0	.0	7.0	7.0	5.0	5.0	4.0	4.0	4.0	6.0
Q**	5.0	6.0	6.0	6.0	6.0	7.0	.0	.0	2.0	.0	2.0	2.0	3.0	3.0	4.0	6.0	6.0	7.0	7.0	7.0
S**	3.0	4.0	5.0	5.0	5.0	6.0	7.0	.0	7.0	.0	7.0	7.0	7.0	5.0	5.0	4.0	3.0	3.0	3.0	5.0
5**	6.0	6.0	6.0	.0	.0	.0	7.0	6.0	6.0	5.0	4.0	4.0	3.0	2.0	7.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 5

A**	5.0	5.0	6.0	5.0	5.0	1.0	7.0	7.0	6.0	7.0	7.0	7.0	6.0	7.0	3.0	6.0	.0	.0	.0	7.0
C**	3.0	3.0	4.0	4.0	4.0	5.0	5.0	5.0	5.0	6.0	6.0	6.0	7.0	7.0	7.0	.0	.0	1.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	7.0	7.0	6.0	6.0	6.0	5.0	4.0	4.0	4.0	6.0
G**	3.0	4.0	4.0	5.0	6.0	5.0	6.0	6.0	6.0	6.0	7.0	.0	.0	2.0	2.0	4.0	5.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	.0	.0	7.0
O**	4.0	5.0	5.0	5.0	6.0	6.0	6.0	6.0	7.0	7.0	.0	.0	1.0	2.0	2.0	2.0	2.0	3.0	2.0	2.0
P**	6.0	6.0	6.0	7.0	6.0	6.0	6.0	6.0	5.0	2.0	7.0	.0	.0	7.0	7.0	6.0	5.0	5.0	4.0	6.0
Q**	4.0	5.0	5.0	6.0	6.0	6.0	6.0	7.0	.0	.0	2.0	2.0	2.0	3.0	2.0	3.0	6.0	6.0	6.0	6.0
S**	3.0	4.0	5.0	5.0	6.0	6.0	7.0	.0	7.0	7.0	7.0	6.0	6.0	5.0	5.0	4.0	4.0	3.0	3.0	5.0
5**	6.0	6.0	6.0	.0	6.0	.0	.0	7.0	6.0	6.0	5.0	5.0	4.0	3.0	2.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 6

A**	5.0	6.0	5.0	6.0	6.0	6.0	1.0	6.0	6.0	7.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	6.0
C**	3.0	4.0	4.0	4.0	5.0	5.0	5.0	6.0	6.0	6.0	6.0	7.0	7.0	.0	7.0	.0	1.0	1.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	2.0	1.0	1.0	.0	7.0	7.0	7.0	6.0	5.0	5.0	4.0	4.0	4.0	6.0
G**	3.0	4.0	5.0	5.0	5.0	6.0	6.0	7.0	7.0	7.0	.0	.0	1.0	2.0	3.0	4.0	5.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	7.0
O**	3.0	4.0	5.0	5.0	6.0	6.0	6.0	6.0	6.0	7.0	.0	1.0	1.0	1.0	2.0	2.0	2.0	2.0	2.0	.0
P**	6.0	6.0	6.0	6.0	5.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	7.0	7.0	5.0	5.0	4.0	4.0	6.0
Q**	3.0	5.0	5.0	6.0	5.0	6.0	6.0	7.0	.0	1.0	1.0	2.0	2.0	2.0	3.0	2.0	6.0	6.0	6.0	6.0
S**	3.0	4.0	4.0	5.0	5.0	6.0	7.0	7.0	7.0	7.0	7.0	7.0	7.0	5.0	5.0	5.0	4.0	3.0	3.0	5.0
5**	6.0	5.0	7.0	7.0	7.0	6.0	6.0	5.0	5.0	4.0	4.0	4.0	2.0	1.0	7.0	7.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 7

A**	5.0	5.0	6.0	5.0	5.0	5.0	1.0	6.0	6.0	7.0	7.0	7.0	7.0	7.0	3.0	5.0	.0	.0	.0	7.0
C**	3.0	4.0	4.0	4.0	5.0	5.0	5.0	5.0	5.0	6.0	6.0	7.0	7.0	7.0	7.0	.0	1.0	1.0	1.0	6.0
D**	6.0	6.0	6.0	5.0	6.0	5.0	2.0	.0	.0	1.0	.0	7.0	6.0	5.0	5.0	5.0	4.0	5.0	4.0	6.0
G**	2.0	3.0	4.0	5.0	5.0	5.0	5.0	6.0	7.0	6.0	.0	7.0	.0	1.0	2.0	4.0	5.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	.0	7.0
O**	3.0	4.0	4.0	5.0	5.0	6.0	5.0	6.0	6.0	7.0	.0	.0	1.0	.0	1.0	2.0	2.0	2.0	3.0	2.0
P**	6.0	5.0	6.0	6.0	5.0	6.0	5.0	6.0	2.0	1.0	1.0	.0	7.0	7.0	6.0	5.0	4.0	5.0	3.0	6.0
Q**	4.0	5.0	5.0	5.0	6.0	6.0	6.0	7.0	.0	1.0	1.0	1.0	2.0	2.0	2.0	3.0	6.0	6.0	7.0	6.0
S**	4.0	4.0	4.0	4.0	5.0	5.0	7.0	7.0	7.0	.0	7.0	7.0	5.0	5.0	5.0	4.0	4.0	3.0	3.0	5.0
5**	5.0	5.0	7.0	.0	7.0	7.0	6.0	6.0	5.0	5.0	5.0	4.0	3.0	3.0	1.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 8

A**	5.0	5.0	5.0	5.0	5.0	1.0	6.0	6.0	7.0	7.0	6.0	7.0	7.0	4.0	5.0	.0	.0	.0	.0	7.0
C**	2.0	3.0	4.0	4.0	5.0	5.0	5.0	6.0	6.0	6.0	6.0	6.0	7.0	7.0	.0	.0	.0	1.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	.0	7.0	7.0	6.0	5.0	5.0	4.0	4.0	4.0	6.0
G**	3.0	4.0	5.0	5.0	6.0	6.0	7.0	6.0	7.0	.0	.0	2.0	2.0	4.0	5.0	.0	.0	.0	.0	7.0
H**	6.0	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	7.0
O**	4.0	5.0	5.0	6.0	5.0	7.0	6.0	7.0	.0	.0	.0	1.0	2.0	1.0	2.0	3.0	3.0	3.0	4.0	2.0
P**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	.0	7.0	6.0	5.0	4.0	4.0	4.0	6.0
Q**	4.0	5.0	5.0	6.0	6.0	6.0	7.0	.0	.0	1.0	2.0	2.0	1.0	2.0	3.0	3.0	6.0	6.0	6.0	6.0
S**	3.0	3.0	5.0	5.0	5.0	6.0	6.0	7.0	.0	7.0	7.0	7.0	5.0	5.0	4.0	4.0	4.0	3.0	2.0	5.0
5**	6.0	5.0	7.0	.0	.0	7.0	6.0	6.0	5.0	5.0	4.0	4.0	4.0	2.0	.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 9

A**	5.0	6.0	5.0	6.0	5.0	6.0	1.0	6.0	6.0	7.0	6.0	7.0	6.0	6.0	3.0	5.0	.0	.0	.0	6.0
C**	3.0	4.0	4.0	4.0	5.0	5.0	5.0	6.0	5.0	6.0	6.0	6.0	6.0	7.0	7.0	7.0	7.0	.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	.0	7.0	6.0	6.0	6.0	6.0	4.0	4.0	4.0	6.0
G**	2.0	3.0	4.0	5.0	5.0	6.0	6.0	6.0	6.0	6.0	7.0	.0	.0	1.0	1.0	4.0	5.0	7.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	7.0
O**	3.0	3.0	2.0	2.0	2.0	2.0	1.0	1.0	.0	.0	6.0	6.0	6.0	6.0	6.0	5.0	5.0	4.0	4.0	4.0
P**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	7.0	.0	.0	7.0	7.0	6.0	5.0	4.0	4.0	4.0	6.0
Q**	4.0	4.0	5.0	6.0	6.0	6.0	6.0	7.0	.0	1.0	1.0	2.0	2.0	2.0	3.0	6.0	6.0	7.0	7.0	6.0
S**	3.0	4.0	4.0	5.0	5.0	5.0	6.0	6.0	7.0	.0	.0	7.0	6.0	5.0	5.0	5.0	3.0	4.0	3.0	5.0
5**	6.0	5.0	7.0	.0	7.0	6.0	6.0	5.0	6.0	5.0	4.0	4.0	3.0	1.0	.0	7.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 10

A**	5.0	5.0	5.0	6.0	5.0	5.0	1.0	6.0	7.0	6.0	7.0	7.0	7.0	6.0	3.0	6.0	.0	.0	.0	6.0
C**	6.0	6.0	3.0	4.0	4.0	5.0	5.0	5.0	5.0	6.0	6.0	6.0	6.0	7.0	.0	.0	.0	.0	.0	6.0
D**	6.0	6.0	5.0	6.0	6.0	5.0	6.0	2.0	.0	.0	.0	7.0	7.0	6.0	5.0	5.0	5.0	5.0	5.0	6.0
G**	3.0	4.0	4.0	4.0	5.0	5.0	6.0	6.0	7.0	.0	.0	.0	1.0	1.0	4.0	5.0	.0	.0	.0	6.0
H**	6.0	5.0	6.0	6.0	6.0	5.0	1.0	.0	6.0	6.0	6.0	5.0	6.0	3.0	6.0	.0	.0	1.0	.0	7.0
O**	3.0	4.0	5.0	5.0	5.0	6.0	6.0	6.0	7.0	7.0	.0	1.0	1.0	2.0	1.0	2.0	3.0	2.0	3.0	4.0
P**	6.0	6.0	6.0	5.0	5.0	6.0	6.0	5.0	2.0	.0	.0	.0	.0	7.0	6.0	5.0	5.0	4.0	4.0	6.0
Q**	4.0	5.0	5.0	6.0	6.0	7.0	7.0	.0	.0	1.0	1.0	2.0	2.0	3.0	3.0	6.0	6.0	.0	6.0	6.0
S**	4.0	4.0	4.0	5.0	4.0	6.0	.0	.0	7.0	7.0	.0	6.0	6.0	6.0	5.0	4.0	4.0	4.0	3.0	5.0
5**	5.0	5.0	6.0	.0	.0	.0	6.0	5.0	5.0	5.0	4.0	4.0	2.0	1.0	.0	.0	.0	1.0	1.0	.0

CHARACTERISTICS OF SET NO. 11

A**	5.0	6.0	5.0	5.0	6.0	5.0	1.0	7.0	7.0	7.0	7.0	7.0	7.0	4.0	5.0	.0	.0	.0	.0	7.0
C**	3.0	3.0	3.0	4.0	5.0	5.0	5.0	6.0	5.0	6.0	6.0	6.0	7.0	.0	.0	1.0	.0	1.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	.0	7.0	6.0	6.0	6.0	5.0	4.0	4.0	4.0	6.0
G**	4.0	3.0	4.0	4.0	5.0	5.0	6.0	6.0	7.0	7.0	7.0	7.0	7.0	.0	.0	1.0	1.0	4.0	6.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	.0	7.0
O**	5.0	5.0	6.0	7.0	6.0	7.0	7.0	7.0	1.0	1.0	1.0	1.0	2.0	2.0	3.0	3.0	4.0	4.0	4.0	2.0
P**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	.0	.0	6.0	6.0	5.0	4.0	4.0	4.0	6.0
Q**	6.0	7.0	7.0	.0	1.0	1.0	1.0	2.0	2.0	2.0	3.0	4.0	5.0	5.0	5.0	6.0	7.0	7.0	7.0	7.0
S**	4.0	3.0	3.0	5.0	5.0	5.0	6.0	6.0	.0	.0	.0	7.0	7.0	6.0	5.0	4.0	4.0	4.0	4.0	5.0
5**	6.0	6.0	7.0	.0	.0	.0	.0	6.0	6.0	5.0	4.0	4.0	4.0	2.0	.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 12

A**	5.0	6.0	6.0	5.0	6.0	6.0	1.0	7.0	7.0	7.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	7.0
C**	3.0	3.0	4.0	4.0	5.0	5.0	5.0	6.0	6.0	6.0	6.0	6.0	6.0	7.0	7.0	.0	.0	1.0	.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	7.0	7.0	6.0	6.0	5.0	5.0	4.0	4.0	6.0
G**	4.0	4.0	4.0	5.0	5.0	6.0	7.0	6.0	7.0	.0	.0	.0	1.0	2.0	3.0	6.0	1.0	.0	7.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	.0	7.0
O**	4.0	5.0	6.0	6.0	6.0	6.0	7.0	7.0	.0	.0	1.0	1.0	2.0	2.0	2.0	3.0	3.0	3.0	4.0	2.0
P**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	.0	7.0	6.0	5.0	4.0	4.0	4.0	6.0
Q**	5.0	5.0	5.0	6.0	6.0	7.0	7.0	.0	.0	1.0	2.0	2.0	2.0	3.0	3.0	4.0	6.0	6.0	6.0	6.0
S**	4.0	4.0	4.0	5.0	5.0	6.0	7.0	.0	.0	7.0	.0	7.0	7.0	5.0	4.0	5.0	4.0	4.0	4.0	5.0
5**	6.0	5.0	7.0	.0	.0	7.0	6.0	6.0	5.0	5.0	4.0	4.0	4.0	2.0	.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 13

A**	5.0	6.0	5.0	5.0	6.0	5.0	1.0	6.0	7.0	6.0	7.0	6.0	7.0	7.0	3.0	6.0	.0	.0	.0	6.0
C**	2.0	3.0	3.0	4.0	4.0	5.0	5.0	5.0	5.0	5.0	6.0	6.0	6.0	7.0	7.0	.0	.0	1.0	.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	7.0	7.0	6.0	6.0	5.0	5.0	5.0	4.0	4.0	6.0
G**	2.0	2.0	3.0	4.0	5.0	5.0	5.0	6.0	6.0	6.0	6.0	7.0	7.0	.0	1.0	2.0	5.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	7.0	6.0	2.0	.0	6.0	6.0	6.0	6.0	6.0	7.0	3.0	6.0	.0	.0	1.0	7.0
O**	2.0	3.0	4.0	4.0	5.0	6.0	6.0	6.0	6.0	7.0	7.0	.0	.0	1.0	1.0	2.0	2.0	3.0	2.0	2.0
P**	6.0	6.0	6.0	7.0	6.0	6.0	6.0	6.0	2.0	1.0	.0	.0	7.0	6.0	7.0	6.0	5.0	5.0	4.0	6.0
Q**	4.0	5.0	5.0	6.0	6.0	6.0	7.0	.0	.0	1.0	2.0	3.0	2.0	3.0	6.0	6.0	7.0	6.0	6.0	6.0
S**	1.0	3.0	4.0	4.0	4.0	4.0	5.0	5.0	6.0	7.0	.0	7.0	.0	7.0	6.0	5.0	4.0	4.0	3.0	5.0
5**	5.0	5.0	5.0	5.0	7.0	1.0	.0	6.0	5.0	5.0	4.0	3.0	1.0	4.0	.0	1.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 14

A**	5.0	5.0	5.0	5.0	6.0	5.0	1.0	6.0	6.0	6.0	7.0	6.0	7.0	6.0	3.0	5.0	.0	.0	.0	7.0
C**	4.0	4.0	5.0	5.0	5.0	5.0	6.0	6.0	6.0	6.0	6.0	6.0	7.0	6.0	7.0	7.0	7.0	.0	.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	7.0	6.0	6.0	6.0	5.0	5.0	5.0	5.0	6.0
G**	3.0	4.0	5.0	5.0	5.0	6.0	6.0	6.0	7.0	7.0	7.0	.0	1.0	1.0	4.0	6.0	.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	.0	7.0
O**	5.0	5.0	5.0	6.0	6.0	6.0	6.0	7.0	.0	.0	1.0	1.0	2.0	1.0	2.0	3.0	2.0	3.0	3.0	2.0
P**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	6.0	1.0	1.0	.0	7.0	7.0	6.0	5.0	4.0	4.0	3.0	6.0
Q**	4.0	5.0	6.0	6.0	6.0	6.0	.0	.0	1.0	1.0	2.0	2.0	2.0	3.0	4.0	6.0	6.0	7.0	7.0	6.0
S**	4.0	5.0	5.0	6.0	6.0	6.0	7.0	.0	7.0	7.0	6.0	6.0	5.0	6.0	5.0	4.0	4.0	4.0	3.0	5.0
5**	6.0	6.0	6.0	6.0	1.0	7.0	7.0	6.0	5.0	6.0	4.0	3.0	3.0	2.0	.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 15

A**	5.0	5.0	5.0	6.0	5.0	5.0	1.0	7.0	6.0	6.0	6.0	6.0	6.0	6.0	3.0	5.0	.0	.0	.0	6.0
C**	4.0	4.0	5.0	5.0	5.0	6.0	5.0	5.0	6.0	6.0	5.0	6.0	7.0	6.0	7.0	.0	.0	.0	1.0	6.0
D**	6.0	6.0	5.0	6.0	6.0	6.0	6.0	2.0	7.0	1.0	.0	7.0	6.0	6.0	6.0	5.0	5.0	5.0	4.0	6.0
G**	5.0	5.0	5.0	5.0	5.0	5.0	6.0	6.0	7.0	.0	1.0	1.0	1.0	1.0	2.0	4.0	6.0	1.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	5.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	.0	7.0
O**	5.0	5.0	6.0	6.0	6.0	6.0	6.0	7.0	7.0	.0	1.0	1.0	2.0	2.0	2.0	2.0	3.0	3.0	4.0	2.0
P**	6.0	5.0	6.0	6.0	6.0	5.0	5.0	5.0	2.0	.0	.0	7.0	7.0	6.0	6.0	5.0	5.0	4.0	4.0	6.0
Q**	4.0	5.0	5.0	6.0	6.0	6.0	7.0	.0	1.0	1.0	2.0	2.0	2.0	3.0	6.0	6.0	6.0	.0	.0	6.0
S**	3.0	4.0	4.0	4.0	5.0	5.0	7.0	7.0	7.0	7.0	7.0	7.0	6.0	5.0	5.0	4.0	4.0	4.0	4.0	5.0
5**	5.0	5.0	5.0	7.0	7.0	6.0	5.0	6.0	5.0	4.0	5.0	4.0	1.0	.0	.0	.0	.0	.0	1.0	.0

CHARACTERISTICS OF SET NO. 16

A**	6.0	5.0	6.0	6.0	6.0	6.0	2.0	6.0	6.0	7.0	6.0	7.0	7.0	7.0	3.0	6.0	.0	.0	.0	7.0
C**	4.0	4.0	5.0	5.0	5.0	5.0	6.0	6.0	6.0	6.0	6.0	6.0	7.0	7.0	.0	.0	1.0	1.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	7.0	6.0	6.0	6.0	6.0	6.0	4.0	4.0	4.0	6.0
G**	4.0	4.0	5.0	5.0	6.0	6.0	6.0	6.0	7.0	7.0	.0	1.0	1.0	1.0	2.0	4.0	6.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	.0	7.0
O**	5.0	5.0	5.0	5.0	6.0	5.0	6.0	6.0	6.0	.0	1.0	1.0	1.0	1.0	1.0	2.0	2.0	2.0	3.0	2.0
P**	6.0	6.0	6.0	6.0	6.0	5.0	6.0	2.0	.0	.0	.0	.0	7.0	6.0	5.0	5.0	5.0	4.0	4.0	6.0
Q**	4.0	5.0	5.0	5.0	6.0	6.0	7.0	.0	1.0	2.0	2.0	2.0	2.0	2.0	2.0	2.0	6.0	6.0	7.0	6.0
S**	4.0	5.0	5.0	6.0	6.0	6.0	7.0	5.0	7.0	6.0	6.0	7.0	6.0	6.0	6.0	5.0	5.0	4.0	3.0	6.0
5**	5.0	5.0	5.0	6.0	1.0	7.0	6.0	6.0	6.0	6.0	5.0	5.0	5.0	2.0	.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 17

A**	5.0	6.0	5.0	6.0	5.0	6.0	5.0	1.0	6.0	6.0	7.0	6.0	7.0	6.0	6.0	3.0	6.0	.0	.0	6.0
C**	3.0	4.0	4.0	5.0	5.0	6.0	5.0	6.0	6.0	6.0	6.0	6.0	6.0	7.0	7.0	.0	.0	.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	7.0	7.0	6.0	6.0	5.0	5.0	5.0	5.0	6.0
G**	4.0	5.0	5.0	5.0	6.0	5.0	6.0	6.0	7.0	7.0	7.0	.0	1.0	1.0	1.0	2.0	4.0	6.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	.0	7.0
O**	3.0	4.0	5.0	6.0	5.0	6.0	7.0	6.0	7.0	7.0	7.0	1.0	1.0	2.0	1.0	2.0	2.0	3.0	3.0	2.0
P**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	7.0	7.0	6.0	6.0	5.0	5.0	4.0	6.0
Q**	3.0	4.0	5.0	5.0	6.0	6.0	6.0	6.0	7.0	1.0	1.0	2.0	1.0	2.0	2.0	3.0	6.0	6.0	7.0	6.0
S**	5.0	4.0	4.0	5.0	6.0	6.0	6.0	7.0	7.0	7.0	7.0	6.0	6.0	6.0	6.0	5.0	4.0	3.0	3.0	5.0
5**	6.0	6.0	6.0	6.0	7.0	.0	7.0	6.0	6.0	6.0	5.0	5.0	3.0	2.0	.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 18

A**	5.0	6.0	5.0	5.0	6.0	5.0	1.0	6.0	6.0	7.0	6.0	6.0	7.0	7.0	3.0	6.0	.0	.0	.0	7.0
C**	3.0	3.0	4.0	4.0	5.0	5.0	5.0	6.0	5.0	6.0	7.0	6.0	7.0	7.0	7.0	.0	.0	1.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	2.0	7.0	.0	.0	.0	7.0	7.0	6.0	5.0	5.0	5.0	4.0	4.0	6.0
G**	3.0	3.0	4.0	5.0	5.0	6.0	6.0	6.0	7.0	.0	.0	1.0	1.0	2.0	3.0	4.0	5.0	.0	.0	5.0
H**	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	.0	.0	7.0
O**	4.0	5.0	5.0	6.0	6.0	6.0	7.0	7.0	7.0	.0	1.0	1.0	1.0	2.0	2.0	2.0	3.0	3.0	4.0	4.0
P**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	7.0	.0	.0	.0	7.0	6.0	6.0	4.0	4.0	4.0	4.0	6.0
Q**	4.0	5.0	5.0	5.0	6.0	6.0	6.0	7.0	.0	.0	1.0	1.0	2.0	2.0	3.0	3.0	6.0	6.0	7.0	6.0
S**	3.0	4.0	5.0	5.0	5.0	5.0	7.0	7.0	7.0	7.0	7.0	7.0	6.0	5.0	5.0	4.0	4.0	3.0	3.0	5.0
5**	6.0	6.0	6.0	7.0	.0	7.0	6.0	6.0	5.0	4.0	3.0	4.0	3.0	2.0	.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 19

A**	5.0	5.0	5.0	5.0	5.0	1.0	7.0	7.0	6.0	6.0	7.0	7.0	7.0	3.0	5.0	.0	.0	.0	.0	6.0
C**	3.0	3.0	4.0	4.0	5.0	5.0	6.0	5.0	6.0	6.0	6.0	6.0	7.0	7.0	7.0	.0	.0	1.0	1.0	6.0
D**	5.0	5.0	5.0	5.0	5.0	1.0	.0	.0	.0	7.0	7.0	7.0	7.0	5.0	5.0	5.0	4.0	4.0	4.0	5.0
G**	3.0	4.0	4.0	4.0	5.0	5.0	6.0	6.0	6.0	7.0	7.0	7.0	1.0	1.0	1.0	4.0	6.0	.0	.0	6.0
H**	6.0	5.0	5.0	6.0	5.0	1.0	.0	5.0	6.0	5.0	5.0	5.0	3.0	6.0	.0	.0	.0	.0	.0	7.0
O**	3.0	4.0	5.0	4.0	5.0	5.0	5.0	6.0	6.0	7.0	.0	.0	1.0	1.0	1.0	2.0	1.0	2.0	3.0	2.0
P**	6.0	6.0	5.0	6.0	5.0	5.0	5.0	1.0	.0	.0	.0	.0	.0	6.0	5.0	5.0	5.0	4.0	4.0	6.0
Q**	3.0	5.0	5.0	5.0	6.0	5.0	6.0	6.0	7.0	.0	1.0	1.0	2.0	1.0	2.0	3.0	6.0	6.0	7.0	6.0
S**	4.0	4.0	4.0	5.0	4.0	5.0	6.0	6.0	6.0	.0	7.0	6.0	6.0	5.0	5.0	4.0	4.0	3.0	3.0	5.0
5**	6.0	5.0	6.0	.0	6.0	.0	7.0	7.0	5.0	5.0	4.0	4.0	3.0	1.0	.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 20

A**	5.0	6.0	5.0	5.0	5.0	6.0	1.0	6.0	7.0	6.0	7.0	6.0	7.0	4.0	6.0	.0	.0	.0	.0	7.0
C**	4.0	3.0	4.0	5.0	5.0	5.0	5.0	5.0	6.0	6.0	7.0	7.0	7.0	7.0	.0	1.0	.0	1.0	2.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	7.0	7.0	6.0	6.0	5.0	5.0	5.0	4.0	6.0
G**	3.0	4.0	4.0	5.0	6.0	5.0	6.0	6.0	7.0	.0	.0	1.0	1.0	2.0	4.0	5.0	.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	5.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	.0	7.0
O**	5.0	5.0	5.0	6.0	6.0	6.0	7.0	7.0	7.0	.0	1.0	1.0	1.0	2.0	2.0	2.0	3.0	3.0	4.0	2.0
P**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	.0	7.0	7.0	6.0	5.0	5.0	3.0	4.0	6.0
Q**	4.0	5.0	5.0	6.0	6.0	7.0	7.0	.0	.0	1.0	1.0	2.0	2.0	3.0	3.0	4.0	6.0	6.0	7.0	6.0
S**	3.0	4.0	5.0	5.0	5.0	7.0	7.0	.0	7.0	.0	7.0	6.0	6.0	6.0	5.0	4.0	4.0	3.0	3.0	5.0
5**	5.0	6.0	5.0	.0	.0	7.0	7.0	6.0	6.0	5.0	5.0	4.0	2.0	2.0	.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 21

A**	5.0	5.0	5.0	5.0	6.0	5.0	6.0	1.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	3.0	5.0	.0	.0	6.0
C**	4.0	5.0	4.0	5.0	5.0	5.0	6.0	6.0	6.0	6.0	6.0	6.0	7.0	7.0	.0	.0	.0	.0	1.0	6.0
D**	5.0	5.0	6.0	5.0	5.0	5.0	6.0	1.0	.0	.0	7.0	7.0	6.0	5.0	5.0	5.0	5.0	5.0	4.0	6.0
G**	4.0	5.0	5.0	5.0	5.0	6.0	6.0	7.0	.0	.0	.0	.0	1.0	1.0	4.0	6.0	.0	.0	.0	7.0
H**	6.0	6.0	5.0	6.0	5.0	6.0	1.0	.0	5.0	6.0	5.0	6.0	5.0	6.0	3.0	6.0	.0	.0	.0	7.0
O**	4.0	5.0	5.0	5.0	5.0	6.0	6.0	7.0	7.0	.0	1.0	1.0	1.0	1.0	1.0	2.0	2.0	3.0	3.0	2.0
P**	5.0	6.0	5.0	5.0	6.0	5.0	5.0	6.0	2.0	2.0	.0	.0	7.0	6.0	6.0	5.0	5.0	4.0	3.0	5.0
Q**	5.0	5.0	5.0	6.0	5.0	7.0	7.0	7.0	1.0	1.0	1.0	2.0	2.0	2.0	3.0	6.0	6.0	7.0	7.0	6.0
S**	4.0	4.0	5.0	4.0	5.0	5.0	6.0	6.0	7.0	7.0	7.0	7.0	6.0	5.0	6.0	5.0	4.0	4.0	3.0	5.0
5**	6.0	5.0	5.0	6.0	.0	6.0	.0	1.0	7.0	6.0	5.0	5.0	4.0	4.0	1.0	7.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 22

A**	5.0	5.0	5.0	5.0	5.0	6.0	1.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	3.0	5.0	.0	.0	.0	6.0
C**	2.0	3.0	4.0	5.0	5.0	5.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	7.0	7.0	.0	.0	2.0	2.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	2.0	1.0	1.0	.0	.0	7.0	7.0	6.0	6.0	5.0	5.0	5.0	4.0	6.0
G**	3.0	4.0	5.0	5.0	5.0	6.0	6.0	6.0	6.0	7.0	.0	1.0	2.0	3.0	4.0	5.0	.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	1.0	.0	.0	7.0
O**	3.0	3.0	5.0	5.0	6.0	6.0	6.0	6.0	7.0	7.0	.0	1.0	1.0	2.0	2.0	2.0	3.0	2.0	2.0	2.0
P**	6.0	6.0	7.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	7.0	7.0	6.0	6.0	5.0	5.0	4.0	6.0
Q**	5.0	5.0	6.0	6.0	6.0	7.0	7.0	.0	1.0	1.0	2.0	2.0	3.0	3.0	3.0	6.0	6.0	7.0	7.0	7.0
S**	2.0	3.0	4.0	5.0	5.0	6.0	6.0	7.0	.0	7.0	6.0	7.0	6.0	6.0	6.0	5.0	4.0	4.0	3.0	6.0
5**	7.0	6.0	6.0	4.0	2.0	.0	7.0	6.0	6.0	5.0	5.0	5.0	3.0	2.0	1.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 23

A**	5.0	5.0	5.0	5.0	5.0	5.0	5.0	1.0	7.0	7.0	7.0	6.0	7.0	6.0	3.0	6.0	.0	.0	.0	7.0
C**	4.0	4.0	4.0	5.0	4.0	5.0	5.0	5.0	6.0	5.0	6.0	6.0	7.0	7.0	.0	.0	.0	.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	2.0	1.0	1.0	.0	7.0	7.0	7.0	6.0	6.0	5.0	5.0	4.0	4.0	6.0
G**	3.0	4.0	5.0	5.0	5.0	5.0	5.0	7.0	7.0	7.0	.0	1.0	1.0	2.0	4.0	5.0	.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	5.0	6.0	6.0	6.0	3.0	6.0	.0	7.0	.0	.0	7.0
O**	4.0	4.0	5.0	5.0	6.0	5.0	6.0	7.0	7.0	.0	.0	1.0	.0	1.0	2.0	2.0	2.0	2.0	3.0	2.0
P**	5.0	6.0	5.0	6.0	5.0	6.0	5.0	1.0	.0	.0	.0	.0	7.0	6.0	5.0	5.0	5.0	4.0	3.0	6.0
Q**	3.0	4.0	5.0	5.0	5.0	6.0	7.0	7.0	7.0	1.0	1.0	1.0	2.0	2.0	3.0	6.0	6.0	6.0	7.0	6.0
S**	4.0	4.0	4.0	4.0	5.0	6.0	6.0	7.0	.0	.0	7.0	7.0	6.0	6.0	5.0	4.0	4.0	3.0	3.0	5.0
5**	5.0	6.0	.0	7.0	6.0	6.0	5.0	6.0	4.0	4.0	4.0	4.0	1.0	.0	.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 24

A**	5.0	6.0	5.0	5.0	5.0	5.0	1.0	6.0	6.0	6.0	6.0	7.0	6.0	7.0	3.0	6.0	.0	1.0	.0	6.0
C**	3.0	4.0	4.0	5.0	5.0	5.0	6.0	6.0	5.0	6.0	6.0	6.0	7.0	7.0	7.0	.0	.0	1.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	7.0	.0	7.0	6.0	7.0	6.0	6.0	5.0	5.0	4.0	6.0
G**	4.0	4.0	4.0	5.0	6.0	6.0	6.0	6.0	6.0	7.0	.0	1.0	1.0	2.0	2.0	4.0	6.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	.0	7.0
O**	4.0	4.0	5.0	5.0	6.0	6.0	6.0	7.0	.0	.0	1.0	1.0	2.0	2.0	2.0	2.0	2.0	4.0	2.0	2.0
P**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	7.0	6.0	6.0	6.0	5.0	4.0	5.0	4.0	6.0
Q**	4.0	4.0	5.0	6.0	6.0	6.0	6.0	6.0	.0	.0	1.0	1.0	2.0	2.0	2.0	2.0	6.0	6.0	7.0	6.0
S**	3.0	4.0	4.0	6.0	5.0	5.0	7.0	7.0	.0	7.0	6.0	6.0	5.0	5.0	4.0	5.0	3.0	3.0	5.0	
5**	6.0	6.0	6.0	.0	7.0	7.0	6.0	6.0	6.0	6.0	5.0	4.0	3.0	2.0	.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 25

A**	6.0	6.0	6.0	6.0	5.0	6.0	2.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	6.0
C**	4.0	4.0	5.0	5.0	5.0	5.0	5.0	6.0	5.0	6.0	6.0	6.0	7.0	6.0	7.0	.0	1.0	1.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	7.0	7.0	6.0	6.0	6.0	5.0	5.0	4.0	4.0	6.0
G**	5.0	5.0	6.0	5.0	6.0	6.0	6.0	6.0	7.0	7.0	1.0	1.0	1.0	2.0	4.0	6.0	.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	7.0
O**	5.0	5.0	5.0	5.0	6.0	5.0	6.0	7.0	7.0	.0	.0	1.0	2.0	1.0	2.0	1.0	2.0	3.0	3.0	2.0
P**	6.0	6.0	6.0	6.0	6.0	5.0	6.0	6.0	2.0	.0	.0	.0	7.0	7.0	6.0	5.0	5.0	4.0	3.0	6.0
Q**	4.0	5.0	6.0	5.0	6.0	6.0	7.0	.0	1.0	1.0	2.0	2.0	2.0	3.0	3.0	6.0	6.0	7.0	7.0	6.0
S**	4.0	5.0	5.0	5.0	6.0	7.0	7.0	7.0	6.0	6.0	6.0	6.0	5.0	5.0	4.0	4.0	4.0	4.0	3.0	5.0
5**	6.0	6.0	6.0	7.0	7.0	6.0	6.0	6.0	6.0	5.0	4.0	3.0	3.0	1.0	.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 26

A**	5.0	5.0	6.0	5.0	6.0	1.0	7.0	7.0	7.0	7.0	7.0	6.0	7.0	3.0	6.0	.0	.0	.0	.0	7.0
C**	2.0	3.0	4.0	4.0	5.0	5.0	5.0	5.0	6.0	6.0	6.0	6.0	7.0	6.0	7.0	.0	.0	1.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	7.0	7.0	7.0	6.0	6.0	5.0	5.0	5.0	5.0	6.0
G**	3.0	4.0	5.0	5.0	5.0	5.0	6.0	6.0	7.0	7.0	7.0	7.0	1.0	1.0	4.0	6.0	.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	7.0
O**	5.0	5.0	5.0	6.0	5.0	6.0	7.0	7.0	7.0	.0	1.0	1.0	2.0	1.0	3.0	3.0	3.0	3.0	3.0	2.0
P**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	7.0	6.0	6.0	5.0	5.0	4.0	4.0	6.0
Q**	5.0	5.0	5.0	6.0	7.0	7.0	7.0	.0	1.0	1.0	2.0	2.0	2.0	3.0	3.0	6.0	6.0	7.0	6.0	6.0
S**	3.0	4.0	5.0	5.0	6.0	7.0	7.0	.0	7.0	7.0	7.0	6.0	6.0	5.0	5.0	4.0	4.0	4.0	4.0	5.0
5**	6.0	5.0	6.0	.0	.0	.0	7.0	6.0	6.0	5.0	5.0	4.0	4.0	2.0	.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 27

A**	5.0	6.0	6.0	5.0	5.0	6.0	1.0	7.0	7.0	6.0	7.0	6.0	6.0	7.0	3.0	6.0	.0	.0	.0	7.0
C**	3.0	4.0	4.0	4.0	4.0	5.0	5.0	6.0	5.0	6.0	6.0	6.0	7.0	7.0	.0	.0	.0	1.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	2.0	1.0	.0	.0	7.0	7.0	6.0	6.0	6.0	5.0	5.0	4.0	4.0	6.0
G**	3.0	4.0	4.0	5.0	5.0	5.0	6.0	6.0	6.0	7.0	7.0	.0	.0	1.0	2.0	4.0	5.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	7.0
O**	3.0	5.0	5.0	6.0	6.0	6.0	7.0	7.0	7.0	.0	.0	1.0	1.0	2.0	2.0	2.0	3.0	3.0	4.0	2.0
P**	6.0	6.0	6.0	6.0	6.0	5.0	6.0	2.0	.0	1.0	.0	.0	7.0	6.0	5.0	5.0	4.0	4.0	4.0	6.0
Q**	4.0	4.0	5.0	6.0	6.0	6.0	7.0	.0	.0	1.0	1.0	2.0	2.0	3.0	2.0	6.0	6.0	7.0	7.0	6.0
S**	3.0	4.0	4.0	3.0	4.0	4.0	5.0	5.0	5.0	6.0	6.0	7.0	7.0	.0	7.0	6.0	5.0	5.0	4.0	5.0
5**	6.0	6.0	6.0	6.0	6.0	6.0	.0	7.0	6.0	5.0	5.0	4.0	3.0	2.0	.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 28

A**	5.0	5.0	5.0	6.0	5.0	5.0	1.0	7.0	7.0	6.0	7.0	6.0	6.0	7.0	3.0	6.0	.0	.0	1.0	6.0
C**	4.0	4.0	4.0	5.0	5.0	5.0	5.0	6.0	5.0	6.0	6.0	7.0	6.0	7.0	7.0	.0	1.0	.0	.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	7.0	7.0	6.0	8.0	6.0	5.0	5.0	4.0	2.0	2.0
G**	3.0	5.0	5.0	5.0	5.0	5.0	6.0	6.0	7.0	7.0	.0	.0	1.0	2.0	4.0	6.0	.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	.0	.0	7.0
O**	3.0	5.0	5.0	5.0	6.0	6.0	7.0	6.0	7.0	.0	.0	1.0	1.0	2.0	2.0	2.0	2.0	3.0	4.0	2.0
P**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	7.0	7.0	6.0	5.0	4.0	5.0	4.0	3.0	6.0
Q**	5.0	6.0	6.0	6.0	6.0	7.0	.0	.0	1.0	1.0	2.0	3.0	2.0	3.0	4.0	5.0	6.0	6.0	6.0	6.0
S**	4.0	4.0	4.0	5.0	5.0	6.0	7.0	7.0	.0	.0	7.0	6.0	6.0	5.0	5.0	5.0	4.0	4.0	3.0	5.0
5**	6.0	6.0	6.0	6.0	.0	7.0	7.0	6.0	5.0	5.0	4.0	4.0	3.0	2.0	.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 29

A**	6.0	6.0	6.0	6.0	5.0	6.0	2.0	6.0	6.0	7.0	7.0	6.0	7.0	7.0	3.0	6.0	.0	.0	.0	7.0
C**	3.0	4.0	4.0	5.0	5.0	5.0	6.0	6.0	6.0	6.0	6.0	7.0	7.0	7.0	.0	7.0	1.0	.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	7.0	7.0	7.0	6.0	6.0	5.0	5.0	4.0	4.0	6.0
G**	5.0	5.0	5.0	5.0	6.0	6.0	6.0	6.0	7.0	.0	7.0	1.0	1.0	2.0	4.0	6.0	.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	.0	.0	7.0
O**	5.0	5.0	5.0	6.0	6.0	6.0	7.0	7.0	.0	.0	1.0	1.0	2.0	2.0	3.0	2.0	3.0	3.0	4.0	5.0
P**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	.0	7.0	6.0	5.0	4.0	4.0	4.0	6.0
Q**	4.0	5.0	6.0	6.0	6.0	7.0	7.0	.0	1.0	1.0	2.0	2.0	3.0	2.0	3.0	4.0	6.0	6.0	7.0	6.0
S**	4.0	4.0	5.0	6.0	6.0	7.0	7.0	7.0	7.0	6.0	7.0	6.0	5.0	6.0	5.0	4.0	5.0	3.0	3.0	6.0
5**	6.0	6.0	6.0	7.0	6.0	7.0	6.0	6.0	6.0	5.0	5.0	4.0	3.0	2.0	.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 30

A**	6.0	5.0	5.0	6.0	5.0	6.0	1.0	6.0	6.0	7.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	6.0
C**	4.0	3.0	4.0	5.0	5.0	5.0	6.0	5.0	6.0	6.0	5.0	6.0	7.0	7.0	.0	.0	.0	1.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	5.0	6.0	2.0	.0	.0	.0	7.0	6.0	6.0	6.0	5.0	5.0	5.0	5.0	6.0
G**	3.0	5.0	6.0	5.0	5.0	5.0	6.0	6.0	7.0	7.0	.0	1.0	1.0	2.0	4.0	6.0	.0	7.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	6.0	6.0	5.0	6.0	6.0	6.0	6.0	2.0	6.0	1.0	.0	7.0
O**	3.0	5.0	5.0	5.0	6.0	5.0	6.0	7.0	7.0	.0	.0	1.0	1.0	1.0	2.0	2.0	2.0	3.0	3.0	2.0
P**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	1.0	.0	.0	7.0	6.0	6.0	5.0	4.0	4.0	6.0
Q**	3.0	5.0	5.0	6.0	6.0	6.0	6.0	7.0	.0	.0	1.0	2.0	2.0	2.0	2.0	3.0	6.0	6.0	7.0	6.0
S**	4.0	5.0	5.0	6.0	6.0	6.0	7.0	7.0	.0	7.0	6.0	7.0	6.0	6.0	5.0	5.0	4.0	4.0	3.0	6.0
5**	6.0	6.0	6.0	6.0	6.0	.0	7.0	7.0	5.0	6.0	4.0	5.0	3.0	4.0	2.0	2.0	1.0	.0	.0	1.0

CHARACTERISTICS OF SET NO. 31

A**	5.0	6.0	5.0	5.0	6.0	6.0	5.0	2.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	3.0	6.0	.0	.0	6.0
C**	4.0	4.0	4.0	4.0	5.0	5.0	6.0	5.0	6.0	6.0	6.0	6.0	7.0	6.0	7.0	.0	.0	.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	5.0	6.0	6.0	2.0	.0	.0	.0	7.0	7.0	6.0	6.0	5.0	5.0	4.0	4.0	6.0
G**	4.0	4.0	5.0	5.0	5.0	6.0	5.0	6.0	7.0	.0	.0	.0	.0	1.0	2.0	4.0	6.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	7.0
O**	3.0	4.0	4.0	5.0	5.0	6.0	6.0	6.0	6.0	7.0	.0	.0	1.0	1.0	1.0	2.0	2.0	2.0	3.0	3.0
P**	6.0	6.0	6.0	5.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	7.0	6.0	5.0	5.0	5.0	4.0	6.0
Q**	4.0	5.0	6.0	5.0	6.0	6.0	6.0	7.0	.0	1.0	2.0	1.0	2.0	3.0	2.0	6.0	6.0	.0	6.0	6.0
S**	3.0	4.0	5.0	5.0	6.0	6.0	6.0	7.0	.0	7.0	7.0	5.0	6.0	5.0	4.0	5.0	4.0	4.0	4.0	5.0
5**	6.0	6.0	5.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	5.0	5.0	3.0	2.0	.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 32

A**	5.0	6.0	5.0	5.0	6.0	5.0	1.0	6.0	6.0	6.0	7.0	6.0	7.0	6.0	3.0	6.0	.0	.0	.0	6.0
C**	4.0	4.0	4.0	5.0	5.0	5.0	6.0	5.0	6.0	5.0	6.0	7.0	7.0	7.0	.0	.0	.0	1.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	7.0	6.0	7.0	6.0	5.0	5.0	5.0	4.0	4.0	6.0
G**	2.0	3.0	5.0	5.0	6.0	5.0	6.0	6.0	6.0	6.0	7.0	.0	1.0	1.0	2.0	4.0	6.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	5.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	7.0
O**	4.0	4.0	5.0	5.0	6.0	6.0	6.0	7.0	7.0	.0	1.0	1.0	2.0	1.0	2.0	2.0	2.0	3.0	4.0	4.0
P**	6.0	6.0	6.0	6.0	6.0	6.0	7.0	5.0	2.0	.0	.0	.0	.0	7.0	6.0	5.0	4.0	4.0	4.0	6.0
Q**	4.0	5.0	5.0	6.0	6.0	6.0	6.0	.0	1.0	1.0	1.0	2.0	2.0	2.0	2.0	3.0	6.0	6.0	7.0	6.0
S**	3.0	3.0	4.0	6.0	5.0	6.0	7.0	.0	7.0	.0	7.0	6.0	6.0	5.0	4.0	5.0	4.0	3.0	3.0	5.0
5**	6.0	6.0	6.0	6.0	6.0	6.0	.0	7.0	5.0	5.0	4.0	3.0	2.0	.0	.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 33

A**	6.0	6.0	6.0	6.0	6.0	6.0	2.0	6.0	6.0	6.0	6.0	6.0	7.0	6.0	3.0	6.0	.0	.0	.0	6.0
C**	3.0	4.0	5.0	5.0	5.0	5.0	5.0	6.0	6.0	6.0	7.0	6.0	7.0	7.0	.0	.0	.0	1.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	2.0	1.0	.0	.0	7.0	7.0	6.0	6.0	6.0	5.0	4.0	4.0	4.0	6.0
G**	4.0	5.0	5.0	5.0	5.0	6.0	6.0	7.0	7.0	7.0	.0	.0	1.0	2.0	4.0	6.0	.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	7.0	6.0	6.0	3.0	6.0	.0	.0	.0	.0	7.0
O**	5.0	5.0	6.0	6.0	6.0	7.0	6.0	7.0	7.0	.0	1.0	1.0	2.0	2.0	2.0	2.0	3.0	3.0	4.0	2.0
P**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	.0	7.0	6.0	5.0	5.0	4.0	4.0	4.0	6.0
Q**	4.0	5.0	6.0	6.0	6.0	6.0	7.0	7.0	.0	1.0	1.0	2.0	2.0	2.0	3.0	3.0	6.0	6.0	6.0	6.0
S**	4.0	4.0	5.0	5.0	6.0	6.0	7.0	7.0	7.0	7.0	6.0	6.0	6.0	5.0	5.0	4.0	4.0	3.0	4.0	5.0
5**	6.0	6.0	6.0	6.0	.0	1.0	7.0	6.0	5.0	5.0	4.0	4.0	4.0	2.0	.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 34

A**	5.0	5.0	5.0	5.0	6.0	1.0	6.0	6.0	7.0	7.0	6.0	7.0	6.0	3.0	5.0	.0	.0	.0	.0	7.0
C**	3.0	3.0	4.0	4.0	5.0	5.0	5.0	6.0	6.0	5.0	6.0	6.0	7.0	7.0	.0	.0	.0	1.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	7.0	7.0	6.0	6.0	6.0	6.0	4.0	4.0	4.0	6.0
G**	3.0	4.0	5.0	5.0	5.0	6.0	5.0	6.0	6.0	7.0	.0	.0	1.0	2.0	1.0	4.0	6.0	.0	.0	6.0
H**	6.0	6.0	6.0	5.0	6.0	5.0	1.0	.0	6.0	5.0	6.0	5.0	6.0	5.0	3.0	6.0	.0	.0	.0	7.0
O**	2.0	3.0	4.0	5.0	5.0	5.0	5.0	5.0	6.0	6.0	7.0	.0	.0	1.0	1.0	1.0	2.0	2.0	2.0	2.0
P**	5.0	5.0	5.0	5.0	5.0	5.0	5.0	5.0	1.0	1.0	1.0	7.0	7.0	6.0	5.0	5.0	5.0	4.0	3.0	5.0
Q**	4.0	4.0	5.0	5.0	6.0	6.0	6.0	7.0	.0	.0	1.0	2.0	2.0	2.0	2.0	6.0	6.0	7.0	6.0	6.0
S**	2.0	2.0	3.0	4.0	5.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	5.0	5.0	4.0	3.0	2.0	2.0	5.0
5**	5.0	5.0	.0	.0	.0	6.0	6.0	6.0	5.0	5.0	4.0	4.0	3.0	1.0	1.0	.0	.0	.0	1.0	.0

CHARACTERISTICS OF SET NO. 35

A**	5.0	7.0	5.0	5.0	6.0	5.0	1.0	7.0	6.0	7.0	6.0	7.0	6.0	7.0	3.0	6.0	.0	.0	.0	6.0
C**	3.0	4.0	3.0	4.0	4.0	5.0	5.0	5.0	6.0	6.0	6.0	7.0	7.0	7.0	.0	.0	1.0	1.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	.0	.0	7.0	6.0	6.0	5.0	5.0	4.0	4.0	4.0	6.0
G**	3.0	4.0	3.0	4.0	5.0	5.0	6.0	6.0	6.0	7.0	7.0	.0	1.0	2.0	2.0	4.0	5.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	.0	7.0
O**	4.0	5.0	5.0	5.0	6.0	6.0	6.0	7.0	7.0	.0	.0	1.0	1.0	2.0	3.0	2.0	3.0	3.0	3.0	.0
P**	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	.0	.0	6.0	6.0	6.0	4.0	4.0	4.0	4.0	6.0
Q**	4.0	5.0	6.0	6.0	6.0	7.0	7.0	.0	1.0	2.0	2.0	2.0	2.0	3.0	4.0	6.0	6.0	7.0	.0	7.0
S**	3.0	3.0	4.0	5.0	5.0	5.0	7.0	7.0	7.0	.0	7.0	7.0	6.0	5.0	5.0	4.0	4.0	3.0	3.0	5.0
5**	5.0	6.0	.0	.0	.0	.0	7.0	6.0	5.0	5.0	4.0	4.0	3.0	2.0	1.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 36

A**	5.0	5.0	6.0	6.0	5.0	6.0	1.0	7.0	7.0	7.0	7.0	6.0	7.0	6.0	3.0	6.0	.0	.0	.0	7.0
C**	3.0	3.0	5.0	4.0	5.0	5.0	5.0	6.0	5.0	6.0	6.0	6.0	7.0	7.0	.0	.0	.0	.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	7.0	6.0	6.0	6.0	6.0	6.0	5.0	4.0	4.0	6.0
G**	3.0	4.0	4.0	5.0	6.0	6.0	6.0	6.0	6.0	7.0	.0	.0	2.0	2.0	2.0	4.0	5.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	.0	7.0
O**	5.0	5.0	6.0	5.0	6.0	6.0	6.0	7.0	7.0	.0	.0	1.0	2.0	2.0	2.0	3.0	2.0	3.0	3.0	.0
P**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	7.0	7.0	6.0	5.0	5.0	4.0	4.0	6.0
Q**	5.0	5.0	6.0	6.0	6.0	6.0	7.0	.0	1.0	2.0	2.0	2.0	2.0	3.0	4.0	6.0	6.0	7.0	7.0	6.0
S**	3.0	4.0	4.0	5.0	5.0	7.0	7.0	7.0	7.0	7.0	7.0	6.0	5.0	6.0	5.0	4.0	4.0	4.0	4.0	5.0
5**	6.0	5.0	5.0	5.0	7.0	.0	7.0	7.0	6.0	5.0	5.0	5.0	4.0	3.0	2.0	.0	.0	7.0	.0	.0

CHARACTERISTICS OF SET NO. 37

A**	6.0	6.0	5.0	6.0	5.0	6.0	1.0	7.0	6.0	6.0	6.0	7.0	6.0	7.0	3.0	6.0	.0	.0	.0	6.0
C**	3.0	4.0	4.0	5.0	5.0	5.0	5.0	5.0	6.0	6.0	6.0	7.0	7.0	7.0	.0	.0	.0	.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	2.0	1.0	.0	.0	7.0	7.0	6.0	6.0	6.0	5.0	5.0	4.0	4.0	6.0
G**	4.0	4.0	5.0	5.0	5.0	5.0	6.0	6.0	7.0	6.0	7.0	.0	.0	1.0	4.0	6.0	.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	.0	7.0
O**	4.0	5.0	5.0	5.0	6.0	7.0	6.0	7.0	7.0	.0	1.0	1.0	1.0	2.0	2.0	2.0	3.0	3.0	4.0	.0
P**	6.0	6.0	5.0	6.0	6.0	6.0	6.0	2.0	1.0	1.0	.0	7.0	7.0	6.0	6.0	5.0	4.0	4.0	3.0	6.0
Q**	4.0	5.0	5.0	6.0	6.0	6.0	7.0	.0	.0	1.0	1.0	1.0	2.0	2.0	3.0	4.0	6.0	6.0	7.0	6.0
S**	3.0	4.0	4.0	5.0	6.0	5.0	7.0	7.0	7.0	7.0	7.0	6.0	6.0	5.0	6.0	5.0	4.0	3.0	3.0	5.0
5**	6.0	6.0	6.0	6.0	6.0	.0	7.0	6.0	6.0	5.0	4.0	3.0	3.0	2.0	2.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 38

A**	6.0	6.0	5.0	6.0	5.0	6.0	2.0	6.0	6.0	7.0	6.0	6.0	6.0	6.0	6.0	2.0	6.0	.0	.0	6.0
C**	3.0	4.0	5.0	5.0	5.0	6.0	5.0	5.0	6.0	6.0	6.0	6.0	6.0	7.0	7.0	.0	1.0	1.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	7.0	7.0	6.0	6.0	6.0	5.0	5.0	4.0	4.0	6.0
G**	3.0	4.0	5.0	5.0	5.0	6.0	6.0	6.0	6.0	7.0	7.0	.0	.0	1.0	2.0	4.0	6.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	.0	7.0
O**	4.0	4.0	5.0	5.0	5.0	6.0	6.0	7.0	7.0	.0	.0	1.0	1.0	2.0	1.0	2.0	2.0	3.0	3.0	2.0
P**	6.0	6.0	6.0	6.0	6.0	7.0	6.0	2.0	.0	1.0	.0	.0	7.0	7.0	5.0	5.0	4.0	5.0	4.0	6.0
Q**	4.0	4.0	5.0	5.0	6.0	6.0	7.0	6.0	1.0	.0	1.0	1.0	2.0	2.0	2.0	3.0	6.0	6.0	7.0	6.0
S**	4.0	5.0	4.0	5.0	5.0	6.0	6.0	.0	7.0	7.0	6.0	6.0	6.0	5.0	4.0	4.0	4.0	4.0	3.0	5.0
5**	6.0	6.0	6.0	6.0	.0	7.0	6.0	6.0	5.0	5.0	4.0	3.0	3.0	2.0	.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 39

A**	5.0	5.0	5.0	5.0	6.0	1.0	6.0	7.0	7.0	6.0	7.0	7.0	6.0	3.0	5.0	.0	.0	.0	.0	7.0
C**	3.0	4.0	4.0	4.0	4.0	5.0	6.0	6.0	6.0	6.0	6.0	6.0	7.0	5.0	.0	.0	.0	.0	.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	.0	.0	6.0	6.0	6.0	6.0	4.0	4.0	4.0	4.0	6.0
G**	3.0	4.0	4.0	4.0	6.0	6.0	6.0	6.0	6.0	7.0	.0	.0	1.0	2.0	2.0	4.0	6.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	.0	7.0
O**	4.0	4.0	4.0	4.0	6.0	6.0	6.0	6.0	6.0	.0	.0	.0	.0	1.0	2.0	2.0	2.0	2.0	3.0	2.0
P**	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	.0	.0	7.0	6.0	6.0	4.0	5.0	4.0	4.0	4.0	6.0
Q**	4.0	4.0	4.0	5.0	6.0	6.0	6.0	7.0	.0	.0	1.0	2.0	2.0	2.0	2.0	3.0	6.0	6.0	7.0	6.0
S**	4.0	4.0	4.0	4.0	4.0	5.0	6.0	7.0	.0	.0	.0	.0	7.0	6.0	5.0	4.0	4.0	4.0	4.0	5.0
5**	6.0	6.0	6.0	6.0	1.0	.0	7.0	6.0	6.0	6.0	5.0	4.0	5.0	2.0	.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 40

A**	6.0	6.0	6.0	6.0	5.0	6.0	2.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	6.0
C**	4.0	4.0	5.0	5.0	5.0	5.0	5.0	6.0	5.0	6.0	6.0	6.0	7.0	6.0	7.0	.0	1.0	1.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	7.0	7.0	6.0	6.0	6.0	5.0	5.0	4.0	4.0	6.0
G**	5.0	5.0	5.0	5.0	6.0	5.0	5.0	6.0	6.0	7.0	.0	1.0	1.0	2.0	3.0	2.0	.0	6.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	7.0
O**	5.0	5.0	5.0	5.0	6.0	5.0	6.0	7.0	7.0	.0	.0	1.0	2.0	1.0	2.0	1.0	2.0	3.0	3.0	2.0
P**	6.0	6.0	6.0	5.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	7.0	.0	7.0	6.0	6.0	5.0	4.0	4.0	6.0
Q**	5.0	5.0	5.0	6.0	6.0	6.0	6.0	7.0	.0	1.0	2.0	2.0	2.0	2.0	2.0	3.0	6.0	6.0	6.0	6.0
S**	4.0	5.0	4.0	5.0	6.0	7.0	7.0	7.0	6.0	6.0	6.0	5.0	6.0	5.0	5.0	5.0	4.0	3.0	3.0	5.0
5**	6.0	5.0	5.0	5.0	7.0	7.0	7.0	6.0	6.0	5.0	5.0	4.0	3.0	2.0	1.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 41

A**	5.0	6.0	5.0	6.0	5.0	6.0	1.0	6.0	7.0	6.0	6.0	7.0	6.0	7.0	3.0	6.0	.0	.0	.0	6.0
C**	4.0	4.0	4.0	5.0	5.0	5.0	6.0	5.0	6.0	6.0	7.0	7.0	7.0	7.0	.0	.0	.0	1.0	1.0	6.0
D**	6.0	5.0	6.0	6.0	6.0	5.0	6.0	6.0	2.0	.0	.0	7.0	7.0	6.0	6.0	5.0	5.0	5.0	4.0	6.0
G**	3.0	4.0	5.0	5.0	5.0	5.0	6.0	7.0	7.0	.0	1.0	1.0	2.0	6.0	.0	6.0	6.0	6.0	6.0	6.0
H**	6.0	5.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	.0	7.0
O**	4.0	5.0	5.0	5.0	5.0	5.0	6.0	6.0	7.0	7.0	.0	1.0	1.0	1.0	2.0	2.0	2.0	2.0	3.0	6.0
P**	6.0	6.0	6.0	6.0	6.0	2.0	6.0	2.0	2.0	2.0	2.0	1.0	7.0	1.0	.0	6.0	6.0	4.0	4.0	6.0
Q**	5.0	5.0	5.0	5.0	5.0	6.0	7.0	5.0	.0	1.0	1.0	2.0	1.0	2.0	2.0	3.0	6.0	6.0	7.0	6.0
S**	4.0	4.0	5.0	5.0	5.0	6.0	7.0	7.0	7.0	7.0	6.0	6.0	5.0	6.0	5.0	4.0	4.0	3.0	4.0	5.0
5**	5.0	6.0	6.0	5.0	6.0	5.0	7.0	6.0	6.0	5.0	5.0	3.0	2.0	1.0	1.0	7.0	7.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 42

A**	5.0	5.0	5.0	5.0	6.0	5.0	1.0	6.0	6.0	6.0	6.0	7.0	6.0	6.0	3.0	5.0	.0	.0	.0	6.0
C**	3.0	4.0	4.0	5.0	4.0	5.0	5.0	6.0	6.0	6.0	6.0	7.0	6.0	7.0	.0	.0	.0	1.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	1.0	.0	7.0	7.0	6.0	6.0	5.0	5.0	5.0	4.0	4.0	6.0
G**	2.0	3.0	5.0	4.0	5.0	5.0	6.0	6.0	6.0	7.0	.0	7.0	1.0	1.0	2.0	4.0	6.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	.0	.0	7.0
O**	4.0	5.0	5.0	6.0	5.0	6.0	7.0	6.0	7.0	7.0	1.0	1.0	1.0	1.0	2.0	3.0	2.0	3.0	4.0	2.0
P**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	7.0	1.0	.0	7.0	6.0	6.0	6.0	5.0	4.0	4.0	6.0
Q**	3.0	4.0	5.0	5.0	6.0	6.0	6.0	7.0	.0	1.0	2.0	1.0	1.0	3.0	6.0	6.0	6.0	7.0	6.0	6.0
S**	4.0	4.0	5.0	4.0	6.0	6.0	7.0	.0	7.0	.0	7.0	6.0	6.0	5.0	4.0	4.0	4.0	4.0	3.0	5.0
5**	6.0	6.0	6.0	1.0	7.0	6.0	6.0	6.0	5.0	4.0	4.0	3.0	1.0	1.0	.0	.0	.0	.0	1.0	.0

CHARACTERISTICS OF SET NO. 43

A**	5.0	5.0	5.0	5.0	5.0	5.0	1.0	6.0	6.0	6.0	7.0	6.0	6.0	6.0	3.0	5.0	.0	.0	.0	6.0
C**	2.0	2.0	3.0	3.0	4.0	5.0	5.0	5.0	5.0	6.0	6.0	6.0	7.0	6.0	7.0	7.0	7.0	.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	7.0	7.0	6.0	6.0	5.0	5.0	5.0	4.0	6.0
G**	4.0	4.0	5.0	5.0	5.0	6.0	7.0	7.0	7.0	.0	.0	1.0	1.0	1.0	4.0	6.0	.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	7.0
O**	3.0	4.0	5.0	5.0	5.0	6.0	6.0	7.0	6.0	7.0	.0	.0	1.0	1.0	1.0	2.0	2.0	2.0	3.0	2.0
P**	7.0	6.0	6.0	6.0	6.0	6.0	5.0	6.0	2.0	.0	1.0	.0	.0	7.0	6.0	5.0	5.0	4.0	4.0	6.0
Q**	3.0	5.0	5.0	6.0	6.0	6.0	7.0	7.0	.0	1.0	1.0	2.0	2.0	3.0	2.0	6.0	6.0	7.0	5.0	6.0
S**	2.0	3.0	3.0	4.0	5.0	5.0	6.0	6.0	5.0	7.0	7.0	7.0	7.0	6.0	5.0	4.0	4.0	3.0	3.0	5.0
5**	6.0	6.0	6.0	.0	6.0	.0	7.0	7.0	6.0	5.0	5.0	4.0	4.0	2.0	.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 44

A**	6.0	5.0	6.0	5.0	5.0	5.0	1.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	6.0
C**	2.0	3.0	5.0	5.0	5.0	5.0	5.0	6.0	6.0	5.0	6.0	6.0	6.0	7.0	.0	1.0	.0	1.0	1.0	6.0
D**	6.0	5.0	6.0	6.0	6.0	5.0	6.0	2.0	.0	.0	7.0	6.0	6.0	6.0	6.0	5.0	5.0	5.0	3.0	6.0
G**	4.0	5.0	5.0	5.0	5.0	5.0	6.0	6.0	5.0	7.0	.0	1.0	1.0	2.0	4.0	6.0	.0	.0	6.0	
H**	6.0	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	6.0	.0	.0	7.0
O**	5.0	5.0	6.0	5.0	5.0	6.0	5.0	6.0	7.0	7.0	1.0	1.0	1.0	2.0	1.0	2.0	2.0	2.0	3.0	2.0
P**	6.0	6.0	5.0	6.0	5.0	6.0	5.0	6.0	6.0	2.0	.0	.0	7.0	6.0	6.0	6.0	5.0	5.0	4.0	6.0
Q**	5.0	5.0	5.0	6.0	5.0	6.0	6.0	.0	1.0	1.0	1.0	2.0	2.0	2.0	3.0	6.0	6.0	6.0	6.0	6.0
S**	4.0	5.0	4.0	5.0	5.0	6.0	6.0	7.0	6.0	7.0	7.0	7.0	6.0	5.0	4.0	4.0	4.0	3.0	3.0	5.0
5**	5.0	6.0	6.0	6.0	5.0	6.0	7.0	7.0	7.0	5.0	6.0	5.0	4.0	3.0	3.0	2.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 45

A**	6.0	5.0	5.0	6.0	5.0	6.0	1.0	7.0	7.0	7.0	7.0	7.0	6.0	7.0	3.0	6.0	.0	.0	.0	7.0
C**	4.0	4.0	4.0	4.0	5.0	5.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	7.0	7.0	.0	.0	.0	.0	6.0
D**	7.0	6.0	6.0	6.0	6.0	6.0	2.0	1.0	1.0	.0	7.0	7.0	7.0	6.0	6.0	5.0	5.0	4.0	4.0	6.0
G**	4.0	5.0	5.0	5.0	5.0	5.0	6.0	6.0	6.0	7.0	.0	.0	.0	1.0	2.0	3.0	6.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	.0	.0	7.0
O**	6.0	6.0	6.0	6.0	7.0	7.0	7.0	.0	1.0	2.0	1.0	2.0	2.0	2.0	3.0	3.0	4.0	5.0	5.0	2.0
P**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	1.0	1.0	.0	.0	7.0	7.0	6.0	5.0	5.0	4.0	4.0	6.0
Q**	4.0	5.0	5.0	6.0	6.0	6.0	7.0	.0	1.0	1.0	2.0	2.0	2.0	2.0	3.0	6.0	6.0	7.0	7.0	6.0
S**	4.0	4.0	5.0	6.0	5.0	6.0	7.0	.0	.0	7.0	.0	7.0	6.0	6.0	5.0	4.0	4.0	4.0	4.0	6.0
5**	6.0	6.0	6.0	.0	.0	.0	7.0	6.0	6.0	5.0	4.0	4.0	4.0	2.0	.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 46

A**	5.0	6.0	5.0	6.0	5.0	5.0	1.0	6.0	7.0	6.0	7.0	7.0	7.0	7.0	3.0	6.0	.0	1.0	7.0	7.0
C**	2.0	3.0	4.0	4.0	5.0	5.0	5.0	5.0	6.0	5.0	6.0	6.0	6.0	7.0	.0	.0	.0	1.0	2.0	6.0
D**	5.0	6.0	6.0	5.0	6.0	2.0	.0	.0	.0	.0	7.0	7.0	6.0	6.0	5.0	5.0	4.0	3.0	3.0	6.0
G**	3.0	3.0	5.0	5.0	6.0	6.0	.0	1.0	1.0	3.0	5.0	.0	.0	.0	5.0	6.0	6.0	6.0	6.0	6.0
H**	6.0	6.0	6.0	5.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	7.0	3.0	6.0	.0	1.0	.0	.0	7.0
O**	5.0	5.0	6.0	6.0	6.0	6.0	7.0	.0	.0	.0	1.0	1.0	2.0	2.0	2.0	3.0	4.0	3.0	5.0	2.0
P**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	4.0	1.0	.0	.0	.0	.0	7.0	6.0	5.0	5.0	4.0	6.0
Q**	5.0	6.0	5.0	6.0	6.0	7.0	.0	1.0	1.0	2.0	1.0	2.0	3.0	4.0	5.0	6.0	6.0	7.0	7.0	6.0
S**	2.0	3.0	4.0	5.0	5.0	7.0	.0	7.0	7.0	7.0	7.0	6.0	6.0	5.0	5.0	4.0	4.0	3.0	3.0	5.0
5**	5.0	5.0	7.0	.0	1.0	7.0	7.0	7.0	6.0	5.0	5.0	5.0	3.0	2.0	1.0	.0	1.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 47

A**	5.0	5.0	5.0	6.0	5.0	5.0	1.0	6.0	7.0	6.0	6.0	7.0	6.0	7.0	3.0	6.0	.0	.0	.0	6.0
C**	4.0	4.0	4.0	5.0	4.0	5.0	5.0	5.0	5.0	6.0	6.0	7.0	7.0	7.0	7.0	.0	.0	.0	.0	6.0
D**	7.0	5.0	6.0	6.0	6.0	6.0	2.0	5.0	.0	.0	.0	7.0	7.0	6.0	5.0	5.0	4.0	4.0	3.0	6.0
G**	3.0	4.0	5.0	5.0	5.0	5.0	7.0	6.0	7.0	.0	.0	.0	2.0	6.0	.0	5.0	6.0	6.0	6.0	6.0
H**	6.0	6.0	7.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	7.0
O**	5.0	5.0	5.0	5.0	6.0	6.0	7.0	7.0	7.0	.0	1.0	1.0	1.0	2.0	2.0	2.0	3.0	3.0	3.0	.0
P**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	7.0	2.0	7.0	.0	.0	7.0	7.0	6.0	5.0	5.0	4.0	3.0	6.0
Q**	5.0	5.0	5.0	6.0	6.0	7.0	7.0	1.0	.0	1.0	2.0	3.0	3.0	3.0	6.0	6.0	7.0	7.0	.0	7.0
S**	5.0	4.0	5.0	5.0	5.0	6.0	6.0	7.0	7.0	7.0	7.0	7.0	6.0	6.0	5.0	5.0	4.0	4.0	4.0	6.0
5**	7.0	7.0	.0	.0	.0	7.0	6.0	6.0	5.0	5.0	4.0	3.0	2.0	1.0	1.0	1.0	.0	1.0	.0	1.0

CHARACTERISTICS OF SET NO. 48

A**	5.0	5.0	6.0	6.0	5.0	6.0	1.0	7.0	6.0	6.0	7.0	7.0	6.0	7.0	3.0	6.0	.0	.0	.0	7.0
C**	4.0	4.0	4.0	4.0	4.0	5.0	6.0	6.0	6.0	6.0	6.0	6.0	.0	.0	.0	.0	.0	.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	2.0	1.0	.0	.0	.0	.0	7.0	6.0	6.0	6.0	5.0	4.0	4.0	4.0	4.0	6.0
G**	4.0	4.0	4.0	4.0	5.0	6.0	6.0	6.0	7.0	.0	.0	.0	.0	1.0	3.0	4.0	5.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	.0	.0	7.0
O**	4.0	4.0	4.0	4.0	5.0	6.0	6.0	6.0	6.0	.0	.0	.0	.0	.0	2.0	2.0	2.0	2.0	2.0	7.0
P**	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	.0	.0	.0	6.0	5.0	4.0	4.0	4.0	4.0	4.0	6.0
Q**	4.0	4.0	4.0	4.0	6.0	6.0	6.0	6.0	.0	.0	.0	.0	2.0	2.0	2.0	2.0	6.0	6.0	7.0	6.0
S**	4.0	4.0	4.0	4.0	5.0	5.0	6.0	7.0	.0	.0	.0	.0	7.0	6.0	5.0	4.0	4.0	4.0	4.0	5.0
5**	6.0	6.0	7.0	1.0	.0	7.0	6.0	6.0	5.0	4.0	4.0	4.0	3.0	1.0	1.0	.0	.0	.0	.0	.0

CHARACTERISTICS OF SET NO. 49

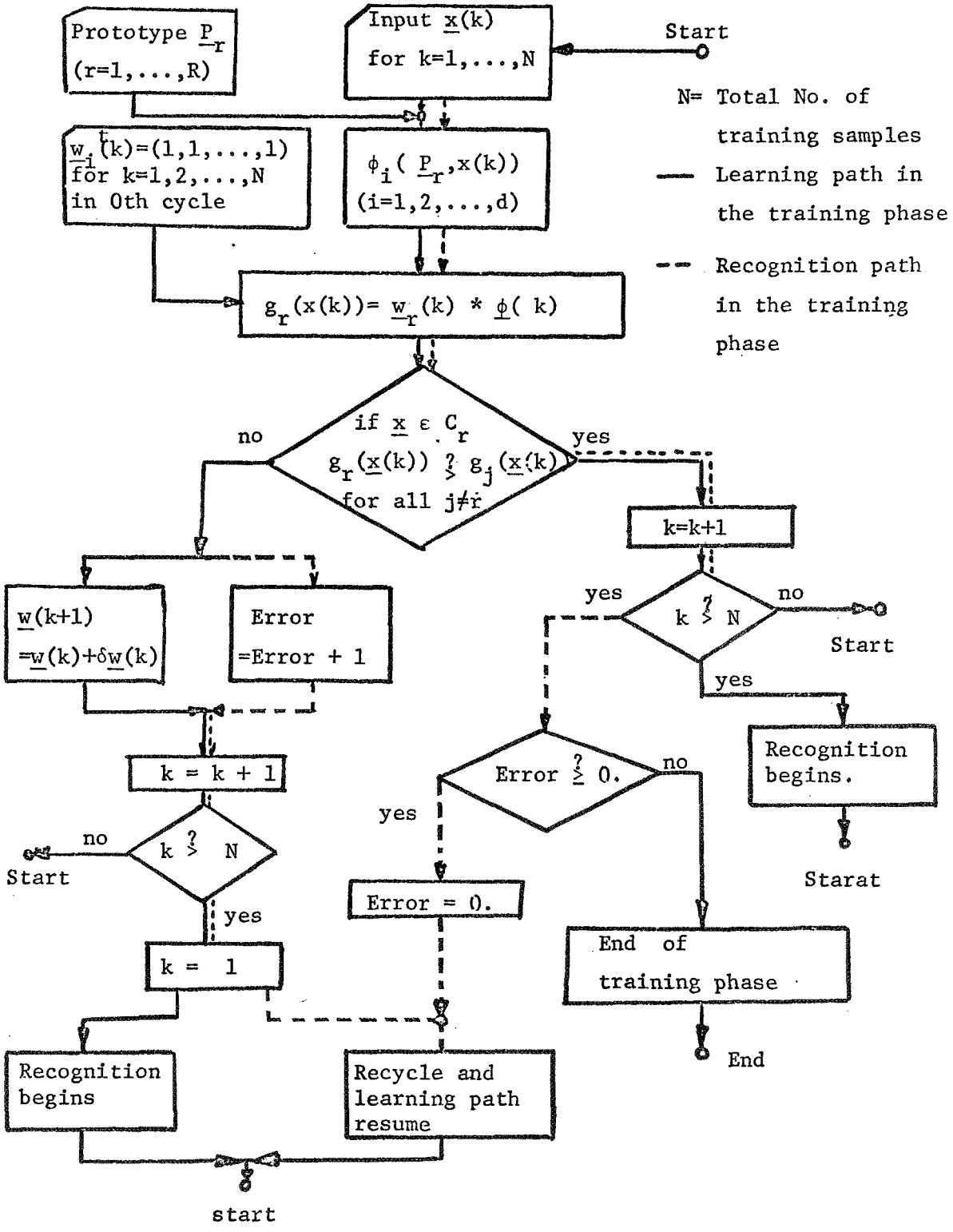
A**	6.0	6.0	6.0	5.0	6.0	5.0	1.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	6.0
C**	3.0	4.0	5.0	4.0	5.0	5.0	5.0	6.0	6.0	6.0	6.0	7.0	6.0	7.0	7.0	.0	.0	.0	1.0	6.0
D**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	7.0	6.0	6.0	6.0	6.0	5.0	4.0	4.0	6.0
G**	3.0	4.0	5.0	5.0	5.0	6.0	6.0	6.0	6.0	7.0	.0	.0	.0	2.0	2.0	4.0	6.0	.0	.0	6.0
H**	6.0	6.0	6.0	6.0	6.0	6.0	1.0	.0	6.0	6.0	6.0	6.0	6.0	6.0	3.0	6.0	.0	.0	.0	7.0
D**	4.0	4.0	5.0	5.0	6.0	6.0	6.0	7.0	7.0	.0	.0	.0	1.0	2.0	2.0	2.0	3.0	2.0	4.0	2.0
P**	6.0	6.0	6.0	6.0	6.0	6.0	6.0	2.0	.0	.0	.0	.0	7.0	6.0	6.0	5.0	4.0	5.0	4.0	6.0
Q**	4.0	5.0	6.0	6.0	6.0	7.0	.0	.0	.0	2.0	2.0	2.0	2.0	3.0	4.0	6.0	6.0	7.0	7.0	6.0
S**	3.0	4.0	4.0	4.0	5.0	7.0	7.0	7.0	7.0	7.0	7.0	6.0	6.0	6.0	5.0	4.0	4.0	3.0	3.0	5.0
5**	6.0	6.0	6.0	7.0	.0	7.0	6.0	6.0	6.0	4.0	4.0	4.0	2.0	2.0	1.0	.0	.0	.0	.0	.0

Appendix C

Digital Computer Program of A Learning Algorithm for An On-line Handwritten Character Recognition

This appendix describes the digital computer program for a learning algorithm for an on-line handwritten character recognition system discussed in Section 3.2 which is a modified of the Zobrak-Sze program. It is written in the MAD language and has been run on IBM 7090. The flow chart and the program listing are given in the following pages.

Flow Chart of the Learning Algorithm for On-line Handwritten Character Recognition



```
          DIMENSION UNKNOW(10*20*50),PATTER(14*20),CONF(14*14),OUT(15),
         1 WEIGHT(14*20),DIFF(14*20),DIM(3),IDCHAR(14),LRS(50),
         2 RESULT(14)
          INTEGER I,II,J,JJ,K,KK,KKK,TR,ERFLAG, DIM,IDCHAR,CONF,LRS,
         1 LRIND,OKFLAG,K1,K2,OUT,VAR,N,D,KI1,KI2,KJ1
          ERFLAG=0
          OKFLAG=0
          READ AND PRINT DATA N,C,D,KI1,KI2,KJ1
          K1=KI1
          READ FORMAT ALPHA, IDCHAR(1)...IDCHAR(N)
          THROUGH ID2, FOR I=1,1,I.G.N
          VECTOR VALUES ALPHA = $10C1*$
ID2       READ FORMAT CHAR, PATTER(I,1)...PATTER( I,D)
          VECTOR VALUES CHAR = $20F1.0*$
          THROUGH ID5, FOR K=1,1,K.G.K1
          THROUGH ID5,FOR I=1,1,I.G.N
ID5       READ FORMAT CHAR, (J=1,1,J.G.D,UNKNOW(I,J,K))
          PRINT FORMAT TC
          VECTOR VALUES TC = $1H1,S12,32HCHARACTERISTICS OF THE IDEAL SET/*$
          THROUGH ID10,FOR I=1,1,I.G.N
ID10      PRINT FORMAT CU, IDCHAR(I),PATTER(I,1)...PATTER(I,D)
          VECTOR VALUES CU = $S5,C1,2H**,20F4.1*$
          THROUGH ID15,FOR K=1,1,K.G.K1
          PRINT FORMAT TS, K
          VECTOR VALUES TS = $//S15,27HCHARACTERISTICS OF SET NO. ,I2/*$
          THROUGH ID15,FOR I=1,1,I.G.N
ID15      PRINT FORMAT CU, IDCHAR(I), (J=1,1,J.G.D,UNKNOW(I,J,K))

          PRINT COMMENT $1$
          VAR=1
R(4)      THROUGH P4, FOR I=1,1,I.G.N
P4        READ FORMAT W5, WEIGHT(I,1)...WEIGHT(I,D)
          VECTOR VALUES W5=$20F4.1*$

          TRANSFER TO S(VAR)
S(1)      CONTINUE
P10       LRIND=0
          K1=KI1
          K2=1
          THROUGH P15, FOR I=1,1,I.G.K1
P15       LRS(I)=I
S(3)      CONTINUE
P(2)      THROUGH P20, FOR I=1,1,I.G.N
          THROUGH P20, FOR J=1,1,J.G.N
P20       CONF(I,J)=0

          KKK=1

C1        WHENEVER KKK.NE.1
          PRINT FORMAT T10,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20
          VECTOR VALUES T10= $1H1///S54,7HWEIGHTS/1HO,S15,20(S1,I2,S2)/1H ,
         1 15H     PATTERN   ,20(S1,1H*,S2)/*$
```

```
          THROUGH C2, FOR I=1,1,I.G.N
C2        PRINT FORMAT W10,IDCHAR(I), WEIGHT(I,1)...WEIGHT(I,D)
          OTHERWISE
          END OF CONDITIONAL
          PRINT FORMAT CC,KKK
          VECTOR VALUES CC=$1HO, 20H TRAINING CYCLE NO. ,I2*$
C11       ERRORS=0.0
          TRIALS=0.0
          JJ=1
          K=LRS(1)

C5        II=1
C10       I=1
          KK=1
          OUT(1)=0
C15       RESULT(I)=0.0
          OUT(I+1)=0
          THROUGH C20, FOR J=1,1,J.G.D
          DIFF(I,J)=.ABS.(PATTER(I,J)-UNKNOW(II,J,K))
          WHENEVER DIFF(I,J).G.4.0, DIFF(I,J)=8.0-DIFF(I,J)
C20       RESULT(I)=RESULT(I)+DIFF(I,J)*WEIGHT(I,J)
          I=I+1
          WHENEVER I.G.N, TRANSFER TO C30
          TRANSFER TO C15

C30       THROUGH C35, FOR I=1,1,I.G.N
          WHENEVER RESULT(I).G.RESULT(II), TRANSFER TO C35
          OUT(KK)=I
          KK=KK+1
C35       CONTINUE

          ALP = 1.0/(KK-2)
          I=1
M10       WHENEVER OUT(I).E.0, TRANSFER TO M27
          WHENEVER OUT(I).E.II, TRANSFER TO M25
          PRINT FORMAT ER, IDCHAR(II),K,IDCHAR(OUT(I))
          VECTOR VALUES ER= $8H ERROR--,C1,I2,12H LOOKS LIKE ,C1*$
          ERFLAG=1
          WHENEVER LRIND.E.0, TRANSFER TO M25
          THROUGH M15,  FOR J=1,1,J.G.D
M15       WEIGHT(OUT(I),J)=WEIGHT(OUT(I),J)+(C*ALP*DIFF(OUT(I),J))

M25       CONF(II,OUT(I))=CONF(II,OUT(I))+1
          I=I+1
          TRANSFER TO M10

M27       WHENEVER ERFLAG.E.0, TRANSFER TO M40
          WHENEVER LRIND.E.0, TRANSFER TO M35
          THROUGH M30, FOR J=1,1,J.G.D
          WEIGHT(II,J)=WEIGHT(II,J)-(C*DIFF(II,J))
M30       WHENEVER WEIGHT(II,J).L.0.0, WEIGHT(II,J)=0.0

M35       ERRORS=ERRORS+1.0
M40       TRIALS=TRIALS+1.0
          ERFLAG=0
          OKFLAG=0
          II=II+1
          WHENEVER II.G.N , TRANSFER TO P25
          TRANSFER TO C10

P25       JJ=JJ+1
```

```
          WHENEVER JJ.G.K1, TRANSFER TO P30
          K=LRS(JJ)
          PRINT COMMENT $0$
          TRANSFER TO C5

P30       ERRATE=ERRORS/TRIALS
          PRINT RESULTS  ERRORS,TRIALS,ERRATE
          WHENEVER LRIND.E.0, TRANSFER TO P32
          LRIND=0
          PRINT FORMAT BB
          VECTOR VALUES BB=$1HO, 29H   RECOGNITION AFTER TRAINING*$
          TRANSFER TO C11
P32       KKK=KKK+1
          LRIND=1
          WHENEVER KKK.G.K2.OR.ERRORS.E.0.0, TRANSFER TO T(VAR)
          TRANSFER TO C1
T(1)      VAR=2
          LRIND=1
          K1=KI2
          K2=KJ1
          THROUGH T11, FOR I=1,1,I.G.K1
T11       LRS(I)=I
          TRANSFER TO P35
T(2)      VAR=3
          TRANSFER TO P10
T(3)      TRANSFER TO P35


P35       PRINT FORMAT TI
          VECTOR VALUES TI = $1H1///S39,51H NUMBER OF TIMES UNKNOWN PATTERN WAS
         1 ECOGNIZED AS ,//S12,105H  A  B  C  D  E  F  G  H  I  J  K  L
         2  M  N  O  P  Q  R  S  T  U  V  W  X  Y  Z  1  2  3  4  5  6
         3 7  8  9/12H   UNKNOWN  ,35(S2,1H*)/12H   PATTERN  ,35(S2,1H*)
         4 //*$
          THROUGH P40,  FOR I=1,1,I.G.N
P40       PRINT FORMAT CO, IDCHAR(I), CONF(I,1)...CONF(I,N)
          VECTOR VALUES CO = $S9,C1,2H**,35I3*$
          PRINT FORMAT T10,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20
          VECTOR VALUES T10= $1H1///S54,7HWEIGHTS/1HO,S15,20(S1,I2,S2)/1H ,
         1 15H     PATTERN   ,20(S1,1H*,S2)/*$
          THROUGH P45,  FOR I=1,1,I.G.N
P45       PRINT FORMAT W10,IDCHAR(I), WEIGHT(I,1)...WEIGHT(I,D)
          VECTOR VALUES W10=$S9,C1,1H*,S3,20F5.1*$
          WHENEVER VAR.NE.3, TRANSFER TO P(VAR)
          TRANSFER TO R(VAR)
R(3)      THROUGH P50, FOR I=1,1,I.G.N
P50       PUNCH FORMAT W5, WEIGHT(I,1)...WEIGHT(I,D)
          END OF PROGRAM
```

Appendix D

Properties of The Matrix $\underline{E}_j$

From (4.9) and (4.14), the matrices $\underline{E}_j$, $\underline{E}_j{}^t\underline{E}_j$, and $(\underline{E}_j{}^t\underline{E}_j)^{-1}$ for all j, (j = 1, 2, . . ., R), of a (R-1) dimensional equilateral simplex with its centroid at the origin have been computed for R = 2 to 10 and are tabulated in the following. The properties (v) and (vi) stated in Section 4.3 have been deduced from this extensive enumeration.

For R = 2 to 10, the values of $\underline{E}_j$, $\underline{F}_R$, $\underline{G}_R$ are tabulated in the following:

(i) For R = 2

$$e_1{}^t = 1.00 \quad , \quad e_2{}^t = -1.00$$

$$E_1 = [2] \quad , \quad E_2 = [-2]$$

$$F_R = E_1{}^t E_1 = E_2{}^t E_2 = 4$$

$$= C_2\,[1] \qquad \text{where } C_2 = 4$$

$$G_R = F_R{}^{-1} = (E_1{}^t E_1)^{-1} = (E_2{}^t E_2)^{-1} = 1/4$$

$$= g_2 \qquad \text{where } g_2 = 1/4$$

(ii) For R = 3

$$\underline{e}_1{}^t = (1.000,\ 0.000)$$

$$\underline{e}_2{}^t = (-0.500,\ 0.866)$$

$$\underline{e}_3{}^t = (-0.500,\ -0.866)$$

$$\underline{E}_1 = \begin{bmatrix} 1.500 & 1.500 \\ -0.866 & 0.866 \end{bmatrix}, \quad \underline{E}_2 = \begin{bmatrix} -1.500 & 0.0 \\ 0.866 & 1.732 \end{bmatrix},$$

$$\underline{E}_3 = \begin{bmatrix} -1.500 & 0.0 \\ -0.866 & -1.732 \end{bmatrix}$$

$$\underline{F}_R = \underline{E}_1^t \underline{E}_1 = \underline{E}_2^t \underline{E}_2 = \underline{E}_3^t E_3$$

$$= C_3 \begin{bmatrix} 1.0 & 0.5 \\ 0.5 & 1.0 \end{bmatrix} \quad \text{where } C_3 = 3$$

$$\underline{G}_R = \underline{F}_R^{-1} = (\underline{E}_1^t \underline{E}_1)^{-1} = (\underline{E}_2^t \underline{E}_2)^{-1} = (\underline{E}_3^t E_3)^{-1}$$

$$= \begin{bmatrix} g_3 & g_3' \\ g_3' & g_3 \end{bmatrix} \quad \text{where } g_3 = 0.44444, \; g_3' = -0.22222$$

(iii) For R = 4

$\underline{e}_1^t = (1.0, 0.0, 0.0)$,

$\underline{e}_2^t = (-0.33333, 0.94281, 0.0)$,

$\underline{e}_3^t = (-0.33333, -0.47140, 0.81650)$,

$\underline{e}_4^t = (-0.33333, -0.47140, -0.81650)$.

$$\underline{E}_1 = \begin{bmatrix} 1.33333 & 1.33333 & 1.33333 \\ -0.94281 & 0.47140 & 0.47140 \\ 0.0 & -0.81650 & 0.81650 \end{bmatrix}$$

$$\underline{E}_2 = \begin{bmatrix} -1.33333 & 0.0 & 0.0 \\ 0.94281 & 1.41421 & 1.41421 \\ 0.0 & -0.81650 & 0.81650 \end{bmatrix}$$

$$\underline{E}_3 = \begin{bmatrix} -1.33333 & 0.0 & 0.0 \\ -0.47140 & -1.41421 & 0.0 \\ 0.81650 & 0.81650 & 1.63299 \end{bmatrix}$$

$$\underline{E}_4^t = \begin{bmatrix} -1.33333 & -0.47140 & -0.81650 \\ 0.0 & -1.41421 & -0.81650 \\ 0.0 & 0.0 & -1.63299 \end{bmatrix}$$

$$\underline{F}_R = \underline{E}_1^t \underline{E}_1 = \underline{E}_2^t \underline{E}_2 = \underline{E}_3^t \underline{E}_3 = \underline{E}_4^t \underline{E}_4$$

$$= C_4 \begin{bmatrix} 1.0 & 0.5 & 0.5 \\ 0.5 & 1.0 & 0.5 \\ 0.5 & 0.5 & 1.0 \end{bmatrix} \qquad \text{where } C_4 = 2.66666$$

$$\underline{G}_R = \underline{F}_R^{-1} = (\underline{E}_1^t \underline{E}_1)^{-1} = (\underline{E}_2^t \underline{E}_2)^{-1} = (\underline{E}_3^t \underline{E}_3)^{-1} = (\underline{E}_4^t \underline{E}_4)^{-1}$$

$$= \begin{bmatrix} g_4 & g_4' & g_4' \\ g_4' & g_4 & g_4' \\ g_4' & g_4' & g_4 \end{bmatrix}$$

where

$$g_4 = 0.56264 \quad , \quad g_4' = -0.18755$$

(iv) For R= 5

$$\underline{e}_1^t = (1.0,\ 0.0,\ 0.0,\ 0.0)\ ,$$

$$\underline{e}_2^t = (-0.25,\ 0.96825,\ 0.0,\ 0.0)\ ,$$

$$\underline{e}_3^t = (-0.25,\ -0.32275,\ 0.91287,\ 0.0)\ ,$$

$$\underline{e}_4^t = (-0.25,\ -0.32275,\ -0.45644,\ 0.79057)\ ,$$

$$\underline{e}_5^t = (-0.25,\ -0.32275,\ -0.45644,\ -0.79057)$$

$$\underline{E}_1 = \begin{bmatrix} 1.25 & 1.25 & 1.25 & 1.25 \\ -0.96825 & 0.32275 & 0.32275 & 0.32275 \\ 0.0 & -0.91287 & 0.45644 & 0.45644 \\ 0.0 & 0.0 & -0.79057 & 0.79057 \end{bmatrix}$$

$$\underline{E}_2 = \begin{bmatrix} -1.25 & 0.0 & 0.0 & 0.0 \\ 0.96828 & 1.29099 & 1.29099 & 1.29099 \\ 0.0 & -0.91287 & 0.45644 & 0.45644 \\ 0.0 & 0.0 & -0.79099 & 0,79057 \end{bmatrix}$$

$$\underline{E}_3 = \begin{bmatrix} -1.25 & 0.0 & 0.0 & 0.0 \\ -0.32275 & -1.29099 & 0.0 & 0.0 \\ 0.91287 & 0.91287 & 1.36931 & 1.36931 \\ 0.0 & 0.0 & -0.79057 & 0.79057 \end{bmatrix}$$

$$\underline{E}_4 = \begin{bmatrix} -1.25 & 0.0 & 0.0 & 0.0 \\ -0.32275 & -1.29099 & 0.0 & 0.0 \\ -0.45644 & -0.45644 & -1.36931 & 0.0 \\ 0.79047 & 0.79057 & 0.79057 & 1.58114 \end{bmatrix}$$

$$\underline{E}_5 = \begin{bmatrix} -1.25 & 0.0 & 0.0 & 0.0 \\ -0.32275 & -1.29099 & 0.0 & 0.0 \\ -0.45644 & -0.45644 & -1.36931 & 0.0 \\ -0.79057 & -0.79057 & -0.79057 & 1.58114 \end{bmatrix}$$

$$\underline{F}_R = \underline{E}_1^t \underline{E}_1 = \underline{E}_2^t \underline{E}_2 = \underline{E}_3^t \underline{E}_3 = \underline{E}_4^t \underline{E}_4 = \underline{E}_5^t \underline{E}_5$$

$$= C_5 \begin{bmatrix} 1.0 & 0.5 & 0.5 & 0.5 \\ 0.5 & 1.0 & 0.5 & 0.5 \\ 0.5 & 0.5 & 1.0 & 0.5 \\ 0.5 & 0.5 & 0.5 & 1.0 \end{bmatrix}$$

where

$C_5 = 2.5$

$$\underline{G}_R = (\underline{E}_1^t \underline{E}_1)^{-1} = (\underline{E}_2^t \underline{E}_2)^{-1} = (\underline{E}_3^t \underline{E}_3)^{-1} = (\underline{E}_4^t \underline{E}_4)^{-1}$$
$$= (\underline{E}_5^t \underline{E}_5)^{-1}$$

$$\begin{bmatrix} g_5 & g_5' & g_5' & g_5' \\ g_5' & g_5 & g_5' & g_5' \\ g_5' & g_5' & g_5 & g_5' \\ g_5' & g_5' & g_5' & g_5 \end{bmatrix}$$

where

$g_5 = 0.64$, $g_5' = -0.16$

(v) For R = 6

$$e_1^t = (1.0,\ 0.0,\ 0.0,\ 0.0,\ 0.0)$$

$$e_2^t = (-0.2,\ 0.97980,\ 0.0,\ 0.0,\ 0.0)$$

$$e_3^t = (-0.2, -0.24495,\ 0.94868,\ 0.0,\ 0.0)$$

$$e_4^t = (-0.2,\ -0.24495,\ -0.31623,\ 0.89443,\ 0.0)$$

$$e_5^t = (-.02,\ -0.24495,\ -0.31623,\ -0.44721,\ 0.77460)$$

$$e_6^t = (-0.2,\ -0.24495,\ -0.31623,\ -0.44721,\ -0.77460)$$

$$E_1 = \begin{bmatrix} 1.200 & 1.200 & 1.200 & 1.200 & 1.200 \\ -0.97980 & 0.24495 & 0.24495 & 0.24495 & 0.24495 \\ 0.0 & -0.94868 & 0.31623 & 0.31623 & 0.31623 \\ 0.0 & 0.0 & -0.89443 & 0.44721 & 0.44721 \\ 0.0 & 0.0 & 0.0 & -0.77460 & 0.77460 \end{bmatrix}$$

$$E_2 = \begin{bmatrix} -1.200 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.97980 & 1.22474 & 1.22474 & 1.22474 & 1.22474 \\ 0.0 & -0.94868 & 0.31623 & 0.31623 & 0.31623 \\ 0.0 & 0.0 & -0.89443 & 0.44721 & 0.44721 \\ 0.0 & 0.0 & 0.0 & -0.77460 & 0.77460 \end{bmatrix}$$

$$\underline{E}_3 = \begin{bmatrix} -1.20 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.24495 & -1.22474 & 0.0 & 0.0 & 0.0 \\ 0.94888 & 0.94888 & 1.26491 & 1.26491 & 1.26491 \\ 0.0 & 0.0 & -0.89443 & 0.44721 & 0.44721 \\ 0.0 & 0.0 & 0.0 & -0.72460 & 0.77460 \end{bmatrix}$$

$$\underline{E}_4 = \begin{bmatrix} -1.200 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.24495 & -1.22474 & 0.0 & 0.0 & 0.0 \\ -0.31623 & -0.31623 & -1.26491 & 0.0 & 0.0 \\ 0.89443 & 0.89443 & 0.89443 & 1.34164 & 1.34164 \\ 0.0 & 0.0 & 0.0 & -0.7746 & 0.7746 \end{bmatrix}$$

$$\underline{E}_5 = \begin{bmatrix} -1.200 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.24495 & -1.22474 & 0.0 & 0.0 & 0.0 \\ -0.31623 & -0.31623 & -1.26491 & 0.0 & 0.0 \\ -0.44721 & -0.44721 & -0.44721 & -1.34164 & 0.0 \\ 0.77460 & 0.77460 & 0.7746 & 0.77460 & 1.54919 \end{bmatrix}$$

$$\underline{E}_6 = \begin{bmatrix} -1.2 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.24495 & -1.22474 & 0.0 & 0.0 & 0.0 \\ -0.31623 & -0.31623 & -1.26491 & 0.0 & 0.0 \\ -0.44721 & -0.44721 & -0.44721 & -1.34164 & 0.0 \\ -0.77460 & -0.77460 & -0.77460 & -0.77460 & -1.54919 \end{bmatrix}$$

$$\underline{F}_R = (\underline{E}_1{}^t\underline{E}_1) = (\underline{E}_2{}^t\underline{E}_2) = (\underline{E}_3{}^t\underline{E}_3) = (\underline{E}_4{}^t\underline{E}_4)$$
$$= (\underline{E}_5{}^t\underline{E}_5) = (\underline{E}_6{}^t\underline{E}_6)$$

$$= C_6 \begin{bmatrix} 1.0 & 0.5 & 0.5 & 0.5 & 0.5 \\ 0.5 & 1.0 & 0.5 & 0.5 & 0.5 \\ 0.5 & 0.5 & 1.0 & 0.5 & 0.5 \\ 0.5 & 0.5 & 0.5 & 1.0 & 0.5 \\ 0.5 & 0.5 & 0.5 & 0.5 & 1.0 \end{bmatrix}$$

where

$$C_6 = 2.39999$$

$$\underline{G}_R = (\underline{E}_1{}^t\underline{E}_1)^{-1} = (\underline{E}_2{}^t\underline{E}_2)^{-1} = (\underline{E}_3{}^t\underline{E}_3)^{-1} = (\underline{E}_4{}^t\underline{E}_4)^{-1}$$
$$= (\underline{E}_5{}^t\underline{E}_5)^{-1} = (\underline{E}_6{}^t\underline{E}_6)^{-1}$$

$$= \begin{bmatrix} g_6 & g_6' & g_6' & g_6' & g_6' \\ g_6' & g_6 & g_6' & g_6' & g_6' \\ g_6' & g_6' & g_6 & g_6' & g_6' \\ g_6' & g_6' & g_6' & g_6 & g_6' \\ g_6' & g_6' & g_6' & g_6' & g_6 \end{bmatrix}$$

where

$$g_6 = 0.67444 \quad , \quad g_6' = -0.13888 .$$

(vi) For R = 7

$$\underline{e}_1^t = (1.0,\ 0.0,\ 0.0,\ 0.0,\ 0.0\ 0.0)$$

$$\underline{e}_2^t = (-0.16667,\ 0.98601,\ 0.0,\ 0.0,\ 0.0,\ 0.0)$$

$$\underline{e}_3^t = (-0.16667,-0.19720,\ 0.96609,\ 0.0,\ 0.0,\ 0.0)$$

$$\underline{e}_4^t = (0.16667,\ -0.19760,-0.24152,\ 0.93541,\ 0.0,\ 0.0)$$

$$\underline{e}_5^t = (-0.16667,-0.19720,-0.24152,-0.31180,0.88192,0.0)$$

$$\underline{e}_6^t = (-0.16667,-0.19720,-0.24152,-0.3118,-0.44096,0.76376)$$

$$\underline{e}_7^t = (-0.16667,-0.19720,-0.24152,-0.3118,-0.44096,-0.76376)$$

$$\underline{E}_1 = \begin{bmatrix} 1.16667 & 1.16667 & 1.16667 & 1.16667 & 1.16667 & 1.16667 \\ -0.98601 & 0.19720 & 0.19720 & 0.1972 & 0.19720 & 0.19720 \\ 0.0 & -0.96609 & 0.24152 & 0.24152 & 0.24152 & 0.24152 \\ 0.0 & 0.0 & -0.93541 & 0.31180 & 0.31180 & 0.31180 \\ 0.0 & 0.0 & 0.0 & -0.88192 & 0.44096 & 0.44096 \\ 0.0 & 0.0 & 0.0 & 0.0 & -0.76376 & 0.76376 \end{bmatrix}$$

$$\underline{E}_2 = \begin{bmatrix} -1.16667 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.98601 & 1.18322 & 1.18322 & 1.18322 & 1.18322 & 1.18322 \\ 0.0 & -0.96609 & 0.24152 & 0.24152 & 0.24152 & 0.24152 \\ 0.0 & 0.0 & -0.93541 & 0.31180 & 0.31180 & 0.31180 \\ 0.0 & 0.0 & 0.0 & -0.88192 & 0.44096 & 0.44096 \\ 0.0 & 0.0 & 0.0 & 0.0 & -0.76376 & 0.76376 \end{bmatrix}$$

$$\underline{E}_3 = \begin{bmatrix} -1.16667 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.19720 & -1.18322 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.96609 & 0.96609 & 1.20761 & 1.20761 & 1.20761 & 1.20761 \\ 0.0 & 0.0 & -0.93541 & 0.31180 & 0.31180 & 0.31180 \\ 0.0 & 0.0 & 0.0 & -0.88192 & 0.44096 & 0.44096 \\ 0.0 & 0.0 & 0.0 & 0.0 & -0.76376 & 0.76376 \end{bmatrix}$$

$$\underline{E}_4 = \begin{bmatrix} -1.16667 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.19720 & -1.18322 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.24152 & -0.24152 & -1.20760 & 0.0 & 0.0 & 0.0 \\ 0.93541 & 0.93541 & 0.93541 & 1.24722 & 1.24722 & 1.24722 \\ 0.0 & 0.0 & 0.0 & -0.88192 & 0.44096 & 0.44096 \\ 0.0 & 0.0 & 0.0 & 0.0 & -0.76376 & 0.76376 \end{bmatrix}$$

$$\underline{E}_5 = \begin{bmatrix} -1.16667 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.19720 & -1.18322 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.24152 & -0.24152 & -1.20761 & 0.0 & 0.0 & 0.0 \\ -0.31180 & -0.31180 & -0.31180 & -1.24722 & 0.0 & 0.0 \\ 0.88192 & 0.88192 & 0.88192 & 0.88192 & 1.32288 & 1.32288 \\ 0.0 & 0.0 & 0.0 & 0.0 & -0.76376 & 0.76376 \end{bmatrix}$$

$$\underline{E}_6 = \begin{bmatrix} -1.16667 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.19720 & -1.18322 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.24152 & -0.24152 & -1.20761 & 0.0 & 0.0 & 0.0 \\ -0.31180 & -0.31180 & -0.31180 & -1.24722 & 0.0 & 0.0 \\ -0.44096 & -0.44096 & -0.44096 & 0.44096 & -1.32288 & 0.0 \\ 0.76376 & 0.76376 & 0.76376 & 0.76376 & 0.76376 & 1.52752 \end{bmatrix}$$

$$\underline{E}_7 = \begin{bmatrix} -1.16667 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.19720 & -1.18322 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.24152 & -0.24152 & -1.20761 & 0.0 & 0.0 & 0.0 \\ -0.31180 & -0.31180 & -0.31180 & -1.24722 & 0.0 & 0.0 \\ -0.44096 & -0.44096 & -0.44096 & -0.44096 & -1.32288 & 0.0 \\ -0.76376 & -0.76376 & -0.76376 & -0.76376 & -0.76376 & -1.52752 \end{bmatrix}$$

$$\underline{F}_R = \underline{E}_1{}^t\underline{E}_1 = \underline{E}_2{}^t\underline{E}_2 = \underline{E}_3{}^t\underline{E}_3 = \underline{E}_4{}^t\underline{E}_4 = \underline{E}_5{}^t\underline{E}_5 = \underline{E}_6{}^t\underline{E}_6 = \underline{E}_7{}^t\underline{E}$$

$$= C_7 \begin{bmatrix} 1 & 1/2 & 1/2 & 1/2 & 1/2 & 1/2 \\ 1/2 & 1 & 1/2 & 1/2 & 1/2 & 1/2 \\ 1/2 & 1/2 & 1 & 1/2 & 1/2 & 1/2 \\ 1/2 & 1/2 & 1/2 & 1 & 1/2 & 1/2 \\ 1/2 & 1/2 & 1/2 & 1/2 & 1 & 1/2 \\ 1/2 & 1/2 & 1/2 & 1/2 & 1/2 & 1 \end{bmatrix}$$

where

$C_7 = 2.33333$

$$\underline{G}_R = (\underline{E}_1{}^t\underline{E}_1)^{-1} = (\underline{E}_2{}^t\underline{E}_2)^{-1} = (\underline{E}_3{}^t\underline{E}_3)^{-1} = (\underline{E}_4{}^t\underline{E}_4)^{-1} = (\underline{E}_5{}^t\underline{E}_5)^{-1}$$

$$= (\underline{E}_6{}^t\underline{E}_6)^{-1} = (\underline{E}_7{}^t\underline{E}_7)^{-1}$$

$$= \begin{bmatrix} g_7 & g_7' & g_7' & g_7' & g_7' & g_7' \\ g_7' & g_7 & g_7' & g_7' & g_7' & g_7' \\ g_7' & g_7' & g_7 & g_7' & g_7' & g_7' \\ g_7' & g_7' & g_7' & g_7 & g_7' & g_7' \\ g_7' & g_7' & g_7' & g_7' & g_7 & g_7' \\ g_7' & g_7' & g_7' & g_7' & g_7' & g_7 \end{bmatrix}$$

where

$g_7 = 0.73449$, $g_7' = -0.12239$

(vii) For R=8

$\underline{e}_1{}^t = (1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0)$

$\underline{e}_2{}^t = (-0.14286, 0.98974, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0)$

$\underline{e}_3{}^t = (-0.14286, -0.16496, 0.97590, 0.0, 0.0, 0.0, 0.0, 0.0)$

$\underline{e}_4{}^t = (-0.14286, -0.16496, -0.19518, 0.95618, 0.0, 0.0, 0.0)$

$\underline{e}_5{}^t = (0.14286, -0.16496, -0.19518, -0.23905, 0.92582, 0.0, 0.0)$

$\underline{e}_6{}^t = (-0.14286, -0.16496, -0.19518, -0.23905, -0.30861, 0.87287, 0.0)$

$\underline{e}_7{}^t = (-0.14286, -0.16496, -0.19518, -0.23905, -0.30861, -0.43644, 0.75593)$

$\underline{e}_8{}^t = (-0.14286, -0.16496, -0.19518, -0.23905, -0.30861, -0.43644, -0.75593)$

$$\underline{E}_1 = \begin{bmatrix} 1.14286 & 1.14286 & 1.14286 & 1.14286 & 1.14286 & 1.14286 & 1.14286 \\ -0.90974 & 0.16496 & 0.16496 & 0.16496 & 0.16496 & 0.16496 & 0.16496 \\ 0.0 & -0.97590 & 0.19518 & 0.19518 & 0.19518 & 0.19518 & 0.19518 \\ 0.0 & 0.0 & -0.96518 & 0.23905 & 0.23905 & 0.23905 & 0.23905 \\ 0.0 & 0.0 & 0.0 & -0.92582 & 0.30861 & 0.30861 & 0.30861 \\ 0.0 & 0.0 & 0.0 & 0.0 & -0.87287 & 0.43644 & 0.43644 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.75593 & 0.75593 \end{bmatrix}$$

$$\underline{E}_2 = \begin{bmatrix} -1.14286 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.98974 & 1.1547 & 1.1547 & 1.1547 & 1.1547 & 1.1547 & 1.1547 \\ 0.0 & -0.97590 & 0.19518 & 0.19518 & 0.19518 & 0.19518 & 1.19518 \\ 0.0 & 0.0 & -0.95618 & 0.23905 & 0.23905 & 0.23905 & 0.23905 \\ 0.0 & 0.0 & 0.0 & -0.92582 & 0.30861 & 0.30861 & 0.30861 \\ 0.0 & 0.0 & 0.0 & 0.0 & -0.87287 & 0.43644 & 0.43644 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.75593 & 0.75593 \end{bmatrix}$$

$$\underline{E}_3 = \begin{bmatrix} -1.14286 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.16496 & -1.15470 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.97590 & 0.97590 & 1.17108 & 1.17108 & 1.17108 & 1.17108 & 1.17108 \\ 0.0 & 0.0 & -0.95618 & 0.23905 & 0.23905 & 0.23905 & 0.23905 \\ 0.0 & 0.0 & 0.0 & -0.92582 & 0.30861 & 0.30861 & 0.30861 \\ 0.0 & 0.0 & 0.0 & 0.0 & -0.87287 & 0.43644 & 0.43644 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.75593 & 0.75593 \end{bmatrix}$$

$$\underline{E}_4 = \begin{bmatrix} -1.14286 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.16496 & -1.15470 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.19518 & -0.19518 & -1.17108 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.95618 & 0.95618 & 0.95618 & 1.19523 & 1.19523 & 1.19523 & 1.19523 \\ 0.0 & 0.0 & 0.0 & -0.92582 & 0.30861 & 0.30861 & 0.30861 \\ 0.0 & 0.0 & 0.0 & 0.0 & -0.87287 & 0.43644 & 0.43644 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.75593 & 0.75593 \end{bmatrix}$$

$$\underline{E}_5 = \begin{bmatrix} -1.14286 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.16496 & -1.15470 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.19518 & -0.19518 & -1.17108 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.23905 & -0.23905 & -0.23905 & -1.19523 & 0.0 & 0.0 & 0.0 \\ -0.92582 & 0.92582 & 0.92582 & 0.92582 & 1.23443 & 1.23443 & 1.23433 \\ 0.0 & 0.0 & 0.0 & 0.0 & -0.87287 & 0.43644 & 0.43644 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.75593 & 0.75593 \end{bmatrix}$$

$$\underline{E}_6 = \begin{bmatrix} -1.14286 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.16496 & -1.15470 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.19518 & -0.19518 & -1.17108 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.23905 & -0.23905 & -0.23905 & -1.19523 & 0.0 & 0.0 & 0.0 \\ -0.30861 & -0.30861 & -0.30861 & -0.30861 & -1.23433 & 0.0 & 0.0 \\ 0.87287 & 0.87287 & 0.87287 & 0.87287 & 0.87287 & 1.30931 & 1.30931 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.75593 & 0.75593 \end{bmatrix}$$

$$\underline{E}_7 = \begin{bmatrix} -1.14286 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.16496 & -1.15470 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.19518 & -0.19518 & -1.17108 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.23905 & -0.23905 & -0.23905 & -1.19523 & 0.0 & 0.0 & 0.0 \\ -0.30861 & -0.30861 & -0.30861 & -0.30861 & -1.23443 & 0.0 & 0.0 \\ -0.43644 & -0.43644 & -0.43644 & -0.43644 & -0.43644 & -1.30931 & 0.0 \\ 0.75593 & 0.75593 & 0.75593 & 0.75593 & 0.75593 & 0.75593 & 1.51186 \end{bmatrix}$$

$$\underline{E}_8 = \begin{bmatrix} -1.14286 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.16496 & -1.15470 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.19518 & -0.19518 & -1.17108 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.23905 & -0.23905 & -0.23905 & -1.19523 & 0.0 & 0.0 & 0.0 \\ -0.30861 & -0.30861 & -0.30861 & -0.30861 & -1.23443 & 0.0 & 0.0 \\ -0.43644 & -0.43644 & -0.43644 & -0.43644 & -0.43644 & -1.30931 & 0.0 \\ -0.75593 & -0.75593 & -0.75593 & -0.75593 & -0.75593 & -0.75593 & -1.51186 \end{bmatrix}$$

$$\underline{F}_R = \underline{E}_1{}^t\underline{E}_1 = \underline{E}_2{}^t\underline{E}_2 = \underline{E}_3{}^t\underline{E}_3 = \underline{E}_4{}^t\underline{E}_4 = \underline{E}_5{}^t\underline{E}_5 = \underline{E}_6{}^t\underline{E}_6 = \underline{E}_7{}^t\underline{E}_7 = \underline{E}_8{}^t\underline{E}_8$$

$$= C_8 \begin{bmatrix} 1.0 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 \\ 0.5 & 1.0 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 \\ 0.5 & 0.5 & 1.0 & 0.5 & 0.5 & 0.5 & 0.5 \\ 0.5 & 0.5 & 0.5 & 1.0 & 0.5 & 0.5 & 0.5 \\ 0.5 & 0.5 & 0.5 & 0.5 & 1.0 & 0.5 & 0.5 \\ 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 1.0 & 0.5 \\ 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 1.0 \end{bmatrix}$$

where

$C_8 = 2.28751$

$\underline{G}_R = (\underline{E}_1{}^t\underline{E}_1)^{-1} = (\underline{E}_2{}^t\underline{E}_2)^{-1} = \underline{E}_3{}^t\underline{E}_3)^{-1} = (\underline{E}_4{}^t\underline{E}_4)^{-1} = (\underline{E}_5{}^t\underline{E}_5)^{-1}$
$= (\underline{E}_6{}^t\underline{E}_6)^{-1} = (\underline{E}_7{}^t\underline{E}_7)^{-1} = (\underline{E}_8{}^t\underline{E}_8)^{-1}$

$$= \begin{bmatrix} g_8 & g_8' & g_8' & g_8' & g_8' & g_8' & g_8' \\ g_8' & g_8 & g_8' & g_8' & g_8' & g_8' & g_8' \\ g_8' & g_8' & g_8 & g_8' & g_8' & g_8' & g_8' \\ g_8' & g_8' & g_8' & g_8 & g_8' & g_8' & g_8' \\ g_8' & g_8' & g_8' & g_8' & g_8 & g_8' & g_8' \\ g_8' & g_8' & g_8' & g_8' & g_8' & g_8 & g_8' \\ g_8' & g_8' & g_8' & g_8' & g_8' & g_8' & g_8 \end{bmatrix}$$

where

$g_8 = 0.76553$ and $g_8' = -0.10936$

(viii) For R = 9

$\underline{e}_1^t$ = (1.00, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0)

$\underline{e}_2^t$ = (-0.12500, 0.99216, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0)

$\underline{e}_3^t$ = (-0.12500, -0.14174, 0.98198, 0.0, 0.0, 0.0, 0.0, 0.0)

$\underline{e}_4^t$ = (-0.12500, -0.14174, -0.16365, 0.96825, 0.0, 0.0, 0.0, 0.0)

$\underline{e}_5^t$ = (-0.12500, -0.14174, -0.16366, -0.19365, 0.94868, 0.0, 0.0, 0.0)

$\underline{e}_6^t$ = (-0.12500, -0.14174, -0.16366, -0.19365, -0.23717, 0.91856, 0.0,0.0)

$\underline{e}_7^t$=(-0,12500,-0.14174,-0.16366,-0.19365,-0.23717,-0.30619,0.86603, 0.0)

$\underline{e}_8^t$=(-0.12500,-0.14174,-0.16366,-0.19365,-0.23717,-0.30619,-0.43301,0.75000)

and

$\underline{e}_9^t$=(-0.12500,-0.14174,-0.16366,-0.19365,-0.23717,-0.30619,-0.43301,-0.75000)

$$\underline{E}_1 = \begin{bmatrix} 1.12500 & 1.12500 & 1.12500 & 1.12500 & 1.12500 & 1.12500 & 1.12500 & 1.12500 \\ -0.99216 & 0.14174 & 0.14174 & 0.14174 & 0.14174 & 0.14174 & 0.14174 & 0.14174 \\ 0.0 & -0.98198 & 0.16366 & 0.16366 & 0.16366 & 0.16366 & 0.16366 & 0.16366 \\ 0.0 & 0.0 & -0.96825 & 0.19365 & 0.19365 & 0.19365 & 0.19365 & 0.19365 \\ 0.0 & 0.0 & 0.0 & -0.94868 & 0.23717 & 0.23717 & 0.23717 & 0.23717 \\ 0.0 & 0.0 & 0.0 & 0.0 & -0.91856 & 0.30619 & 0.30619 & 0.30619 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.86603 & 0.43301 & 0.43301 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.75000 & 0.75000 \end{bmatrix}$$

$$\underline{E}_2 = \begin{bmatrix} -1.12500 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.99216 & 1.13389 & 1.13389 & 1.13389 & 1.13389 & 1.13389 & 1.13389 & 1.13389 \\ 0.0 & -0.98198 & 0.16366 & 0.16366 & 0.16366 & 0.16366 & 0.16366 & 0.16366 \\ 0.0 & 0.0 & -0.96825 & 0.19365 & 0.19365 & 0.19365 & 0.19365 & 0.19365 \\ 0.0 & 0.0 & 0.0 & -0.94868 & 0.23717 & 0.23717 & 0.23717 & 0.23717 \\ 0.0 & 0.0 & 0.0 & 0.0 & -0.91856 & 0.30619 & 0.30619 & 0.30619 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.86603 & 0.43301 & 0.43301 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.7500 & 0.7500 \end{bmatrix}$$

$$\underline{E}_3 = \begin{bmatrix} -0.12500 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.14174 & -1.13389 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.98198 & 0.98198 & 1.14564 & 1.14564 & 1.14564 & 1.14564 & 1.14564 & 1.14564 \\ 0.0 & 0.0 & -0.96825 & 0.19365 & 0.19365 & 0.19365 & 0.19365 & 0.19365 \\ 0.0 & 0.0 & 0.0 & -0.94868 & 0.23717 & 0.23717 & 0.23717 & 0.23717 \\ 0.0 & 0.0 & 0.0 & 0.0 & -0.91856 & 0.30619 & 0.30619 & 0.30619 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.86603 & 0.43301 & 0.43301 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.75000 & 0.75000 \end{bmatrix}$$

$$\underline{E}_4 = \begin{bmatrix} -1.12500 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.14174 & -1.13389 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.16366 & -0.16366 & -1.14564 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.96825 & 0.96825 & 0.96825 & 1.16189 & 1.16189 & 1.16189 & 1.16189 & 1.16189 \\ 0.0 & 0.0 & 0.0 & -0.94868 & 0.23717 & 0.23717 & 0.23717 & 0.23717 \\ 0.0 & 0.0 & 0.0 & 0.0 & -0.91856 & 0.30619 & 0.30619 & 0.30619 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.86603 & 0.43301 & 0.43301 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.75000 & 0.75000 \end{bmatrix}$$

$$\underline{E}_5 = \begin{bmatrix} -1.12500 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.14174 & -1.13389 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.16366 & -0.16366 & -1.14564 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.19365 & -0.19365 & -0.19365 & -1.16189 & 0.0 & 0.0 & 0.0 & 0.0 \\ 0.94868 & 0.94868 & 0.94868 & 0.94868 & 1.18585 & 1.18585 & 1.18585 & 1.18585 \\ 0.0 & 0.0 & 0.0 & 0.0 & -0.91850 & 0.30619 & 0.30619 & 0.30619 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.86603 & 0.43301 & 0.43301 \\ 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.75000 & 0.75000 \end{bmatrix}$$

$$\underline{E}_6 = \begin{bmatrix}
-1.12500 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.14174 & -1.13389 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.16366 & -0.16366 & -1.14564 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.19363 & -0.19363 & -0.19363 & -1.16189 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.23717 & -0.23717 & -0.23717 & -0.23717 & -1.18585 & 0.0 & 0.0 & 0.0 \\
0.91856 & 0.91856 & 0.91856 & 0.91856 & 0.91856 & 1.22474 & 1.22474 & 1.22474 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.86603 & 0.43301 & 0.43301 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.75000 & 0.75000
\end{bmatrix}$$

$$\underline{E}_7 = \begin{bmatrix}
-1.12500 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.14174 & -1.13389 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.16366 & -0.16366 & -1.14564 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.19365 & -0.19365 & -0.19365 & -1.16189 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.23717 & -0.23717 & -0.23717 & -0.23717 & -1.18585 & 0.0 & 0.0 & 0.0 \\
-0.30619 & -0.30619 & -0.30619 & -0.30619 & -0.30619 & -1.22475 & 0.0 & 0.0 \\
0.86603 & 0.86603 & 0.86603 & 0.86603 & 0.86603 & 0.86603 & 1.29904 & 1.29904 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.75 & 0.75
\end{bmatrix}$$

$$\underline{E}_8 = \begin{bmatrix} -1.12500 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.14174 & -1.13389 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.16306 & -0.16360 & -1.14564 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.19365 & -0.19365 & -0.19365 & -1.16189 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.23717 & -0.23717 & -0.23717 & -0.23717 & -1.18585 & 0.0 & 0.0 & 0.0 \\ -0.30619 & -0.30619 & -0.30619 & -0.30619 & -0.30619 & -1.22475 & 0.0 & 0.0 \\ -0.43301 & -0.43301 & -0.43301 & -0.43301 & -0.43301 & -0.43301 & 1.29904 & 0.0 \\ 0.75 & 0.75 & 0.75 & 0.75 & 0.75 & 0.75 & 0.75 & 1.500 \end{bmatrix}$$

$$\underline{E}_9 = \begin{bmatrix} -1.12500 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.14174 & -1.13490 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.16366 & -0.16366 & -1.14564 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.19365 & -0.19365 & -0.19365 & -1.16190 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.23717 & -0.23717 & -0.23717 & -0.23717 & -1.18585 & 0.0 & 0.0 & 0.0 \\ -0.30619 & -0.30619 & -0.30619 & -0.30619 & -0.30619 & -1.22475 & 0.0 & 0.0 \\ -0.43301 & -0.43301 & -0.43301 & -0.43301 & -0.43301 & -0.43301 & -1.29904 & 0.0 \\ -0.75 & -0.75 & -0.75 & -0.75 & -0.75 & -0.75 & -0.75 & -1.5 \end{bmatrix}$$

$$\underline{F}_R = \underline{E}_1{}^t\underline{E}_1 = \underline{E}_2{}^t\underline{E}_2 = \underline{E}_3{}^t\underline{E}_3 = \underline{E}_4{}^t\underline{E}_4 = \underline{E}_5{}^t\underline{E}_5 = \underline{E}_6{}^t\underline{E}_6 = \underline{E}_7{}^t\underline{E}_7 = \underline{E}_8{}^t\underline{E}_8 = \underline{E}_9{}^t\underline{E}_9$$

$$F_R = C_9 \begin{bmatrix} 1.0 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 \\ 0.5 & 1.0 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 \\ 0.5 & 0.5 & 1.0 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 \\ 0.5 & 0.5 & 0.5 & 1.0 & 0.5 & 0.5 & 0.5 & 0.5 \\ 0.5 & 0.5 & 0.5 & 0.5 & 1.0 & 0.5 & 0.5 & 0.5 \\ 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 1.0 & 0.5 & 0.5 \\ 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 1.0 & 0.5 \\ 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 1.0 \end{bmatrix}$$

where $C_9 = 2.25$

$$\underline{G}_R = (\underline{E}_1{}^t\underline{E}_1)^{-1} = (\underline{E}_2{}^t\underline{E}_2)^{-1} = (\underline{E}_3{}^t\underline{E}_3)^{-1} = (\underline{E}_4{}^t\underline{E}_4)^{-1} = (\underline{E}_5{}^t\underline{E}_5)^{-1} = (\underline{E}_6{}^t\underline{E}_6)^{-1}$$
$$= (\underline{E}_7{}^t\underline{E}_7)^{-1} = (\underline{E}_8{}^t\underline{E}_8)^{-1} = (\underline{E}_9{}^t\underline{E}_9)^{-1}$$

$$= \begin{bmatrix} g_9 & g_9' & g_9' & g_9' & g_9' & g_9' & g_9' & g_9' \\ g_9' & g_9 & g_9' & g_9' & g_9' & g_9' & g_9' & g_9' \\ g_9' & g_9' & g_9 & g_9' & g_9' & g_9' & g_9' & g_9' \\ g_9' & g_9' & g_9' & g_9 & g_9' & g_9' & g_9' & g_9' \\ g_9' & g_9' & g_9' & g_9' & g_9 & g_9' & g_9' & g_9' \\ g_9' & g_9' & g_9' & g_9' & g_9' & g_9 & g_9' & g_9' \\ g_9' & g_9' & g_9' & g_9' & g_9' & g_9' & g_9 & g_9' \\ g_9' & g_9' & g_9' & g_9' & g_9' & g_9' & g_9' & g_9 \end{bmatrix}$$

where $g_9 = 0.79012$ and $g_3' = -0.09877$

(ix) For R = 10

$\underline{e}_1^t$ = (1.00, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0)

$\underline{e}_2^t$ = (-0.11111, 0.99381, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0)

$\underline{e}_3^t$ = (-0.11111, -0.12423, 0.98601, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0)

$\underline{e}_4^t$ = (-0.11111, -0.12423, -0.14086, 0.97590, 0.0 0.0, 0.0, 0.0, 0.0)

$\underline{e}_5^t$ = (-0.11111, -0.12423, -0.14086, -0.16265, 0.96225, 0.0, 0.0, 0.0, 0.0)

$\underline{e}_6^t$ = (-0.11111, -0.12423, -0.14086, -0.16265, -0.19245, 0.94281, 0.0,0.0,0.0)

$\underline{e}_7^t$ = (-0.11111,-0.12423,-0.14086,-0.16265,-0.19245,-0.23570,0.91287,0.0, 0.0)

$\underline{e}_8^t$ = (-0.11111,-0.12423,-0.14086,-0.16265,-0.19245,-0.23570,-0.30429,0.86066,0.0)

$\underline{e}_9^t$ = (-0.11111,-0.12423,-0.14086,-0.16265,-0.19245,-0.23570,-0.30429,-0.43033,0.74536)

and

$\underline{e}_{10}^t$ =(-0.11111,-0.12423,-0.14086,-0.16265,-0.19245,-0.23570,-0.30429,-0.43033,-0.74536)

$$\underline{E}_1 = \begin{bmatrix}
1.11111 & 1.11111 & 1.11111 & 1.11111 & 1.11111 & 1.11111 & 1.11111 & 1.11111 & 1.11111 \\
-0.99381 & 0.12423 & 0.12423 & 0.12423 & 0.12423 & 0.12423 & 0.12423 & 0.12423 & 0.12423 \\
0.0 & -0.98601 & 0.14086 & 0.14086 & 0.14086 & 0.14086 & 0.14086 & 0.14086 & 0.14086 \\
0.0 & 0.0 & -0.97590 & 0.16265 & 0.16265 & 0.16265 & 0.16265 & 0.16265 & 0.16265 \\
0.0 & 0.0 & 0.0 & -0.96225 & 0.19245 & 0.19245 & 0.19245 & 0.19245 & 0.19245 \\
0.0 & 0.0 & 0.0 & 0.0 & -0.94281 & 0.23570 & 0.23570 & 0.23570 & 0.23570 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.91287 & 0.30429 & 0.30429 & 0.30429 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.86066 & 0.43033 & 0.43033 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.74536 & 0.74536
\end{bmatrix}$$

$$\underline{E}_2 = \begin{bmatrix}
-1.11111 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.99381 & 1.11803 & 1.11803 & 1.11803 & 1.11803 & 1.11803 & 1.11803 & 1.11803 & 1.11803 \\
0.0 & -0.98601 & 0.14086 & 0.14086 & 0.14086 & 0.14086 & 0.14086 & 0.14086 & 0.14086 \\
0.0 & 0.0 & -0.97590 & 0.16265 & 0.16265 & 0.16265 & 0.16265 & 0.16265 & 0.16265 \\
0.0 & 0.0 & 0.0 & -0.96225 & 0.19245 & 0.19245 & 0.19245 & 0.19245 & 0.19245 \\
0.0 & 0.0 & 0.0 & 0.0 & -0.94281 & 0.23570 & 0.23570 & 0.23570 & 0.23570 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.91287 & 0.30429 & 0.30429 & 0.30429 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.86066 & 0.43033 & 0.43033 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.74536 & 0.74536
\end{bmatrix}$$

$$\underline{E}_3 = \begin{bmatrix}
-1.11111 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.12423 & -1.11803 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.98601 & 0.98601 & 1.12687 & 1.12687 & 1.12687 & 1.12687 & 1.12687 & 1.12687 & 1.12687 \\
0.0 & 0.0 & -0.97590 & 0.16265 & 0.16265 & 0.16265 & 0.16265 & 0.16265 & 0.16265 \\
0.0 & 0.0 & 0.0 & -0.96225 & 0.19245 & 0.19245 & 0.19245 & 0.19245 & 0.19245 \\
0.0 & 0.0 & 0.0 & 0.0 & -0.94281 & 0.23570 & 0.23570 & 0.23570 & 0.23570 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.91287 & 0.30429 & 0.30429 & 0.30429 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.86066 & 0.43033 & 0.43033 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.74536 & 0.74536
\end{bmatrix}$$

$$\underline{E}_4 = \begin{bmatrix}
-1.11111 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.12423 & -1.11803 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.14086 & -0.14086 & -1.12687 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.97590 & 0.97590 & 0.97590 & 1.13855 & 1.13855 & 1.13855 & 1.13855 & 1.13855 & 1.13855 \\
0.0 & 0.0 & 0.0 & -0.96225 & 0.19245 & 0.19245 & 0.19245 & 0.19245 & 0.19245 \\
0.0 & 0.0 & 0.0 & 0.0 & -0.94281 & 0.23570 & 0.23570 & 0.23570 & 0.23570 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.91287 & 0.30429 & 0.30429 & 0.30429 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.86066 & 0.43033 & 0.43033 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.74536 & 0.74536
\end{bmatrix}$$

$$\underline{E}_5 = \begin{bmatrix}
-1.11111 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.12423 & -1.1180 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.14086 & -0.14086 & -1.12687 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.16265 & -0.16265 & -0.16265 & -1.13855 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.96225 & 0.96225 & 0.96225 & 0.96225 & 1.15470 & 1.15470 & 1.15470 & 1.15470 & 1.15470 \\
0.0 & 0.0 & 0.0 & 0.0 & -0.94281 & 0.23570 & 0.23570 & 0.23570 & 0.23570 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.91287 & 0.30429 & 0.30429 & 0.30429 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.86066 & 0.43033 & 0.43033 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.74536 & 0.74536
\end{bmatrix}$$

$$\underline{E}_6 = \begin{bmatrix}
-1.11111 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.12423 & -1.11803 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.14086 & -0.14086 & -1.12687 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.16265 & -0.16265 & -0.16265 & -1.13855 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.19245 & -0.19245 & -0.19245 & -0.19245 & -1.15470 & 0.0 & 0.0 & 0.0 & 0.0 \\
0.94281 & 0.94281 & 0.94281 & 0.94281 & 0.94281 & 1.17851 & 1.17851 & 1.17851 & 1.17851 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.91287 & 0.30429 & 0.30429 & 0.30429 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.86066 & 0.43033 & 0.43033 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.74536 & 0.74536
\end{bmatrix}$$

$$\underline{E}_7 = \begin{bmatrix}
-1.11111 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.12423 & -1.11903 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.14086 & -0.14086 & -1.12687 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.16265 & -0.16265 & -0.16265 & -1.13855 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.19245 & -0.19245 & -0.19245 & -1.19245 & -1.15470 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.23570 & -0.23570 & -0.23570 & -0.23570 & -0.23570 & -1.17851 & 0.0 & 0.0 & 0.0 \\
0.91287 & 0.91287 & 0.91287 & 0.91287 & 0.91287 & 0.91287 & 1.21716 & 1.21716 & 1.21716 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.86066 & 0.43033 & 0.43033 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.74536 & 0.74536
\end{bmatrix}$$

$$\underline{E}_8 = \begin{bmatrix}
-1.11111 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.12423 & -1.11803 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.14086 & -0.14086 & -1.12687 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.16265 & -0.16265 & -0.16265 & -1.13855 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.19245 & -0.19245 & -0.19245 & -0.19245 & -1.15470 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.23570 & -0.23570 & -0.23570 & -0.23570 & -0.23570 & -1.17851 & 0.0 & 0.0 & 0.0 \\
-0.30429 & -0.30429 & -0.30429 & -0.30429 & -0.30429 & -0.30429 & -1.21716 & 0.0 & 0.0 \\
0.86066 & 0.86066 & 0.86066 & 0.86066 & 0.86066 & 0.86066 & 0.86066 & 1.29099 & 1.29099 \\
0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & -0.74536 & 0.74536
\end{bmatrix}$$

$$\underline{E}_9 = \begin{bmatrix}
-1.11111 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.12423 & -1.11803 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.14086 & -0.14086 & -1.12687 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.16265 & -0.16265 & -0.16265 & -1.13855 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.19245 & -0.19245 & -0.19245 & -0.19245 & -1.15470 & 0.0 & 0.0 & 0.0 & 0.0 \\
-0.23570 & -0.23570 & -0.23570 & -0.23570 & -0.23570 & -1.17851 & 0.0 & 0.0 & 0.0 \\
-0.30429 & -0.30429 & -0.30429 & -0.30429 & -0.30429 & -0.30429 & -1.21716 & 0.0 & 0.0 \\
-0.43033 & -0.43033 & -0.43033 & -0.43033 & -0.43033 & -0.43033 & -0.43033 & -1.29099 & 0.0 \\
0.74536 & 0.74536 & 0.74536 & 0.74536 & 0.74536 & 0.74536 & 0.74536 & 0.74536 & 1.49071
\end{bmatrix}$$

$$\underline{E}_{10} = \begin{bmatrix} -1.11111 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.12423 & -1.11803 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.14086 & -0.14086 & -1.12087 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.16265 & -0.16265 & -0.16265 & -1.13855 & 0.0 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.19245 & -0.19245 & -0.19245 & -0.19245 & -1.15470 & 0.0 & 0.0 & 0.0 & 0.0 \\ -0.23570 & -0.23570 & -0.23570 & -0.23570 & -0.23570 & -1.17851 & 0.0 & 0.0 & 0.0 \\ -0.30429 & -0.30429 & -0.30429 & -0.30429 & -0.30429 & -0.30429 & -1.21716 & 0.0 & 0.0 \\ -0.43033 & -0.43033 & -0.43033 & -0.43033 & -0.43033 & -0.43033 & -0.43033 & -1.29099 & 0.0 \\ -0.74536 & -0.74536 & -0.74536 & -0.74536 & -0.74536 & -0.74536 & -0.74536 & -0.74536 & -1.49071 \end{bmatrix}$$

$$\underline{F}_R = \underline{E}_1{}^t\underline{E}_1 = \underline{E}_2{}^t\underline{E}_2 = \underline{E}_3{}^t\underline{E}_3 = \underline{E}_4{}^t\underline{E}_4 = \underline{E}_5{}^t\underline{E}_5 = \underline{E}_6{}^t\underline{E}_6 = \underline{E}_7{}^t\underline{E}_7 = \underline{E}_8{}^t\underline{E}_8 = \underline{E}_9{}^t\underline{E}_9 = \underline{E}_{10}{}^t\underline{E}_{10}$$

$$= C_{10} \begin{bmatrix} 1.0 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 \\ 0.5 & 1.0 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 \\ 0.5 & 0.5 & 1.0 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 \\ 0.5 & 0.5 & 0.5 & 1.0 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 \\ 0.5 & 0.5 & 0.5 & 0.5 & 1.0 & 0.5 & 0.5 & 0.5 & 0.5 \\ 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 1.0 & 0.5 & 0.5 & 0.5 \\ 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 1.0 & 0.5 & 0.5 \\ 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 1.0 & 0.5 \\ 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 0.5 & 1.0 \end{bmatrix}$$

where

$C_{10} = 2.22222$

$$\underline{G}_R = (\underline{E}_1{}^t\underline{E}_1)^{-1} = (\underline{E}_2{}^t\underline{E}_2)^{-1} = (\underline{E}_3{}^t\underline{E}_3)^{-1} = (\underline{E}_4{}^t\underline{E}_4)^{-1} = (\underline{E}_5{}^t\underline{E}_5)^{-1}$$
$$= (\underline{E}_6{}^t\underline{E}_6)^{-1} = (\underline{E}_7{}^t\underline{E}_7)^{-1} = (\underline{E}_8{}^t\underline{E}_8)^{-1} = (\underline{E}_9{}^t\underline{E}_9)^{-1} = (\underline{E}_{10}{}^t\underline{E}_{10})^{-1}$$

$$= \begin{bmatrix} g_{10} & g'_{10} & g'_{10} & g'_{10} & g'_{10} & g'_{10} & g'_{10} & g'_{10} & g'_{10} \\ g'_{10} & g_{10} & g'_{10} & g'_{10} & g'_{10} & g'_{10} & g'_{10} & g'_{10} & g'_{10} \\ g'_{10} & g'_{10} & g_{10} & g'_{10} & g'_{10} & g'_{10} & g'_{10} & g'_{10} & g'_{10} \\ g'_{10} & g'_{10} & g'_{10} & g_{10} & g'_{10} & g'_{10} & g'_{10} & g'_{10} & g'_{10} \\ g'_{10} & g'_{10} & g'_{10} & g'_{10} & g_{10} & g'_{10} & g'_{10} & g'_{10} & g'_{10} \\ g'_{10} & g'_{10} & g'_{10} & g'_{10} & g'_{10} & g_{10} & g'_{10} & g'_{10} & g'_{10} \\ g'_{10} & g'_{10} & g'_{10} & g'_{10} & g'_{10} & g'_{10} & g_{10} & g'_{10} & g'_{10} \\ g'_{10} & g'_{10} & g'_{10} & g'_{10} & g'_{10} & g'_{10} & g'_{10} & g_{10} & g'_{10} \\ g'_{10} & g'_{10} & g'_{10} & g'_{10} & g'_{10} & g'_{10} & g'_{10} & g'_{10} & g_{10} \end{bmatrix}$$

where

$$g_{10} = 0.81008 ,$$

and $$g'_{10} = -0.09001 .$$

Appendix E

DIGITAL COMPUTER PROGRAM OF THE PROPOSED MULTI-CLASS LEARNING ALGORITHM

This appendix presents the digital computer program for the multi-class learning algorithm proposed in Section 4.0 which is a generalization of the Ho-Kashyap algorithm. The program was written in the modified Fortran IV E level language and has been run on IBM 360/50 University of Pittsburgh Time-Sharing System. The entire program consists of the following four parts.

1) Multi-class learning algorithm,
2) generalized inversed matrix (GINVS),
3) Andrees algorithm (ADE), and
4) generation of matrix $\underline{E}$ and its inverse of equilateral simplex vertex vector (SIMPLX).

The last three parts used as subroutines for the multi-class learning algorithm. The programs for the generalized inverse matrix and Andree's algorithm are adapted from Brand's work[6].

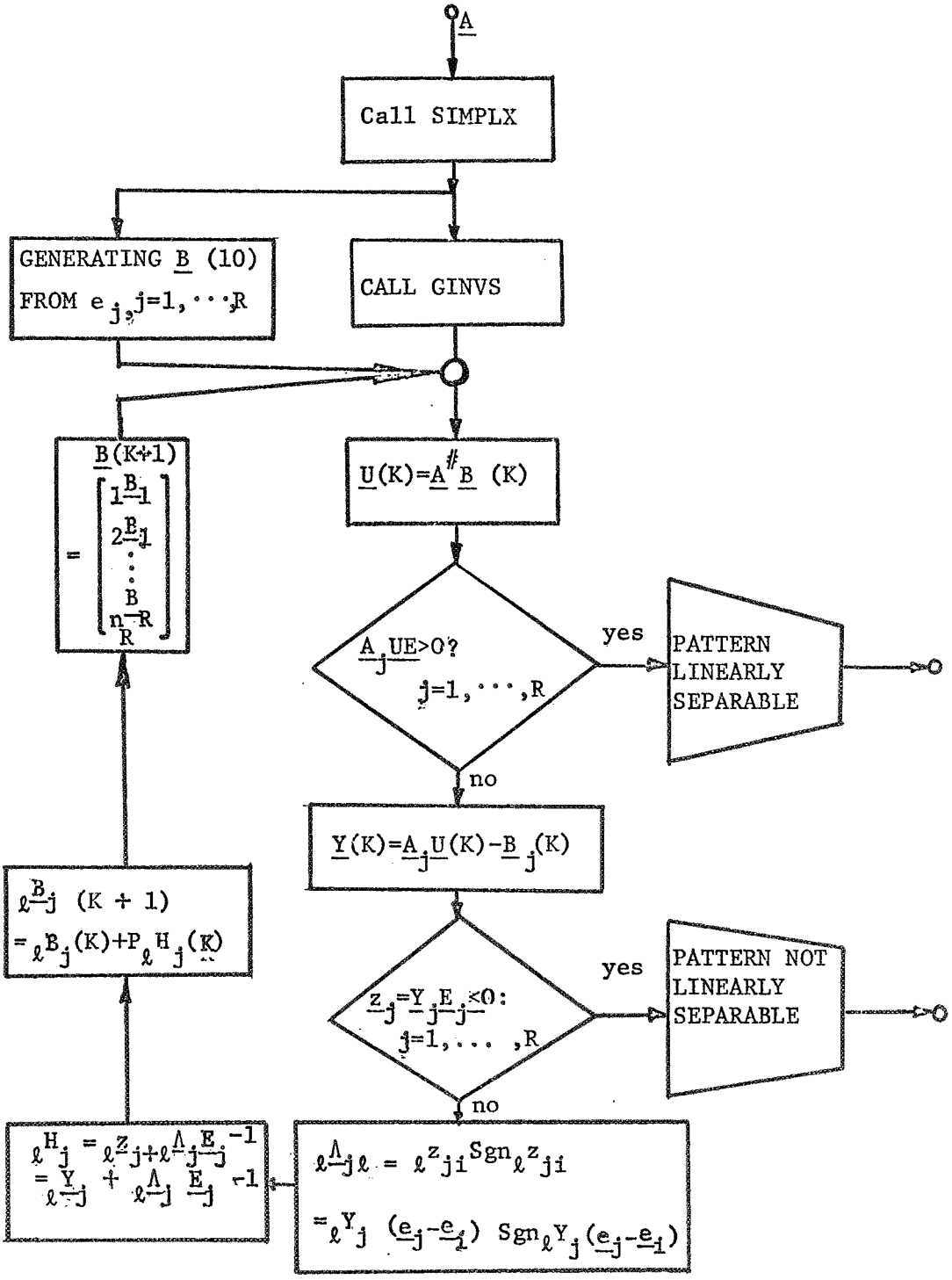
The essence of the multi-class learning algorithm is to determine a solution matrix $\underline{U}$ of a set of matrix inequalities $\underline{A}_j\underline{U}\ \underline{E}_j > \underline{0}$ for all $j = 1, 2, \ldots, R$. The complete algorithm is summarized as follows:

$$
\begin{cases}
\underline{U} = \underline{A}^{\#}\,\underline{B}(0), \\
\underline{Y}(K) = \underline{A}\,\underline{U}(K) - \underline{B}(K), \\
\underline{B}(K+1) = \underline{B}(K) + p\,H(K), \\
\underline{H}_j(K) = \{\,\underline{Z}_j(K) + \underline{\Lambda}_j(K)\,\}\,\underline{E}_j^{-1}, \\
\underline{Z}_j(K) = \underline{Y}_j(K)\,E_j, \\
{}_{\ell}\underline{\Lambda}_{jq}(K) = ({}_{\ell}Z_{jq}(K)\ \mathrm{Sgn}\ ({}_{\ell}Z_{jq}(K)) \\
\underline{U}(K+1) = U(K) + p\underline{A}^{\#}\,\underline{H}(K)
\end{cases}
$$

$$
\underline{B}_j(0) = \beta \begin{bmatrix} \underline{e}_j^t \\ \underline{e}_j^t \\ \cdot \\ \cdot \\ \underline{e}_j^t \end{bmatrix}, \qquad \beta > 0
$$

The program terminates when either a solution $\underline{U}$ is obtained so that $\underline{A}_j\,\underline{U}\,\underline{E}_j > \underline{0}$ for all $j = 1,2,\ldots,R$, or else, the sample patterns not linearly separable if $\underline{Y}_j\,\underline{E}_j \leq \underline{0}$ for all $j = 1,2,\ldots,R$. The flow chart and computer program listing are given below:

FLOW CHART FOR THE PROPOSED MULTI CLASS ALGORITHM



```
C
C       MULTICLASS LEARNING MACHINE
        COMMON A(250,21),AI(21,250),G(10,10),E(9,90),NN(10),
       1N,ID,IR,EI(9,90)
        DIMENSION Z(250,9),AV(21,9),BV(250,9),Y(250,9)
        CALL SIMPLX
        READ(5,500)N,ID,IR
500     FORMAT(1013)
        READ(5,610)((A(I,J),J=1,ID),I=1,N)
610     FORMAT(21F3.0)
510     FORMAT(9F10.5)
        CALL GINVS
        WRITE(6,5000)
5000    FORMAT(/'INVERSED INPUT MATRIX='/)
        WRITE(6,510)((AI(I,J),I=1,ID),J=1,N)
        WRITE(6,500)N
        S=1.0
        NR=IR-1
        INR=IR*NR
         M=0
         READ(5,500)(NN(J),J=1,IR)
        DO 1 I=1,IR
        KK=NN(I)
        DO 1 J=1,KK
        M=M+1
        DO 1 L=1,NR
        TEMP=S*G(I,L)
        BV(M,L)=TEMP
1       CONTINUE
        ISW=3
        P=1.0
        IT=0
2       CONTINUE
        ISW1=1
        DO 12 I=1,ID
        DO 12 J=1,NR
        AV(I,J)=0.0
        DO 12 K=1,N
        TEMP1=AI(I,K)
        TEMP2=BV(K,J)
        TEMP3=TEMP1*TEMP2
        AV(I,J)=AV(I,J)+TEMP3
12      CONTINUE
        WRITE(6,1010)IT
1010    FORMAT(/I3,' ,ITERATION'/)
        DO 3 I=1,N
        DO 3 J=1,NR
        Y(I,J)=0.0
```

```
      DO 3 K=1,ID
      TEMP1=A(I,K)
      TEMP2=AV(K,J)
      TEMP3=TEMP1*TEMP2
      Y(I,J)=Y(I,J)+TEMP3
3     CONTINUE
      WRITE(6,1040)
1040  FORMAT(/'U(I,J)='/)
      WRITE(6,510)((AV(I,J),J=1,NR),I=1,ID)
      WRITE(6,1050)
1050  FORMAT(/'B(I,J)='/)
      WRITE(6,510)((BV(I,J),J=1,NR),I=1,N)
7     CONTINUE
      K1=1
      K2=0
      L=0
      DO 16 I=1,IR
      KK=NN(I)
      K2=KK+K2
      DO 10 M=1,NR
      L=L+1
      DO 10 J=K1,K2
      Z(J,M)=0.0
      DO 10 K=1,NR
      TEMP1=Y(J,K)
      TEMP2=E(K,L)
      TEMP3=TEMP1*TEMP2
      Z(J,M)=Z(J,M)+TEMP3
10    CONTINUE
      K1=KK+K1
16    CONTINUE
      GO TO (6,30),ISW1
6     CONTINUE
      WRITE(6,1020)
1020  FORMAT(/'A(J)*U*E(J)=')
      WRITE(6,510)((Z(I,J),J=1,NR),I=1,N)
      DO 5 I=1,N
      DO 5 J=1,NR
      TEMZ=Z(I,J)
      IF(TEMZ)11,11,5
5     CONTINUE
      ISW=1
      GO TO 35
11    CONTINUE
      IT=IT+1
      DO 4 I=1,N
      DO 4 J=1,NR
      TEMP1=Y(I,J)
```

```
      TEMP2=BV(I,J)
      TEMP3=TEMP1-TEMP2
      Y(I,J)=TEMP3
4     CONTINUE
      ISW1=2
      GO TO 7
30    CONTINUE
      WRITE(6,1030)
1030  FORMAT(/'Y(J)*E(J)='/)
      WRITE(6,510)((Z(I,J),J=1,NR),I=1,N)
      DO 21 I=1,N
      DO 21 J=1,NR
      TEMZ=Z(I,J)
      IF(TEMZ)21,21,35
21    CONTINUE
      ISW=2
35    CONTINUE
      GO TO (60,65,22),ISW
22    CONTINUE
      DO 20 I=1,N
      DO 20 J=1,NR
      TEMP=ABS(Z(I,J))
      Z(I,J)=Z(I,J)+TEMP
20    CONTINUE
      K1=1
      K2=0
      L=0
      DO 40 I=1,IR
      KK=NN(I)
      K2=KK+K2
      DO 17 M=1,NR
      L=L+1
      DO 17 J=K1,K2
      Y(J,M)=0.0
      DO 17 K=1,NR
      Y(J,M)=Y(J,M)+Z(J,K)*EI(K,L)
17    CONTINUE
      K1=KK+K1
40    CONTINUE
      DO 50 I=1,N
      DO 50 J=1,NR
      Y(I,J)=P*Y(I,J)
      BV(I,J)=BV(I,J)+Y(I,J)
50    CONTINUE
      IF(IT-40)80,80,85
80    CONTINUE
      GO TO 2
85    CONTINUE
```

```
      WRITE(4,4000)((BV(I,J),J=1,NR),I=1,N)
4000  FORMAT(2F10.5)
      GO TO 100
60    CONTINUE
      WRITE(6,1070)
1070  FORMAT(/'**********PATTERN LINEARLY SEPARABLE**')
      GO TO 90
65    CONTINUE
      WRITE(6,1080)
1080  FORMAT(/'**********PATTERN NON-SEPARABLE*******')
90    CONTINUE
100   END
```

```
C
      SUBROUTINE GINVS
      COMMON A(250,21),AI(21,250),G(10,10),E(9,90),NN(10),
     1N,ID,IR,EI(9,90)
      DIMENSION T(21,21),U(21,21),C(21,21),B(21,21),
     1D(21,21),X(21,21)
C     SELECT SMALLEST DIMENSION
      IOUT=2
      INORM=0
      DO 750 I=1,ID
      X(I)=0.0
750   CONTINUE
      K1=ID
      K2=N
      ERROR=0.1
1     CONTINUE
      M=K2
      K=K1
      IF(K-M)2,3,3
    2 KA=K
      KB=M
      JI=1
      DO 20 I=1,KA
      DO 20 J=1,KA
      B(I,J)=0.0
      DO 20 K=1,KB
      B(I,J)=B(I,J)+A(K,I)*A(K,J)
   20 C(I,J)=B(I,J)
      GO TO 4
    3 KA=M
      KB=K
      JI=2
      DO 21 I=1,KA
      DO 21 J=1,KA
      B(I,J)=0.0
      DO 21 K=1,KB
      B(I,J)=B(I,J)+A(I,K)*A(J,K)
   21 C(I,J)=B(I,J)
   4  CONTINUE
      CALL  ADE(B,B,T,KA,L,ERROR)
      WRITE(6,100) L
      IF(L-KA)5,16,16
    5 CONTINUE
C     CONSTRUCT U MATRIX
      DO 10 I=1,KA
      DO 10 J=1,KA
      IF(B(I,I)-0.0)7,7,8
    7 CONTINUE
```

```
      U(I,J)=-T(I,J)
      GO TO 9
    8 CONTINUE
      U(I,J)=0.0
    9 CONTINUE
      U(I,I)=B(I,I)
   10 CONTINUE
C     COMPUTE UTGU=C
      DO 26 I=1,KA
      DO 26 J=1,KA
      T(I,J)=0.0
      DO 26 K=1,KA
   26 TII,J)=T(I,J)+C(I,K)*U(K,J)
      DO 27 I=1,KA
      DO 27 J=1,KA
      C(I,J)=0.0
      DO 27 K=1,KA
   27 C(I,J)=C(I,J)+U(K,I)*T(K,J)
C     SET ROWS & COLS OF C =  0.0
       J=0
C
      DO 12 I=1,KA
      IF(B(I,I)-0.0)12,11,12
   11 CONTINUE
      DO 18 J=1,KA
      C(I,J)=0.0
      C(J,I)=0.0
   18 CONTINUE
      J=0.0
   12 CONTINUE
C     EXTRACT D MATRIX FROM C
      NX=0.0
      DO 15 I=1,KA
      DO 15 J=1,KA
      IF(C(J,I)-0.0)13,15,13
   13 CONTINUE
      NX=NX+1
      MX=0.0
      DO 15 K=J,KA
      IF(C(K,I)-0.0)14,15,14
   14 CONTINUE
      MX=MX+1
      D(MX,NX)=C(K,I)
      C(K,I)=0.0
   15 CONTINUE
      CALL  ADE(D,C,T,MX,L1,ERROR)
      LX=0.0
      DO 37 I=1,MX
```

```
   30 CONTINUE
      LX=LX+1
      IF(B(LX,LX)-0.0)31,31,33
   31 CONTINUE
      DO 32 KT=1,KA
      T(LX,KT)=0.0
   32 CONTINUE
      IF(LX-KA)30,37,37
   33 CONTINUE
      K=0.0
      DO 37 J=1,MX
   34 CONTINUE
      K=K+1
      IF(B(K,K)-0.0)35,35,36
   35 CONTINUE
      T(LX,K)=0.0
      IF(K-KA)34,37,37
   36 CONTINUE
      T(LX,K)=D(I,J)
   37 CONTINUE
C     CONSTRUCT UDUT
      DO 38 I=1,KA
      DO 38 J=1,KA
      C(I,J)=0.0
      DO 38 K=1,KA
   38 C(I,J)=C(I,J)+U(I,K)*T(K,J)
      DO 39 I=1,KA
      DO 39 J=1,KA
      B(I,J)=0.0
      DO 39 K=1,KA
   39 B(I,J)=B(I,J)+C(I,K)*U(J,K)
C     COMPUTE INVERSE
   16 GO TO(40,41),JI
   40 DO 42 I=1,KA
      DO 42 J=1,KB
      AI(I,J)=0.0
      DO 42 K=1,KA
   42 AI(I,J)=AI(I,J)+B(I,K)*A(J,K)
      GO TO 43
   41 DO  17 I=1,KB
      DO 17 J=1,KA
      AI(I,J)=0.0
      DO 17 K=1,KA
      AI(I,J)=AI(I,J)+A(K,I)*B(K,J)
   17 CONTINUE
   43 CONTINUE
      GO TO (69,599),IOUT
  599 CONTINUE
```

```
      M=KB
      DO 601 I=1,KA
      DO 601 J=1,KA
      B(I,J)=0.0
      DO 601 K=1,M
      B(I,J)=B(I,J)+AI(I,K)*A(K,J)
601   CONTINUE
      KK=1
      DO 604 I=1,KA
      DO 604 J=1,KA
      TEMP=B(I,J)-B(J,I)
      IF(ABS(TEMP)-0.001)604,604,605
604   CONTINUE
      ITGI=1
      GO TO 669
605   CONTINUE
      WRITE(6,1000)TEMP,I,J
1000  FORMAT(F10.5,2I3)
      ERROR=ERROR*0.1
      ITGI=2
669   CONTINUE
      GO TO (69,700),ITGI
700     CONTINUE
       INORM=INORM+1
       DO 702 I=1,KA
       C(I,I)=0.0
       DO 702 J=1,KB
       C(I,I)=C(I,I)+AI(I,J)
702    CONTINUE
       DO 703 I=1,KA
       X(INORM)=X(INORM)+C(I,I)*C(I,I)
703    CONTINUE
       X(INORM)=SQRT(X(INORM))
       R(INORM)=ERROR*10.
       WRITE(6,1010)X(INORM),R(INORM)
1010   FORMAT('X(INORM)=',F10.5,'R(INORM)=',F10.8//)
       IF (KA-L)769,769,1
769    CONTINUE
       S=X(1)
       DO 704 I=1,INORM
       IF(S-X(I))704,708,708
708    CONTINUE
       S=X(I)
       JN=I
704    CONTINUE
       ERROR=R(JN)
       IOUT=1
       WRITE(6,1020)S
```

```
1020    FORMAT(/'S=',F10.5/)
        GO TO 1
   69 CONTINUE
  100 FORMAT(2I3)
      RETURN
      END
```

```
C
      SUBROUTINE  ADE(B,E,T,KA,L,ERROR)
      DIMENSION E(21,21),B(21,21),T(21,21),NS(21)
      COMMON A(250,21),AI(21,250),G(10,10),S(9,90),NN(10),
     1 N,ID,IR,EI(9,90)
C
      IK=0
      DO 7 I=1,KA
      DO 7 J=1,KA
      NS(I)=0
      T(I,J)=0.0
      T(I,I)=1.0
C
    7 CONTINUE
      DO 69 LSI=1,KA
    1 CONTINUE
      X=0.0
      DO 10 K=1,KA
      IF(NS(K)-K)8,10,10
    8 CONTINUE
      IF(B(K,K)-X)10,9,9
    9 CONTINUE
      X=B(K,K)
      IK=K
   10 CONTINUE
      NS(IK)=IK
      X=1.0/SQRT(X)
      DO 11 J=1,KA
      IF(IK-J)6,11,6
    6 CONTINUE
      B(IK,J)=B(IK,J)*X
      B(J,IK)=B(J,IK)*X
      T(IK,J)=T(IK,J)*X
      T(J,IK)=T(J,IK)*X
   11 CONTINUE
      T(IK,IK)=T(IK,IK)*X
      B(IK,IK)=1.0
      DO 13 I=1,KA
      DO 13 J=1,KA
      IF(J-IK)33,13,33
   33 CONTINUE
      IF(I-IK)12,13,12
   12 CONTINUE
      B(I,J)=B(I,J)-B(IK,J)*B(I,IK)
   13 CONTINUE
      DO 21 I=1,KA
      DO 21 J=1,KA
      IF(I-IK)24,21,24
```

```
   24 CONTINUE
      T(I,J)=T(I,J)-T(IK,J)*B(I,IK)
   21 CONTINUE
      DO 14 J=1,KA
      B(J,IK)=0.0
      B(IK,J)=0.0
   14 CONTINUE
      B(IK,IK)=1.0
      MT=0
      L=0
      DO 17 J=1,KA
      IF(B(J,J)-ERROR)2,2,15
    2 CONTINUE
      B(J,J)=0.0
      MT=MT+1
      GO TO 17
   15 CONTINUE
      IF(B(J,J)-1.0)17,16,17
   16 CONTINUE
      MT=MT+1
      L=L+1
   17 CONTINUE
      IF(MT-KA)69,3,69
   69 CONTINUE
    3 CONTINUE
C     CHECK RANK
      IF(L-KA)19,18,19
   18 CONTINUE
      DO 20 I=1,KA
      DO 20 J=1,KA
      B(I,J)=0
      DO 20 K=1,KA
   20 B(I,J)=B(I,J)+T(K,I)*T(K,J)
   19 CONTINUE
      RETURN
      END
```

```
C
      SUBROUTINE SIMPLX
      COMMON A(250,21),AI(21,250),G(10,10),E(9,90),NN(10),
     1N,ID,IR,EI(9,90)
      DIMENSION EE(90,10)
      ID=9
      N=9
      IR=10
      NR=IR-1
      INR=NR*IR
      X=IR/(NR+0.0)
      DO 20 I=1,IR
      DO 20 J=1,NR
      NCJ=IR-J
      Y=NCJ/(NCJ+1.0)
      YY=X*Y
      Z=SQRT(YY)
      IF(I-J)5,10,15
    5 CONTINUE
      G(I,J)=0.0
      GO TO 20
   10 CONTINUE
      G(I,J)=Z
      GO TO 20
   15 CONTINUE
      G(I,J)=(-1.0/NCJ)*Z
   20 CONTINUE
C     GENERATING THE MATRICS OF E(I,J)
      L=1
      DO 30 I=1,IR
      DO 30 K=1,IR
      DO 31 J=1,NR
      IF(K-I)35,30,35
   35 CONTINUE
      TEMP1=G(I,J)
      TEMP2=G(K,J)
      TEMP3=TEMP1-TEMP2
      EE(L,J)=TEMP3
   31 CONTINUE
      L=L+1
   30 CONTINUE
      DO 40 I=1,NR
      DO 40 J=1,INR
       E(I,J)=EE(J,I)
40    CONTINUE
      K1=0
      K2=1
      DO 70 I=1,IR
```

```
      L=1
      K1=NR+K1
      DO 56 M=K2,K1
      DO 55 J=1,NR
      A(J,L)=0.0
      A(J,L)=E(J,M)
55    CONTINUE
      L=L+1
56    CONTINUE
      CALL GINVS
      L=1
      DO 60 J=K2,K1
      DO 61 M=1,NR
      EI(M,J)=AI(M,L)
61    CONTINUE
      L=L+1
60    CONTINUE
      K2=K2+NR
70    CONTINUE
      RETURN
      END
```

BIBLIOGRAPHY

1. Arkadev, A. G. and Braverman, E. M., Computers and Pattern Recognition. Washington, D. C.: Thompson Book Co., 1967.

2. Bernstein, M. I., "A Method for Recognizing Handprinted Characters in Real Time", in Pattern Recognition, edited by L. N. Kanal, Washington, D. C.: Thompson Book Company, pp. 109-140, 1968.

3. Bernstein, M. I., "Handprinted Input for On-Line Systems". TM-3937/000/00, System Development Corp., Santa Monica, Calif., April 1968.

4. Blaydon, C. C., "Recursive Algorithms for Pattern Recognition", Ph.D. Dissertation, Harvard University, 1967.

5. Brain, A. E., Hart, P. E. and Munson, J. H., "Graphical-Data-Processing Research Study and Experimental Investigation", Tech. Rept. ECOM-01901-25. Stanford Research Institute, Menlo Park, California, December 1966.

6. Brand, T. E., The Feasibility of Applying Pattern Recognition Concepts for An Aircraft Approach Progress Monitor, M.S. Thesis, Department of Electrical Engineering, University of Pittsburgh, 1968.

7. Brown, R. M., "On Line Computer Recognition of Handprinted Characters", IEEE Transactions on Electronic Computers, Vol. EC-13, No. 6, pp. 750-752, December 1964.

8. Chaplin, W. G. and Levadi, V. S., "A Generalization of the Linear Threshold Decision Algorithm to Multiple Classes", in Computers and Information Sciences II, edited by J. Tou, New York: Academic Press, pp. 337-355, 1967.

9. Cheng, G. C., Ledley, R. S., Pollock, D. K. and Rosenfeld, A., editors, Pictorial Pattern Recognition. Washington, D. C.: Thompson Book Co., 1968.

10. Chow, C. K., "A Recognition Method Using Neighborhood Dependence", IRE Transactions on Electronic Computers, Vol. EC-11, No. 5, 683-690, October 1962.

11. Chow, C. K. and Liu, C. N., "An Approach to Structure Adaptation in Pattern Recognition", Proceeding of National Electronics Conference, Vol. 22, pp. 573-578, 1966.

12. Dimond, T. L., "Devices for Reading Handwritten Characters", Proceedings of Eastern Joint Computer Conference, Vol. 12, pp. 232-237, December 1957.

13. Doyle, W., "Recognition of Sloppy, Hand-Printed Characters", Proceedings of Western Joint Computer Conference, Vol. 17, pp. 133-142, May 1960.

14. Eden, M. and Halle, M., "The Characterization of Cursive Writing", in Information Theory, Fourth London Symposium edited by C. Cherry, London, England. Butterworthe, pp. 287-299, 1961.

15. Feder, J. and Freeman, H., "Digital Curve Matching Using a Contour Correlation Algorithm", IEEE International Convention Record, part 3, pp. 69-85, 1966.

16. Fischer, G. L., Pollock, D. K., Roddeck, B., and Stevens, M. E., editors, Optical Character Recognition. Washington, D. C.: Spartan Books, 1962.

17. Forejt, D. A. "The Method of Arbitrary Choice for Training Linear Perceptrons", Ph.D. Dissertation, University of Pittsburgh, 1967.

18. Freeman, H., "On Encoding of Arbitrary Geometric Configurations", IRE Transactions on Electronic Computers, Vol. EC-10, pp. 260-268, June 1961.

19. Freeman, H., "Techniques for Digital Computer Analysis of Chain Encoded Arbitrary Plane Curves", Proceedings of National Electronics Conference, Vol. 17, pp. 421-432, 1961.

20. Freeman, H., "On Digital Computer Classification of Geometric Line Patterns", Proceedings of National Electronics Conferences, Vol. 18, pp. 312-324, 1962.

21. Freeman, H. and Garder, L., "A Pictorial Jigsaw Puzzle: The Computer Solution of a Problem in Pattern Recognition", IEEE Transactions on Electronic Computers, Vol. EC-13, No. 2, pp. 118-127, April 1964.

22. Freeman, H. and Morse, S. P., "On Searching Contour Map for a Given Terrain Elevation Profile", Journal of Franklin Institute, Vol. 284, No. 1, pp. 1-25, July 1967.

23. Frishkopf, L. S. and Harmon, L. D., "Machine Reading of Cursive Script", in Information Theory, Fourth London Symposium, edited by C. Cherry, London, England. Butterworths, pp. 300-316, 1961.

24. Fu, K. S., Sequential Methods in Pattern Recognition and Machine Learning. New York, N. Y.: Academic Press, 1968.

25. Fu, K. S. and Chen, C. H., "A Sequential Decision Approach to Problems in Pattern Recognition and Learning", Proceedings of National Electronics Conference, Vol. 20, pp. 626-630, 1964.

26. Fu, K. S. and W. G.Wee, "On Generalization of Adaptive Algorithms and Application of the Fuzzy Sets Concepts to Pattern Classification", TR EE67-7, School of Electrical Engineering, Purdue University, June 1967.

27. Greanias, E. C., Meagher, P. F., Norman, R. J. and Essinger, P., "Recognition of Handwritten Numerals by Contour Analysis", IBM Journal of Research and Development, Vol. 7, No. 1, pp. 14-21, January 1963.

28. Groner, G. F., "Real-Time Recognition of Handprinted Text", Proceedings of AFIPS Conference, Vol. 29, pp. 591-601, 1966.

29. Highleyman, W. H., "Linear Decision Functions with Applications to Pattern Recognition", Proceedings of IRE, Vol. 50, No. 6, pp. 1501-1514, June 1962.

30. Highleyman, W. H., "Data for Character Recognition Studies", IEEE Transactions on Electronic Computers, Vol. EC-12, No. 2, pp. 135-136, April 1963.

31. Ho, Y. C. and Agrawala, A.K., "On Pattern Recognition Algorithms, Introduction and Survey", Proceedings of IEEE, Vol. 56, No. 12, pp. 2101-2114, December 1968.

32. Ho, Y. C. and Kashyap, R. L., "An Algorithm for Linear Inequalities and its Applications", IEEE Transactions on Electronic Computers, Vol. EC-14, No. 5, pp. 683-688, October, 1965.

33. Kanal, L. N., editor, Pattern Recognition. Washington, D. C.: Thompson Book Co., 1968.

34. Kolers, P. A. and Eden, M., editors, Recognizing Patterns. Cambridge, Mass.: The MIT Press, 1968.

35. Kovalevsky, V. A., editor, Character Readers and Pattern Recognition. New York, N. Y.: Spartan Books, 1968.

36. Kuhl, F., "Classification and Recognition of Handprinted Characters", IEEE International Convention Record, part 4, pp. 75-93, 1963.

37. Lindgren, N., "Machine Recognition of Human Language, Part III - Cursive Script Recognition", IEEE Spectrum, pp. 104-116, May 1965.

38. MacDonald, J. S., "Experimental Studies of Handwriting Signals", Sc. D. Dissertation, Massachusetts Institute of Technology, 1964.

39. Mermelstein, P., "Computer Recognition of Connected Handwritten Words", Sc. D. Dissertation, MIT, 1964.

40. Munson, J. H., Duda, R. O. and Hart, P. E., "Experiments with Highleyman's Data", IEEE Transactions on Computers, Vol. C-17, pp. 399-401, April 1968.

41. Munson, J. H., "The Recognition of Hand-Printed Text", in Pattern Recognition, edited by L. N. Kanal, Washington, D. C.: Thompson Book Co., pp. 115-140, 1968.

42. Nagy, G., "State of the Art in Pattern Recognition", Proceedings of IEEE, Vol. 56, No. 5, pp. 836-862. May 1968.

43. Nilsson, N. J., Learning Machines. New York, N. Y.: McGraw-Hill Book Co., 1965.

44. Sebestyen, G. S., Decision-Making Processes in Pattern Recognition. New York, N. Y.: The MacMillan Company, 1962.

45. Sezari, N. and Katagiri, H., "Character Recognition by Follow Method", Proceedings of IEEE, Vol. 53, No. 5, p. 510, May 1965.

46. Simek, J. G. and Tunis, C. J., "Handprinting Input Device for Computer Systems", IEEE Spectrum, pp. 72-81, July 1967.

47. Smith, F. W., "Pattern Classifier Design by Linear Programming", IEEE Transactions on Computers, Vol. C-17, No. 4, pp. 367-372, April, 1968.

48. Sprinrad, R. J., "Machine Recognition of Handprinted Block Letters", Ph.D. Dissertation, MIT, Cambridge, Mass., June 1963.

49. Teitelman, W., "Real-Time Recognition of Hand-Drawn Characters", Proceedings of AFIPS Conference, Vol. 26, part 1, pp. 559-575, 1964.

50. Wee, W. G. and Fu, K. S., "An Extension of the Generalized Inverse Algorithm to Multi-Class Pattern Classification", IEEE Transactions on Systems Science and Cybernetics, Vol. SEC-4, No. 2, pp. 192-194, July 1968

51. Wee, G., "A Survey of Pattern Recognition", IEEE Proceedings of Seventh Symposium on Adaptive Processes, pp. 2e1-2e13, 1968.

52. Zahn, C. T., "Two-Dimensional Pattern Description and Recognition Via Curvature Points", SLAC Report No. 70, Stanford Linear Accelerator Center, Stanford University, Stanford, California, December 1966.

53. Zobrak, M. J., "A Method for Rapid Recognition of Hand Drawn Line Patterns", M. S. Thesis, Department of Electrical Engineering, University of Pittsburgh, 1966. (Also, Tech. Report No. 2, Learning Research and Development Center, Univ. of Pittsburgh, 1966)

54. Zobrak, M. J. and T. W. Sze, "A Method of Recognition of Hand Drawn Line Patterns", Proceedings of the First Annual Princeton Conference on Information Science and Systems, pp. 240-244, 1967.