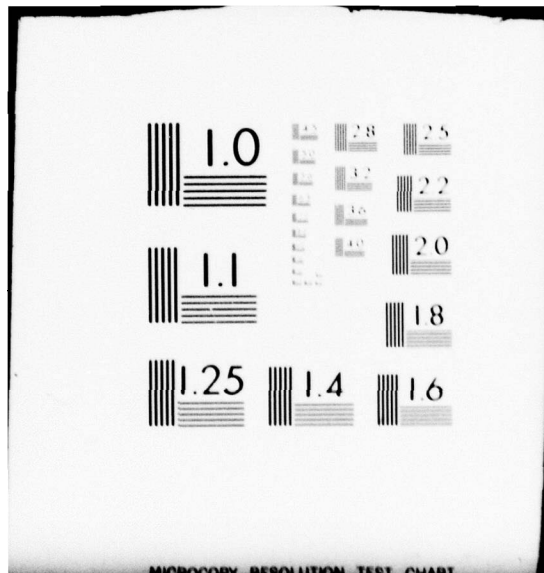


AD-A069 298 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OHIO SCH--ETC F/G 9/2
SEGMENTATION OF TOUCHING ENGLISH LETTERS.(U)
MAR 79 R E BENTKOWSKI
UNCLASSIFIED AFIT/GE/EE/79-1 NL

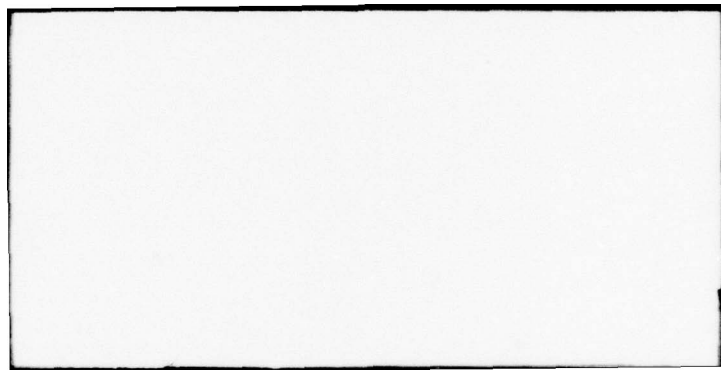
1 OF 2

AD A069298





MICROCOPY RESOLUTION TEST CHART



DISCLAIMER NOTICE

**THIS DOCUMENT IS BEST QUALITY
PRACTICABLE. THE COPY FURNISHED
TO DDC CONTAINED A SIGNIFICANT
NUMBER OF PAGES WHICH DO NOT
REPRODUCE LEGIBLY.**

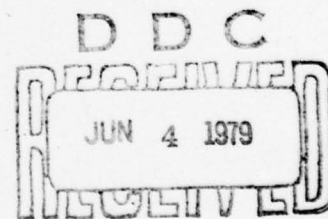


SEGMENTATION OF TOUCHING
ENGLISH LETTERS

THESIS

AFIT/GE/EE/79-1

Roy E. Bentkowski
2nd Lt USAF



aw
A

Approved for public release; distribution unlimited

79 05 30 254

14

AFIT/GE/EE/79-1

6

SEGMENTATION OF TOUCHING
ENGLISH LETTERS.

9

Master's THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air Training Command
in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

12 189p.

10

Edward by
Roy E. Bentkowski, B.S.
2nd Lt USAF

Graduate Electrical Engineering

11 March 1979

Accession For	
NTIS - G&I	<input checked="" type="checkbox"/>
DDC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or special
A	234A

Approved for public release; distribution unlimited.

012 225

LB

Acknowledgements

I wish to thank Captain Troy Sponaugle for the original recognition programs and the data files. I am also very grateful to Captain John Leary. His familiarity with the computer programs was an immense help to me. A very special thanks to Dr. Matthew Kabrisky for his valuable guidance and encouragement throughout my thesis.

Contents

	<u>Page</u>
Acknowledgements	ii
List of Figures	iv
List of Tables.	v
Abstract	vi
I. Introduction	1
Background	1
Objectives	2
II. Approach	3
Pattern Recognition Algorithm	3
Window Shapes	5
Data	7
LETTERS	7
SENDATA	8
Programs	9
MAKFLTR	9
PROTOBLD	9
CMPRSEN	9
III. Results	11
Single Alphabet Prototypes	11
Multiple Alphabet Prototypes	16
IV. Conclusions and Recommendations	20
Bibliography	23
Appendix A: LETTERS	24
Appendix B: SENDATA	26
Appendix C: MAKFLTR	49
Appendix D: PROTOBLD	71
Appendix E: CMPRSEN	80
Appendix F: Data Output For Various Standard Deviations	103
Appendix G: Data Output For Complex and Amplitude Spectra: Sentences 1 - 4	110
Vita	127

List of Figures

<u>Figure</u>		<u>Page</u>
1	Interaction of Data Files and Computer Programs	4
2	Various Trapezoidal Window Shapes	7
3	Various Gaussian Window Shapes	8

List of Tables

<u>Table</u>	<u>Page</u>
I. Distance and Figure of Merit Data for Various Window Shapes on Sentence 1	13
II. Complex Spectrum - Lowest Distance Achieved by Prototype Filters	14
III. Amplitude Spectrum - Lowest Distance Achieved by Prototype Filters	15
IV. Multiple Alphabet Prototypes - False Trigger Chart	19

Abstract

This paper examines the problem of building a machine to read uncontrolled type fonts set with essentially no space between letters (within words). The consequence of this type of data, which represents the usual format of printed text, is that the data vectors produced by the optical scanner contain multiple letters and/or fragments of letters that cannot be easily separated.

An algorithm based on a variant of running cross-correlation between prototype letters and successively "windowed" fragments of the sentence is employed. The algorithm computes the Euclidean distance between prototypes and the sentence fragment in a filtered Fourier domain.

It is shown that appropriate normalization and windowing techniques allow perfect recognition of touching letters within words. This occurs even when no a priori knowledge of letter location within the word is available, provided that suitable prototypes can be established.

Multiple alphabet prototypes were then built and used to examine widely differing type fonts. Techniques to set acceptance thresholds were evaluated and the behavior of the resulting recognition system tabulated. A number of false triggers did occur in this case and these were discussed. Recommendations for further improvements in the system are suggested.

Introduction

Background

A reading machine (to read uncontrolled, linotype-set fonts) based on correlation mathematics (Refs 3 and 4) has never been successfully operated, apparently because of the nature of the noise in the data stream that such systems are required to process. The noise arises from multiple mechanisms, including shape variations, and has never been well defined because the processes producing it seem to have no useful description. Empirical data for this noise does not seem to exist in the literature. Recognition algorithms are unable to reliably operate a threshold detection machine, even for isolated letters where the best prototype has been employed. In the more realistic case of touching (or nearly touching) letters, as set by linotype or equivalent machines, an additional noise consequent to multiple complete and incomplete signals in the observation window further reduces the accuracy of all known algorithms.

In a previous thesis by Captain Paul Kabler (Ref 2), 150 alphabets with widely varying fonts were digitized, and these serve as the basis data set for this thesis. The Fourier transforms of these vectors were obtained in the unpublished work of Captain Troy Sponaugle (Ref 5), and were used to test various isolated letters in a correlation-type recognition system against the kinds of noise mentioned above.

Captain Sponaugle then constructed various data sentences from the two-space letter data. These sentences were each made from a distinct alphabet with fixed spacing between the letters. Further improvements of the computer algorithms enabled reading the sentence by passing a window across the sentence by single element increments. This program and the data files are the basic research tools for this thesis.

Objectives

The major objective of this thesis is to solve the segmentation problem of touching English letters. A method of shaping the windows to de-emphasize the energy in the right and left edges of the letters is used. This method eliminates some of the noise introduced by the neighboring letters, and in fact, produces more reliable recognition.

The computer performance is evaluated by comparing the computer output with the actual content of the window; this enables acceptance thresholds to be set for each letter. Therefore, a simple decision rule can be implemented in this recognition system.

Approach

This section begins by summarizing the logic executed by the actual pattern recognition algorithm. It then continues by discussing the various window shaping functions that can be applied by the programs. The types of window shaping functions used include:

Unshaped (or Full)
Trapezoid
Gaussian

This section concludes with a summary of the data files and computer programs (Ref 5) used in this thesis. The two data files are:

LETTERS
SENDATA

The three computer programs used are:

MAKFLTR
PROTOBLD
CMRSEN

Figure 1 gives a flowchart of the way that the programs and data files interact. In the figure, the circles represent the data files and the rectangular boxes represent the computer programs. The triangular boxes are files built by the indicated programs for use in later programs.

Pattern Recognition Algorithm

A brief description of the pattern recognition algorithm used in this thesis follows. This algorithm is used in the MAKFLTR and CMRSEN program, which is discussed later in this thesis.

A window is brought in as a digitized 32 x 32 element

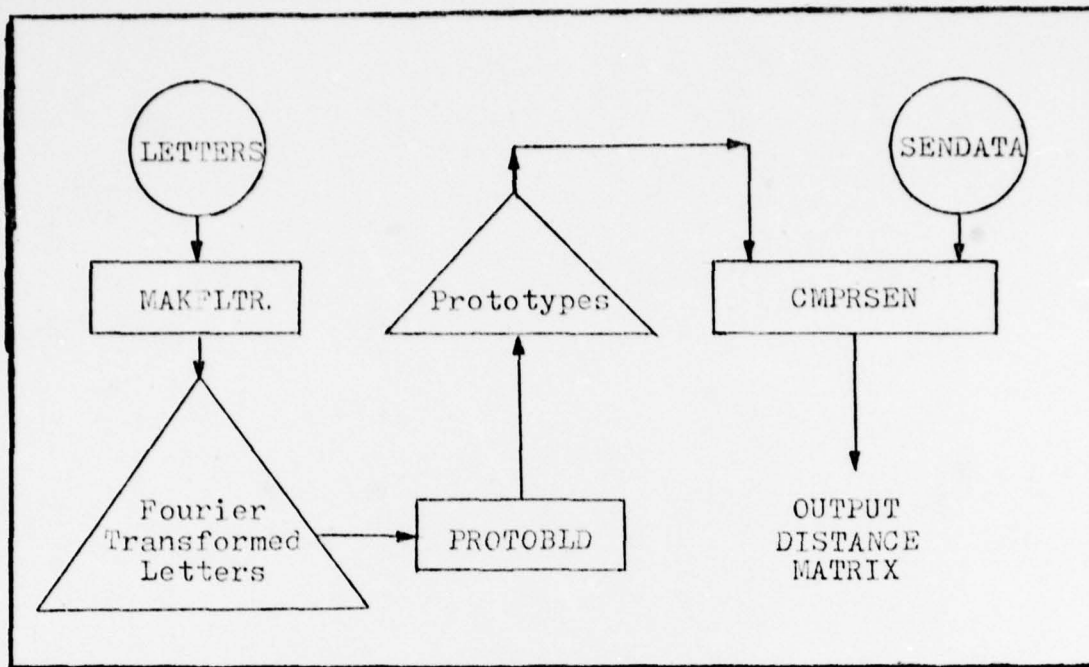


Fig 1. Interaction of Data Files and Computer Programs

array, consisting of zeros and ones. This array is then input to a discrete Fourier transform routine, FOURT (Ref 1), which obtains the two-dimensional Fourier transform of the input array. The discrete two-dimensional Fourier transform is calculated as follows:

$$O(r,s) = \sum_{k=1}^K \sum_{l=1}^L I(k,l) \exp \left[-j2\pi \left(\frac{kr}{K} + \frac{ls}{L} \right) \right]$$

where $j = \sqrt{-1}$, $I(k,l)$ = input array, and $O(r,s)$ = output array.

The array that is output from the FOURT routine is then further processed. First it is filtered in order to eliminate the spatial harmonics greater than the fourth (Ref 6). This allows reduction of the signal space to 81 - space.

The array is then energy normalized to insure that the energy of each vector is unity. Energy normalization is computed as follows:

$$N(r,s) = O(r,s) / \left[\sum_{r=1}^2 \sum_{s=1}^2 O^2(r,s) \right]^{1/2}$$

where $O(r,s)$ is output array from FOURT routine and $N(r,s)$ is the normalized array.

The normalized array is then reordered into a one-dimensional array. This new array contains the DC term followed by the real part of the first harmonic, the imaginary part of the first harmonic, the real part of the second harmonic, etc. This is the same format used for all data vectors.

The next step in the algorithm is to calculate the Euclidean distance between the input and a prototype. This is simply done using the following equation:

$$\text{Distance} = \left[\sum_{i=1}^{91} [W^2(i) - P^2(i)] \right]^{1/2}$$

where $W(i)$ = one-dimensional window array and $P(i)$ = one-dimensional prototype array. Thus when the input array is exactly the same as the prototype array the distance is equal to zero - or a perfect match.

Window Shapes

The three window functions used in this thesis are the unshaped (or full), trapezoidal and Gaussian shapes. These various functions are applied to windows in both the MAKFLTR and CMRSEN programs (see the Program section below).

Each of these window functions are calculated to obtain 32 normalized values, with the highest value being unity. The values are put into an array such that the unity values are centered within it. Then each of the 32 columns within the window are multiplied by the corresponding array value. This process causes a de-emphasis of the window edges, thus reducing the noise introduced by the neighboring letters. This process also reduces the signal but not enough to significantly change the recognition distances.

In the case of the unshaped (or full) window function the above described array contains all ones. This allows the window to be read in directly, with no change made. It is used only to see the effect the neighboring letters have on recognition distances.

In this thesis the trapezoid plateau length was varied in order to study window shape affects. The various values used for the plateau length were 1, 5, 10, 15, 20, 25, and 30 columns. Examples of these window functions are given in Figure 2.

The Gaussian window function was obtained using the following equation:

$$W(x) = \begin{cases} \exp[-(x-16)^2/2s^2] & x=1,2,\dots,16 \\ \exp[-(x-17)^2/2s^2] & x=17,18,\dots,32 \end{cases}$$

where $W(x)$ is the Gaussian window array, x is the column position along the window, and s is the standard deviation.

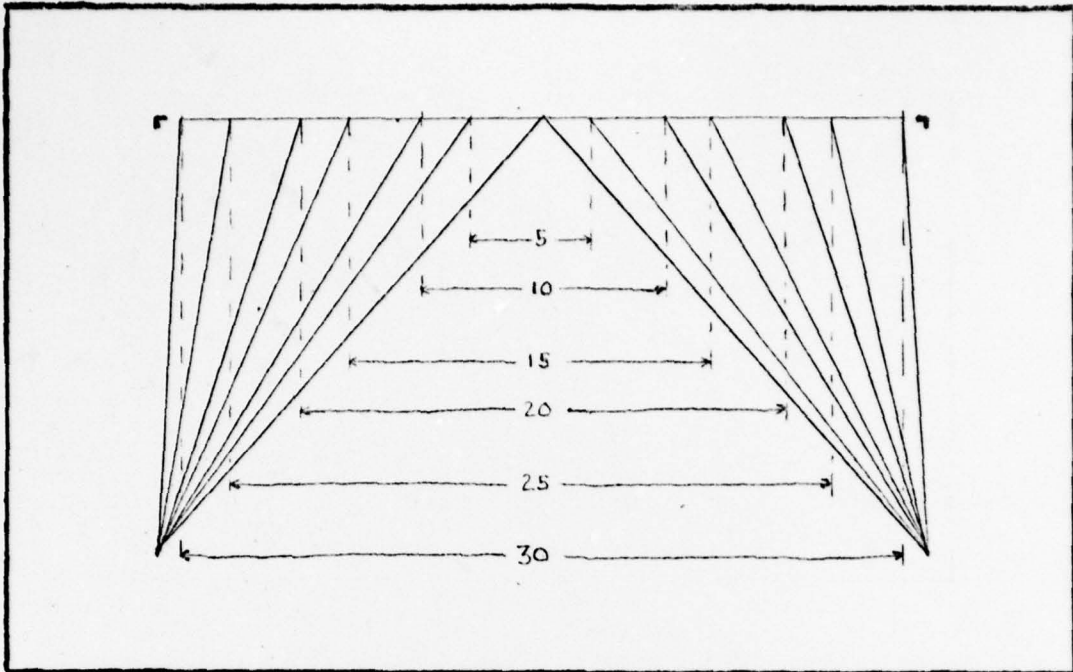


Fig 2. Various Trapezoidal Window Shapes

The range of standard deviations used were values of 2, 5, and 10 columns. Examples of these window functions are given in Figure 3.

The above described process is executed in the subroutine TRPMLT. This routine is listed in the MAKFLTR and CMPRSN programs (see Appendix C, pg. 69; Appendix E, pg. 101 respectively). It should be noted that these listings contain only the Gaussian logic, but any type window can be implemented by simply substituting the new function logic.

Data

LETTERS. This data file contains the digitized letters of 150 various font alphabets. Each of these stored letters are isolated and centered in a 32 x 32 element window.

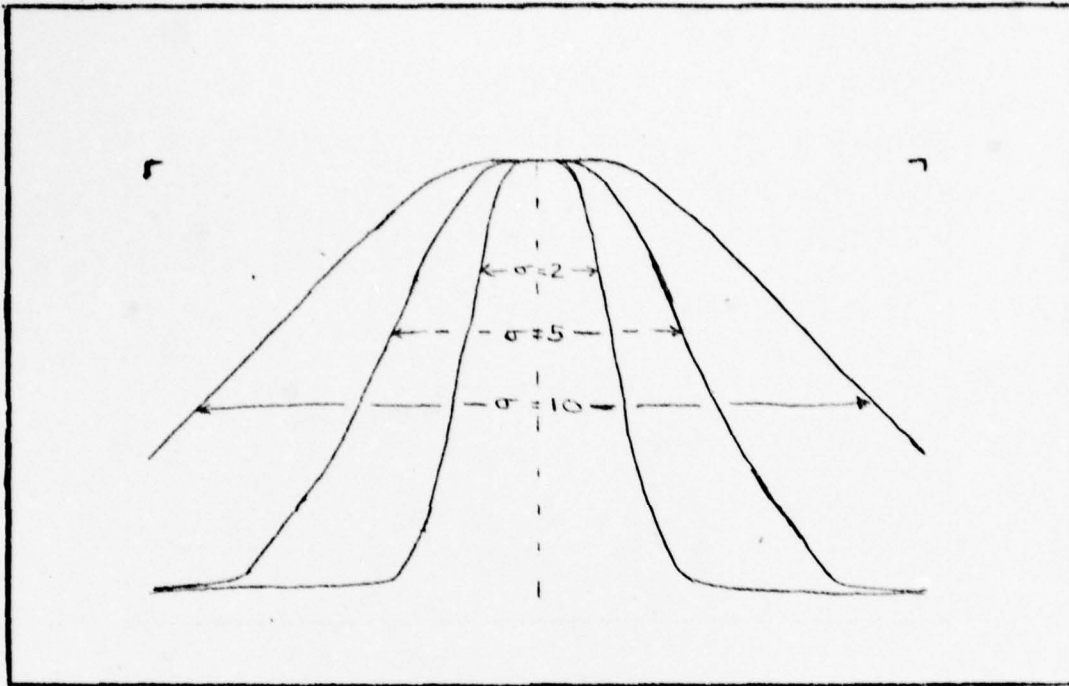


Fig 3 Various Gaussian Window Shapes

This data file is used by the MAKPLTR program discussed below. Some examples of the data in this file are shown in Appendix A.

SENDATA. This file contains various numbered sentences. Each sentence is 133 columns long and 32 rows high. These sentences consist of letters in a line, in the manner of an ordinary sentence but with no spacing between letters. Each letter within a sentence comes from a single alphabet so there is no mixing of fonts within the sentence.

This data file is used by the CMPRSEN program discussed below. A complete list of the sentences used, and their characteristics are shown in Appendix B.

Programs

MAKFLTR. This program produces the Fourier transformed letters necessary to build the prototypes. It applies a shaped window to each isolated letter in the data file LETTERS. This shaped letter is then Fourier transformed and stored in the format used by the PROTOBLD program discussed below. A complete listing of this program is included in Appendix C.

PROTOBLD. This program uses the file built by the MAKFLTR program. It uses the shaped Fourier letters to build average prototypes. The averaging process used is the summing of the Fourier components for each letter and dividing by the number of alphabets used. The formula used is as follows:

$$P(K) = \frac{1}{N} \sum_{i=1}^N L_i(K)$$

Where $L_i(K)$ is the K^{th} component of the Fourier representation of the i^{th} alphabet-version of a letter. $P(K)$ is the K^{th} component of the Fourier representation of the prototype.

This program also permits the user to specify which alphabets are to be used in the averaging process. A complete listing of this program is included in Appendix D.

CMRSEN. This program contains the recognition algorithm. It requires, as input, both the prototypes and a data sentence. The sentence is read by passing a 32 x 32 element window along the sentence. The input at each window position is run through the algorithms discussed in the previous sections.

These Euclidean distances are output from the CMRSEN

program in two formats. The first format is a list of the distance each prototype is from each window along the sentence. This allows easy observation of each distance between prototypes and the contents of the window. The second format is a listing of the best five choices (i.e. lowest distances) for each window position along the sentence.

There is an additional capability of using either complex or amplitude Fourier spectra. However the same Euclidean distance logic is used for either spectra. A complete listing of the program is included in Appendix E.

Results

Single Alphabet Prototypes

Two measurements are used to evaluate the performance of the recognition system. The first is the Euclidean distance to the correct choice. The second measurement is the figure of merit (FOM), which is defined as the ratio of the distances from the second choice to the first choice. Therefore, the FOM has a range of values greater than unity as long as the first choice is also the correct choice.

The first prototypes were built from a single, unshaped alphabet, identified as alphabet #10. This is the same alphabet used to construct sentences numbered one to four (see Appendix B, pg. 26). Thus we have perfect prototypes for each letter in alphabet #10; in the sense that the sentences are constructed from the same letters as the prototypes.

Using the above prototypes, sentence number one ("ABCDEFGH") was read in with the unshaped (or full) function being used. This allowed a standard for further comparisons. The only noise in the data, in this case, is provided by the adjacent letters in the scan windows. This sentence was also read in using the various trapezoidal-shaped function described in the previous section. The lowest recognition distances and highest FOM to the correct choices occurred when the trapezoid plateau length was equal to five columns.

New prototypes were then constructed with a trapezoidal

shape of plateau length equal to five columns and various trapezoidal shaped functions were applied to read sentence number one using these new prototypes. This time the best recognition distances occurred for trapezoidal function shape with plateau length of one column. These empirical results suggested a function shape with a sharper peak.

A Gaussian function was chosen next. The various Gaussian prototypes were each run through the different Gaussian window shapes. From these runs the best recognition occurred when the prototypes and window function had a standard deviation of two columns. The above results are summarized in Table I. Note especially the very low, virtually zero, distances to the correct prototype.

The data in Appendix F are the actual computer output of the five lowest distances for each window position along sentence number one. The three outputs are for both prototype and window standard deviations equal to 2, 5, and 10 columns. These data show how the recognition distances depend upon the standard deviations.

These prototypes are then run along the sentences numbered one to four with a Gaussian function shaped window of standard deviation of two columns. Tables II and III summarize the output from the complex and amplitude spectra, respectively. The tables give a listing of the lowest distance achieved by each prototype filter on each of the four sentences. It shows how the correct prototype distances fall to nearly zero. It should

TABLE I

Distance and Figure of Merit Data for Various Window Shapes on Sentence 1

Prototypes Window	A10			A10/T5			A10/G2		
	FULL		T5	T1		G2			
Letter	DX	FOM	DX	FOM	DX	FOM	DX	FOM	
B	.786	1.085	.376	1.495	.353	1.530	.000	und	
C	1.053	1.174	.536	1.664	.507	1.712	.000	und	
D	.798	1.189	.408	1.716	.366	1.861	.000	und	
E	.932	1.048	.524	1.389	.521	1.376	.001	413.	
F	.792	1.229	.450	1.798	.443	1.725	.004	103.	
G	.487	1.608	.303	2.327	.248	2.605	.000	und	

KEY

DX → distance to first choice FOM = Figure of Merit = $\frac{\text{Distance to second choice}}{\text{Distance to first choice}}$

An → A: Alphabet n = alphabet number used

Tm → T: Trapezoid m = length of trapezoid plateau

Cj → G: Gaussian j = standard deviation of Gaussian window

und = undefined

TABLE II

Complex Spectrum - Lowest Distance Achieved by Prototype Filters

Sentence Number	A	B	C	D	E	F	G	H	I	J	K	L	M
1	.638*	.000	.000	.000	.001	.004	.000	.541	.219	.330	.564	.763	.710
2	.590	.793	.519	.305	.672	.514	.392	.942*	.066	.000	.000	.824†	.000
3	.475	.556	.556	.652	.506	.434	.669	.637	.172	.366	.500	.761	.551
4	.490	.846	.701	.284	.777	.828	.635	.861	.687	.759	.428	.755	.402
	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	.789	.431	.412	.723	.506	.572	.183	.624	.439	.828	.628	.649	.699
2	.000	.000	.738	.741	.603	.635	.223	.689	.502	.735	.317	.515	.683
3	.550	.671	.421*	.000	.000	.014	.000	.000	.000	.366	.556	.409	.737
4	.317	.554	.876	.827	.830	.621	.710	.736	.295	.000	.000	.000	.000

Prototypes - Alphabet #10, Gaussian Shape (Standard Deviation = 2)

Window - Gaussian Shape (Standard Deviation = 2)

* Letter never centered in window

† Bad data letter

TABLE III

Amplitude Spectrum - Lowest Distance Achieved by Prototype Filters

Sentence Number	A	B	C	D	E	F	G	H	I	J	K	L	M
1	.246*	.000	.000	.000	.000	.002	.000	.355	.116	.131	.289	.416	.206
2	.204	.524	.207	.187	.467	.391	.211	.500*	.051	.000	.000	.395†	.000
3	.204	.366	.330	.244	.306	.259	.363	.358	.120	.105	.206	.352	.180
4	.222	.586	.112	.197	.536	.501	.298	.485	.399	.348	.212	.367	.164
	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	.358	.223	.251	.346	.309	.252	.092	.244	.258	.329	.290	.221	.420
2	.000	.000	.485	.275	.343	.461	.057	.330	.181	.336	.182	.230	.399
3	.326	.345	.213*	.000	.000	.012	.000	.000	.000	.249	.282	.209	.411
4	.272	.294	.554	.275	.439	.360	.345	.286	.133	.000	.000	.000	.000

Prototypes - Alphabet #10, Gaussian Shape (Standard Deviation = 2)

Window - Gaussian Shape (Standard Deviation = 2)

* Letter never centered in window

† Bad data letter

be noted that the way the sentence is read in causes the first letter of each sentence never to be centered in the window. This is the reason that the letters A, H and P do not drop to zero. This was not corrected in this project (by adding leading columns of zeros to the sentences) because of time considerations. The letter L, in sentence number two is incomplete because when the sentence was built the lower bar was omitted (see Appendix B, pg. 28). These data errors can be easily corrected once the sentence building program is restored to the computer files.

Multiple Alphabet Prototypes

The next step was to introduce multiple alphabet prototypes. The alphabets used were Kabler's numbers 10, 11, 12, 13, 14 and 15 (Ref 2). These are some of the alphabets (number 15 was excluded) that were used to construct the sentence data (see Appendix B). Note that with multiple alphabets an additional source of noise, really a "shape" noise, is added to the problem. As a consequence one would expect the performance of the system to be degraded.

In the single alphabet prototype case, it was found that the Gaussian shaped window gave perfect results. Therefore, the various Gaussian shaped prototypes were built from the six alphabets. Each of these prototypes were run along sentence number one with the various Gaussian functions applied. In this case, the best recognition (i.e. lowest distances, highest FOM) occurred when the

prototypes had a standard deviation of ten columns and the window had a standard deviation of five columns.

These best prototypes were then run along all 22 of the data sentences. Both the complex and amplitude spectra output was obtained. However, as expected, the distances in this case did not drop to zero.

Some examples of the raw data output from the above discussion is contained in Appendix G. These data are the lowest five distances for each window position along sentences numbered one to four. This Appendix contains the output for both the complex and amplitude spectra.

Acceptance thresholds were then set for each letter. This would allow a simple decision scheme to make the detection choice. Therefore, for each window, where the letter was relatively centered within the window, the distances were recorded for each of the 26 letters. This was repeated for each letter in every sentence until five distances (one for each alphabet), were obtained for each letter, and for each spectra. The highest number for each letter was then set as the acceptance threshold. Therefore, all letters were detailed by the threshold mechanism.

Using the Euclidean distances output for each prototype along the sentences, every number below that letter threshold was marked. When both spectra agreed (i.e. triggered at the same sentence position) the program declared that letter was in the window. But when this was done there were a large number of false triggers.

It was then observed that the letters from alphabet #11 were determining thresholds in most of the cases. These sentences were made of atypically thin letters. Thresholds were reset omitting this alphabet and there was a substantial decrease in the number of false triggers.

A summary of these false triggers is given in Table IV. Five letters (I, J, O, Q, W) were consistently falsely triggered. Other false triggers did occur for the different alphabets, and these are entered in the table.

TABLE IV
 Multiple Alphabet Prototypes - False Trigger Chart

Alphabet Number	Sentence Numbers	False Trigger Letters
10	1 - 4	A, B, C, N, R, T
12	10 - 13	C, D, H, L, U
13	14 - 17	D, H, R
14	18 - 22	A, B, E, L, R, S

Note: Letters I, J, O, Q, W missed in all alphabets

Conclusions and Recommendations

The single alphabet prototype results showed that a perhaps interesting fragment of the segmentation problem can be solved. Since the distances fall to nearly zero only on the correct letters, a simple threshold detector can be implemented. The detector can be used on either the complex or amplitude spectra.

The multiple alphabet prototype results turned out fairly well. The problem of false triggers that occurred could be attacked in a number of ways. First, and most obvious, is based on the fact that the prototypes were an average of the letters in six alphabets. These alphabets included an atypically thin font (alphabet #11) and a thick font (alphabet #14). This wide variation in font might very well account for many of the false triggers obtained. A method of ameliorating this problem is suggested below.

It would be possible to add special logic for the troublesome letters (i.e. I, J, O, Q, W). An example of this occurs when the B and I filters trigger at approximately the same location. Since the I filter would normally trigger in multiple neighboring windows, the special logic would trigger the B filter only.

Another example of special logic could be used in the following case. The recognition system triggers B, I, and E filters, within a small number of window locations. The additional logic would use the average column lengths of

B, I and E, taking into account the fact that an I could never fit between these letters, and override the I triggering.

There are other types of special logic that might be implemented. Perhaps logic based on English constraints could be included such as the rule that U always follows a Q. It is also possible to adaptively vary the window size and re-evaluate the decision computation when an ambiguity arises. However, to keep the generality of this process it is to be hoped the amount of special logic can be kept to a minimum.

The obvious next step in this project would be to go through the 150 alphabets and divide the fonts into separate categories according to the thickness of line strokes. Then prototypes for each letter could be built in these categories and tested against various sentences within the category. New thresholds could then be set as discussed previously and the results studied.

It is also necessary to develop a font detector for the overall recognition scheme. This device would enable the machine to decide the most suitable prototype to use for any input sentence. It is hoped the number of prototypes required would be considerably less than the total number of fonts read by the machine. For instance, by observation of the first Fourier transformed window of the sentence, the DC term could be used to decide the most appropriate line thickness prototype to use.

Another recommendation would be to investigate the behavior of this recognition system with arbitrary sequences

of letters within the sentence, since this thesis is concerned with the noise introduced by neighboring letters. However, the sentence data used in this thesis is of a set sequence of letters. For example, the letter C is always between letters B and D. Therefore the many combinations at different letter triplets should be investigated.

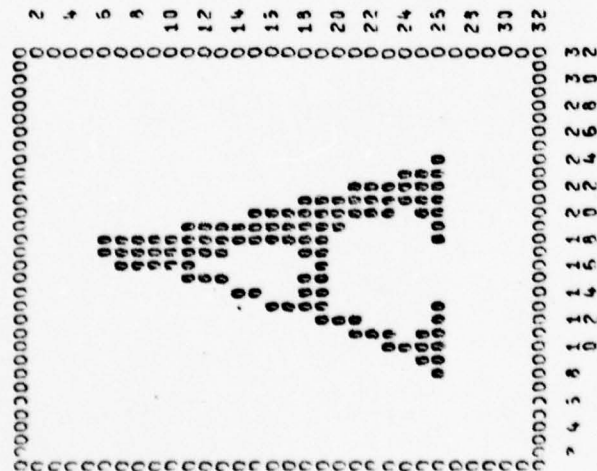
Bibliography

1. Haller, Mark. The Cooley-Tukey Fast Fourier Transform in USASI Basic Fortran. A Computer Program for the CDC 6600. Wright-Patterson Air Force Base, Ohio: ASD Computer Center, 1972.
2. Kabler, Paul W. Fourier Spatial Frequency Components as Descriptors for Recognition of Variable Font Type-script. Unpublished MS Thesis, Wright-Patterson Air Force Base, Ohio: Air Force Institute of Technology, December 1975.
3. McLachlan, Dan, Jr. "The Role of Optics in Applying Correlation Functions To Pattern Recognition", Journal of the Optical Society of America, 52: 454-459 (April 1962).
4. Pirich, Andrew R. A Survey of Optical Character Recognition (OCR) Technology. Technical Report, Griffis Air Force Base, New York: Rome Air Development Center, March 1973 (AD909342L).
5. Sponagle, Troy. Electrical Engineering Student (personal communication) Wright-Patterson Air Force Base, Ohio: Air Force Institute of Technology, January - March 1979.
6. Tallman, O. H. The Classification of Visual Images by Spatial Filtering. PHD Dissertation, Wright-Patterson Air Force Base, Ohio: Air Force Institute of Technology, June 1969 (AD858866).

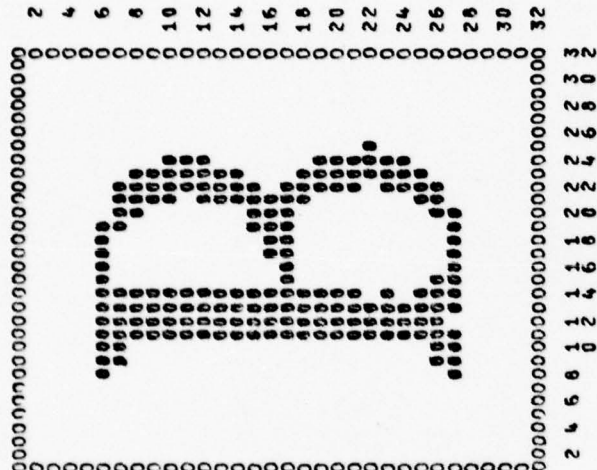
Appendix A

LETTERS

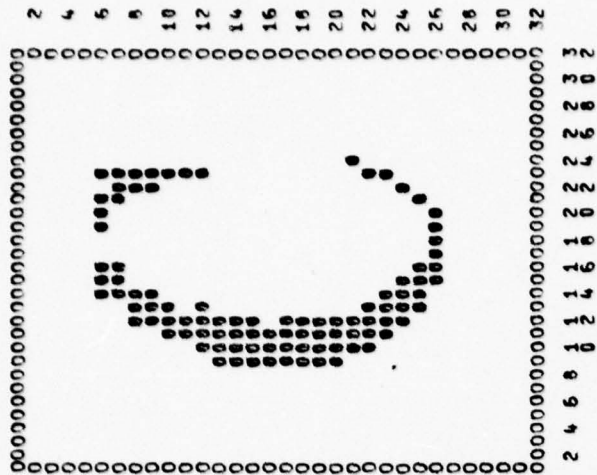
*** ALPHABET 10, LETTER 1 ***



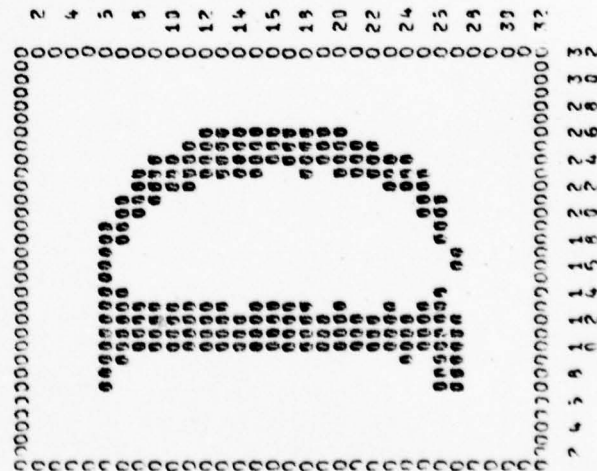
*** ALPHABET 10, LETTER 2 ***



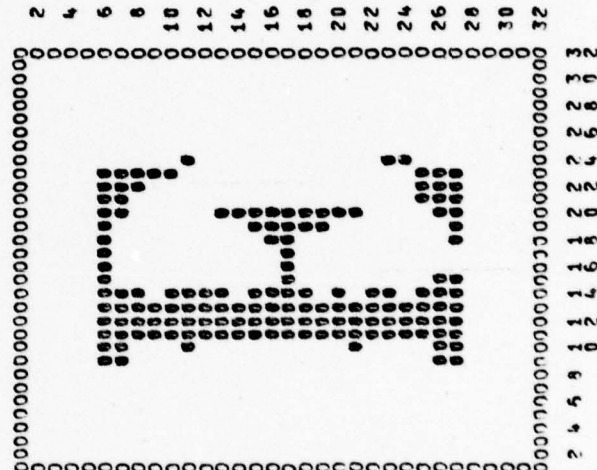
*** ALPHABET 10, LETTER 3 ***



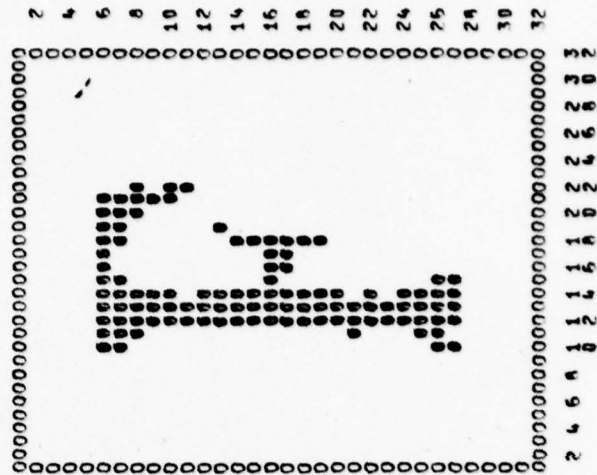
*** ALPHABET 10, LETTER 4 ***



*** ALPHABET 10, LETTER 5 ***



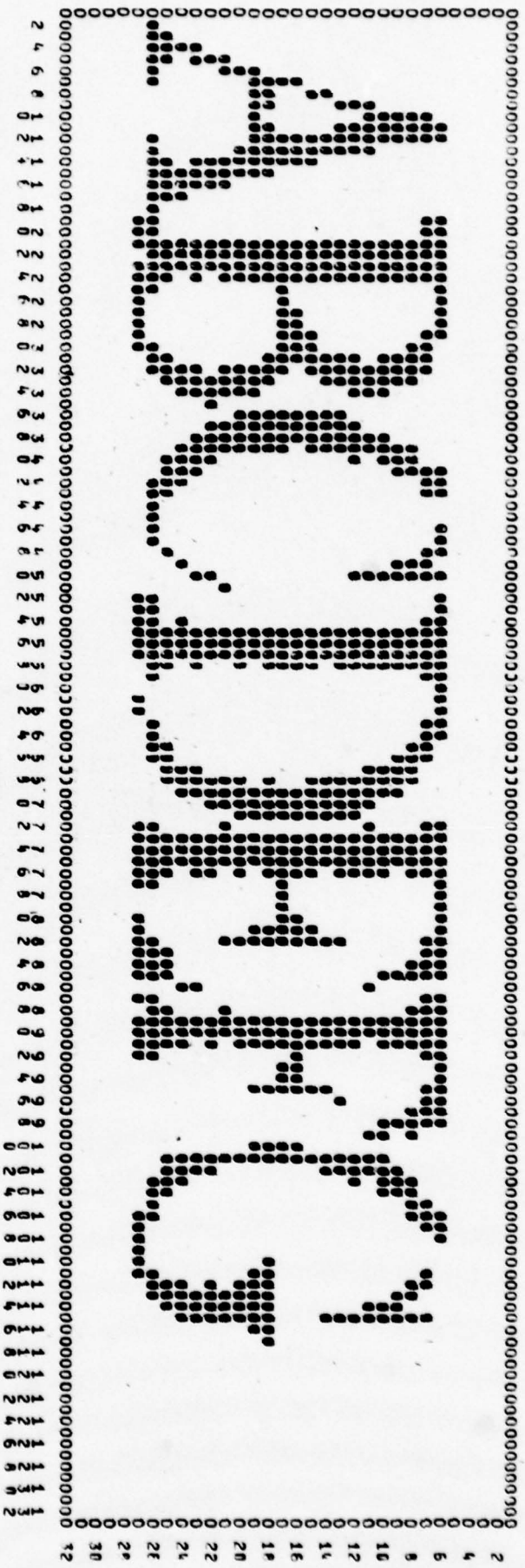
*** ALPHABET 10, LETTER 6 ***



Appendix B

SENDATA

CHARACTERISTICS OF SENTENCE DATA SENTENCE NUMBER 1

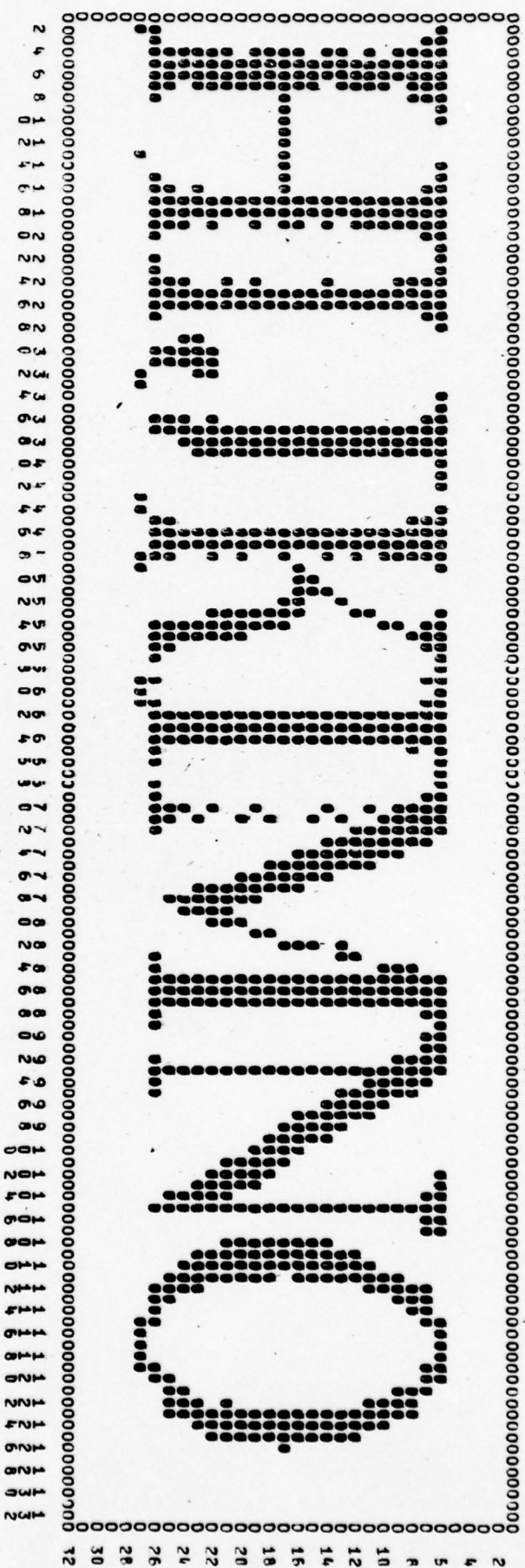


SENTENCE # 1 COLUMN 1 TO 33
LETTERS APPEARING IN THE SENTENCE ARE:

COLUMNS	WIDTH	ALPHABET	LETTER
2-10	10	S	S
11-17	7	U	U
18-21	4	N	N
22-24	3	S	S
25-27	3	H	H
28-30	3	I	I
31-33	3	E	E

2
4
5
6
9
10
12
14
16
18
20
22
24
26
28
30
32

CHARACTERISTICS OF SENTENCE DATA SENTENCE NUMBER 2

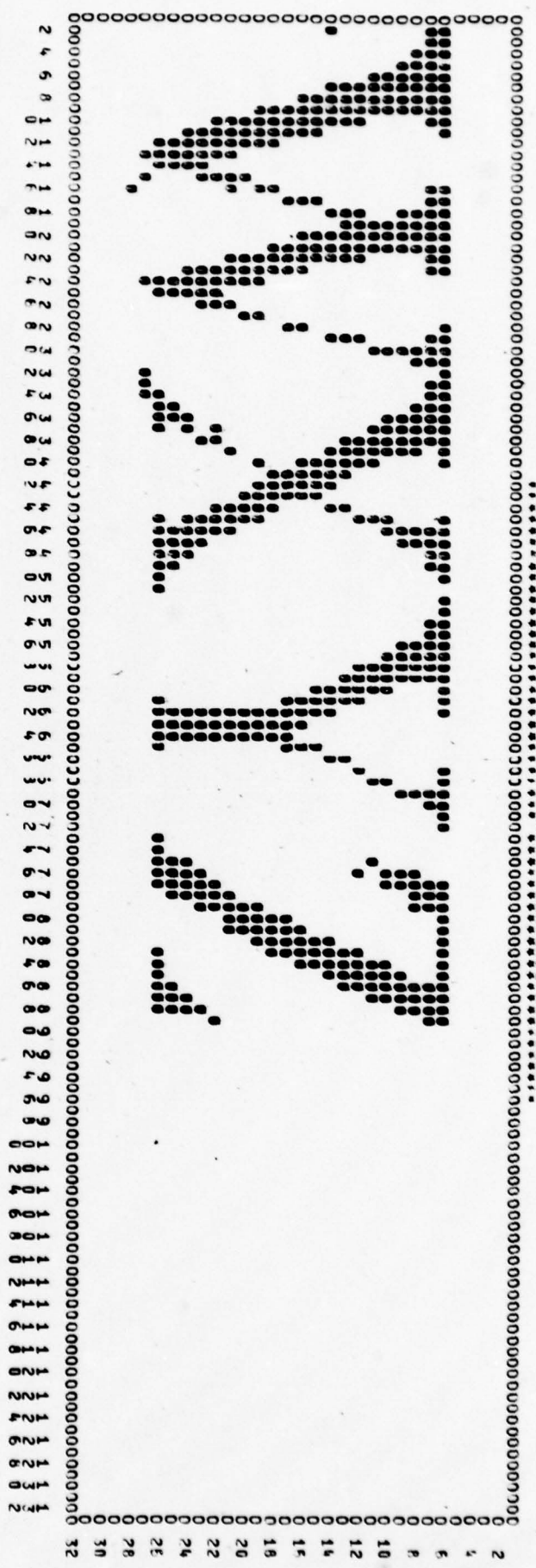


SENTENCE 4 2 COLUMN 1 TO 13
LETTERS APPEARING IN THE SENTENCE ARE:

SENTENCE	4	2	COLUMN	1	TO	13
2	21	20	7	10	9	9
23	42	10	10	10	10	11
43	59	17	8	10	11	12
66	67	22	10	10	13	13
88	89	16	10	10	14	14
90-107						
108-126						

COLUMNS WITH ALPHABET LETTER

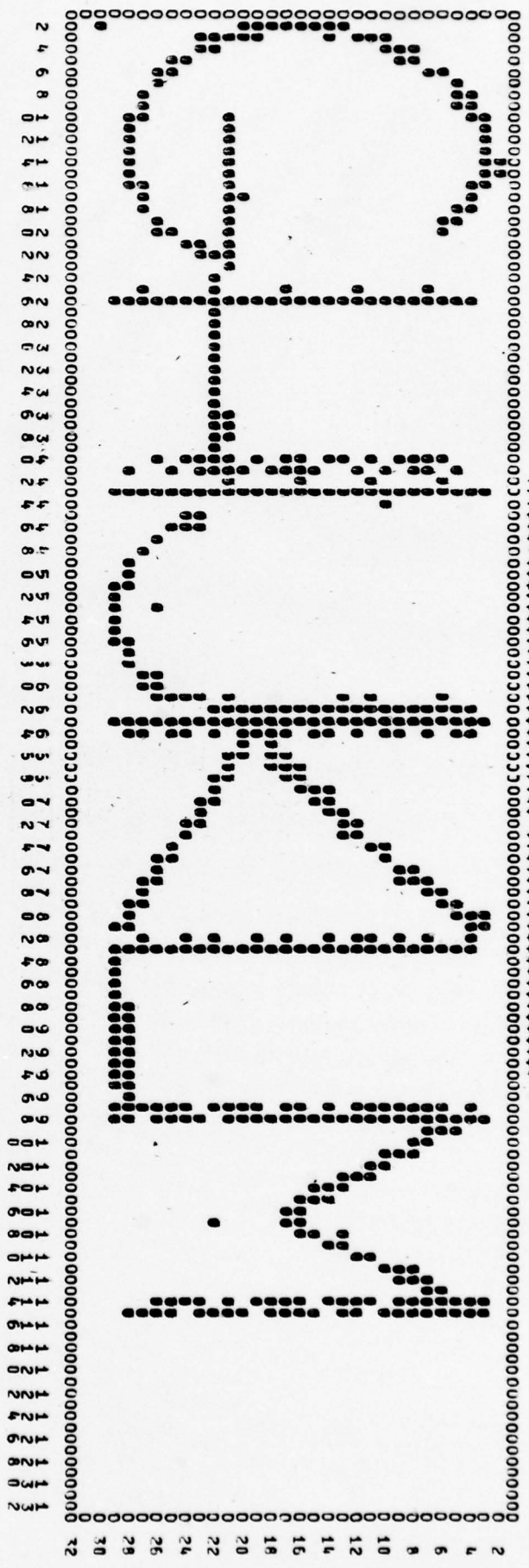
CHARACTERISTICS OF SENTENCE DATA SENTENCE NUMBER



SENTENCE APPEARING IN COLUMN IN THE SENTENCE ARE:

COLUMNS	WIDTH	ALPHABET	LETTER
2-31	30	10	23
32-72	20	10	24
73-86	14	10	25

 CHARACTERISTICS OF SENTENCE DATA: SENTENCE NUMBER 5.

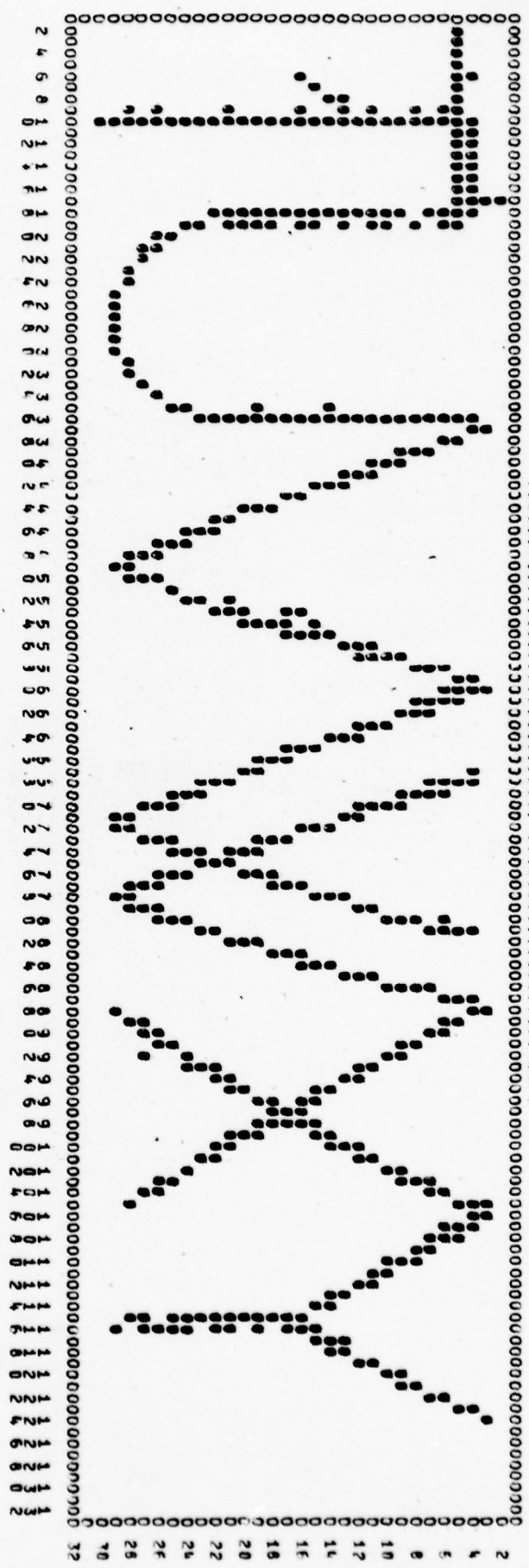


SENTENCE # 5 COLUMN IN THE SENTENCE ARE:

 COLUMNS WITH ALPHABET LETTER

26-27	22	11	7
23-24	19	11	6
25-26	18	11	9
27-28	15	11	10
29-30	13	11	11
31-32	11	11	12
33-34	11	11	13

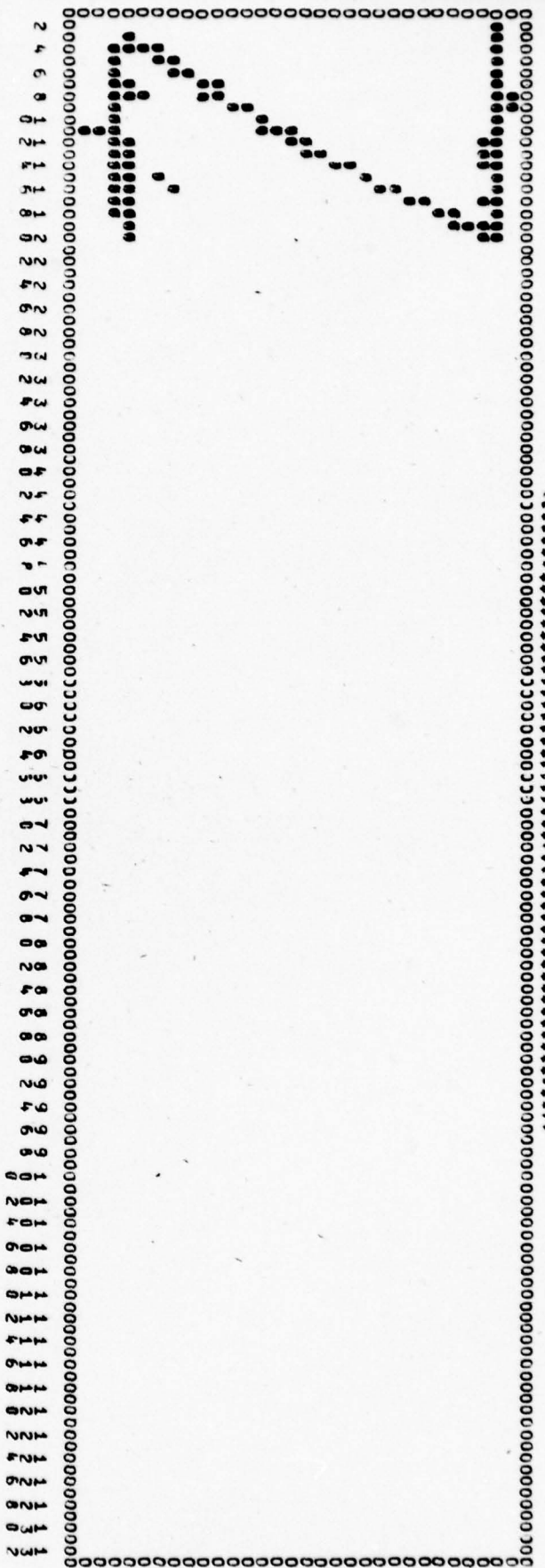
CHARACTERISTICS OF SENTENCE DATA SENTENCE NUMBER



SENTENCE APPEARING IN THE SENTENCE AREA

COLUMNS WITH ALPHABET LETTER	SENTENCE NUMBER
2-17	11
18-36	15
37-59	13
60-87	23
88-126	11
	19
	20
	21
	22
	23
	24
	25

CHARACTERISTICS OF SENTENCE DATA: SENTENCE NUMBER 9.

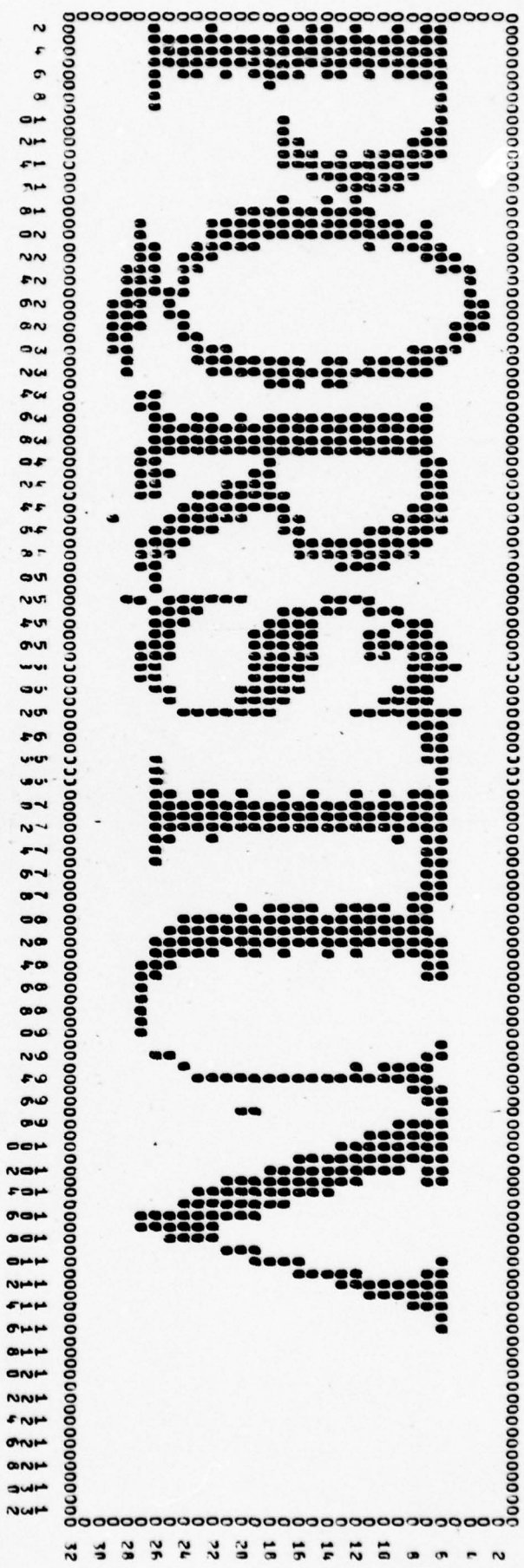


SENTENCE LENGTH COLUMN 1 TO 133
LETTERS APPEARING IN THE SENTENCE AREA

COLUMN WIDTH ALPHABET LETTER

2 - 20 19 11 26

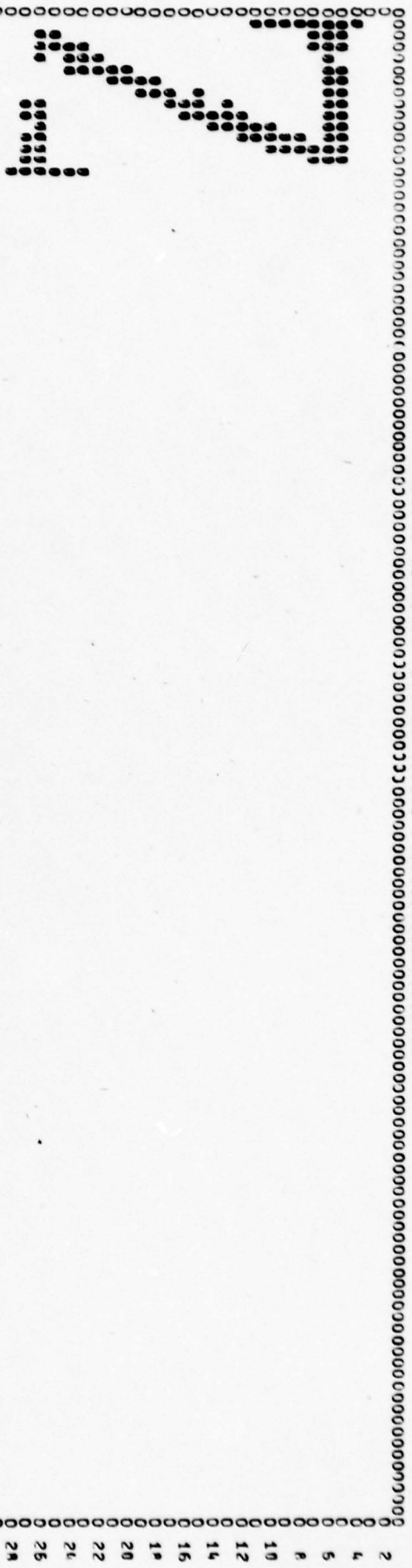
CHARACTERISTICS OF SENTENCE DATA: SENTENCE NUMBER 12.



SENTENCE # 12 COLUMN IN THE SENTENCE ARE:

COLUMNS	WIDTH	ALPHABET LETTER
1-16	16	16
17-33	17	16
34-51	18	16
52-68	17	16
69-85	17	20
86-102	17	21
103-119	17	22

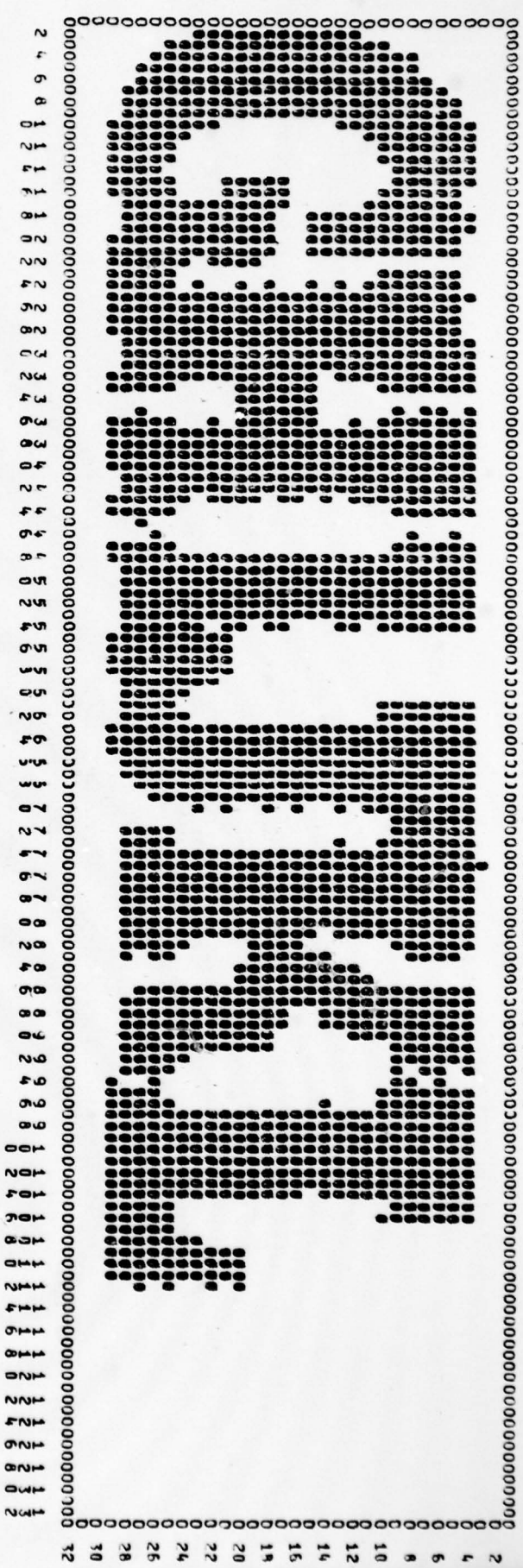
CHARACTERISTICS OF SENTENCE DATA: SENTENCE NUMBER 17.



SENTENCE 8 17 COLUMN 1 TO 17 LETTERS APPEARING IN THE SENTENCE ARE:

 COLUMNS WITH ALPHABET LETTER
 2 - 15 14 13 26

CHARACTERISTICS OF SENTENCE DATA SENTENCE NUMBER 13



SENTENCE # 19 COLUMN 1 TO 133
LETTERS APPEARING IN THE SENTENCE ARE:

COLUMNS	WIDTH	ALPHABET	LETTER
21	21	14	7
22	22	14	8
23	23	14	9
24	24	14	10
25	25	14	11
26	26	14	12

2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32

Appendix C

MAKFLTR

```

PROGRAM MAIN(INPUT,OUTPUT,TAPE3,TAPE4)
INTEGER ALPHSTR,ALPHSTP,LTRSTRT,LTRSTOP,LTRSIZE,NSPACE,TAPEIN,
%TAPEOUT $ LOGICAL ADJUST,RANDIN $ REAL WRATIO,HRATIO
INTEGER ENORM,DC
LOGICAL      LTR,VEXPAND,XFORM,IMAGE,FLTR,INVERSE,STORE
1  FORMAT (1HT)
LTRSIZE=32  $  NSPACE=81  $  TAPEIN=3  $  TAPEOUT=4
ALPHSTR=1   $  ALPHSTP=150  $  LTRSTRT=1  $  LTRSTOP=26
ADJUST=.F.  $  RANDIN=.T.  $  LTR=.F.  $  VEXPAND=.F.  $  XFORM=.F.
IMAGE=.F.   $  FLTR=.F.   $  INVERSE=.F.  $  STORE=.T.
ENORM=0  $  DC=0
WRITE 1
IF (RANDIN) CALL PREPLTR(TAPEIN)
CALL PREPFL(TAPEOUT)
CALL      FORTAPE(ALPHSTR,ALPHSTP,LTRSTRT,LTRSTOP,LTRSIZE,
% NSPACE,ADJUST,WRATIO,HRATIO,TAPEIN,RANDIN,TAPEOUT,LTR,
% VEXPAND,XFORM,IMAGE,FLTR,INVERSE,STORE,ENORM,DC)
STOP
END
SUBROUTINE FORTAPE(ALPHSTR,ALPHSTP,LTRSTRT,LTRSTOP,LTRSIZE,
% NSPACE,ADJUST,WRATIO,HRATIO,TAPEIN,RANDIN,TAPEOUT,LTR,
% VEXPAND,XFORM,IMAGE,FLTR,INVERSE,STORE,ENORM,DC)
INTEGER ALPHSTR,ALPHSTP,LTRSTRT,LTRSTOP,LTRSIZE,NSPACE,TAPEIN,
%TAPEOUT $ LOGICAL ADJUST,RANDIN $ REAL WRATIO,HRATIO
LOGICAL      LTR,VEXPAND,XFORM,IMAGE,FLTR,INVERSE,STORE
INTEGER LETTER(32,32),NUMLTRS,RECNUM
COMMON/LETTERS/LETTER,NUMLTRS,RECNUM
INTEGER RECFLTR
REAL FLTRLTR(180),PROTO(180)
COMMON/PROFLTR/FLTRLTR,PROTO,RECFLTR
COMMON/XFORMS/FORTLTR
COMPLEX FORTLTR(64,64)
INTEGER ALPHNUM,LTRNUM,          WIDTH,HEIGHT,NUMROWS,NUMCOLS
1  FORMAT (1H1)
3  FORMAT (1H,"ALPHABET ",I3," COMPLETE.")
WRITE 1
NUMWRDS=NSPACE+4
DO 110 ALPHNUM=ALPHSTR,ALPHSTP
DO 100 LTRNUM=LTRSTRT,LTRSTOP
CALL GETLTR(ALPHNUM,LTRNUM,TAPEIN,RANDIN)
IF (LTR)      CALL PRNTLTR(ALPHNUM,LTRNUM,.F.,.T.,.F.,.T.)
CALL XFORMIT(LTRSIZE,WIDTH,HEIGHT,ADJUST,WRATIO,HRATIO,NSPACE,
% NUMROWS,NUMCOLS,ALPHNUM,LTRNUM,VEXPAND,XFORM,IMAGE,INVERSE)
IF (XFORM)
%CALL PRNTCPX(FORTLTR,NUMROWS,NUMCOLS,1,NUMROWS,1,NUMCOLS,
%1,ALPHNUM ,LTRNUM,0,.T.,.T.)
CALL FILTER (FORTLTR,NUMROWS,NUMCOLS,FLTRLTR,NSPACE)
IF (FLTR) CALL PRNTFLR(ALPHNUM,LTRNUM,FLTELTR,NSPACE,3,.F.)
IF (ENORM.EQ.1) CALL ENORMFL (FLTRLTR,FLTRLTR,NSPACE,DC)
FLTRLTR(NSPACE+1)=WIDTH
FLTRLTR(NSPACE+2)=HEIGHT
FLTRLTR(NSPACE+3)=NUMROWS
FLTRLTR(NSPACE+4)=NUMCOLS
KEY=(ALPHNUM-1)*26+LTRNUM
IF (STORE) CALL WRITMS(TAPEOUT,FLTRLTR,NUMWRDS,KEY)
100 CONTINUE
IF (STORE) WRITE 3,(ALPHNUM)
110 CONTINUE
IF (.NOT.RANDIN) REWIND TAPEIN
C  IF (STORE) REWIND TAPEOUT
RETURN

```

```

SUBROUTINE ARRANGE (TEMP, NROWS, NCOLS, TAMP, NUMROWS, NUMCOLS)
COMPLEX TEMP(NROWS, NCOLS), TAMP(NUMROWS, NUMCOLS)
INTEGER NROWS, NCOLS, NUMROWS, NUMCOLS
DO 100 NCOL=1, NUMCOLS
DO 100 NROW=1, NUMROWS
TAMP(NROW, NCOL)=TEMP(NROW, NCOL)
100 CONTINUE
RETURN
END
SUBROUTINE ARRANGE (TEMP, NROWS, NCOLS, TAMP, NUMROWS, NUMCOLS)
INTEGER TEMP(NROWS, NCOLS), TAMP(NUMROWS, NUMCOLS)
INTEGER NROWS, NCOLS, NUMROWS, NUMCOLS
DO 100 NCOL=1, NUMCOLS
DO 100 NROW=1, NUMROWS
TAMP(NROW, NCOL)=TEMP(NROW, NCOL)
100 CONTINUE
RETURN
END
SUBROUTINE ENORMFL (FLTRLTR, NORMLTR, NSPACE, DC)
REAL FLTRLTR(NSPACE), NORMLTR(NSPACE)
INTEGER NSPACE, DC
INTEGER START, NVECTOR      $ REAL SUMSQRS
START=2
IF (DC.EQ.1) START=1
SUMSQRS=0.0
DO 100 NVECTOR=START, NSPACE
100 SUMSQRS=SUMSQRS+FLTRLTR(NVECTOR)**2
SUMSQRS=SUMSQRS**.5
IF (SUMSQRS.EQ.0.0) SUMSQRS=1.0
DO 110 NVECTOR=START, NSPACE
110 NORMLTR(NVECTOR)=FLTRLTR(NVECTOR)/SUMSQRS
IF (DC.EQ.0) NORMLTR(1)=1
RETURN
END
SUBROUTINE FILTER (FORTLTR, NUMROWS, NUMCOLS, FLTRLTR, NSPACE)
INTEGER NUMROWS, NUMCOLS, NSPACE
COMPLEX FORTLTR(NUMROWS, NUMCOLS)
REAL FLTRLTR(NSPACE)
INTEGER NHRMNC, NUMWRDS, RIGHT, LEFT, DOWN, NROW, NCOL
NHRMNC=NSPACE**.5
RIGHT=NHRMNC/2+1
LEFT=NUMCOLS-RIGHT+2
DOWN=RIGHT
FLTRLTR(1)=REAL(FORTLTR(1,1))
IF (NHRMNC.EQ.1) RETURN
NUMWRDS=0
DO 100 NCOL=2, RIGHT
NUMWRDS=NUMWRDS+2
FLTRLTR(NUMWRDS)=REAL(FORTLTR(1, NCOL))
100 FLTRLTR(NUMWRDS+1)=AIMAG(FORTLTR(1, NCOL))
DO 220 NROW=2, DOWN
DO 210 NCOL=LEFT, NUMCOLS
NUMWRDS=NUMWRDS+2
FLTRLTR(NUMWRDS)=REAL(FORTLTR(NROW, NCOL))
FLTRLTR(NUMWRDS+1)=AIMAG(FORTLTR(NROW, NCOL))
210 CONTINUE
DO 200 NCOL=1, RIGHT
NUMWRDS=NUMWRDS+2
FLTRLTR(NUMWRDS)=REAL(FORTLTR(NROW, NCOL))
FLTRLTR(NUMWRDS+1)=AIMAG(FORTLTR(NROW, NCOL))
200 CONTINUE
220 CONTINUE
RETURN
END
SUBROUTINE GETLTR (ALPHNUM, LTRNUM, TAPENUM, RANDOM)
INTEGER ALPHNUM, LTRNUM, TAPENUM
LOCAL RANDOM

```

```

COMMON/LOADMS/ALPHPAC
DATA RECNUM/1/
IF (.NOT.RANDOM) GO TO 7
LTRSTRT=(LTRNUM-1)*32+1
KEY=(ALPHNUM-1)*26+LTRNUM
CALL READMS(TAPENUM,ALPHPAC(LTRSTRT),32,KEY)
GO TO 60
7 CONTINUE
RECRQD=ALPHNUM
IF ((RECRQD-RECNUM).EQ.-1) GO TO 60
RECSKP=IABS(RECRQD-RECNUM)
IF (RECRQD-RECNUM) 10,30,20
10 DO 15 I=1,RECSKP
15 BACKSPACE TAPENUM
GO TO 30
20 DO 25 I=1,RECSKP
BUFFER IN (TAPENUM,1) (ALPHPAC(1),ALPHPAC(832))
CALL CHCKTPE(ALPHNUM,LTRNUM,TAPENUM)
25 CONTINUE
30 BUFFER IN (TAPENUM,1) (ALPHPAC(1),ALPHPAC(832))
RECNUM=RECRQD+1
CALL CHCKTPE(ALPHNUM,LTRNUM,TAPENUM)
60 LTRSTRT=(LTRNUM-1)*32
MASKER=MASK(1)
DO 80 NROW=1,32
TEMP=ALPHPAC(LTRSTRT+NROW)
DO 70 NCOL=1,32
IF (TEMP.AND.MASKER) 65,68
65 LETTER(NROW,NCOL)=1
GO TO 63
68 LETTER(NROW,NCOL)=0
69 TEMP=SHIFT(TEMP,1)
70 CONTINUE
80 CONTINUE
RETURN
END
SUBROUTINE PREPLTR(LTRTAPE)
INTEGER LTRTAPE
INTEGER LINDEK(3901)
CALL OPENMS(LTRTAPE,LINDEK,3901,0)
RETURN
END
SUBROUTINE PREPFL(FLRTAP)
INTEGER FLRTAP
INTEGER MINDEK(3901)
CALL OPENMS(FLRTAP,MINDEK,3901,0)
RETURN
END
SUBROUTINE REPLACE(TRPLTR,FORTLTR,NUMROWS,NUMCOLS)
INTEGER NUMROWS,NUMCOLS
DIMENSION TRPLTR(32,32)
COMPLEX FORTLTR(NUMROWS,NUMCOLS)
DO 100 NROW=1,NUMROWS
DO 100 NCOL=1,NUMCOLS
100 FORTLTR(NROW,NCOL)=TRPLTR(NROW,NCOL)
RETURN
END
SUBROUTINE SIZEIT(LTRSIZE,WIDTH,HEIGHT)
INTEGER LTRSIZE,WIDTH,HEIGHT
COMMON/LETTERS/LETTER,NUMLTRS,RECNUM
INTEGER LETTER(32,32),NUMLTRS,RECNUM
INTEGER NCOL,NROW,STRICOL,TEMP,STRIROW

```

```

DO 100 NCOL=1,32
DO 100 NROW=1,32
IF (LETTER(NROW,NCOL).EQ.1) GO TO 110
100 CONTINUE
110 STRTCOL=NCOL
DO 120 NCOL=1,32
TEMP=33-NCOL
DO 120 NROW=1,32
IF (LETTER(NROW,TEMP).EQ.1) GO TO 130
120 CONTINUE
130 WIDTH=TEMP-STRTCOL+1
IF (WIDTH.GE.1) GO TO 135
WIDTH=HEIGHT=0
RETURN
135 CONTINUE
DO 140 NROW=1,32
DO 140 NCOL=1,LTRSIZE
IF (LETTER(NROW,NCOL).EQ.1) GO TO 150
140 CONTINUE
150 STRTROW=NROW
DO 160 NROW=1,LTRSIZE
TEMP=LTRSIZE+1-NROW
DO 160 NCOL=1,LTRSIZE
IF (LETTER(TEMP,NCOL).EQ.1) GO TO 170
160 CONTINUE
170 HEIGHT=TEMP-STRTROW+1
RETURN
END
SUBROUTINE CPXREAL (REALARY,CPXARY,NUMROWS,NUMCOLS)
REAL REALARY(NUMROWS,NUMCOLS) $ INTEGER NUMROWS,NUMCOLS
COMPLEX CPXARY(NUMROWS,NUMCOLS)
DO 100 NROW=1,NUMROWS
DO 100 NCOL=1,NUMCOLS
100 REALARY(NROW,NCOL)=CPXARY(NROW,NCOL)
RETURN
END
SUBROUTINE FRMTNUM(FMT,A,B,C)
INTEGER FMT,A,B,C
GO TO (10,20,30,40,50),FMT
1 FORMAT (/ " FOURIER TRANSFORM OF ALPHABET ",I3," LETTER " ,I2,/)
10 WRITE 1,(A,B)
RETURN
2 FORMAT (/ " FOURIER TRANSFORM OF SENTENCE ",I2," STARTING AT COLUMN
% ",I3,/)
20 WRITE 2,(A,B)
RETURN
3 FORMAT (/ " FOURIER XFORM OF ALPHABET ",I2," LETTER ",I2,
% " FILTERED TO ",I3," SPACE",/)
30 WRITE 3,(A,B,C)
RETURN
4 FORMAT (" ALPHABET ",I3," LETTER ",I2,/)
40 WRITE 4,(A,B)
RETURN
50 CONTINUE
RETURN
END
SUBROUTINE GLPRT(M,N,A,STORE,NEG,NFORM,SKIPAGE)
DIMENSION DEN(20,7),A(M,N),STORE(N,7)
DIMENSION FMT1(1),FMT2(2),FMT3(2)
LOGICAL SKIPAGE
C THE MAIN PROGRAM MUST SUPPLY FMT1,FMT2,& FMT3
C FOR EXAMPLE IF M=32, USE THE FOLLOWING DATA CARDS IN MAIN PROGRAM.
C DATA FMT1/20H(1H ,32I2) /
C DATA FMT2,FMT3/10H(1H ,32A1),10H(1H+,32A1)/
C IF M IS LT 43, THE FOLLOWING CARD MAY BE USED. E.G. M=32 IS USED.
C DATA FMT1,20H(1H ,32I2) /

```

```

C      DATA FMT1/20H(1H,7X,6I2) /
C      NEG = 0 IS USED TO PRESENT THE MAX VALUE AS BLANK (WHITE);
C      WHILE NEG NOT = 0 IS TO PRESENT THE MAX VALUE AS BLACK.
C      THE CHOICE USUALLY DEPENDS UPON THE OPERATOR,S PREFERENCE
C      RECOMMEND NEG BE CHOSEN SUCH THAR THE BACKGROUND (MOST
C      OF THE FIELD) IS BLANK. IN OTHER WORDS, IF THE PICTURE IS
C      HIGH VALUES ON A LOW VALUE (BLACK) BACKGROUND, RECOMMEND
C      NEG NOT = 0. THIS WILL CAUSE THE HIGH VALUES TO BE BLACK
C      AND THE BACKGROUND TO BE BLANK.
      INTEGER NEG,NFORM
      INTEGER STORE,DEN
      INTEGER DUMMY

      DATA FMT1/10H(1H ,64I2)/
      DATA FMT2/20H(1H ,"#",=(A1),"#") /
      DATA FMT3/20H(1H+,1X ,=(A1)) /
10     FORMAT (1H ,=( "#"))
20     FORMAT (1H0)
      DATA (DEN(1,J),J=1,7)/7(1H ) /
      DATA (DEN(2,J),J=1,7)/1H-,6(1H )/
      DATA (DEN(3,J),J=1,7)/1H=,6(1H )/
      DATA (DEN(4,J),J=1,7)/1H+,6(1H )/
      DATA (DEN(5,J),J=1,7)/1H),6(1H )/
      DATA (DEN(6,J),J=1,7)/1H1,6(1H )/
      DATA (DEN(7,J),J=1,7)/1HZ,6(1H )/
      DATA (DEN(8,J),J=1,7)/1HX,6(1H )/
      DATA (DEN(9,J),J=1,7)/1HA,6(1H )/
      DATA (DEN(10,J),J=1,7)/1HM,6(1H )/
      DATA (DEN(11,J),J=1,7)/1H0,1H-,5(1H )/
      DATA (DEN(12,J),J=1,7)/1H0,1H=,5(1H )/
      DATA (DEN(13,J),J=1,7)/1H0,1H+,5(1H )/
      DATA (DEN(14,J),J=1,7)/1H0,1H),5(1H )/
      DATA (DEN(15,J),J=1,7)/1H0,1H+,1H.,4(1H )/
      DATA (DEN(16,J),J=1,7)/1H0,1H+,1H.,1H=,3(1H )/
      DATA (DEN(17,J),J=1,7)/1H0,1HX,1H.,1H-,2(1H )/
      DATA (DEN(18,J),J=1,7)/1H0,1HX,1H.,1HH,1H0,2(1H )/
      DATA (DEN(19,J),J=1,7)/1H0,1HX,1H.,1HH,1HB,2(1H )/
      DATA (DEN(20,J),J=1,7)/1H0,1HX,1H.,1HH,1HB,1HV,1HA/
      DMAX=A(1,1)
      DMIN=A(1,1)
      DO 100 I=1,M
      DO 100 J=1,N
      DMAX=AMAX1(DMAX,A(I,J))
      DMIN=AMIN1(DMIN,A(I,J))
100    CONTINUE
      WRITE 20
      IF (SKIPAGE) PRINT 1
      PRINT 4
      IF (NFORM .EQ. 0) GO TO 200
      WRITE 10,(N+2)
      DO 120 I=1,M
      L = 0
      DO 130 J=1,N
      IF (NEG.EQ.0) DUMMY = 20. - 19.*(A(I,J)-DMIN)/(DMAX-DMIN)
      IF (NEG.NE.0) DUMMY = 1. + 19.*(A(I,J)-DMIN)/(DMAX-DMIN)
      L = MAX0( DUMMY, L )
      IF (DUMMY.GT.20) DUMMY=20
      IF (DUMMY.LT.1) DUMMY=1
      DO 140 K=1,7
140     STORE(J,K)=DEN(DUMMY,K)
130    CONTINUE
      PRINT FMT2,N,(STORE(J,1),J=1,N)
      IF ( L .LE. 10 ) GO TO 120
      PRINT FMT3,N,(STORE(J,2),J=1,N)
      IF ( L .LE. 14 ) GO TO 120
      PRINT FMT3,N,(STORE(J,3),J=1,N)
      IF ( L .LE. 15 ) GO TO 120

```

```

PRINT FMT3,N,(STORE(J,4),J=1,N)
IF ( L .LE. 16 ) GO TO 120
PRINT FMT3,N,(STORE(J,5),J=1,N)
IF ( L .LE. 19 ) GO TO 120
PRINT FMT3,N,(STORE(J,6),J=1,N)
PRINT FMT3,N,(STORE(J,7),J=1,N)
120 CONTINUE
WRITE 10,(N+2)
GO TO 225
200 DO 220 I=1,M
DO 210 J=1,N
IF (NEG.EQ.0) DUMMY=99.-98.*(A(I,J)-DMIN)/(DMAX-DMIN)
IF (NEG.NE.0) DUMMY= 1.+98.*(A(I,J)-DMIN)/(DMAX-DMIN)
IF (DUMMY .GT. 99) DUMMY=99
IF (DUMMY .LT. 1 ) DUMMY=1
210 STORE(J,1)=DUMMY
PRINT FMT1,(STORE(J,1),J=1,N)
220 CONTINUE
225 PRINT 3
1 FORMAT (1H1)
3 FORMAT (1H )
4 FORMAT (1HT)
RETURN
END
SUBROUTINE IVRTPX(LETTER,NUMROWS,NUMCOLS)
COMPLEX LETTER(NUMROWS,NUMCOLS)
INTEGER NUMROWS,NUMCOLS
REAL TEMP1,TEMP2
DO 100 NROW=1,NUMROWS
DO 100 NCOL=1,NUMCOLS
J=NROW+NCOL
IF (MOD(J,2).NE.1) GO TO 100
TEMP1=REAL(LETTER(NROW,NCOL))*(-1.0)
TEMP2=AIMAG(LETTER(NROW,NCOL))*(-1.0)
LETTER(NROW,NCOL)=CMPLX(TEMP1,TEMP2)
100 CONTINUE
RETURN
END
SUBROUTINE PRNTCPX(ARRAY,ROWSIN,COLSIN,STRTROW,STOPROW,STRTCOL,
% STOPCOL,FMT,A,B,C,SEQHRMC,SKIPAGE)
INTEGER ROWSIN,COLSIN,STRTROW,STOPROW,STRTCOL,STOPCOL,=FMT,A,B,C
COMPLEX ARRAY(ROWSIN,COLSIN)
LOGICAL SEQHRMC,SKIPAGE
INTEGER ROWSTRT,ROWSTOP,COLSTRT,COLSTOP,TEMP,NPAGE,NSPLITS,HALF,
%CINDEX(50),RINDEX(50)
1 FORMAT (1HT)
5 FORMAT (1H1)
10 FORMAT (/ ,5X, =(2X,I3,3X))
20 FORMAT (1H0,I3,2X,=(F7.3,1X))
30 FORMAT (1H ,5X, =(F7.3,1X))
40 FORMAT (/)
50 FORMAT (1H ,"ARRAY PRINTED IS FROM ROW ",I3," TO ",I3," COLUMN ",
% I3," TO ",I3)
70 FORMAT (1H ," INDEXING REQUESTED IS STRAIGHT SEQUENTIAL."//)
80 FORMAT (1H ," INDEXING REQUESTED IS HARMONIC NUMBER."//)
WRITE 1
NUMCOLS=16
IF (.NOT.SEQHRMC) GO TO 120
DO 100 NCOL=STRTCOL,STOPCOL
100 CINDEX(NCOL)=NCOL
DO 110 NROW=STRTROW,STOPROW
110 RINDEX(NROW)=NROW
GO TO 170
120 CONTINUE
TEMP=STOPROW-STRTROW+1
HALF=TEMP/2+STRTROW

```

```

DO 130 NROW=STRTRW,HALF
RINDEX(NROW)=TEMP
TEMP=TEMP-2
130 CONTINUE
HALF=HALF+1 $ TEMP=TEMP+2
DO 140 NROW=HALF,STOPROW
TEMP=TEMP+2
RINDEX(NROW)=TEMP
140 CONTINUE
TEMP=STOPCOL-STRTCOL+1
HALF=TEMP/2+STRTCOL
DO 150 NCOL=STRTCOL,HALF
CINDEX(NCOL)=TEMP
TEMP=TEMP-2
150 CONTINUE
HALF=HALF+1 $ TEMP=TEMP+2
DO 160 NCOL=HALF,STOPCOL
TEMP=TEMP+2
CINDEX(NCOL)=TEMP
160 CONTINUE
170 CONTINUE
NSPLITS=(STOPCOL-STRTCOL)/NUMCOLS+1
COLSTRT=STRTCOL
DO 220 NPAGE=1,NSPLITS
COLSTOP=COLSTRT+NUMCOLS-1
IF (NSPLITS.EQ.NPAGE) COLSTOP=STOPCOL
TEMP=COLSTOP-COLSTRT+1
IF (SKIPAGE) WRITE 5
WRITE 50,(STRTRW,STOPROW,COLSTRT,COLSTOP)
IF (SEQHRMC) 180,190
180 WRITE 70
GO TO 200
190 WRITE 80
200 CONTINUE
CALL FRMNUM(FMT,A,B,C)
WRITE 10,(NUMCOLS,(CINDEX(NCOL),NCOL=COLSTRT,COLSTOP))
DO 210 NROW=STRTRW,STOPROW
WRITE 20,(RINDEX(NROW),TEMP,(REAL(ARRAY(NROW,NCOL)),NCOL=COLSTRT,
% COLSTOP))
WRITE 30,(TEMP,(AIMAG(ARRAY(NROW,NCOL)),NCOL=COLSTRT,COLSTOP))
210 CONTINUE
COLSTRT=COLSTOP+1
WRITE 40
220 CONTINUE
RETURN
END
SUBROUTINE PRNTEXP(FORTLTR,NUMROWS,NUMCOLS,ALPHNUM,LTRNUM,
% WRATIO,HRATIO,WIDTH,HEIGHT)
COMPLEX FORTLTR(NUMROWS,NUMCOLS)
INTEGER ALPHNUM,LTRNUM,WIDTH,HEIGHT,NUMCOLS,NUMROWS
REAL WRATIO,HRATIO
1 FORMAT(1H0,"EXPANSION OF ALPHABET "I3" LETTER "I2/1H "WIDTH RATIO=
%"F6.2" HEIGHT RATIO="F6.2,/,"1H "LETTER SIZE IS "I2" COLUMNS WIDE
%BY " I2" ROWS HIGH"/1H ,"WINDOW SIZE IS "I2" COLUMNS WIDE BY "I2
%" ROWS HIGH"///
%,=(1H ,=(I1)/)
2 FORMAT(1HT)
WRITE 2
WRITE 1,(ALPHNUM,LTRNUM,WRATIO,HRATIO,
%WIDTH,HEIGHT,NUMCOLS,NUMROWS,NUMROWS,(NUMCOLS,(IFIX(REAL(FORTLTR
%(NROW,NCOL))) ,NCOL=1,NUMCOLS),NROW=1,NUMROWS))
RETURN
END
SUBROUTINE PRNTPFLR (ALPHNUM,LTRNUM,FLTRLTR,NSPACE,FMT,SKIPAGE)
INTEGER ALPHNUM,LTRNUM,NSPACE,FMT $ LOGICAL SKIPAGE
REAL FLTRLTR(NSPACE)

```

```

INTEG ER NHRMNC S, NZER OES, DOWN, INDEX, NROW, START, STOP
REAL EMPTY(7)
INTEG ER INDEX(13)
1  FORMAT (1HT)
5  FORMAT (1H1)
10  FORMAT (/ , 5X, 13(4X, I2, 4X))
20  FORMAT (1H- , I2, 3X, 13(F9.5, 1X))
30  FORMAT (1H , 5X, 13(F9.5, 1X))
40  FORMAT (/)
50  FORMAT (/ " NSPACE=" , I3, " WHICH IS INCORRECT (NOT AN ODD NUMBER) F
%OR ALPHABET " , I3, " LETTER " , I2, /)
WRITE 1
IF (MOD(NSPACE , 2) .EQ. 1) GO TO 90
WRITE 50, (NSPACE, ALPHNUM, LTRNUM)
RETURN
90  CONTINUE
IF (SKIPAGE) WRITE 5
DATA (INDEX(I), I=1, 13)/13, 11, 9, 7, 5, 3, 1, 3, 5, 7, 9, 11, 13/
DATA (EMPTY(I), I=1, 7)/7*0.0/
CALL FRMTNUM(FMT, ALPHNUM, LTRNUM, NSPACE)
NHRMNC S=NSPACE** .5
START=(13-NHRMNC S)/2+1
STOP=START+NHRMNC S-1
WRITE 10, (INDEX(I), I=START, STOP)
DOWN=NHRMNC S/2+1
NZER OES=NHRMNC S/2
INDX=1
WRITE 20, (INDX , (EMPTY(I), I=1, NZER OES), FLTRLTR(1),
% (FLTRLTR(NVECTOR), NVECTOR=2, NHRMNC S, 2))
NZER OES=NZER OES+1
WRITE 30, ( (EMPTY(I), I=1, NZER OES), (FLTRLTR(NVECTOR),
% NVECTOR=3, NHRMNC S, 2))
START=NHRMNC S+1
DO 100 NROW=2, DOWN
INDX =INDX +2 $ STOP=START+NHRMNC S* 2-1
WRITE 20, (INDX, (FLTRLTR(NVECTOR), NVECTOR=START, STOP, 2))
START=START+1
WRITE 30, ( FLTRLTR(NVECTOR), NVECTOR=START, STOP, 2)
START=STOP+1
100 CONTINUE
WRITE 40
RETURN
END
SUBROUTINE PRNTIMG(FORTLTR, NUMROWS, NUMCOLS, ALPHNUM, LTRNUM)
COMPLEX FORTLTR(NUMROWS, NUMCOLS) $ INTEG ER NUMROWS, NUMCOLS
INTEG ER ALPHNUM, LTRNUM
REAL STORE(64, 7)
REAL BRIGHT(64, 64)
30  FORMAT (1H "FOURIER TRANSFORM OF ALPHABET "I3" LETTER "I2" CONVERTE
% TO BRIGHTNESS LEVELS"/ , 1H "ROWS="I3" COLUMNS="I3)
DO 92 NROW=1, NUMROWS
DO 92 NCOL=1, NUMCOLS
92  BRIGHT (NROW, NCOL)=(REAL (FORTLTR (NROW, NCOL))**2+AIMAG (FORTLTR (NROW,
% NCOL))**2)** .5
CALL ARRANGE (BRIGHT, 64, 64, BRIGHT, NUMROWS, NUMCOLS)
CALL GLPRT (NUMROWS, NUMCOLS, BRIGHT, STORE, 1, 1, .F.)
WRITE 30, (ALPHNUM, LTRNUM, NUMROWS, NUMCOLS)
RETURN
END
SUBROUTINE PRNTLTR(ALPHNUM, LTRNUM, DUMPIT, PRINTIT, ALPHSKP, SKIPAGE)
INTEG ER ALPHNUM, LTRNUM
LOGICAL DUMPIT, PRINTIT, ALPHSKP, SKIPAGE
INTEG ER LETTER(32, 32), NUMLTRS, RECNUM
COMMON /LETTERS/ LETTER, NUMLTRS, RECNUM
INTEG ER ADJCOL, NUMROW, NUMCOL, COLSIZE, NUMPAGE, OLDALPH, TITLE(70)
INTEG ER KEEPER(32, 100), KEEPER1(32, 100)

```

```

1   FORMAT (1HT)
98  FORMAT (1H+, 3(21X,I3,9X,I2,7X))
99  FORMAT (/)
100 FORMAT (/ ,1X,3(8X,3A1,1X,8A1,1X,3X,A1,1X,6A1 ,1X,2X,1X,3A1,3X))
101 FORMAT (1H ,3(7X,33A1,2X))
102 FORMAT (1H+,3(7X,32A1,1X,I2))
103 FORMAT (/ ,1X,3(7X,16(1X,I1),3X))
104 FORMAT (1H1)
106 FORMAT (1X,3(7X,4(1X,A1),12(1X,I1),3X))
107 FORMAT (1H+,3(7X,33A1,2X))
    DATA NUMLTRS,NUMPAGE/0,0/
    WRITE 1
    IF (NUMLTRS.EQ.0) OLDALPH=ALPHNUM
    IF (SKIPAGE) NUMPAGE=0
    IF ((NUMLTRS.EQ.0).AND.(NUMPAGE.EQ.0).OR .SKIPAGE) WRITE 104
    IF (PRINTIT) GO TO 105
    IF (DUMPIT.OR.(ALPHSKP.AND.(OLDALPH.NE.ALPHNUM))) 120,105
105  NUMLTRS=NUMLTRS + 1
    ADJCOL=33*(NUMLTRS-1) +1
    COLSIZE=ADJCOL+31
    DO 117 NUMROW=1, 32
    U=0
    DO 116 NUMCOL=ADJCOL,COLSIZE
    U=J+1
    IF (LETTER(NUMROW,J ).EQ.0) 115,110
110  KEEPER (NUMROW,NUMCOL)="#"
    KEEPER1(NUMROW,NUMCOL)="0"
    GO TO 116
115  KEEPER (NUMROW,NUMCOL)=KEEPER1(NUMROW,NUMCOL)=" "
116  CONTINUE
117  CONTINUE
    TITLE(NUMLTRS*2+63)=LTRNUM
    TITLE(NUMLTRS*2+62)=ALPHNUM
    IF (PRINTIT) GO TO 120
    IF (NUMLTRS.LT.3) RETURN
120  IF (NUMLTRS.EQ.0) RETURN
    COLSIZE=NUMLTRS*33
    DO 140 I=1,32
    DO 130 J=1,COLSIZE,33
130  KEEPER1(I,J)="0"
    DO 135 J=32,COLSIZE,33
135  KEEPER1(I,J)="0"
140  CONTINUE
    DO 145 I=1,32,31
    DO 145 J=1,COLSIZE
145  KEEPER1(I,J)="0"
    DO 165 I=2,32,2
    DO 165 J=33,COLSIZE,33
    KEEPER (I,J)=" "
165  KEEPER1(I,J)=I
    DO 170 I=1,32,2
    DO 170 J=33,COLSIZE,33
170  KEEPER (I,J)=KEEPER1(I,J)=" "
    DATA (TITLE(J),J=1,63)/3* (" ", " ", " ", "A", "L", "P", "4", "A", "B",
% "E", "T", " ", " ", "L", "E", "T", "T", "E", "R", " ", " ", " ", " ") /
    DATA (KEEPER1(33,NUMCOL),NUMCOL=2,96,2)/3* (2,4,6,8,1,1,1,1,1,2,2,
% 2,2,2,3,3) /
    DATA (KEEPER1(34,NUMCOL),NUMCOL=2,96,2)/3* (" ", " ", " ", " ", " ", 0,2,4,
% 6,8,0,2,4,6,8,0,2) /
    COLSIZE=NUMLTRS*21
    WRITE 100,(TITLE(NUMCOL),NUMCOL=1,COLSIZE)
    COLSIZE=NUMLTRS*2+63
    WRITE 98,(TITLE(I),I=64,COLSIZE)
    WRITE 99
    COLSIZE=NUMLTRS*33
    DO 180 I=1,32

```

```

WRITE 101, (KEEPER (I, J), J=1, COLSIZE)
IF (MOD(I, 2).EQ.0) GO TO 175
WRITE 107, (KEEPER1(I, J), J=1, COLSIZE)
GO TO 180
175 WRITE 102, (KEEPER1(I, J), J=1, COLSIZE)
180 CONTINUE
COLSIZE=32*NUMLTRS
WRITE 103, (KEEPER1(33 , NUMCOL), NUMCOL=2, COLSIZE, 2)
WRITE 106, (KEEPER1(34 , NUMCOL), NUMCOL=2, COLSIZE, 2)
WRITE 99
NUMLTRS=0
NUMPAGE=NUMPAGE+1
IF (ALPHSKP.AND. (OLDALPH.NE.ALPHNUM)) 190, 195
190 NUMPAGE=0
OLDALPH=ALPHNUM
WRITE 104
GO TO 105
195 IF ((NUMPAGE.EQ.2) .OR.DUMPIT) NUMPAGE=0
200 RETURN
END
SUBROUTINE XFORMIT(LTRSIZE, WIDTH, HEIGHT, ADJUST, WRATIO, HRATIO,
% NSPACE, NUMROWS, NUMCOLS, ALPHNUM, LTRNUM, VEXPAND, XFORM, IMAGE, INVERSE)
INTEGER LTRSIZE, WIDTH, HEIGHT, NSPACE, NUMROWS, NUMCOLS, ALPHNUM, LTRNUM
LOGICAL ADJUST $ REAL WRATIO, HRATIO
LOGICAL VEXPAND, XFORM, IMAGE, INVERSE
INTEGER LETTER(32, 32), NUMLTRS, RECNUM
COMMON/LETTERS/LETTER, NUMLTRS, RECNUM
INTEGER RECFLTR
REAL FLTRLTR(180), PROTO(180)
COMMON/PROFLTR/FLTRLTR, PROTO, RECFLTR
COMMON/XFORMS/FORTLTR
COMPLEX FORTLTR(64, 64)
COMPLEX WORKER(64)
INTEGER POWERS(7)
INTEGER NN(2), NUMCOLS, NUMROWS, NCOL, NROW, FLAG, TOP, BOTTOM, ITEMP,
% COL, ROW, LEFT, RIGHT, IFORM, ISIGN
2 FORMAT (1HT)
DIMENSION TRPLTR(32, 32)
DATA POWERS/2, 4, 8, 16, 32, 64, 128/
WRITE 2
FLAG=0
90 CONTINUE
IF (ADJUST) GO TO 100
NUMROWS=NUMCOLS=LTRSIZE
WIDTH=32
HEIGHT=22
CALL TRPMLT(LETTER, TRPLTR)
CALL REPLACE (TRPLTR, FORTLTR, NUMROWS, NUMCOLS)
GO TO 235
100 CONTINUE
CALL SIZEIT(LTRSIZE, WIDTH, HEIGHT)
NUMROWS=HEIGHT*HRATIO
NUMCOLS=WIDTH*WRATIO
NRINGS=NSPACE** .5
IF (NUMROWS.LT.NRINGS) NUMROWS=NRINGS
IF (NUMCOLS.LT.NRINGS) NUMCOLS=NRINGS
IF (MOD(NUMROWS, 2).EQ.1) NUMROWS=NUMROWS+1
IF (MOD(NUMCOLS, 2).EQ.1) NUMCOLS=NUMCOLS+1
TOP =IABS(NUMROWS-LTRSIZE)/2
IF (NUMROWS.LT.LTRSIZE) GO TO 150
BOTTOM=TOP+1+LTRSIZE
ITEMP=TOP
DO 140 NCOL=1, LTRSIZE
TOP=ITEMP
DO 110 NROW=1, TOP
110 FORTLTR(NROW, NCOL)=0

```

```

120 DO 120 NROW=1,LTRSIZE
    TOP=TOP+1
120 FORTLTR(TOP,NCOL)=LETTER(NROW,NCOL)
    DO 130 NROW=BOTTOM,NUMROWS
130     FORTLTR(NROW,NCOL)=0
140 CONTINUE
    GO TO 170
150 CONTINUE
    TOP=TOP+1
    BOTTOM=TOP+NUMROWS-1
    DO 160 NCOL=1,LTRSIZE
        ROW=0
    DO 160 NROW=TOP,BOTTOM
        ROW=ROW+1
160 FORTLTR(ROW,NCOL)=LETTER(NROW,NCOL)
170 CONTINUE
    LEFT=IABS(NUMCOLS-LTRSIZE)/2
    IF(NUMCOLS.LT.LTRSIZE) GO TO 220
    RIGHT=LEFT+LTRSIZE+1
    DO 210 NROW=1,NUMROWS
    DO 180 NCOL=RIGHT,NUMCOLS
180 FORTLTR(NROW,NCOL)=0
    ITEMP=RIGHT
    DO 190 NCOL=1,LTRSIZE
        ITEMP=ITEMP-1
        COL=LTRSIZE-NCOL+1
190 FORTLTR(NROW,ITEMP)=FORTLTR(NROW,COL)
    DO 200 NCOL=1,LEFT
200 FORTLTR(NROW,NCOL)=0
210 CONTINUE
    GO TO 232
220 CONTINUE
    LEFT=LEFT+1
    RIGHT=LEFT+NUMCOLS-1
    DO 230 NROW=1,NUMROWS
        COL=0
    DO 230 NCOL=LEFT,RIGHT
        COL=COL+1
230 FORTLTR(NROW,COL)=FORTLTR(NROW,NCOL)
232 CONTINUE
    CALL ARNGCPX(FORTLTR,64,64,FORTLTR,NUMROWS,NUMCOLS)
235 CONTINUE
    IF (FLAG.GT.0) GO TO 245
    IF (VEXPAND) CALL PRNTEXP(FORTLTR,NUMROWS,NUMCOLS,ALPHNUM,LTRNUM,
%RATIO,HRATIO,WIDTH,HEIGHT)
    IF (XFORM.OR.IMAGE)
% CALL IVRTCPX(FORTLTR,NUMROWS,NUMCOLS)
    NN(1)=NUMROWS $ NN(2)=NUMCOLS $NDIM=2 $ ISIGN=-1 $ IFORM=0
    COL=ROW=0
    DO 240 ITEMP=1,7
        IF (NUMROWS.EQ.POWERS(ITEMP)) ROW=1
240 IF (NUMCOLS.EQ.POWERS(ITEMP)) COL=1
245 CONTINUE
    IF (ROW.EQ.1.AND.COL.EQ.1) GO TO 250
    CALL FOURT (FORTLTR,NN,NDIM,ISIGN,IFORM,WORKER)
    GO TO 255
250 CALL FOURT (FORTLTR,NN,NDIM,ISIGN,IFORM,0.0)
C
C THIS IS THE PLACE TO PUT IN AN INVERSE PRINT ROUTINE THAT
C FILTERS THE FOURIER TRANSFORM FROM THE MIDDLE AND THEN CALL GLPRT
C
255 CONTINUE
    IF (FLAG.GT.0) RETURN
    IF (IMAGE) CALL PRINTING(FORTLTR,NUMROWS,NUMCOLS,ALPHNUM,LTRNUM)
    IF (XFORM)
%CALL PRNTPX(FORTLTR,NUMROWS,NUMCOLS-1,NUMROWS-1,NUMCOLS-

```

%,ALPHNUM ,LTRNUM,0,.T.,.T.)

FLAG=FLAG+1

IF (XFORM.OR.IMAGE) GO TO 90

RETURN

END

SUBROUTINE FOURT(DATA,NN,NDIM,ISIGN,IFORM,WORK)

THE COOLEY-TUKEY FAST FOURIER TRANSFORM IN USASI BASIC FORTRAN

TRANSFORM(K1,K2,...) = SUM(DATA(J1,J2,...)*EXP(ISIGN*2*PI*SQRT(-1)
((J1-1)(K1-1)/NN(1)+(J2-1)*(K2-1)/NN(2)+...))), SUMMED FOR ALL
J1, K1 BETWEEN 1 AND NN(1), J2, K2 BETWEEN 1 AND NN(2), ETC.
THERE IS NO LIMIT TO THE NUMBER OF SUBSCRIPTS. DATA IS A
MULTIDIMENSIONAL COMPLEX ARRAY WHOSE REAL AND IMAGINARY
PARTS ARE ADJACENT IN STORAGE, SUCH AS FORTRAN IV PLACES THEM.
IF ALL IMAGINARY PARTS ARE ZERO (DATA ARE DISGUISED REAL), SET
IFORM TO ZERO TO CUT THE RUNNING TIME BY UP TO FORTY PERCENT.
OTHERWISE, IFORM = +1. THE LENGTHS OF ALL DIMENSIONS ARE
STORED IN ARRAY NN, OF LENGTH NDIM. THEY MAY BE ANY POSITIVE
INTEGERS, THO THE PROGRAM RUNS FASTER ON COMPOSITE INTEGERS, AND
ESPECIALLY FAST ON NUMBERS RICH IN FACTORS OF TWO. ISIGN IS +1
OR -1. IF A -1 TRANSFORM IS FOLLOWED BY A +1 ONE (OR A +1
BY A -1) THE ORIGINAL DATA REAPPEAR, MULTIPLIED BY NTOT (=NN(1)*
NN(2)*...). TRANSFORM VALUES ARE ALWAYS COMPLEX, AND ARE RETURNED
IN ARRAY DATA, REPLACING THE INPUT. IN ADDITION, IF ALL
DIMENSIONS ARE NOT POWERS OF TWO, ARRAY WORK MUST BE SUPPLIED,
COMPLEX OF LENGTH EQUAL TO THE LARGEST NON 2**K DIMENSION.
OTHERWISE, REPLACE WORK BY ZERO IN THE CALLING SEQUENCE.
NORMAL FORTRAN DATA ORDERING IS EXPECTED, FIRST SUBSCRIPT VARYING
FASTEST. ALL SUBSCRIPTS BEGIN AT ONE.

RUNNING TIME IS MUCH SHORTER THAN THE NAIVE NTOT**2, BEING
GIVEN BY THE FOLLOWING FORMULA. DECOMPOSE NTOT INTO
 $2^{**}K_2 * 3^{**}K_3 * 5^{**}K_5 * \dots$. LET SUM2 = $2^{**}K_2$, SUMF = $3^{**}K_3 + 5^{**}K_5$
+ ... AND NF = $K_3 + K_5 + \dots$. THE TIME TAKEN BY A MULTI-
DIMENSIONAL TRANSFORM ON THESE NTOT DATA IS $T = T_0 + NTOT*(T_1 +$
 $T_2*SUM2 + T_3*SUMF + T_4*NF)$. ON THE CDC 3300 (FLOATING POINT ADD TIME
OF SIX MICROSECONDS), $T = 3000 + NTOT*(500 + 43*SUM2 + 68*SUMF +$
 $320*NF)$ MICROSECONDS ON COMPLEX DATA. IN ADDITION, THE
ACCURACY IS GREATLY IMPROVED, AS THE RMS RELATIVE ERROR IS

BOUNDED BY $3*2^{**}(-B)*SUM(FACTOR(J)^{**}1.5)$, WHERE B IS THE NUMBER
OF BITS IN THE FLOATING POINT FRACTION AND FACTOR(J) ARE THE
PRIME FACTORS OF NTOT.

PROGRAM BY NORMAN BRENNER FROM THE BASIC PROGRAM BY CHARLES
RADER. RALPH ALTER SUGGESTED THE IDEA FOR THE DIGIT REVERSAL.
MIT LINCOLN LABORATORY, AUGUST 1967. THIS IS THE FASTEST AND MOST
VERSATILE VERSION OF THE FFT KNOWN TO THE AUTHOR. SHORTER PRO-
GRAMS FOUR1 AND FOUR2 RESTRICT DIMENSION LENGTHS TO POWERS OF TWO.
SEE-- IEEE AUDIO TRANSACTIONS (JUNE 1967), SPECIAL ISSUE ON FFT.

THE DISCRETE FOURIER TRANSFORM PLACES THREE RESTRICTIONS UPON THE
DATA.

1. THE NUMBER OF INPUT DATA AND THE NUMBER OF TRANSFORM VALUES
MUST BE THE SAME.
2. BOTH THE INPUT DATA AND THE TRANSFORM VALUES MUST REPRESENT
EQUISPACED POINTS IN THEIR RESPECTIVE DOMAINS OF TIME AND
FREQUENCY. CALLING THESE SPACINGS DELTAT AND DELTAF, IT MUST BE
TRUE THAT $DELTA F = 2*PI/(NN(I)*DELTA T)$. OF COURSE, DELTAT NEED NOT
BE THE SAME FOR EVERY DIMENSION.
3. CONCEPTUALLY AT LEAST, THE INPUT DATA AND THE TRANSFORM OUTPUT
REPRESENT SINGLE CYCLES OF PERIODIC FUNCTIONS.

EXAMPLE 1. THREE-DIMENSIONAL FORWARD FOURIER TRANSFORM OF A

```

C      EXAMPLE 1. THREE-DIMENSIONAL FORWARD FOURIER TRANSFORM OF A
C      COMPLEX ARRAY DIMENSIONED 32 BY 25 BY 13 IN FORTRAN IV.
C      DIMENSION DATA(32,25,13),WORK(50),NN(3)
C      COMPLEX DATA
C      DATA NN/32,25,13/
C      DO 1 I=1,32
C      DO 1 J=1,25
C      DO 1 K=1,13

C      1 DATA(I,J,K)=COMPLEX VALUE
C      CALL FOURT(DATA,NN,3,-1,1,WORK)

C      EXAMPLE 2. ONE-DIMENSIONAL FORWARD TRANSFORM OF A REAL ARRAY OF
C      LENGTH 64 IN FORTRAN II.
C      DIMENSION DATA(2,64)
C      DO 2 I=1,64
C      DATA(1,I)=REAL PART
C      2 DATA(2,I)=0.
C      CALL FOURT(DATA,64,1,-1,0,0)

C      DIMENSION DATA(1),NN(1),IFACT(32),WORK(1)
C      TWOPI=6.283185307
C      WR=0.
C      WI=0.
C      WSTPI=0.
C      WSTPR=0.
C      IF(NDIM-1)920,1,1
C      1 NTOT=2
C      DO 2 IDIM=1,NDIM
C      IF(NN(IDIM))920,920,2
C      2 NTOT=NTOT*NN(IDIM)

C      MAIN LOOP FOR EACH DIMENSION

C      NP1=2
C      DO 910 IDIM=1,NDIM
C      N=NN(IDIM)
C      NP2=NP1*N
C      IF(N-1)920,900,5

C      FACTOR N

C      5 M=N
C      NTWO=NP1
C      IF=1
C      IDIV=2
C      10 IQUOT=M/IDIV
C      IREM=M-IDIV*IQUOT
C      IF(IQUOT-IDIV)50,11,11
C      11 IF(IREM)20,12,20

C      12 NTWO=NTWO+NTWO
C      M=IQUOT
C      GO TO 10

C      20 IDIV=3
C      30 IQUOT=M/IDIV
C      IREM=M-IDIV*IQUOT
C      IF(IQUOT-IDIV)60,31,31
C      31 IF(IREM)40,32,40
C      32 IFACT(IF)=IDIV
C      IF=IF+1
C      M=IQUOT
C      GO TO 30

C      40 IDIV=IDIV+2
C      GO TO 30

C      50 IF(IREM)60,51,60
C      51 NTWO=NTWO+NTWO

```

```

60      IFACT(IF)=M
C
C      SEPARATE FOUR CASES--
C          1. COMPLEX TRANSFORM OR REAL TRANSFORM FOR THE 4TH, 5TH, ETC.
C             DIMENSIONS.
C          2. REAL TRANSFORM FOR THE 2ND OR 3RD DIMENSION. METHOD--
C             TRANSFORM HALF THE DATA, SUPPLYING THE OTHER HALF BY CON-
C             JUGATE SYMMETRY.
C          3. REAL TRANSFORM FOR THE 1ST DIMENSION, N ODD. METHOD--
C             TRANSFORM HALF THE DATA AT EACH STAGE, SUPPLYING THE OTHER
C             HALF BY CONJUGATE SYMMETRY.
C          4. REAL TRANSFORM FOR THE 1ST DIMENSION, N EVEN. METHOD--
C             TRANSFORM A COMPLEX ARRAY OF LENGTH N/2 WHOSE REAL PARTS
C             ARE THE EVEN NUMBERED REAL VALUES AND WHOSE IMAGINARY PARTS
C
C             ARE THE ODD NUMBERED REAL VALUES. SEPARATE AND SUPPLY
C             THE SECOND HALF BY CONJUGATE SYMMETRY.
C
70      NON2=NP1*(NP2/NTWO)
        ICASE=1
        IF(IDIM-4)71,90,90
71      IF(IFORM)72,72,90
72      ICASE=2
        IF(IDIM-1)73,73,90
73      ICASE=3
        IF(NTWO-NP1)90,90,74
74      ICASE=4
        NTWO=NTWO/2
        N=N/2
        NP2=NP2/2
        NTOT=NTOT/2
        I=3
        DO 80 J=2,NTOT
        DATA(J)=DATA(I)
80      I=I+2
90      I1RNG=NP1
        IF(ICASE-2)100,95,100
95      I1RNG=NP0*(1+NPREV/2)
C
C      SHUFFLE ON THE FACTORS OF TWO IN N. AS THE SHUFFLING
C      CAN BE DONE BY SIMPLE INTERCHANGE, NO WORKING ARRAY IS NEEDED
C
100     IF(NTWO-NP1)600,600,110
110     NP2HF=NP2/2
        U=1
        DO 150 I2=1,NP2,NON2

120     IF(J-I2)120,130,130
        I1MAX=I2+NON2-2
        DO 125 I1=I2,I1MAX,2
        DO 125 I3=I1,NTOT,NP2
        U3=J+I3-I2
        TEMPR=DATA(I3)
        TEMPI=DATA(I3+1)
        DATA(I3)=DATA(J3)
        DATA(I3+1)=DATA(J3+1)
        DATA(J3)=TEMPR
        DATA(J3+1)=TEMPI
125     DATA(J3+1)=TEMPI
130     M=NP2HF
140     IF(J-M)150,150,145
145     U=J-M
        M=M/2
        IF(M-NON2)150,140,140
150     U=J+M
C

```

C MAIN LOOP FOR FACTORS OF TWO. PERFORM FOURIER TRANSFORMS OF
C LENGTH FOUR, WITH ONE OF LENGTH TWO IF NEEDED. THE TWIDDLE FACTOR
C $W = \text{EXP}(\text{ISIGN} * 2 * \text{PI} * \text{SQRT}(-1) * M / (4 * \text{MMAX}))$. CHECK FOR $W = \text{ISIGN} * \text{SQRT}(-1)$
C AND REPEAT FOR $W = \text{ISIGN} * \text{SQRT}(-1) * \text{CONJUGATE}(W)$.
C

```
NON2T=NON2+NON2
IPAR=NTWO/NP1
310 IF(IPAR-2) 350, 330, 320
320 IPAR=IPAR/4
GO TO 310
330 DO 340 I1=1, I1RNG, 2
DO 340 J3=I1, NON2, NP1
DO 340 K1=J3, NTOT, NON2T

K2=K1+NON2
TEMPR=DATA(K2)
TEMPI=DATA(K2+1)
DATA(K2)=DATA(K1)-TEMPR
DATA(K2+1)=DATA(K1+1)-TEMPI
DATA(K1)=DATA(K1)+TEMPR
DATA(K1+1)=DATA(K1+1)+TEMPI
340 DATA(K1+1)=DATA(K1+1)+TEMPI
350 MMAX=NON2
360 IF(MMAX-NP2HF) 370, 600, 600
370 LMAX=MAX0(NON2T, MMAX/2)
IF(MMAX-NON2) 405, 405, 380
380 THETA=-TWOPI*FLOAT(NON2)/FLOAT(4*MMAX)
IF(ISIGN) 400, 390, 390
390 THETA=-THETA
400 WR=COS(THETA)
WI=SIN(THETA)
WSTPR=-2.*WI*WI
WSTPI=2.*WR*WI
405 DO 570 L=NON2, LMAX, NON2T
M=L
IF(MMAX-NON2) 420, 420, 410
410 W2R=WR*WR-WI*WI
W2I=2.*WR*WI
W3R=W2R*WR-W2I*WI
W3I=W2R*WI+W2I*WR
420 DO 530 I1=1, I1RNG, 2
DO 530 J3=I1, NON2, NP1
KMIN=J3+IPAR*M
IF(MMAX-NON2) 430, 430, 440
430 KMIN=J3
440 KDIF=IPAR*MMAX

450 KSTEP=4*KDIF
DO 520 K1=KMIN, NTOT, KSTEP
K2=K1+KDIF
K3=K2+KDIF
K4=K3+KDIF
IF(MMAX-NON2) 460, 460, 480
460 U1R=DATA(K1)+DATA(K2)
U1I=DATA(K1+1)+DATA(K2+1)
U2R=DATA(K3)+DATA(K4)
U2I=DATA(K3+1)+DATA(K4+1)
U3R=DATA(K1)-DATA(K2)
U3I=DATA(K1+1)-DATA(K2+1)
IF(ISIGN) 470, 475, 475
470 U4R=DATA(K3+1)-DATA(K4+1)
U4I=DATA(K4)-DATA(K3)
GO TO 510
475 U4R=DATA(K4+1)-DATA(K3+1)
U4I=DATA(K3)-DATA(K4)
GO TO 510
480 T2R=W2R*DATA(K2)-W2I*DATA(K2+1)
```

```

T2I=W2R*DATA(K2+1)+W2I*DATA(K2)
T3R=WR*DATA(K3)-WI*DATA(K3+1)
T3I=WR*DATA(K3+1)+WI*DATA(K3)
T4R=W3R*DATA(K4)-W3I*DATA(K4+1)
T4I=W3R*DATA(K4+1)+W3I*DATA(K4)
U1R=DATA(K1)+T2R
U1I=DATA(K1+1)+T2I
U2R=T3R+T4R
U2I=T3I+T4I
U3R=DATA(K1)-T2R
U3I=DATA(K1+1)-T2I

IF(ISIGN)490,500,500
490 U4R=T3I-T4I
    U4I=T4R-T3R
    GO TO 510
500 U4R=T4I-T3I
    U4I=T3R-T4R
510 DATA(K1)=U1R+U2R
    DATA(K1+1)=U1I+U2I
    DATA(K2)=U3R+U4R
    DATA(K2+1)=U3I+U4I
    DATA(K3)=U1R-U2R
    DATA(K3+1)=U1I-U2I
    DATA(K4)=U3R-U4R
520 DATA(K4+1)=U3I-U4I
    KMIN=4*(KMIN-J3)+J3
    KDIF=KSTEP
    IF(KDIF-NP2)450,530,530
530 CONTINUE
    M=MMAX-M
    IF(ISIGN)540,550,550
540 TEMPR=WR
    WR=-WI
    WI=-TEMPR
    GO TO 550
550 TEMPR=WR
    WR=WI
    WI=TEMPR
560 IF(M-LMAX)565,565,410
565 TEMPR=WR
    WR=WR*WSTPR-WI*WSTPI+WR
570 WI=WI*WSTPR+TEMPR*WSTPI+WI

IPAR=3-IPAR
MMAX=MMAX+MMAX
GO TO 350

C
C MAIN LOOP FOR FACTORS NOT EQUAL TO TWO. APPLY THE TWIDDLE FACTOR
C W=EXP(ISIGN*2*PI*SQRT(-1)*(J2-1)*(J1-J2)/(NP2*IFP1)), THEN
C PERFORM A FOURIER TRANSFORM OF LENGTH IFACT(IF), MAKING USE OF
C CONJUGATE SYMMETRIES.
C
600 IF(NTWO-NP2)605,700,700
605 IFP1=NON2
    IF=1
    NP1HF=NP1/2
610 IFP2=IFP1/IFACT(IF)
    J1RNG=NP2
    IF(ICASE-3)612,611,612
611 U1RNG=(NP2+IFP1)/2
    U2STP=NP2/IFACT(IF)
    J1RG2=(J2STP+IFP2)/2
612 U2MIN=1+IFP2
    IF(IFP1-NP2)615,640,640

```

```

        THETA=-TWOPI*FLOAT(J2-1)/FLOAT(NP2)
        IF(ISIGN)625,620,620
620    THETA=-THETA
625    SINTH=SIN(THETA/2.)
        MSTPR=-2.*SINTH*SINTH
        MSTPI=SIN(THETA)
        WR=MSTPR+1.
        WI=MSTPI
        U1MIN=J2+IFP1
        DO 635 J1=J1MIN,J1RNG,IFP1
            I1MAX=J1+I1RNG-2
        DO 630 I1=J1,I1MAX,2
            DO 630 I3=I1,NTOT,NP2
                U3MAX=I3+IFP2-NP1
            DO 630 J3=I3,J3MAX,NP1
                TEMPR=DATA(J3)
        630    DATA(J3)=DATA(J3)*WR-DATA(J3+1)*WI
            DATA(J3+1)=TEMPR*WI+DATA(J3+1)*WR
            TEMPR=WR
            WR=WR*MSTPR-WI*MSTPI+WR
        635    WI=TEMPR*MSTPI+WI*MSTPR+WI
        640    THETA=-TWOPI/FLOAT(IFACT(IF))
            IF(ISIGN)650,645,645
        645    THETA=-THETA
        650    SINTH=SIN(THETA/2.)
            MSTPR=-2.*SINTH*SINTH
            MSTPI=SIN(THETA)
            KSTEP=2*N/IFACT(IF)
            KRANG=KSTEP*(IFACT(IF)/2)+1
            DO 698 I1=1,I1RNG,2
                DO 698 I3=I1,NTOT,NP2
                    DO 690 KMIN=1,KRANG,KSTEP
                        U1MAX=I3+J1RNG-IFP1
                    DO 680 J1=I3,J1MAX,IFP1
                        U3MAX=J1+IFP2-NP1
                    DO 680 J3=J1,J3MAX,NP1
                        U2MAX=J3+IFP1-IFP2
                        K=KMIN+(J3-J1+(J1-I3)/IFACT(IF))/NP1HF
            IF(KMIN-1)655,655,665
        655    SUMR=0.

            SUMI=0.
            DO 660 J2=J3,J2MAX,IFP2
                SUMR=SUMR+DATA(J2)
        660    SUMI=SUMI+DATA(J2+1)
            WORK(K)=SUMR
            WORK(K+1)=SUMI
            GO TO 680
        665    KCONJ=K+2*(N-KMIN+1)
            J2=J2MAX
            SUMR=DATA(J2)
            SUMI=DATA(J2+1)
            OLDSR=0.
            OLDSI=0.
            J2=J2-IFP2
        670    TEMPR=SUMR
            TEMPI=SUMI
            SUMR=TWO*WR*SUMR-OLDSR+DATA(J2)
            SUMI=TWO*WR*SUMI-OLDSI+DATA(J2+1)
            OLDSR=TEMPR
            OLDSI=TEMPI
            U2=J2-IFP2
            IF(J2-J3)675,675,670
        675    TEMPR=WR*SUMR-OLDSR+DATA(J2)
            TEMPI=WI*SUMI
            WORK(K)=TEMPR-TEMPI

```

```
WORK(KCONJ)=TEMPR+TEMPI
TEMPR=WR*SUMI-OLD SI+DATA(J2+1)
TEMPI=WI*SUMR
WORK(K+1)=TEMPR+TEMPI
WORK(KCONJ+1)=TEMPR-TEMPI
680 CONTINUE
```

```
IF(KMIN-1)685,685,686
685 WR=WSTPR+1.
WI=WSTPI
GO TO 690
686 TEMPR=WR
WR=WR*WSTPR-WI*WSTPI+WR
WI=TEMPR*WSTPI+WI*WSTPR+WI
690 TWOWR=WR+WR
IF(ICASE-3)692,691,692
691 IF(IFP1-NP2)695,692,692
692 K=1
I2MAX=I3+NP2-NP1
DO 693 I2=I3,I2MAX,NP1
DATA(I2)=WORK(K)
DATA(I2+1)=WORK(K+1)
693 K=K+2
GO TO 698
```

```
C
C COMPLETE A REAL TRANSFORM IN THE 1ST DIMENSION, N ODD, BY CON-
C JUGATE SYMMETRIES AT EACH STAGE.
```

```
695 U3MAX=I3+IFP2-NP1
DO 697 J3=I3,J3MAX,NP1
U2MAX=J3+NP2-J2STP
DO 697 J2=J3,J2MAX,J2STP
U1MAX=J2+J1RG2-IFP2
U1CNJ=J3+J2MAX+J2STP-J2
DO 697 J1=J2,J1MAX,IFP2
K=1+J1-I3
DATA(J1)=WORK(K)
DATA(J1+1)=WORK(K+1)
```

```
IF(J1-J2)697,697,696
696 DATA(J1CNJ)=WORK(K)
DATA(J1CNJ+1)=-WORK(K+1)
697 U1CNJ=J1CNJ-IFP2
698 CONTINUE
IF=IF+1
IFP1=IFP2
IF(IFP1-NP1)700,700,610
```

```
C
C COMPLETE A REAL TRANSFORM IN THE 1ST DIMENSION, N EVEN, BY CON-
C JUGATE SYMMETRIES.
```

```
700 GO TO (900,800,900,701),ICASE
701 NHALF=N
```

```
N=N+N
THETA=-TWOPI/FLOAT(N)
IF(ISIGN)703,702,702
```

```
702 THETA=-THETA
703 SINTH=SIN(THETA/2.)
WSTPR=-2.*SINTH*SINTH
WSTPI=SIN(THETA)
WR=WSTPR+1.
WI=WSTPI
IMIN=3
UMIN=2*NHALF-1
GO TO 725
710 U=IMIN
```

```

DO 720 I=IMIN,NTOT,NP2
SUMR=(DATA(I)+DATA(J))/2.
SUMI=(DATA(I+1)+DATA(J+1))/2.
DIFR=(DATA(I)-DATA(J))/2.

DIFI=(DATA(I+1)-DATA(J+1))/2.
TEMPR=WR*SUMI+WI*DIFR
TEMPI=WI*SUMI-WR*DIFR
DATA(I)=SUMR+TEMPR
DATA(I+1)=DIFI+TEMPI
DATA(J)=SUMR-TEMPR
DATA(J+1)=-DIFI+TEMPI
720 J=J+NP2
IMIN=IMIN+2
JMIN=JMIN-2
TEMPR=WR
WR=WR*WSTPR-WI*WSTPI+WR
WI=TEMPR*WSTPI+WI*WSTPR+WI
725 IF(IMIN-JMIN)710,730,740
730 IF(ISIGN)731,740,740
731 DO 735 I=IMIN,NTOT,NP2
735 DATA(I+1)=-DATA(I+1)
740 NP2=NP2+NP2
NTOT=NTOT+NTOT
U=NTOT+1
IMAX=NTOT/2+1
745 IMIN=IMAX-2*NHALF
I=IMIN
GO TO 755
750 DATA(J)=DATA(I)
DATA(J+1)=-DATA(I+1)
755 I=I+2
U=J-2
IF(I-IMAX)750,760,760
760 DATA(J)=DATA(IMIN)-DATA(IMIN+1)
DATA(J+1)=0.

IF(I-J)770,780,780
765 DATA(J)=DATA(I)
DATA(J+1)=DATA(I+1)
770 I=I-2
U=J-2
IF(I-IMIN)775,775,765
775 DATA(J)=DATA(IMIN)+DATA(IMIN+1)
DATA(J+1)=0.
IMAX=IMIN
GO TO 745
780 DATA(1)=DATA(1)+DATA(2)
DATA(2)=0.
GO TO 900

```

```

C
C COMPLETE A REAL TRANSFORM FOR THE 2ND OR 3RD DIMENSION BY
C CONJUGATE SYMMETRIES.
C

```

```

800 IF(I1RNG-NP1)805,900,900
805 DO 860 I3=1,NTOT,NP2
I2MAX=I3+NP2-NP1
DO 860 I2=I3,I2MAX,NP1
IMIN=I2+I1RNG
IMAX=I2+NP1-2
JMAX=2*I3+NP1-IMIN
IF(I2-I3)820,820,810
810 UMAX=JMAX+NP2
820 IF(IDIM-2)850,850,830
830 J=JMAX+NP0
DO 840 I=IMIN,IMAX,2

```

```

DATA(I)=DATA(J)
DATA(I+1)=-DATA(J+1)

840  U=J-2
850  U=JMAX
      DO 860 I=IMIN,IMAX,NP0
DATA(I)=DATA(J)
DATA(I+1)=-DATA(J+1)
860  U=J-NP0
C
C   END OF LOOP ON EACH DIMENSION
C
900  NP0=NP1
      NP1=NP2
910  NPREV=N
920  RETURN
      END
SUBROUTINE TRPMLT(LETTER,TRPLTR)
INTEGER LETTER(32,32)
DIMENSION TRPLTR(32,32),TRPWDW(32)
1  FORMAT(1H1)
2  FORMAT(1H ,24(" "),/1H ,"* TRAPEZOID PARAMETERS *",/1H ,24(" "))
3  FORMAT(1H ,/,," LENGTH OF TRAPEZOID PLATEAU = ",I2)
4  FORMAT(1H ,/,," VARIANCE IS = ",F5.2)
5  FORMAT(1H ,/,," TRAPEZOID WINDOW ARRAY IS: ")
6  FORMAT("      ",10F6.3)
7  FORMAT(1H ,/,," THIS IS A NORMALIZED WINDOW ")
      IF (PSKIP.EQ."OK") GO TO 45
C   DATA NORMALIZED WINDOW SET IPLEN=0 AND TRPHT=VARIANCE
      IPLEN = 0      $      TRPHT = 5.0
C
C   THIS PART BUILDS THE TRAPEZOID
C
      INCLN=(32-IPLEN)/2
      IF (IPLEN.NE.0) GO TO 9
      DO 8 I=1,16
8  TRPWDW(17-I)=TRPWDW(16+I)=EXP(-((I-1)**2)/(2.*((TRPHT)**2)))
      WRITE 1
      WRITE 7
      GO TO 44
9  IF (INCLN.EQ.0) GO TO 35
      DO 10 I=1,INCLN
      TRPWDW(I)=TRPWDW(33-I)=(I/(INCLN+1.))*TRPHT
10  CONTINUE
      DO 20 I=1,IPLN
      TRPWDW(I+INCLN)=1.
20  CONTINUE
      IF((32-(2*INCLN)).EQ.IPLN) GO TO 43
      IODD=INCLN+1
      DO 30 I=1,IODD
      TRPWDW(33-I)=(I/(INCLN+2.))*TRPHT
30  CONTINUE
      GO TO 43
35  DO 40 I=1,16
40  TRPWDW(I)=TRPWDW(33-I)=1.
43  CONTINUE
      WRITE 1
      WRITE 2
      WRITE 3,IPLN
44  WRITE 4,TRPHT
      WRITE 5
      WRITE 6,TRPWDW
C
C   THIS PART DOES THE MULTIPLICATION
C

```

```
40 DO 50 I=1,32
DO 50 J=1,32
50 TRPLTR(J,I)=TRPDW(I)*(FLOAT(LETTER(J,I)))
60 CONTINUE
PSKIP="OK"
RETURN
END
```

Appendix D

PROTOBLD

```

PROGRAM MAIN(INPUT,OUTPUT,TAPE4,TAPE17)
C PROGRAM MAIN(INPUT=402B,OUTPUT=1002B,TAPE4=300B,TAPE10=300B)
1 FORMAT (1HT)
INTEGER ALPHSTR,ALPHSTP,EXCPTNS(20),NEXCPTS ,TOTALPH,LTRSTRT,
% LTRSTOP,NUMSUB,RANGE(2,10),NSPACE(10),TAPEIN,TAPEOUT,ENORM,DC
% ,LSPACE
LOGICAL RANDIN,NEWFILE,ADD,CHANGE,DELETE
WRITE 1
ALPHSTR=1 $ ALPHSTP=150 $ LTRSTRT=1 $ LTRSTOP=26 $ TAPEIN=4
RANDIN=.T. $ NEWFILE=.T.$ ADD=.F. $ CHANGE=.F. $ DELETE=.F.
ENORM=1 $ DC=0 $ TAPEOUT=17 $ NUMSUB=1 $ NEXCPTS=0$ LSPACE=81
XXXX=99 $ ENORM=1$ NUMSUB=1$ LSPACE=81$ NEXCPTS=6
DATA EXCPTNS/24,62,82,101,106/,XXXX/1./
DATA NSPACE/10*81/
DATA RANGE/1,32/
CALL PREPFLR(4)
CALL PREPRTO(TAPEOUT)
CALL PRTORLD(ALPHSTR,ALPHSTP,EXCPTNS,NEXCPTS,TOTALPH,
% LTRSTRT,LTRSTOP,NUMSUB,RANGE,NSPACE,ENORM,DC,TAPEIN,RANDIN,LSPACE
% ,TAPEOUT,NEWFILE,ADD,CHANGE,DELETE)
CALL PRTOTBL(LTRSTRT,LTRSTOP,TOTPRTO,.T.,.F.,.T.,.F.,NSPACE,
% ENORM,DC,TAPEOUT,.F.,0)
STOP
END
SUBROUTINE ENORMFL (FLTRLTR,NORMLTR,NSPACE,DC)
REAL FLTRLTR(NSPACE),NORMLTR(NSPACE)
INTEGER NSPACE,DC
INTEGER START,NVECTOR $ REAL SUMSQRS
START=2
IF (DC.EQ.1) START=1
SUMSQRS=0.0
DO 100 NVECTOR=START,NSPACE
100 SUMSQRS=SUMSQRS+FLTRLTR(NVECTOR)**2
SUMSQRS=SUMSQRS** .5
IF (SUMSQRS.EQ.0.0) SUMSQRS=1.0
DO 110 NVECTOR=START,NSPACE
110 NORMLTR(NVECTOR)=FLTRLTR(NVECTOR)/SUMSQRS
IF (DC.EQ.0) NORMLTR(1)=1
RETURN
END
SUBROUTINE GETFLTR (ALPHNUM,LTRNUM,LSPACE,NSPACE,TAPENUM,RANDOM)
INTEGER ALPHNUM,LTRNUM,LSPACE,NSPACE,TAPENUM
LOGICAL RANDOM
INTEGER RECFLTR
REAL FLTRLTR(180),PROTO(180)
COMMON/PROFLTR/FLTRLTR,PROTO,RECFLTR
INTEGER RECRQD,RECSKIP,KEY
DATA RECFLTR/1/
NUMWRDS=LSPACE+2
IF (.NOT.RANDOM) GO TO 95
KEY=(ALPHNUM-1)*26+LTRNUM
CALL READMS(TAPENUM,FLTRLTR,NUMWRDS,KEY)
GO TO 150
95 CONTINUE
RECRQD=(ALPHNUM-1)*26+LTRNUM
RECSKIP=IABS(RECRQD-RECFLTR)
IF (RECRQD-RECFLTR) 100,140,120
100 DO 110 I=1,RECSKIP
110 BACKSPACE TAPENUM
GO TO 140
120 DO 130 I=1,RECSKIP
130 READ (TAPENUM)
140 READ (TAPENUM) (FLTRLTR(I),I=1,NUMWRDS)

```

```

RECFLTR=RECROD+1
150 CALL REDUCE (FLTRLTR,LSPACE,FLTRLTR,NSPACE)
RETURN
END
SUBROUTINE PREPFLR (FORTAPE)
INTEGER FORTAPE
INTEGER FINDEX (3901)
CALL OPENMS (FORTAPE,FINDEX,3901,0)
RETURN
END
SUBROUTINE PREPRTO (PROTAPE)
INTEGER PROTAPE
INTEGER PINDEX (288)
CALL OPENMS (PROTAPE,PINDEX,288,0)
RETURN
END
SUBROUTINE PRTOBLD (ALPHSTR,ALPHSTP,EXCPTNS,NEXCPTS,TOTALPH,
% LTRSTRT,LTRSTOP,NUMSUB,RANGE,NSPACE,ENORM,DC,TAPEIN,RANDIN,SPACE
% ,TAPEOUT,NEWFILE,ADD,CHANGE,DELETE)
INTEGER ALPHSTR,ALPHSTP,EXCPTNS (20),NEXCPTS ,TOTALPH,LTRSTRT,
% LTRSTOP,NUMSUB,RANGE (2,10),NSPACE (10),TAPEIN,TAPEOUT,ENORM,DC
% ,LSPACE
LOGICAL RANDIN,NEWFILE,ADD,CHANGE,DELETE
INTEGER OLDLTR,LTRNUM,CNTALPH (27),PROTNUM,SUBLTRS,KEY,PSPACE,
% NCOL,NROW,ALPHNUM, WIDTH,NWORD,NVECTOR,NEXISTS
REAL PROTOS (169,10)
INTEGER MTABLE (27,5),TABLE (15,10),ETABLE (260,14)
COMMON/TABLES/ MTABLE,ETABLE
INTEGER RECFLTR
REAL FLTRLTR (180),PROTO (180)
COMMON/PROFLTR/FLTRLTR,PROTO,RECFLTR
DATA MTABLE/135*0/
DATA LTRINDX,OLDLTR,CNTALPH/27,27,27*0/
OLDLTR=LTRSTRT
IF (NEWFILE) GO TO 90
CALL READMS (TAPEOUT,MTABLE,135,1)
IF (ADD.OR.CHANGE) GO TO 90
IF (.NOT.DELETE) GO TO 90
DO 80 LTRNUM=LTRSTRT,LTRSTOP
DO 80 NCOL=1,5
80 MTABLE (LTRNUM,NCOL)=0
90 DO 230 LTRNUM=LTRSTRT,LTRSTOP
IF (CNTALPH (OLDLTR).GE.MTABLE (OLDLTR,5)) GO TO 105
C IF THE OLDLTR EXISTS SAVE THE OLDLTR PROTOTYPE INFORMATION
CALL WRITMS (TAPEOUT,TABLE,NUMWRDS,KEY)
DO 100 PROTNUM=1,SUBLTRS
100 IF (TABLE (1,PROTNUM).EQ.1) CALL WRITMS (TAPEOUT,PROTOS (1,PROTNUM)
% ,TABLE (8,PROTNUM),TABLE (2,PROTNUM))
105 IF (CNTALPH (LTRNUM).EQ.0) GO TO 120
C IF THE NEW LETTER ALREADY EXISTS BUT IS NOT COMPLETE READ IT
C INTO CORE TO BE WORKED ON
KEY=MTABLE (LTRNUM,2)
SUBLTRS=MTABLE (LTRNUM,3)
NUMWRDS=SUBLTRS*15
CALL READMS (TAPEOUT,TABLE,NUMWRDS,KEY)
DO 115 PROTNUM=1,SUBLTRS
IF (TABLE (1,PROTNUM).EQ.0) GO TO 110
CALL READMS (TAPEOUT,PROTOS (1,PROTNUM),TABLE (8,PROTNUM),
% TABLE (2,PROTNUM))
GO TO 115
110 DO 112 NROW=1,PSPACE
112 PROTOS (NROW,PROTNUM)=0
115 CONTINUE
GO TO 160
120 CONTINUE

```

```

MTABLE (LTRNUM, 1) = 1
MTABLE (LTRNUM, 2) = KEY = LTRNUM + 1
MTABLE (LTRNUM, 3) = SUBLTRS = NUMSUB
NUMWRDS = SUBLTRS * 15
MTABLE (LTRNUM, 5) = TOTALPH
DO 150 NCOL = 1, SUBLTRS
LTRINDX = LTRINDX + 1
TABLE (1, NCOL) = 0
TABLE (2, NCOL) = LTRINDX
TABLE (3, NCOL) = LTRNUM
TABLE (4, NCOL) = NCOL
TABLE (5, NCOL) = RANGE (1, NCOL)
TABLE (6, NCOL) = RANGE (2, NCOL)
TABLE (7, NCOL) = RANGE (2, NCOL) - RANGE (1, NCOL) + 1
TABLE (8, NCOL) = PSPACE = NSPACE (NCOL)
TABLE (9, NCOL) = ENORM
TABLE (10, NCOL) = DC
DO 130 NROW = 1, 15
130 TABLE (NROW, NCOL) = 0
DO 140 NROW = 1, PSPACE
140 PROTOS (NROW, NCOL) = 0.0
150 CONTINUE
160 CONTINUE
DO 200 ALPHNUM = ALPHSTR, ALPHSTP
IF (NEXCPTS.EQ.0) GO TO 175
DO 170 NROW = 1, NEXCPTS
170 IF (ALPHNUM.EQ.EXCPTNS (NROW)) GO TO 200
175 CNTALPH (LTRNUM) = CNTALPH (LTRNUM) + 1
CALL GETFLTR (ALPHNUM, LTRNUM, LSPACE, LSPACE, TAPEIN, RANDIN)
WIDTH = FLTRLTR (LSPACE + 1)
DO 190 PROTNUM = 1, SUBLTRS
IF ((WIDTH.LT.TABLE (5, PROTNUM)).OR.(WIDTH.GT.TABLE (5, PROTNUM)))
% GO TO 190
PSPACE = TABLE (8, PROTNUM)
TABLE (12, PROTNUM) = TABLE (12, PROTNUM) + 1
MASKER = MASK (1)
NWORD = 60 - MOD (ALPHNUM - 1, 60)
MASKER = SHIFT (MASKER, NWORD)
NWORD = (ALPHNUM - 1) / 60 + 13
TABLE (NWORD, PROTNUM) = MASKER.OR.TABLE (NWORD, PROTNUM)
CALL REDUCE (FLTRLTR, LSPACE, PROTO, PSPACE)
C# XXXX=99.$ IF (ENORM.EQ.1) CALL ENORMFL (PROTO, PROTO, PSPACE, DC)
C# XXXX=99$ CALL LEAR1 (LTRNUM, PROTNUM, ENORM, DC, PROTO, PSPACE)
DO 180 NVECTOR = 1, PSPACE
180 PROTOS (NVECTOR, PROTNUM) = PROTO (NVECTOR) + PROTOS (NVECTOR, PROTNUM)
TABLE (1, PROTNUM) = 1
190 CONTINUE
200 CONTINUE
OLDLTR = LTRNUM
IF (CNTALPH (LTRNUM).GE.TOTALPH) GO TO 215
GO TO 230
205 NEXISTS = 0
DO 220 PROTNUM = 1, SUBLTRS
IF (TABLE (1, PROTNUM).EQ.0) GO TO 220
NEXISTS = NEXISTS + 1
PSPACE = TABLE (8, PROTNUM)
DO 210 NVECTOR = 1, PSPACE
210 PROTOS (NVECTOR, PROTNUM) = PROTOS (NVECTOR, PROTNUM) / TABLE (12, PROTNUM)
XXXX=99.$ IF (ENORM.EQ.1) CALL ENORMFL (PROTOS (1, PROTNUM),
( PROTOS (1, PROTNUM), PSPACE, DC) $XXXX=99.
TABLE (11, PROTNUM) = TABLE (12, PROTNUM)
C# XXXX=99$ CALL LEAR2 (PROTOS (1, PROTNUM), PROTNUM, PSPACE, DC, ENORM)
CALL WRITHS (TAPEOUT, PROTOS (1, PROTNUM), PSPACE, TABLE (2, PROTNUM))
220 CONTINUE
MTABLE (LTRNUM, 4) = NEXISTS
(8) WRITMS (TAPEOUT, TABLE, NUMWRDS, KEY)

```

```

230 CONTINUE
    MTABLE(27,1)=LTRINDX
    CALL WRITMS(TAPEOUT,MTABLE,135,1)
C? XXXX=99$CALL PRTOSTX(LTRSTRT,LTRSTOP,LSPACE,TAPEIN,RANDIN,.F.,
C? 1 .F.,.F.,NSPACE,ENORM,DC,TAPEOUT,.F.,.F.,.F.,1,1)
    RETURN
    END
SUBROUTINE PRTOTBL(LTRSTRT,LTRSTOP,TOTPRTO,PRINTIT,SUMMARY,
%SKIPAGE,
%OVERRIDE,OSPACE,OENORM,ODC,TAPENUM,NEWTAPE,TAPETMP)
    INTEGER LTRSTRT,LTRSTOP,TOTPRTO,TAPENUM,TAPETMP,OSPACE(10)
% ,OENORM,ODC
    LOGICAL PRINTIT,SKIPAGE,OVERRIDE,NEWTAPE,SUMMARY
    INTEGER LETTERS(26),MASKER,LTRNUM,KEY,SUBLTRS,PROTNUM,
% NSPACE,PSPACE,NCOL,NROW,NALPHS,NTEMP,ALPHNUM,ALPHSTR,ALPHSTP,
% ALPH(150),ENORM,DC,NINDEX(300)
    INTEGER MTABLE(27,5),TABLE(15,10),ETABLE(260,14)
    COMMON/TABLES/ MTABLE,ETABLE
    INTEGER RECFLTR
    REAL FLTRLTR(180),PROTO(180)
    COMMON/PROFLTR/FLTRLTR,PROTO,RECFLTR
1   FORMAT(1H0)
5   FORMAT(1H1)
10  FORMAT(1H ,41X,54("****"),/,1H ,41X,"CHARACTERISTICS OF PROTOTYPE FIL
% E "I2", LETTERS "I2" TO "I2,/,1H ,41X,54("****"))
11  FORMAT(1H0,42("****")/1H ,"LTR SUBLTR RANGE SPACE ENORM DC TERM #ALPH
% "/1H ,42("****"))
12  FORMAT(1H ,"CLASS #"I2", LETTER "A1" *** DOES NOT EXIST****")
15  FORMAT(="****")
20  FORMAT(1H ,"CLASS #"I2", LETTER "A1")
25  FORMAT(1H0,"TOTAL # OF ALPHABETS="I3
% ". # OF (POSSIBLE) SUBCLASSES="I2,/" RANGE OF(POSSIBLE) "
% "SUBCLASSES IN ASCENDING ORDER IS:"I2,"-"I2,9(",",I2,"-"I2))
30  FORMAT(1H ,"# OF (ACTUAL) PROTOTYPES ="I2". CHARACTERISTICS FOR "
% "THESE PROTOTYPES ARE SHOWN BELOW."/ )
35  FORMAT(1H ,"LTR SUBLTR RANGE SPACE ENORM DC TERM #ALPH ALPHABETS
% IN THE PROTOTYPE")
40  FORMAT(1H ,1X,A1,3X,I2,3X,I2,"-",I2,3X,I3,3X,A3,4X,A3,4X,I3,3X,
% 22(I3,"")/,6(1H ,44X,22(I3,"")/))
41  FORMAT(1H ,1X,A1,3X,I2,3X,I2,"-",I2,3X,I3,3X,A3,4X,A3,4X,I3,3X)
45  FORMAT(1H ,42("----"))
50  FORMAT(1H ,1X,A1,3X,I2,3X,I2,"-",I2,3X,I3,3X,A3,4X,A3,4X,I3,3X,
% "THIS SUBLTR DOES NOT EXIST, NO ALPHABETS IN THE RANGE")
    DATA LETTERS/"A","B","C","D","E","F","G","H","I","J","K","L","M",
% "N","O","P","Q","R","S","T","U","V","W","X","Y","Z"/
    TOTPRTO=0
    IF (OVERRIDE.AND.NEWTAPE) CALL OPENMS(TAPETMP,NINDEX,238,0)
    CALL READMS(TAPENUM,MTABLE,135,1)
    MASKER=MASK(1)
    IF (SKIPAGE) WRITE 5
    IF (PRINTIT.AND.SUMMARY) PRINTIT=.F.
    IF (PRINTIT.OR.SUMMARY) WRITE 10,(TAPENUM,LTRSTRT,LTRSTOP)
    IF (SUMMARY) WRITE 11
    DO 160 LTRNUM=LTRSTRT,LTRSTOP
    IF (.NOT.PRINTIT) GO TO 85
    WRITE 1
    WRITE 15,(20)
    IF (MTABLE(LTRNUM,1).EQ.1) GO TO 80
    WRITE 12,(LTRNUM,LETTERS(LTRNUM))
    WRITE 15,(20)
    GO TO 150
80  WRITE 20,(LTRNUM,LETTERS(LTRNUM))
    WRITE 15,(20)
85  IF (MTABLE(LTRNUM,1).EQ.0) GO TO 160
    KEY=MTABLE(LTRNUM,2)
    SUBLTRS=MTABLE(LTRNUM,3)

```

```

SUBLTRS=TABLE(LTRNUM,5)
NUMWRDS=SUBLTRS*15
CALL READMS(TAPENUM, TABLE, NUMWRDS, KEY)
IF (.NOT.PRINTIT) GO TO 90
WRITE 25, (MTABLE(LTRNUM,5),
%      SUBLTRS, (TABLE( 5, NCOL), TABLE(6, NCOL), NCOL=1, SUBLTRS))
WRITE 30, (MTABLE(LTRNUM,4))
WRITE 15, (71)
WRITE 35
WRITE 15, (71)
90  CONTINUE
DO 150 PROTNUM=1, SUBLTRS
  IF (TABLE(1, PROTNUM).EQ.0) GO TO 142
  TOTPRTO=TOTPRTO +1
  IF (.NOT. OVERRIDE) GO TO 100
  PSPACE=TABLE(8, PROTNUM)
  TABLE(8, PROTNUM)=NSPACE=OSPACE(PROTNUM)
  TABLE(9, PROTNUM)=OENORM
  TABLE(10, PROTNUM)=ODC
  IF (.NOT. NEWTAPE) GO TO 100
  KEY=TABLE(2, PROTNUM)
  CALL READMS(TAPENUM, PROTO, PSPACE, KEY)
  CALL REDUCE (PROTO, PSPACE, PROTO, NSPACE)
  IF (OENORM.EQ.1) CALL ENORMFL(PROTO, PROTO, NSPACE, ODC)
  CALL WRITMS(TAPETMP, PROTO, NSPACE, KEY)
100  CONTINUE
DO 110 NCOL=2, 15
110  ETABLE(TOTPRTO , NCOL-1)=TABLE(NCOL, PROTNUM)
  IF (SUMMARY) GO TO 142
  IF (.NOT.PRINTIT) GO TO 150
  NALPHS=0
  DO 140 NWRDS=1,3
  NTEMP=TABLE(12+NWRDS, PROTNUM)
  ALPHSTR= (NWRDS-1)*60+1
  ALPHSTP=ALPHSTR+59
  IF (ALPHSTP.GT.150) ALPHSTP=150
  DO 130 ALPHNUM= ALPHSTR, ALPHSTP
  IF (NTEMP.AND.MASKER) 125, 127
125  NALPHS=NALPHS+1
  ALPH(NALPHS)=ALPHNUM
127  NTEMP=SHIFT(NTEMP, 1)
130  CONTINUE
140  CONTINUE
142  NCOL=TABLE(12, PROTNUM)
  ENORM=DC="OUT"
  IF (TABLE(9, PROTNUM).EQ.1) ENORM="IN "
  IF (TABLE(10, PROTNUM).EQ.1) DC="IN "
  IF (TABLE(1, PROTNUM).EQ.0) 145, 147
145  WRITE 50, (LETTERS(LTRNUM), PROTNUM, TABLE(5, PROTNUM), TABLE(6, PROTNUM
%), TABLE(8, PROTNUM), ENORM, DC, NCOL)
  IF (SUMMARY) 150, 149
147  IF (.NOT.SUMMARY) GO TO 148
  WRITE 41, (LETTERS(LTRNUM), PROTNUM, TABLE(5, PROTNUM), TABLE(6, PROTNUM
%), TABLE(8, PROTNUM), ENORM, DC, NCOL)
  GO TO 150
148  WRITE 40, (LETTERS(LTRNUM), PROTNUM, TABLE(5, PROTNUM), TABLE(6, PROTNUM
%), TABLE(8, PROTNUM), ENORM, DC, NCOL , (ALPH(NROW), NROW=1, NALPHS))
149  WRITE 45
150  CONTINUE
160  CONTINUE
  RETURN
  END
  SUBROUTINE REDUCE (INFLTR, INSPACE, OUTFLTR, OTSPACE)
  INTEGER INSPACE, OTSPACE
  REAL      INFLTR(INSPACE), OUTFLTR(OTSPACE)
  INTEGER INHRMNC, OUTHRMNC, SKIP, DOWN, OUTVCTR, INVCTR, START, STOP

```

```

85 DO 85 NVECTOR=1,OTSPACE
OUTFLTR(NVECTOR)=INFLTR(NVECTOR)
GO TO 125
90 CONTINUE
INHRMNC= INSPACE*.5 $ OUTHRMC=OTSPACE *.5
SKIP=(INHRMNC-OUTHRMC)*2
DOWN=OUTHRMC/2+1
OUTFLTR(1)=INFLTR(1)
IF (OTSPACE.EQ.1) RETURN
DO 100 OUTVCTR=2,OUTHRMC
100 OUTFLTR(OUTVCTR)=INFLTR(OUTVCTR)
INVCTR=START=OUTHRMC+1
DO 120 NROW=2,DOWN
INVCTR=INVCTR+SKIP
STOP=START+OUTHRMC*2-1
DO 110 OUTVCTR=START,STOP
OUTFLTR(OUTVCTR)=INFLTR(INVCTR)
INVCTR=INVCTR+1
110 CONTINUE
START=STOP+1
120 CONTINUE
125 CONTINUE
DO 130 I=1,4
130 OUTFLTR(OTSPACE+I)=INFLTR(INSPACE+I)
RETURN
END
SUBROUTINE PRSTOX(LTRSTRT,LTRSTOP,LSPACE,LRTAPE,RANDOM,PRINTIT,
% SKIPAGE,OVERIDE,OSPACE,OENORM,ODC,PROTAPE,COROLTN,HSTOGRM,BARS,
% NDEV,NDIVS,WRATIO,HRATIO)
INTEGER LTRSTRT, LTRSTOP,OSPACE(10),PROTAPE,LSPACE,NDEV,NDIVS,
%LRTAPE,OENORM,ODC
REAL WRATIO,HRATIO
LOGICAL PRINTIT,SKIPAGE,OVERIDE,COROLTN,HSTOGRM,BARS,RANDOM
INTEGER MTABLE(27,5),TABLE(15,10),ETABLE(260,14)
INTEGER RECFLTR
REAL FLTRLTR(180),PROTO(180)
COMMON/PROFLTR/FLTRLTR,PROTO,RECFLTR
COMMON/TABLES/ MTABLE,ETABLE
INTEGER ALPHSTR,ALPHSTP,NWRDS,PROTNUM,NTEMP,TOTPRTO,ALPHNUM
%,PENORM,PDC,TOTALPH,PSPACE,SUBLTR
MASKER=MASK(1)
CALL PRTOTBL(LTRSTRT,LTRSTOP,TOTPRTO,PRINTIT,.F.,SKIPAGE,OVERIDE,
%OSPACE,OENORM,ODC,PROTAPE,.F.,0)
DO 120 PROTNUM=1,TOTPRTO
LTRNUM =ETABLE(PROTNUM,2)
IF ((LTRNUM.LT.LTRSTRT).OR.(LTRNUM.GT.LTRSTOP)) GO TO 120
SUBLTR= ETABLE(PROTNUM,3)
TOTALPH=ETABLE(PROTNUM,11)
PSPACE= ETABLE (PROTNUM,7)
PENORM=ETABLE(PROTNUM,8)
PDC=ETABLE(PROTNUM,9)
CALL READMS(PROTAPE,PROTO,PSPACE,ETABLE(PROTNUM,1))
IF(IQ.NE.23)PRINT*," PROT ",PROTO $ IQ=23
DO 110 NWRDS=1,3
ALPHSTR= (NWRDS-1) *60+1
ALPHSTP = ALPHSTR+59
NTEMP=ETABLE(PROTNUM,11+NWRDS)
DO 100 ALPHNUM= ALPHSTR,ALPHSTP
IF(NTEMP.AND.MASKER) 95,97
C5 CALL STATS(ALPHNUM,ALPHNUM,TOTALPH,LTRNUM,SUBLTR,LSPACE,PSPACE,
CCCCC%CLRTAPE,RANDOM,PENORM,PDC,COROLTN,HSTOGRM,BARS,NDEV,NDIVS,CCCCCCC
CCCCC%PROTAPE,WRATIO,HRATIO)CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
95 CALL GETFLTR(ALPHNUM,LTRNUM,LSPACE,PSPACE,LRTAPE,RANDOM)
CALL STATX(FLTRLTR,PSPACE,TOTALPH)
97 NTEMP=SHIFT(NTEMP,1)
100 CONTINUE

```

```

110 CONTINUE
    CALL MAHPAK(PROTO,PSPACE)
    CALL WRITMS(PROTAPE,PROTO,PSPACE,ETABLE(PROTNUM,1))
120 CONTINUE
    RETURN
    END
    SUBROUTINE STATX(FVEC,NENT,NTOT)
C*****          INSERT TO PROTO BUILD 2/8/78      JRL
    COMMON/STX/ STNDDEV(81,3),FMEAN(81,3)
    DIMENSION FVEC(NENT),FT(2,81)
    IF (INUM.EQ.0) GOTO 60
    DO 50 J=1,169
50  FT(1,J)=FT(2,J) = 0.0
60  CONTINUE
    INUM=INUM+1
    DO 100 J=1,NENT
    CALL DOIT(FVEC(J),STNDEV(J),FMEAN(J),FT(1,J),INUM)
100 CONTINUE
    IF(INUM.EQ.NTOT) INUM = 0
    RETURN
    END
    SUBROUTINE DOIT(FIN,FSD,FM,FT,INUM)
C*****          INSERT TO PROTO BUILD 2/8/78      JRL
    DIMENSION FT(1)
    FT(1)=FT(1) + FIN**2
    FT(2)= FT(2) + FIN
    FM=FT(2)/INUM
    FSD = (FT(1)/INUM) - ((FT(2)/INUM)**2)
    FSD = SQRT(ABS(FSD))
    RETURN
    END
    SUBROUTINE MAHPAK(PROTO,PSPACE)
C*****          INSERT TO PROTO BUILD 2/8/78      JRL
    COMMON/STX/ STNDDEV(81,3),FMEAN(81,3)
    INTEGER PSPACE
    DIMENSION PROTO(PSPACE)
    IF(IQ.NE.23) PRINT*," FM ",FMEAN
    IF(IQ.NE.23) PRINT*," SDEV ",STNDEV $ IO=23
    DO 20 I=1,PSPACE
20  PROTO(I)=PACK(FMEAN(I),STNDEV(I))
    RETURN
    END
    REAL FUNCTION PACK(A,B)
C*****          INSERT TO PROTO BUILD 2/8/78      JRL
    IA=A*1E5 $ IB = B*1E5
    PACK=(SHIFT(IA,30).AND.MASK(30)).OR..NOT.MASK(30).AND.IB
    IF(IA.LT.0) PACK=PACK.OR.MASK(1)
    IF(IB.LT.0) PACK = PACK.OR.SHIFT(MASK(1),30)
    RETURN
    END
    SUBROUTINE LEAR1(NEXT,ISET,NORM,IDC,FVEC,NDIM)
C*****          INSERT TO PRTOBLD 2/17/78      JRL
    COMMON/STX1/FT(2,81,3),INUM(3)
    DIMENSION FVEC(NDIM)
    COMMON/STX/SDEV(81,3),FM(81,3)
    DATA IGO/0/
    IF(NEXT.NE.IGO)GOTO 20
10  CONTINUE
C*****          UPDATE ONE OF 3SPECIFIED MEANAND SDEV VECTORS
    INUM(ISET)=INUM(ISET)+1 $ IX=INUM(ISET)
    IF(NORM.EQ.1) CALL ENORMFL(FVEC,FVEC,NDIM,IDC)
    DO 15 J= 1,NDIM
    CALL DOIT(FVEC(J),SDEV(J,ISET),FM(J,ISET),FT(1,J,ISET),IX)
15  CONTINUE
    RETURN
20  CONTINUE

```

C***** REINITIALIZE FOR EACH NEW SET OF 3 VECTOR STREAMS

IGO=NEXT

DO 30 J=1,81

DO 30 K=1,3

INUM(K)=0

30 FT(1,J,K)=FT(2,J,K)=0.0

GOTO 10

END

SUBROUTINE LEAR2(PROTO,ISET,PSPACE,DC,NORM)

C***** INSERT TO PRTOBLD 2/17/78 JRL

COMMON/STX1/FT(2,81,3),INUM(3)

COMMON/STX/SD(81,3),FM(81,3)

INTEGER PSPACE,START

DIMENSION PROTO(1)

START=28 IF(DC.EQ.1) START=1

IF(NORM.EQ.1) CALL ENORMFL(FM(1,ISET),FM(1,ISET),PSPACE,DC)

DO 10 I=START,PSPACE

SD(I,ISET)=FT(1,I,ISET)/INUM(ISET)-FM(I,ISET)**2

10 SD(I,ISET)=SQRT(ABS(SD(I,ISET)))

IF(IQ.NE.23) PRINT*," PR ",(PROTO(K),K=1,PSPACE)

IF(IQ.NE.23) PRINT*," FM " , (FM(K,ISET),K=1,PSPACE)

IF(IQ.NE.23) PRINT*," SD ",(SD(K,ISET),K=1,PSPACE)

II=II+1 \$ IF(II.GT.1) IQ=23

DO 20 I=1,PSPACE

20 PROTO(I)=PACK(FM(I,ISET),SD(I,ISET))

RETURN

END

Appendix E

CMRSEN

```

PROGRAM MAIN(INPUT,OUTPUT,TAPE10,TAPE20,TAPE31,TAPE55)
INTEGER STRTCOL,STOPCOL,NSKIPS,SENTNUM,SENTAPE,TAPETMP,
% OSPACE(10),OENORM,ODC,LTRSTRT,LTRSTOP,PROTAPE,TOPRTOS,SENTEMP
LOGICAL OVERRIDE,BETWEEN,WINDOWS,PRNTALL,PRNTSUM,ADJUST
REAL WRATIO,HRATIO
INTEGER LWIDTH,HWIDTH,SKWIDTH
LWIDTH=2 $ HWIDTH=32 $ SKWIDTH=2
STRTCOL=1 $ STOPCOL=133 $NSKIPS=1 $ SENTNUM=4 $ SENTAPE=50
C DATA NEXT LINE VARIES WHICH SENTENCE USED FOR DATA
SENTNUM=1
OVERRIDE=.T. $ OENORM=1 $ ODC=0 $ LTRSTRT=1 $ LTRSTOP=26
PROTAPE=10 $ PRNTALL=.T. $ BETWEEN=.T. $ WINDOWS=.T.
NBTWTOP=5
TOPRTOS=5 $ TAPETMP=31 $ ADJUST=.T. $ WRATIO=1.5
HRATIO=1.5 $ PRNTSUM=.F. $ SENTEMP=55
DATA OSPACE/10*81/
CALL PREPSEN(SENTAPE)
CALL PREPRTO(PROTAPE)
CALL CMPRSEN(STRTCOL,STOPCOL,NSKIPS,SENTNUM,SENTAPE,OVERRIDE,
%OSPACE,OENORM,ODC,LTRSTRT,LTRSTOP,PROTAPE,PRNTALL,PRNTSUM,BETWEEN,
%WINDOWS,TOPRTOS,TAPETMP,ADJUST,WRATIO,HRATIO,SENTEMP,LWIDTH
% ,HWIDTH,SKWIDTH,NBTWTOP)
STOP
END
SUBROUTINE CMPRSEN(STRTCOL,STOPCOL,NSKIPS,SENTNUM,SENTAPE,OVERRIDE,
%OSPACE,OENORM,ODC,LTRSTRT,LTRSTOP,PROTAPE,PRNTALL,PRNTSUM,BETWEEN,
%WINDOWS,TOPRTOS,TAPETMP,ADJUST,WRATIO,HRATIO,SENTEMP,LWIDTH
% ,HWIDTH,SKWIDTH,NBTWTOP)
INTEGER STRTCOL,STOPCOL,NSKIPS,SENTNUM,SENTAPE,TAPETMP,
% OSPACE(10),OENORM,ODC,LTRSTRT,LTRSTOP,PROTAPE,TOPRTOS,SENTEMP
LOGICAL OVERRIDE,BETWEEN,WINDOWS,PRNTALL,PRNTSUM,ADJUST
REAL WRATIO,HRATIO
INTEGER PRNTCOL,LTRMASK,SUBMASK,DXMASK,NSPACE,ENORM,DC,
% LSTPRTO,TOTPRTO,PROTNUM,NCOL,FLAG,IAPENUM,KEY,PSPACE,
% PENORM,PDC,START,STOP,PRTOCNT,LASTROW,I,J,XFRMCOL,NSPLITS,NPAGE,
% STRTROW,STOPROW,SUBLTR(52),LTR(52),DX(52),TEMP,LOW,HIGH,SUBCNT
% ,TOPNUM,LETTERS(26),ENDCOL,SINDEX(134)
% ,WMASK ,WIDTH,LWIDTH,HWIDTH,SKWIDTH,LRANGE,HRANGE
COMMON/LETTERS/LETTER,NUMLTRS,RECNUM
INTEGER LETTER(32,32),NUMLTRS,RECNUM
COMMON/RESULTS/RESULTS
INTEGER RESULTS(52,133)
COMMON/PROFLTR/FLTRLTR,PROTO,RECFLTR
INTEGER RECFLTR
REAL FLTRLTR(180),PROTO(180)
COMMON/TABLES/ MTABLE,TABLE,ETABLE
INTEGER MTABLE(27,5),TABLE(10,15),ETABLE(260,14)
REAL WINDOW(180)
EQUIVALENCE(WINDOW,FLTRLTR)
5 FORMAT(1HT)
10 FORMAT(1H1)
15 FORMAT(1H ,136(" *"),/1H ,24X,"RESULTS OF ALGORITHM TO SOLVE THE SE
%GMENTATION PROBLEM FOR NON-ISOLATED TEXT CHARACTERS"/
%1H ,136(" *"),/)//)
16 FORMAT(1H ,"SENTENCE "I3" COLUMN "I3" TO "I3" EVALUATED AGAINST"/
% " PROTOTYPE SET "I2" LETTERS "I2" TO "I2" WITH A LETTER-TO-WINDOW
%WIDTH RATIO OF "F4.2, /" AND A LETTER-TO-WINDOW HEIGHT RATIO OF "
%F4.2, /," SENTENCE WINDOWS RANGE FROM "I2" TO "I2" COLUMNS WIDE"
% " WITH SKIPS IN WINDOW SIZE OF "I2" COLUMNS", /," AND SKIPS BETWEEN
% WINDOWS WITHIN THE SENTENCE OF "I2" SPACES"//)
17 FORMAT(1H ,"FINAL RESULTS OF")
18 FORMAT(1H0,"USER REQUESTED PRINTOUT OF THE CLOSEST "I3" PROTOTYPES
% FOR THE FINAL DECISION MATRIX."/) HOWEVER, FOR THE PROTOTYPE FILE

```

```

% AND RANGE OF LETTERS SPECIFIED, ONLY "I3" PROTOTYPES EXIST." /
% USER REQUEST FOR THE FINAL DECISION MATRIX REDUCED TO THE CLOSE
% XT "I3" PROTOTYPES" //)
20  FORMAT (1H0,"PROTOTYPES IN THE MATRIX ARE: "26(A1,I1,1X),/,3
% (1H ,30X,26(A1,I1,1X)/))
25  FORMAT (1H ,"RESULTS MATRIX SHOWING THE PROTOTYPE-TO-WINDOW "
%/" DISTANCES FOR PROTOTYPES WHOSE WIDTH VARIATION ENABLES"
%/" THE PROTOTYPE TO BE COMPARED TO WINDOWS OF WIDTH "I2" COLUMNS")
26  FORMAT (1H ,"MATRIX IS SORTED BY PROTOTYPE. PROTOTYPE-TO-WINDOW"
% /" DISTANCE IS SHOWN AS IT VARIES CONTINUOUSLY OVER THE SENTENCE"
% /" COLUMNS USING WINDOWS OF WIDTH "I2" BEGINNING AND ENDING AT"
%/" THE COLUMNS SHOWN BY THE MATRIX LEFTMOST COLUMN")
27  FORMAT (1H0,60X,"MATRIX "I2" OF "I2)
30  FORMAT ( 1H ,10(" *"),=(9(" *")),/" COLUMNS * ",
% =(3X,A1,I1,4X),/,1H ,10(" *"),=(9(" *")))
32  FORMAT (1H ,I3"--I3," * "=(F7.3,2X))
35  FORMAT (1H ,I3"--I3" * "=(A1,I1,"-",F5.3,1X))
40  FORMAT (1H ,"MATRIX IS SORTED BY DISTANCE WITHIN THE SENTENCE"
%/" COLUMNS. THE "I2" TO "I2" CLOSEST IDENTIFYING PROTOTYPES ARE"
%/" SHOWN FOR EACH WINDOW OF WIDTH "I2" BEGINNING AND ENDING AT"
%/" THE SENTENCE COLUMN INDICATED BY THE LEFTMOST MATRIX COLUMN.")
41  FORMAT (1H ,"MERGED RESULTS MATRIX SHOWING THE PROTOTYPE-TO-WINDOW"
%/" DISTANCES FOR ALL PROTOTYPES WHOSE RANGE VARIATION ENABLES THE"
%/" PROTOTYPE TO BE COMPARED WITH WINDOWS OF WIDTH "I2" TO "I2"
%/" COLUMNS. THE MATRIX IS SORTED BY DISTANCE WITHIN THE SENTENCE"
%/" COLUMNS TO SHOW THE TOP "I2" PROTOTYPE CHOICES REGARDLESS OF"
%/" WINDOW SIZE. ALL WINDOWS BEGIN AT THE SAME SENTENCE COLUMNS"
%/" WHICH ARE SHOWN IN THE LEFTMOST MATRIX COLUMN")
42  FORMAT (1H ,"MATRIX OF THE "I2" TO "I2" CLOSEST IDENTIFYING "
% "PROTOTYPES FOR WINDOWS OF WIDTH "I2" SORTED BY DISTANCE")
43  FORMAT ( 1H 11 (" *"),=(9 (" *")),/,1H ,"SENTENCE * ",
%=("GUESS "I2,1X),/,1H ,"COLUMNS * "=(("PROTO/DX " ),/,1H 11 (" *"),
% =(9 (" *")))
45  FORMAT (1H 2X,I3,2X" * "=(A1,I1,"-",F5.3,1X))
46  FORMAT ( 1H 11 (" *"),=(16 (" *"))/,1H ,"SENTENCE * ",
%=(" CHOICE #"I2" " ),/,1H ,"COLUMNS * "=(("PROTO-WIDTH-DX " ),
% /,1H ,11 (" *"),=(16 (" *")))
50  FORMAT (1H ,2X,I3,6X,=(A1,I1,2X,I2,2X,F5.3,3X))
DATA LETTERS/"A","B","C","D","E","F","G","H","I","J","K","L","M",
% "N","O","P","Q","R","S","T","U","V","W","X","Y","Z"/
LTRSIZE=32 $ LENGTH=133
PRNTCOL=13 $ NRNTCOL=7
LTRMASK=MASK(8)
SUBMASK=SHIFT(MASK(8),52)
WMASK=SHIFT(MASK(8),44)
DXMASK=SHIFT(MASK(22),22)
WRITE 10
WRITE 15
CALL OPENMS(SENTMP,SINDEX,134,0)
CALL GETSEN (32,SENTNUM,SENTAPE)
CALL PRTOTBL(LTRSTRT,LTRSTOP,TOTPRTO,PRNTALL,PRNTSUM,.T.,OVERIDE,
% OSPACE,0ENORM,ODC,PROTAPE,OVERIDE,TAPETMP)
TAPENUM=PROTAPE
IF (OVERIDE) TAPENUM=TAPETMP
TOPNUM=TOPPRTO
IF (TOPNUM.LE.TOTPRTO) GO TO 80
TOPNUM=TOTPRTO
80  CONTINUE

C
C
C
C
I=26
I=SHIFT(I,52)
TEMP=2000000
DO 70 NCOL=STRCOL-STOPCOL-NSKIPS

```

```

70 DO 70 NROW=1, TOPNUM
RESULTS(NROW, NCOL)=I.OR.TEMP
CONTINUE
C
C SORT OF THE ETABLE BY WIDTH AND NSPACE
C
IF (TOTPRTO.EQ.1) GO TO 123
LSTPRTO=TOTPRTO-1
100 FLAG=0
DO 120 PROTNUM=1, LSTPRTO
IF ((ETABLE(PROTNUM,6).GT.ETABLE(PROTNUM+1,6)).OR.
% ((ETABLE(PROTNUM,6).EQ.ETABLE(PROTNUM+1,6)).AND.
% (ETABLE(PROTNUM,7).GE.ETABLE(PROTNUM+1,7)))) GO TO 120
DO 110 NCOL=1, 14
TEMP=ETABLE(PROTNUM, NCOL)
ETABLE(PROTNUM, NCOL)=ETABLE(PROTNUM+1, NCOL)
ETABLE(PROTNUM+1, NCOL)=TEMP
110 CONTINUE
FLAG=1
120 CONTINUE
IF (FLAG.EQ.1) GO TO 100
123 CONTINUE
C
C BEGIN LOOPING ON THE PROTOTYPES CONTAINED IN THE EXIST TABLE
C PRODUCED BY PRTOTBL
C
WIDTH=0
DC=0
NSPACE=0
ENORM=0
PRTOCNT=0
NLOOKS=1
DO 380 LHWIDTH=1, NLOOKS
NWIDHT=4WIDTH-SKWIDTH*(LHWIDTH-1)
DO 370 PROTNUM=1, TOTPRTO
LRANGE=ETABLE(PROTNUM, 4)
HRANGE=ETABLE(PROTNUM, 5)
IF ((NWIDHT.LT.LRANGE).OR.(NWIDHT.GT.HRANGE)) GO TO 370
KEY=ETABLE(PROTNUM, 1)
PSPACE=ETABLE(PROTNUM, 7)
PENORM=ETABLE(PROTNUM, 8)
PDC=ETABLE(PROTNUM, 9)
CALL READMS(TAPENUM, PROTO, PSPACE, KEY)
C
C IF THE WIDTH OF THE PROTOTYPE CHANGES PRINTOUT THE INTERMEDIATE
C RESULTS BEFORE CONTINUING WITH THE NEXT PROTOTYPE AND TRANSFORM THE
C SENTENCE USING THE NEW WINDOW WIDTH
C IF THE ENERGY NORMALIZATION OR THE DC TERM IN THE NORMALIZED
C WINDOWS REQUIRES A CHANGE DUE TO A CHANGE IN THE PROTOTYPE
C CHARACTERISTICS, THE ENTIRE SENTENCE MUST BE TRANSFORMED AGAIN
C
IF ((WIDTH.NE.NWIDHT).OR.(PENORM.NE.ENORM).OR.(DC.NE.PDC))
% CALL FORTSEN(LTRSIZE, NWIDHT, ADJUST, WRATIO, HRATIO, PSPACE, PENORM,
% PDC, STRTCOL, STOPCOL, NSKIPS, SENTNUM, SENTEMP)
IF ((WIDTH.NE.NWIDHT).OR.(PENORM.NE.ENORM).OR.(DC.NE.PDC))
% NSPACE=PSPACE
C
C IF THE SPACE OF THE PROTOTYPE CHANGES REDUCE THE SPACE OF THE
C WINDOWS CONTAINED IN THE TEMPORARY SENTENCE FILE OF XFORMS
C
IF (NSPACE.EQ.PSPACE) GO TO 350
DO 340 XFRMCOL=STRTCOL, ENDCOL, NSKIPS
CALL READMS(SENTEMP, WINDOW, NSPACE, XFRMCOL)
CALL REDUCE(WINDOW, NSPACE, WINDOW, PSPACE)
CALL WRITMS(SENTEMP, WINDOW, PSPACE, XFRMCOL)
340 CONTINUE

```

```

340 CONTINUE
C
C UPDATE ALL VARIABLE FOR THE CURRENT PROTOTYPE
C
IF ((PRTOCNT.NE.0).AND.(WIDTH.NE.NWIDTH)) GO TO 125
350 CONTINUE
WIDTH=NWIDTH $ NSPACE=PSPACE $ ENORM=PENORM $ DC=PDC
PRTOCNT=PRTOCNT+1
NROW=PRTOCNT+TOPNUM
ENDCOL=STOPCOL
IF ((ENDCOL+WIDTH-1).GT.LENGTH) ENDCOL=LENGTH-WIDTH+1
C
C STORE THE RESULTS OF THE PROTOTYPE AGAINST ALL THE SENTENCE
C WINDOW TRANSFORMS
C
DO 360 XFRMCOL=STRTCOL,ENDCOL,NSKIPS
CALL READMS(SENTEMP,WINDOW,NSPACE,XFRMCOL)
RESULTS(NROW,XFRMCOL)=EUCLID(PROTO,WINDOW,NSPACE,DC)*1000000
I=SHIFT(ETABLE(PROTNUM,2),52)
U=SHIFT(ETABLE(PROTNUM,3),44)
I=I.OR.J
U=SHIFT(WIDTH,36)
I=I.OR.J
RESULTS(NROW,XFRMCOL)=I.OR.RESULTS(NROW,XFRMCOL)
360 CONTINUE
C
C IF THERE ARE BLANK COLUMNS IN THE RESULTS ARRAY, SET THE DISTANCE
C TO 2.0 AND THE LETTER TO Z AND THE SUBLTR TO 0
C
I=26
I=SHIFT(I,52)
TEMP=2000000
DO 362 NCOL=XFRMCOL,STOPCOL,NSKIPS
RESULTS(NROW,NCOL)=I.OR.TEMP
362 CONTINUE
365 IF (PROTNUM.EQ.TOTPRTO) GO TO 125
GO TO 370
C
C CONTINUE TO LOOP ON THE NEXT PROTOTYPE
C
125 STRTROW=TOPNUM+1
STOPROW=STRTROW+PRTOCNT-1
LASTROW=STOPROW-1
C
C PRINTOUT OF THE PROTOTYPES OF WIDTH XX AND THE WINDOWS OF THE
C SENTENCES THEY ARE BEING EVALUATED AGAINST
C
IF (STRTROW.EQ.STOPROW) GO TO 295
C
C SORT BY INCREASING LETTER PROTOTYPE FOR DISPLAY OF EACH LETTERS
C PERFORMANCE BY SENTENCE COLUMN. THIS IS THE FIRST OF THE OPTIONAL
C PRINTOUTS FOR THE INTERMEDIATE RESULTS
C
155 FLAG=0
DO 170 NROW=STRTROW,LASTROW
IF ((LTRMASK.AND.RESULTS(NROW)).LE.(LTRMASK.AND.RESULTS(NROW+1)))
% GO TO 170
FLAG=1
DO 160 XFRMCOL=STRTCOL,ENDCOL,NSKIPS
TEMP=RESULTS(NROW,XFRMCOL)
RESULTS(NROW,XFRMCOL)=RESULTS(NROW+1,XFRMCOL)
RESULTS(NROW+1,XFRMCOL)=TEMP
160 CONTINUE
170 CONTINUE
IF (FLAG.EQ.1) GO TO 155
DO 180 J=STRTROW,STOPROW
LTR(J)=SHIFT(LTRMASK.AND.RESULTS(J,STRTCOL),8)

```

```

SUBLTR(J)=SHIFT(SUBMASK.AND.RESULTS(J,STRTCOL),16)
180 CONTINUE
WRITE 10
WRITE 25,(WIDTH)
WRITE 25,(WIDTH)
NSPLITS=(PRTOCNT-1)/PRNTCOL+1
START=STRTRW
DO 210 NPAGE=1,NSPLITS
STOP=START+PRNTCOL-1
IF (NPAGE.EQ.NSPLITS) STOP=STOPROW
SUBCNT=STOP-START+1
IF (NPAGE.GT.1) WRITE 10
WRITE 27,(NPAGE,NSPLITS)
WRITE 20,( LETTERS(LTR(J)),SUBLTR(J),J=START,STOP)
WRITE 30,(SUBCNT,SUBCNT,(LETTERS(LTR(J)),SUBLTR(J),
% J=START,STOP),SUBCNT)
DO 200 XFRMCOL=STRTCOL,ENDCOL,NSKIPS
DO 190 J=START,STOP
190 DX(J)=DXMASK.AND.RESULTS(J,XFRMCOL)
WRITE 32,(XFRMCOL,XFRMCOL+WIDTH-1,
% SUBCNT,(DX(J)/1000000.0,J=START,STOP))
200 CONTINUE
START=STOP+1
210 CONTINUE
C
C SORT BY DISTANCE WITHIN THE SENTENCE COLUMN FOR THE SECOND OUTPUT
C OF THE INTERMEDIATE RESULTS
C
CALL SORTIT(STRTCOL,ENDCOL,NSKIPS,STRTRW,STOPROW)
START=STRTRW
TEMP=NBTWTOP
IF (NBTWTOP.GT.PRTOCNT) NBTWTOP=PRTOCNT
NSPLITS=(NBTWTOP-1)/PRNTCOL+1
LOW=1
WRITE 10
WRITE 25,(WIDTH)
WRITE 40,(1,SUBCNT,WIDTH)
WRITE 20,(LETTERS(LTR(J)),SUBLTR(J),J=STRTRW,STOPROW)
DO 260 NPAGE=1,NSPLITS
STOP=START+PRNTCOL-1
IF (NPAGE.EQ.NSPLITS) STOP=STRTRW+NBTWTOP-1
SUBCNT=STOP-START+1
HIGH=LOW+SUBCNT-1
IF (NPAGE.GT.1) WRITE 10
WRITE 27,(NPAGE,NSPLITS)
WRITE 42,(LOW,HIGH,WIDTH)
WRITE 43,(SUBCNT,SUBCNT,(NROW,NROW=LOW,HIGH),SUBCNT,SUBCNT)
DO 250 XFRMCOL=STRTCOL,ENDCOL,NSKIPS
DO 240 J=START,STOP
LTR(J)=SHIFT(LTRMASK.AND.RESULTS(J,XFRMCOL),8)
SUBLTR(J)=SHIFT(SUBMASK.AND.RESULTS(J,XFRMCOL),16)
DX(J)=DXMASK.AND.RESULTS(J,XFRMCOL)
240 CONTINUE
WRITE 35,(XFRMCOL,XFRMCOL+WIDTH-1,SUBCNT,(LETTERS(LTR(J)),
% SUBLTR(J),DX(J)/1000000.0,J=START,STOP))
250 CONTINUE
START=STOP+1
LOW=HIGH+1
260 CONTINUE
NBTWTOP=TEMP
295 CONTINUE
370 CONTINUE
380 CONTINUE
RETURN
END
SUBROUTINE ARNGCPX(TEMP,NROWS,NCOLS,TAMP,NUMROWS,NUMCOLS)

```

```

COMPLEX TEMP(NROWS,NCOLS),TAMP(NUMROWS,NUMCOLS)
INTEGER NROWS,NCOLS,NUMROWS,NUMCOLS
DO 100 NCOL=1,NUMCOLS
DO 100 NROW=1,NUMROWS
TAMP(NROW,NCOL)=TEMP(NROW,NCOL)
100 CONTINUE
RETURN
END
SUBROUTINE ENORMFL (FLTRLTR,NORMLTR,NSPACE,DC)
REAL FLTRLTR(NSPACE),NORMLTR(NSPACE)
INTEGER NSPACE,DC
INTEGER START,NVECTOR $ REAL SUMSQRS
START=2
IF (DC.EQ.1) START=1
SUMSQRS=0.0
DO 100 NVECTOR=START,NSPACE
100 SUMSQRS=SUMSQRS+FLTRLTR(NVECTOR)**2
SUMSQRS=SUMSQRS**.5
IF (SUMSQRS.EQ.0.0) SUMSQRS=1.0
DO 110 NVECTOR=START,NSPACE
110 NORMLTR(NVECTOR)=FLTRLTR(NVECTOR)/SUMSQRS
IF (DC.EQ.0) NORMLTR(1)=1
RETURN
END
REAL FUNCTION EUCLID(FLTR1,FLTR2,NSPACE,DC)
INTEGER NSPACE,DC
REAL FLTR1(NSPACE),FLTR2(NSPACE)
REAL SUMSQRS $ INTEGER START $ LOGICAL AMPSPEC
C DATA TO GET COMPLEX SPECTRUM -- AMPSPEC=FALSE
C DATA TO GET AMPLITUDE SPECTRUM -- AMPSPEC=TRUE
AMPSPEC=.FALSE.
SUMSQRS=0.0
START=2
IF (DC.EQ.1) START=1
IF (AMPSPEC) GO TO 50
C THIS BRANCH FINDS EUCLIDIAN DISTANCE - COMPLEX SPECTRUM
DO 25 NVECTOR=START,NSPACE
SUMSQRS=SUMSQRS+(FLTR1(NVECTOR)-FLTR2(NVECTOR))**2
25 CONTINUE
GO TO 110
C THIS BRANCH FINDS EUCLIDIAN DISTANCE - AMPLITUDE SPECTRUM
50 DO 100 NVECTOR=START,NSPACE,2
AFLTR1=SQRT((FLTR1(NVECTOR)**2)+(FLTR1(NVECTOR+1)**2))
AFLTR2=SQRT((FLTR2(NVECTOR)**2)+(FLTR2(NVECTOR+1)**2))
SUMSQRS=SUMSQRS+(AFLTR1-AFLTR2)**2
100 CONTINUE
110 EUCLID=SUMSQRS**.5
RETURN
END
SUBROUTINE FILTER(FORTLTR,NUMROWS,NUMCOLS,FLTRLTR,NSPACE)
INTEGER NUMROWS,NUMCOLS,NSPACE
COMPLEX FORTLTR(NUMROWS,NUMCOLS)
REAL FLTRLTR(NSPACE)
INTEGER NHRMNC,NUMWRDS,RIGHT,LEFT,DOWN,NROW,NCOL
NHRMNC=NSPACE**.5
RIGHT=NHRMNC/2+1
LEFT=NUMCOLS-RIGHT+2
DOWN=RIGHT
FLTRLTR(1)=REAL(FORTLTR(1,1))
IF (NHRMNC.EQ.1) RETURN
NUMWRDS=0
DO 100 NCOL=2,RIGHT
NUMWRDS=NUMWRDS+2
FLTRLTR(NUMWRDS)=REAL(FORTLTR(1,NCOL))
100 FLTRLTR(NUMWRDS+1)=AIMAG(FORTLTR(1,NCOL))
DO 220 NROW=2,DOWN

```

```

DO 210 NCOL=LEFT,NUMCOLS
NUMWRDS=NUMWRDS+2
FLTRLTR(NUMWRDS )=REAL (FORTLTR(NROW,NCOL))
FLTRLTR(NUMWRDS+1)=AIMAG (FORTLTR(NROW,NCOL))
210 CONTINUE
DO 200 NCOL=1,RIGHT
NUMWRDS=NUMWRDS+2
FLTRLTR(NUMWRDS )=REAL (FORTLTR(NROW,NCOL))
FLTRLTR(NUMWRDS+1)=AIMAG (FORTLTR(NROW,NCOL))
200 CONTINUE
220 CONTINUE
RETURN
END
SUBROUTINE FORTSEN(LTRSIZE,WIDTH,ADJUST,WRATIO,HRATIO,NSPACE,
% ENORM,DC,STRTCOL,STOPCOL,NSKIPS,SENTNUM,TAPENUM)
INTEGER LTRSIZE,WIDTH,NSPACE,STRTCOL,STOPCOL,NSKIPS,SENTNUM,
% TAPENUM,ENORM,DC
LOGICAL ADJUST $ REAL WRATIO,HRATIO
INTEGER RECFLTR
REAL FLTRLTR(180),PROTO(180)
COMMON/PROFLTR/FLTRLTR,PROTO,RECFLTR
COMMON/XFORMS/FORTLTR
COMPLEX FORTLTR(64,64)
INTEGER ROW,COL,NROW,NCOL,ENDCOL,XFRMCOL,LENGTH
LENGTH=133
ENDCOL=STOPCOL
IF ((ENDCOL+WIDTH-1).GT.LENGTH) ENDCOL=LENGTH-WIDTH+1
DO 130 XFRMCOL=STRTCOL,ENDCOL ,NSKIPS
CALL MIDDLE(LTRSIZE,XFRMCOL,WIDTH)
CALL XFORMIT(LTRSIZE,NIDTH,HEIGHT,ADJUST,WRATIO,HRATIO,NSPACE
%,NUMROWS,NUMCOLS,0,0,.F.,.F.,.F.,.F.,SENTNUM)
CALL FILTER(FORTLTR,NUMROWS,NUMCOLS,FLTRLTR,NSPACE)
IF((ENORM.EQ.1).OR.(DC.EQ.1)) CALL ENORMFL (FLTRLTR,FLTRLTR,NSPACE,
% DC)
CALL WRITMS(TAPENUM ,FLTRLTR,NSPACE,XFRMCOL)
130 CONTINUE
RETURN
END
SUBROUTINE GETFLTR (ALPHNUM,LTRNUM,LSPACE,NSPACE,TAPENUM,RANDOM)
INTEGER ALPHNUM,LTRNUM,LSPACE,NSPACE,TAPENUM
LOGICAL RANDOM
INTEGER RECFLTR
REAL FLTRLTR(180),PROTO(180)
COMMON/PROFLTR/FLTRLTR,PROTO,RECFLTR
INTEGER RECRQD,RECSKIP,KEY
DATA RECFLTR/1/
NUMWRDS=LSPACE+4
IF (.NOT.RANDOM) GO TO 95
KEY=(ALPHNUM-1)*26+LTRNUM
CALL READMS(TAPENUM,FLTRLTR,NUMWRDS,KEY)
GO TO 150
95 CONTINUE
RECRQD=(ALPHNUM-1)*26+LTRNUM
RECSKIP=IABS(RECRQD-RECFLTR)
IF (RECRQD-RECFLTR) 100,140,120
100 DO 110 I=1,RECSKIP
110 BACKSPACE TAPENUM
GO TO 140
120 DO 130 I=1,RECSKIP
130 READ (TAPENUM)
140 READ (TAPENUM) (FLTRLTR(I),I=1,NUMWRDS)
RECFLTR=RECRQD+1
150 CALL REDUCE (FLTRLTR,LSPACE,FLTRLTR,NSPACE)
RETURN
END
SUBROUTINE GETFORM(LTRSIZE,SENTNUM,TAPENUM)

```

AD-A069 298 AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OHIO SCH--ETC F/G 9/2

SEGMENTATION OF TOUCHING ENGLISH LETTERS.(U)

MAR 79 R E BENTKOWSKI

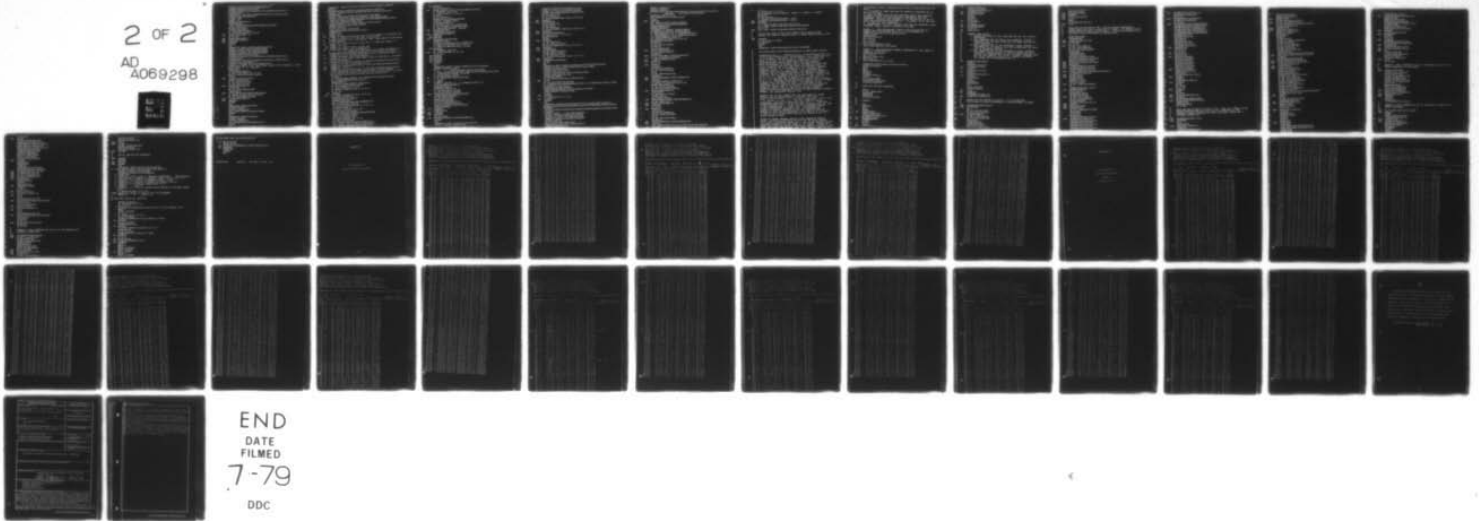
AFIT/GE/EE/79-1

UNCLASSIFIED

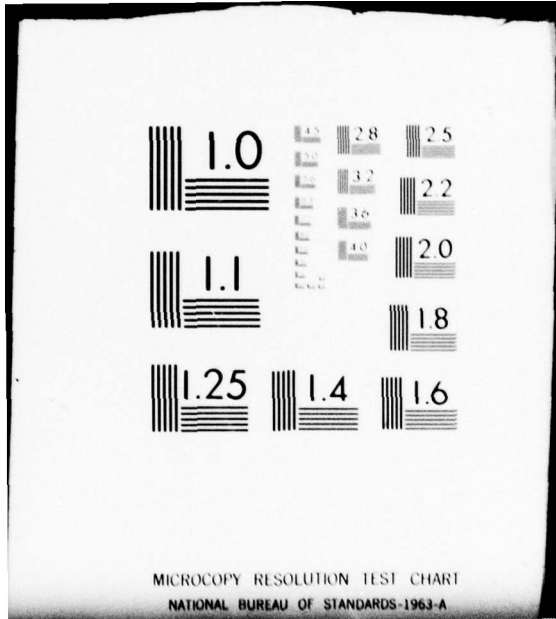
NL

2 OF 2

AD
A069298



END
DATE
FILMED
7-79
DDC



```
SUBROUTINE GETSENT(LTRSIZE,SENTNUM,TAPENUM)
INTEGER LENGTH,LTRSIZE,SENTNUM,TAPENUM
COMMON/SENTNCE/SAMPLE,HOLDIT
INTEGER SAMPLE(32,133),HOLDIT(254),NUMCHAR,TITLE(30,4),
% HOLDSSEN(133)
EQUIVALENCE (HOLDIT(1),NUMCHAR),(HOLDIT(2),TITLE(1,1)),
% (HOLDIT(122),HOLDSSEN(1))
```

```
INTEGER TEMP
LENGTH =133
NUMWRDS=LENGTH+120+1
MASKER=MASK(1)
CALL READMS(TAPENUM,HOLDIT,NUMWRDS,SENTNUM)
DO 130 NCOL=1,133
TEMP=HOLDSSEN(NCOL)
DO 130 NROW=1,LTRSIZE
IF (TEMP.AND.MASKER) 110,100
100 SAMPLE(NROW,NCOL) =0
GO TO 120
110 SAMPLE(NROW,NCOL)=1
120 TEMP=SHIFT(TEMP,1)
130 CONTINUE
RETURN
END
```

```
SUBROUTINE MIDDLE(LTRSIZE,XFRMCOL,WIDTH)
INTEGER LENGTH,LTRSIZE,XFRMCOL,WIDTH
COMMON/LETTERS/LETTER,NUMLTRS,RECNUM
INTEGER LETTER(32,32),NUMLTRS,RECNUM
COMMON/SENTNCE/SAMPLE,HOLDIT
INTEGER SAMPLE(32,133),HOLDIT(254),NUMCHAR,TITLE(30,4),
% HOLDSSEN(133)
EQUIVALENCE (HOLDIT(1),NUMCHAR),(HOLDIT(2),TITLE(1,1)),
% (HOLDIT(122),HOLDSSEN(1))
INTEGER STOP,NROW,NCOL,LEDGE,REDGE,TEMP
1 FORMAT(1H0,"WIDTH ("I3") PLUS START COLUMN ("I3") EXCEEDS THE SENT
%ENCE LENGTH OF "I3" COLUMNS")
```

```
LENGTH=133
STOP=XFRMCOL+WIDTH-1
IF (STOP.LE.LENGTH) GO TO 100
WRITE 1,(WIDTH,XFRMCOL,LENGTH)
RETURN
100 LEDGE=(LTRSIZE-WIDTH)/2
REDGE=LEDGE+WIDTH+1
DO 140 NROW=1,LTRSIZE
DO 110 NCOL=1,LEDGE
110 LETTER(NROW,NCOL)=0
TEMP=LEDGE
DO 120 NCOL=XFRMCOL,STOP
TEMP=TEMP+1
120 LETTER(NROW,TEMP)=SAMPLE(NROW,NCOL)
IF (REDGE.GT.LTRSIZE) GO TO 140
DO 130 NCOL=REDGE,LTRSIZE
130 LETTER(NROW,NCOL)=0
140 CONTINUE
RETURN
END
```

```
SUBROUTINE PREPRTO(PROTAPE)
INTEGER PROTAPE
INTEGER PINDEX(288)
CALL OPENMS(PROTAPE,PINDEX,288,0)
RETURN
END
```

```
SUBROUTINE PREPSEN(SENTAPE)
INTEGER SENTAPE
INTEGER SINDEXT(244)
CALL OPENMS(SENTAPE,SINDEXT,244,0)
RETURN
END
```

```

SUBROUTINE PRTOTBL (LTRSTRT, LTRSTOP, TOTPRTO, PRINTIT, SUMMARY,
%SKIPAGE,
%OVERRIDE, OSPACE, OENORM, ODC, TAPENUM, NEWTAPE, TAPETMP)
INTEGER LTRSTRT, LTRSTOP, TOTPRTO, TAPENUM, TAPETMP, OSPACE(10)
%, OENORM, ODC
LOGICAL PRINTIT, SKIPAGE, OVERRIDE, NEWTAPE, SUMMARY
INTEGER LETTERS(26), MASKER, LTRNUM, KEY, SUBLTRS, PROTNUM,
% NSPACE, PSPACE, NCOL, NROW, NALPHS, NTEMP, ALPHNUM, ALPHSTR, ALPHSTP,
% ALPH(150), ENORM, DC, NINDEX(300)
INTEGER MTABLE(27,5), TABLE(15,10), ETABLE(260,14)
COMMON/TABLES/ MTABLE, TABLE, ETABLE
INTEGER RECFLTR
REAL FLTRLTR(180), PROTO(180)
COMMON/PROFLTR/FLTRLTR, PROTO, RECFLTR
1   FORMAT (1H0)
5   FORMAT (1H1)
10  FORMAT (1H ,41X,54("")), /, 1H ,41X, "CHARACTERISTICS OF PROTOTYPE FIL
%E "I2", LETTERS "I2" TO "I2, /, 1H ,41X, 54(")")
11  FORMAT (1H0,42(" ")) / 1H , "LTR SUBLTR RANGE SPACE ENORM DC TERM #ALPH
% "/1H ,42(" "))
12  FORMAT (1H , "CLASS #"I2", LETTER "A1" *** DOES NOT EXIST***")
15  FORMAT (= (" "))
20  FORMAT (1H , "CLASS #"I2", LETTER "A1")
25  FORMAT (1H0, "TOTAL # OF ALPHABETS="I3
% " " # OF (POSSIBLE) SUBCLASSES="I2, /" RANGE OF (POSSIBLE) "
% "SUBCLASSES IN ASCENDING ORDER IS:"I2, "-"I2,9(", ", I2, "-"I2))
30  FORMAT (1H , "# OF (ACTUAL) PROTOTYPES ="I2". CHARACTERISTICS FOR "
% "THESE PROTOTYPES ARE SHOWN BELOW." / )
35  FORMAT (1H , "LTR SUBLTR RANGE SPACE ENORM DC TERM #ALPH ALPHABETS
% IN THE PROTOTYPE")
40  FORMAT (1H , 1X, A1, 3X, I2, 3X, I2, "-" , I2, 3X, I3, 3X, A3, 4X, A3, 4X, I3, 3X,
% 22(I3, ", ") /, 6(1H ,44X, 22(I3, ", ") /)
41  FORMAT (1H , 1X, A1, 3X, I2, 3X, I2, "-" , I2, 3X, I3, 3X, A3, 4X, A3, 4X, I3, 3X)
45  FORMAT (1H ,42("-"))
50  FORMAT (1H , 1X, A1, 3X, I2, 3X, I2, "-" , I2, 3X, I3, 3X, A3, 4X, A3, 4X, I3, 3X,
% "THIS SUBLTR DOES NOT EXIST, NO ALPHABETS IN THE RANGE")
DATA LETTERS/"A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M",
% "N", "O", "P", "Q", "R", "S", "T", "U", "V", "W", "X", "Y", "Z"/
TOTPRTO=0
IF (OVERRIDE.AND.NEWTAPE) CALL OPENMS (TAPETMP, NINDEX, 288, 0)
CALL READMS (TAPENUM, MTABLE, 135, 1)
MASKER=MASK(1)
IF (PRINTIT.AND.SUMMARY) PRINTIT=.F.
DO 160 LTRNUM=LTRSTRT, LTRSTOP
IF (.NOT.PRINTIT) GO TO 85
IF (MTABLE(LTRNUM,1).EQ.1) GO TO 80
GO TO 150
80  CONTINUE
85  IF (MTABLE(LTRNUM,1).EQ.0) GO TO 160
KEY=MTABLE(LTRNUM,2)
SUBLTRS=MTABLE(LTRNUM,3)
NUMWRDS=SUBLTRS*15
CALL READMS (TAPENUM, TABLE, NUMWRDS, KEY)
DO 150 PROTNUM=1, SUBLTRS
IF (TABLE(1,PROTNUM).EQ.0) GO TO 142
TOTPRTO=TOTPRTO +1
IF (.NOT. OVERRIDE) GO TO 100
PSPACE=TABLE(8,PROTNUM)
TABLE(8,PROTNUM)=NSPACE=OSPACE(PROTNUM)
TABLE(9,PROTNUM)=OENORM
TABLE(10,PROTNUM)=ODC
IF (.NOT.NEWTAPE) GO TO 100
KEY=TABLE(2,PROTNUM)
CALL READMS (TAPENUM, PROTO, PSPACE, KEY)
CALL REDUCE (PROTO, PSPACE, PROTO, NSPACE)
CALL WRITMS (TAPETMP, PROTO, NSPACE, KEY)

```

```

100 CONTINUE
DO 110 NCOL=2,15
110  ETABLE(TOTPRTO,NCOL-1)=TABLE(NCOL,PROTNUM)
    IF (SUMMARY) GO TO 142
    IF (.NOT.PRINTIT) GO TO 150
    NALPHS=0
    DO 140 NWRDS=1,3
    NTEMP=TABLE(12+NWRDS,PROTNUM)
    ALPHSTR= (NWRDS-1)*60+1
    ALPHSTP=ALPHSTR+59
    IF (ALPHSTP.GT.150) ALPHSTP=150
    DO 130 ALPHNUM= ALPHSTR,ALPHSTP
    IF (NTEMP.AND.MASKER) 125,127
125  NALPHS=NALPHS+1
    ALPH(NALPHS)=ALPHNUM
127  NTEMP=SHIFT(NTEMP,1)
    130 CONTINUE
    140 CONTINUE
142  NCOL=TABLE(12,PROTNUM)
    ENORM=DC="OUT"
    IF (TABLE(9,PROTNUM).EQ.1) ENORM="IN "
    IF (TABLE(10,PROTNUM).EQ.1) DC="IN "
    IF (TABLE(1,PROTNUM).EQ.0) 145,147
    145 CONTINUE
    IF (SUMMARY) 150,149
147  IF (.NOT.SUMMARY) GO TO 148
    GO TO 150
    148 CONTINUE
    149 CONTINUE
    150 CONTINUE
    160 CONTINUE
    RETURN
    END
SUBROUTINE REDUCE (INFLTR,INSPACE,OUTFLTR,OTSPACE)
INTEGER INSPACE,OTSPACE
REAL      INFLTR(INSPACE),OUTFLTR(OTSPACE)
INTEGER INHRMNC,OUTHRMC,SKIP,DOWN,OUTVCTR,INVCTR,START,STOP
IF (INSPACE.NE.OTSPACE) GO TO 90
DC 85 NVECTOR=1,OTSPACE
85  OUTFLTR(NVECTOR)=INFLTR(NVECTOR)
    GO TO 125
90  CONTINUE
    INHRMNC= INSPACE** .5 $  OUTHRMC=OTSPACE** .5
    SKIP=(INHRMNC-OUTHRMC)*2
    DOWN=OUTHRMC/2+1
    OUTFLTR(1)=INFLTR(1)
    IF (OTSPACE.EQ.1) RETURN
    DO 100 OUTVCTR=2,OUTHRMC
100  OUTFLTR(OUTVCTR)=INFLTR(OUTVCTR)
    INVCTR=START=OUTHRMC+1
    DO 120 NROW=2,DOWN
    INVCTR=INVCTR+SKIP
    STOP=START+OUTHRMC*2-1
    DO 110 OUTVCTR=START,STOP
    OUTFLTR(OUTVCTR)=INFLTR(INVCTR)
    INVCTR=INVCTR+1
110  CONTINUE
    START=STOP+1
120  CONTINUE
125  CONTINUE
    DO 130 I=1,4
130  OUTFLTR(OTSPACE+I)=INFLTR(INSPACE+I)
    RETURN
    END
SUBROUTINE SIZEIT (LTRSIZE,WIDTH,HEIGHT)
INTEGER LTRSIZE WIDTH HEIGHT

```

```

INTEGER LTRSIZE,WIDTH,HEIGHT
COMMON/LETTERS/LETTER,NUMLTRS,RECNUM
INTEGER LETTER(32,32),NUMLTRS,RECNUM
INTEGER NCOL,NROW,STRTCOL,TEMP,STRTRON
DO 100 NCOL=1,32
DO 100 NROW=1,32
IF (LETTER(NROW,NCOL).EQ.1) GO TO 110
100 CONTINUE
110 STRTCOL=NCOL
DO 120 NCOL=1,32
TEMP=33-NCOL
DO 120 NROW=1,32
IF (LETTER(NROW,TEMP).EQ.1) GO TO 130
120 CONTINUE
130 WIDTH=TEMP-STRTCOL+1
IF (WIDTH.GE.1) GO TO 135
WIDTH=HEIGHT=0
RETURN
135 CONTINUE
DO 140 NROW=1,32
DO 140 NCOL=1,LTRSIZE
IF (LETTER(NROW,NCOL).EQ.1) GO TO 150
140 CONTINUE
150 STRTRON=NROW
DO 160 NROW=1,LTRSIZE
TEMP=LTRSIZE+1-NROW
DO 160 NCOL=1,LTRSIZE
IF (LETTER(TEMP,NCOL).EQ.1) GO TO 170
160 CONTINUE
170 HEIGHT=TEMP-STRTRON+1
RETURN
END
SUBROUTINE SORTIT(STRTCOL,STOPCOL,NSKIPS,STRTRON,STOPROW)
INTEGER STRTCOL,STOPCOL,NSKIPS,STRTRON,STOPROW
COMMON/RESULTS/RESULTS
INTEGER RESULTS(52,133)
INTEGER FLAG,XFRMCOL,TEMP,LASTROW,DXMASK
DXMASK=SHIFT(MASK(22),22)
LASTROW=STOPROW-1
DO 120 XFRMCOL=STRTCOL,STOPCOL,NSKIPS
100 FLAG=0
DO 110 NROW=STRTRON,LASTROW
IF ((DXMASK.AND.RESULTS(NROW,XFRMCOL)).LE.(DXMASK.AND.RESULTS(NROW
%+1,XFRMCOL))) GO TO 110
TEMP=RESULTS(NROW,XFRMCOL)
RESULTS(NROW,XFRMCOL)=RESULTS(NROW+1,XFRMCOL)
RESULTS(NROW+1,XFRMCOL)=TEMP
FLAG=1
110 CONTINUE
IF (FLAG.EQ.1) GO TO 100
120 CONTINUE
RETURN
END
SUBROUTINE XFORMIT(LTRSIZE,WIDTH,HEIGHT,ADJUST,WRATIO,HRATIO,
%NSPACE,NUMROWS,NUMCOLS,ALPHNUM,LTRNUM,VEXPAND,XFORM,IMAGE,INVERSE,
%SENTNUM)
INTEGER LTRSIZE,WIDTH,HEIGHT,NSPACE,NUMROWS,NUMCOLS,ALPHNUM,LTRNUM
%,SENTNUM
LOGICAL ADJUST $ REAL WRATIO,HRATIO
LOGICAL VEXPAND,XFORM,IMAGE,INVERSE
INTEGER LETTER(32,32),NUMLTRS,RECNUM
COMMON/LETTERS/LETTER,NUMLTRS,RECNUM
INTEGER RECFLTR
REAL FLTRLTR(180),PROTO(180)
COMMON/PROFLTR/FLTRLTR,PROTO,RECFLTR
COMMON/XFORMS/FORTLTR

```

```

COMPLEX WORKER(64)
INTEGER POWERS(7)
INTEGER NN(2), NUMCOLS, NUMROWS, NCOL, NROW, FLAG, TOP, BOTTOM, ITEMP,
%          COL, ROW,          LEFT, RIGHT, IFORM, ISIGN
DIMENSION TRPLTR(32,32)
DATA POWERS/2,4,8,16,32,64,128/
FLAG=0
100 CONTINUE
CALL SIZEIT(LTRSIZE,WIDTH,HEIGHT)
CALL TRPLT(LETTER,TRPLTR,SENTNUM)
NUMROWS=NUMCOLS=32
NRINGS=NSPACE*.5
IF(NUMROWS.LT.NRINGS) NUMROWS=NRINGS
IF(NUMCOLS.LT.NRINGS) NUMCOLS=NRINGS
IF(MOD(NUMROWS,2).EQ.1) NUMROWS=NUMROWS+1
IF(MOD(NUMCOLS,2).EQ.1) NUMCOLS=NUMCOLS+1
TOP=IABS(NUMROWS-LTRSIZE)/2
IF(NUMROWS.LT.LTRSIZE) GO TO 150
BOTTOM=TOP+1+LTRSIZE
ITEMP=TOP
DO 140 NCOL=1,LTRSIZE
TOP=ITEMP
DO 110 NROW=1, TOP
110 FORTLTR(NROW,NCOL)=0
DO 120 NROW=1,LTRSIZE
TOP=TOP+1
120 FORTLTR(TOP,NCOL)=TRPLTR(NROW,NCOL)
DO 130 NROW=BOTTOM,NUMROWS
130 FORTLTR(NROW,NCOL)=0
140 CONTINUE
GO TO 170
150 CONTINUE
TOP=TOP+1
BOTTOM=TOP+NUMROWS-1
DO 160 NCOL=1,LTRSIZE
ROW=0
DO 160 NROW=TOP,BOTTOM
ROW=ROW+1
160 FORTLTR(ROW,NCOL)=TRPLTR(NROW,NCOL)
170 CONTINUE
LEFT=IABS(NUMCOLS-LTRSIZE)/2
IF(NUMCOLS.LT.LTRSIZE) GO TO 220
RIGHT=LEFT+LTRSIZE+1
DO 210 NROW=1,NUMROWS
DO 180 NCOL=RIGHT,NUMCOLS
180 FORTLTR(NROW,NCOL)=0
ITEMP=RIGHT
DO 190 NCOL=1,LTRSIZE
ITEMP=ITEMP-1
COL=LTRSIZE-NCOL+1
190 FORTLTR(NROW,ITEMP)=FORTLTR(NROW,COL)
DO 200 NCOL=1,LEFT
200 FORTLTR(NROW,NCOL)=0
210 CONTINUE
GO TO 232
220 CONTINUE
LEFT=LEFT+1
RIGHT=LEFT+NUMCOLS-1
DO 230 NROW=1,NUMROWS
COL=0
DO 230 NCOL=LEFT,RIGHT
COL=COL+1
230 FORTLTR(NROW,COL)=FORTLTR(NROW,NCOL)
232 CONTINUE
CALL ARNGCPX(FORTLTR,64,64,FORTLTR,NUMROWS,NUMCOLS)
235 CONTINUE

```

```

230 CONTINUE
IF (FLAG.GT.0) GO TO 245
NN(1)=NUMROWS $ NN(2)=NUMCOLS $NDIM=2 $ ISIGN=-1 $ IFORM=0
COL=ROW=0
DO 240 ITEMP=1,7
IF (NUMROWS.EQ.POWERS(ITEMP)) ROW=1
240 IF (NUMCOLS.EQ.POWERS(ITEMP)) COL=1
245 CONTINUE
IF ( ROW.EQ.1.AND. COL.EQ.1) GO TO 250
CALL FOURT (FORTLTR,NN,NDIM,ISIGN,IFORM,WORKER)
GO TO 255
250 CALL FOJRT (FORTLTR,NN,NDIM,ISIGN,IFORM,0.0)
C
C THIS IS THE PLACE TO PUT IN AN INVERSE PRINT ROUTINE THAT
C FILTERS THE FOURIER TRANSFORM FROM THE MIDDLE AND THEN CALL 3LPRT
C
255 CONTINUE
IF (FLAG.GT.0) RETURN
FLAG=FLAG+1
RETURN
END
SUBROUTINE FOURT (DATA,NN,NDIM,ISIGN,IFORM,WORK)

```

```

C
C THE COOLEY-TUKEY FAST FOURIER TRANSFORM IN USASI BASIC FORTRAN
C
C TRANSFORM(K1,K2,...) = SUM(DATA(J1,J2,...)*EXP(ISIGN*2*PI*SQRT(-1)
C *((J1-1)*(K1-1)/NN(1)+(J2-1)*(K2-1)/NN(2)+...))), SUMMED FOR ALL
C U1, K1 BETWEEN 1 AND NN(1), J2, K2 BETWEEN 1 AND NN(2), ETC.
C THERE IS NO LIMIT TO THE NUMBER OF SUBSCRIPTS. DATA IS A
C MULTIDIMENSIONAL COMPLEX ARRAY WHOSE REAL AND IMAGINARY
C PARTS ARE ADJACENT IN STORAGE, SUCH AS FORTRAN IV PLACES THEM.
C IF ALL IMAGINARY PARTS ARE ZERO (DATA ARE DISGUISED REAL), SET
C IFORM TO ZERO TO CUT THE RUNNING TIME BY UP TO FORTY PERCENT.
C OTHERWISE, IFORM = +1. THE LENGTHS OF ALL DIMENSIONS ARE
C STORED IN ARRAY NN, OF LENGTH NDIM. THEY MAY BE ANY POSITIVE
C INTEGERS, THO THE PROGRAM RUNS FASTER ON COMPOSITE INTEGERS, AND
C ESPECIALLY FAST ON NUMBERS RICH IN FACTORS OF TWO. ISIGN IS +1
C OR -1. IF A -1 TRANSFORM IS FOLLOWED BY A +1 ONE (OR A +1
C BY A -1) THE ORIGINAL DATA REAPPEAR, MULTIPLIED BY NTOT (=NN(1)*
C NN(2)*...). TRANSFORM VALUES ARE ALWAYS COMPLEX, AND ARE RETURNED
C IN ARRAY DATA, REPLACING THE INPUT. IN ADDITION, IF ALL
C DIMENSIONS ARE NOT POWERS OF TWO, ARRAY WORK MUST BE SUPPLIED,
C COMPLEX OF LENGTH EQUAL TO THE LARGEST NON 2**K DIMENSION.
C OTHERWISE, REPLACE WORK BY ZERO IN THE CALLING SEQUENCE.
C NORMAL FORTRAN DATA ORDERING IS EXPECTED, FIRST SUBSCRIPT VARYING
C FASTEST. ALL SUBSCRIPTS BEGIN AT ONE.

```

```

C
C RUNNING TIME IS MUCH SHORTER THAN THE NAIVE NTOT**2, BEING
C GIVEN BY THE FOLLOWING FORMULA. DECOMPOSE NTOT INTO
C 2**K2 * 3**K3 * 5**K5 * ..... LET SUM2 = 2**K2, SUMF = 3**K3 + 5**K5
C + ... AND NF = K3 + K5 + ..... THE TIME TAKEN BY A MULTI-
C DIMENSIONAL TRANSFORM ON THESE NTOT DATA IS T = T0 + NTOT*(T1+
C T2*SUM2+T3*SUMF+T4*NF). ON THE CDC 3300 (FLOATING POINT ADD TIME
C OF SIX MICROSECONDS), T = 3000 + NTOT*(500+43*SUM2+58*SUMF+
C 320*NF) MICROSECONDS ON COMPLEX DATA. IN ADDITION, THE
C ACCURACY IS GREATLY IMPROVED, AS THE RMS RELATIVE ERROR IS
C BOUNDED BY 3*2**(-B)*SUM(FACTOR(J)**1.5), WHERE B IS THE NUMBER
C OF BITS IN THE FLOATING POINT FRACTION AND FACTOR(J) ARE THE
C PRIME FACTORS OF NTOT.

```

```

C
C PROGRAM BY NORMAN BRENNER FROM THE BASIC PROGRAM BY CHARLES
C RADER. RALPH ALTER SUGGESTED THE IDEA FOR THE DIGIT REVERSAL.
C MIT LINCOLN LABORATORY, AUGUST 1967. THIS IS THE FASTEST AND MOST
C VERSATILE VERSION OF THE FFT KNOWN TO THE AUTHOR. SHORTER PRO-
C GRAMS FOUR1 AND FOUR2 RESTRICT DIMENSION LENGTHS TO POWERS OF TWO.
C SEE -- IEEE AUDIO TRANSACTIONS (JUNE 1967), SPECIAL ISSUE ON FFT.

```

C THE DISCRETE FOURIER TRANSFORM PLACES THREE RESTRICTIONS UPON THE
C DATA.

C 1. THE NUMBER OF INPUT DATA AND THE NUMBER OF TRANSFORM VALUES
C MUST BE THE SAME.

C 2. BOTH THE INPUT DATA AND THE TRANSFORM VALUES MUST REPRESENT
C EQUISPACED POINTS IN THEIR RESPECTIVE DOMAINS OF TIME AND
C FREQUENCY. CALLING THESE SPACINGS DELTAT AND DELTAF, IT MUST BE
C TRUE THAT $DELTAF = 2 * \pi / (NN(I) * DELTAT)$. OF COURSE, DELTAT NEED NOT
C BE THE SAME FOR EVERY DIMENSION.

C 3. CONCEPTUALLY AT LEAST, THE INPUT DATA AND THE TRANSFORM OUTPUT
C REPRESENT SINGLE CYCLES OF PERIODIC FUNCTIONS.

C EXAMPLE 1. THREE-DIMENSIONAL FORWARD FOURIER TRANSFORM OF A
C COMPLEX ARRAY DIMENSIONED 32 BY 25 BY 13 IN FORTRAN IV.

C DIMENSION DATA(32,25,13),WORK(50),NN(3)

C COMPLEX DATA

C DATA NN/32,25,13/

C DO 1 I=1,32

C DO 1 J=1,25

C DO 1 K=1,13

C 1 DATA(I,J,K)=COMPLEX VALUE

C CALL FOURT(DATA,NN,3,-1,1,WORK)

C EXAMPLE 2. ONE-DIMENSIONAL FORWARD TRANSFORM OF A REAL ARRAY OF
C LENGTH 64 IN FORTRAN II.

C DIMENSION DATA(2,64)

C DO 2 I=1,64

C DATA(1,I)=REAL PART

C 2 DATA(2,I)=0.

C CALL FOURT(DATA,64,1,-1,0,0)

C DIMENSION DATA(1),NN(1),IFACT(32),WORK(1)

C TWOPI=6.283185307

C WR=0.

C WI=0.

C WSTPI=0.

C WSTPR=0.

C IF(NDIM-1)920,1,1

C 1 NTOT=2

C DO 2 IDIM=1,NDIM

C IF(NN(IDIM))920,920,2

C 2 NTOT=NTOT*NN(IDIM)

C MAIN LOOP FOR EACH DIMENSION

C NP1=2

C DO 910 IDIM=1,NDIM

C N=NN(IDIM)

C NP2=NP1*N

C IF(N-1)920,900,5

C FACTOR N

C 5 M=N

C NTWO=NP1

C IF=1

C IDIV=2

C 10 IQUOT=M/IDIV

C IREM=M-IDIV*IQUOT

C IF(IQUOT-IDIV)50,11,11

C 11 IF(IREM)20,12,20

C 12 NTWO=NTWO+NTWO

C M=IQUOT

C GO TO 10

C 20 IDIV=3

C IQUOT=M/IDIV

```

30  IQUOT=M/IDIV
    IREM=M-IDIV*IQUOT
    IF(IQUOT-IDIV) 60, 31, 31
31  IF(IREM) 40, 32, 40
32  IFACT(IF)=IDIV
    IF=IF+1
    M=IQUOT
    GO TO 30
40  IDIV=IDIV+2
    GO TO 30
50  IF(IREM) 60, 51, 60
51  NTHO=NTHO+NTWO
    GO TO 70
60  IFACT(IF)=M
C
C  SEPARATE FOUR CASES--
C    1. COMPLEX TRANSFORM OR REAL TRANSFORM FOR THE 4TH, 5TH, ETC.
C      DIMENSIONS.
C    2. REAL TRANSFORM FOR THE 2ND OR 3RD DIMENSION. METHOD--
C      TRANSFORM HALF THE DATA, SUPPLYING THE OTHER HALF BY CON-
C      JUGATE SYMMETRY.
C    3. REAL TRANSFORM FOR THE 1ST DIMENSION, N ODD. METHOD--
C      TRANSFORM HALF THE DATA AT EACH STAGE, SUPPLYING THE OTHER
C      HALF BY CONJUGATE SYMMETRY.
C    4. REAL TRANSFORM FOR THE 1ST DIMENSION, N EVEN. METHOD--
C      TRANSFORM A COMPLEX ARRAY OF LENGTH N/2 WHOSE REAL PARTS
C      ARE THE EVEN NUMBERED REAL VALUES AND WHOSE IMAGINARY PARTS
C      ARE THE ODD NUMBERED REAL VALUES. SEPARATE AND SUPPLY
C      THE SECOND HALF BY CONJUGATE SYMMETRY.
C
70  NON2=NP1*(NP2/NTWO)
    ICASE=1
    IF(IDIM-4) 71, 90, 90
71  IF(IFORM) 72, 72, 90
72  ICASE=2
    IF(IDIM-1) 73, 73, 90
73  ICASE=3
    IF(NTWO-NP1) 90, 90, 74
74  ICASE=4
    NTWO=NTWO/2
    N=N/2
    NP2=NP2/2
    NTOT=NTOT/2
    I=3
    DO 80 J=2, NTOT
    DATA(J)=DATA(I)
80  I=I+2
90  I1RNG=NP1
    IF(ICASE-2) 100, 95, 100
95  I1RNG=NP0*(1+NPREV/2)
C
C  SHUFFLE ON THE FACTORS OF TWO IN N. AS THE SHUFFLING
C  CAN BE DONE BY SIMPLE INTERCHANGE, NO WORKING ARRAY IS NEEDED
C
100 IF(NTWO-NP1) 600, 600, 110
110 NP2HF=NP2/2
    U=1
    DO 150 I2=1, NP2, NON2
    IF(J-I2) 120, 130, 130
120 I1MAX=I2+NON2-2
    DO 125 I1=I2, I1MAX, 2
    DO 125 I3=I1, NTOT, NP2
    U3=J+I3-I2
    TEMPR=DATA(I3)
    TEMPI=DATA(I3+1)
    DATA(I3)=DATA(J3)
    DATA(I3+1)=DATA(J3+1)

```

```

DATA(J3)=TEMPR
125 DATA(J3+1)=TEMPI
130 M=NP2HF
140 IF(J-M)150,150,145
145 U=J-M
M=M/2
IF(M-NON2)150,140,140
150 U=J+M
C
C MAIN LOOP FOR FACTORS OF TWO. PERFORM FOURIER TRANSFORMS OF
C LENGTH FOUR, WITH ONE OF LENGTH TWO IF NEEDED. THE TWIDDLE FACTOR
C W=EXP(ISIGN*2*PI*SQRT(-1)*M/(4*MMAX)). CHECK FOR W=ISIGN*SQRT(-1)
C AND REPEAT FOR W=ISIGN*SQRT(-1)*CONJUGATE(W).
C
NON2T=NON2+NON2
IPAR=NTWO/NP1
310 IF(IPAR-2)350,330,320
320 IPAR=IPAR/4
GO TO 310
330 DO 340 I1=1,I1RNG,2
DO 340 J3=I1,NON2,NP1
DO 340 K1=J3,NTOT,NON2T
K2=K1+NON2
TEMPR=DATA(K2)
TEMPI=DATA(K2+1)
DATA(K2)=DATA(K1)-TEMPR
DATA(K2+1)=DATA(K1+1)-TEMPI
DATA(K1)=DATA(K1)+TEMPR
340 DATA(K1+1)=DATA(K1+1)+TEMPI
350 MMAX=NON2
360 IF(MMAX-NP2HF)370,600,600
370 LMAX=MAX0(NON2T,MMAX/2)
IF(MMAX-NON2)405,405,380
380 THETA=-TWOPI*FLOAT(NON2)/FLOAT(4*MMAX)
IF(ISIGN)400,390,390
390 THETA=-THETA
400 WR=COS(THETA)
WI=SIN(THETA)
WSTPR=-2.*WI*WI
WSTPI=2.*WR*WI
405 DO 570 L=NON2,LMAX,NON2T
M=L
IF(MMAX-NON2)420,420,410
410 W2R=WR*WR-WI*WI
W2I=2.*WR*WI
W3R=W2R*WR-W2I*WI
W3I=W2R*WI+W2I*WR
420 DO 530 I1=1,I1RNG,2
DO 530 J3=I1,NON2,NP1
KMIN=J3+IPAR*M
IF(MMAX-NON2)430,430,440
430 KMIN=J3
440 KDIF=IPAR*MMAX
450 KSTEP=4*KDIF
DO 520 K1=KMIN,NTOT,KSTEP
K2=K1+KDIF
K3=K2+KDIF
K4=K3+KDIF
IF(MMAX-NON2)460,460,480
460 U1R=DATA(K1)+DATA(K2)
U1I=DATA(K1+1)+DATA(K2+1)
U2R=DATA(K3)+DATA(K4)
U2I=DATA(K3+1)+DATA(K4+1)
U3R=DATA(K1)-DATA(K2)
U3I=DATA(K1+1)-DATA(K2+1)
IF(ISIGN)470,475,475

```

```

470  U4R=DATA(K3+1)-DATA(K4+1)
    U4I=DATA(K4)-DATA(K3)
    GO TO 510
475  U4R=DATA(K4+1)-DATA(K3+1)
    U4I=DATA(K3)-DATA(K4)
    GO TO 510
480  T2R=W2R*DATA(K2)-W2I*DATA(K2+1)
    T2I=W2R*DATA(K2+1)+W2I*DATA(K2)
    T3R=WR*DATA(K3)-WI*DATA(K3+1)
    T3I=WR*DATA(K3+1)+WI*DATA(K3)
    T4R=W3R*DATA(K4)-W3I*DATA(K4+1)
    T4I=W3R*DATA(K4+1)+W3I*DATA(K4)
    U1R=DATA(K1)+T2R
    U1I=DATA(K1+1)+T2I
    U2R=T3R+T4R
    U2I=T3I+T4I
    U3R=DATA(K1)-T2R
    U3I=DATA(K1+1)-T2I
    IF(ISIGN)490,500,500
490  U4R=T3I-T4I
    U4I=T4R-T3R
    GO TO 510
500  U4R=T4I-T3I
    U4I=T3R-T4R
510  DATA(K1)=U1R+U2R
    DATA(K1+1)=U1I+U2I
    DATA(K2)=U3R+U4R
    DATA(K2+1)=U3I+U4I
    DATA(K3)=U1R-U2R
    DATA(K3+1)=U1I-U2I
    DATA(K4)=U3R-U4R
    DATA(K4+1)=U3I-U4I
520  DATA(K4+1)=U3I-U4I
    KMIN=4*(KMIN-J3)+J3
    KDIF=KSTEP
    IF(KDIF-NP2)450,530,530
530  CONTINUE
    M=MMAX-M
    IF(ISIGN)540,550,550
540  TEMPR=WR
    WR=-WI
    WI=-TEMPR
    GO TO 550
550  TEMPR=WR
    WR=WI
    WI=TEMPR
560  IF(M-LMAX)565,565,410
565  TEMPR=WR
    WR=WR*WSTPR-WI*WSTPI+WR
    WI=WI*WSTPR+TEMPR*WSTPI+WI
570  IPAR=3-IPAR
    MMAX=MMAX+MMAX
    GO TO 350

```

```

C
C  MAIN LOOP FOR FACTORS NOT EQUAL TO TWO. APPLY THE TWIDDLE FACTOR
C  W=EXP(ISIGN*2*PI*SQRT(-1)*(J2-1)*(J1-J2)/(NP2*IFP1)), THEN
C  PERFORM A FOURIER TRANSFORM OF LENGTH IFACT(IF), MAKING USE OF
C  CONJUGATE SYMMETRIES.
C

```

```

600  IF(NTWO-NP2)605,700,700
605  IFP1=NON2
    IF=1
    NP1HF=NP1/2
610  IFP2=IFP1/IFACT(IF)
    U1RNG=NP2
    IF(ICASE-3)612,611,612

```

```

011 U2STP=NP2/IFACT(IF)
U1RG2=(J2STP+IFP2)/2
612 U2MIN=1+IFP2
IF(IFP1-NP2)615,640,640
615 DO 635 J2=J2MIN,IFP1,IFP2
THETA=-TWOPI*FLOAT(J2-1)/FLOAT(NP2)
IF(ISIGN)625,620,620
620 THETA=-THETA
625 SINTH=SIN(THETA/2.)
WSTPR=-2.*SINTH*SINTH
WSTPI=SIN(THETA)
WR=WSTPR+1.
WI=WSTPI
U1MIN=J2+IFP1
DO 635 J1=J1MIN,J1RNG,IFP1
I1MAX=J1+I1RNG-2
DO 630 I1=J1,I1MAX,2
DO 630 I3=I1,NTOT,NP2
U3MAX=I3+IFP2-NP1
DO 630 J3=I3,J3MAX,NP1
TEMPR=DATA(J3)
DATA(J3)=DATA(J3)*WR-DATA(J3+1)*WI
630 DATA(J3+1)=TEMPR*WI+DATA(J3+1)*WR
TEMPR=WR
WR=WR*WSTPR-WI*WSTPI+WR
635 WI=TEMPR*WSTPI+WI*WSTPR+WI
640 THETA=-TWOPI/IFACT(IF)
IF(ISIGN)650,645,645
645 THETA=-THETA
650 SINTH=SIN(THETA/2.)
WSTPR=-2.*SINTH*SINTH
WSTPI=SIN(THETA)
KSTEP=2*N/IFACT(IF)
KRANG=KSTEP*(IFACT(IF)/2)+1
DO 698 I1=1,I1RNG,2
DO 698 I3=I1,NTOT,NP2
DO 690 KMIN=1,KRANG,KSTEP
U1MAX=I3+J1RNG-IFP1
DO 680 J1=I3,J1MAX,IFP1
U3MAX=J1+IFP2-NP1
DO 680 J3=J1,J3MAX,NP1
U2MAX=J3+IFP1-IFP2
K=KMIN+(J3-J1+(J1-I3)/IFACT(IF))/NP1HF
IF(KMIN-1)655,655,665
655 SUMR=0.
SUMI=0.
DO 660 J2=J3,J2MAX,IFP2
SUMR=SUMR+DATA(J2)
660 SUMI=SUMI+DATA(J2+1)
WORK(K)=SUMR
WORK(K+1)=SUMI
GO TO 680
665 KCONJ=K+2*(N-KMIN+1)
U2=J2MAX
SUMR=DATA(J2)
SUMI=DATA(J2+1)
OLDSR=0.
OLDSI=0.
U2=J2-IFP2
670 TEMPR=SUMR
TEMPI=SUMI
SUMR=TWOHR*SUMR-OLDSR+DATA(J2)
SUMI=TWOHR*SUMI-OLDSI+DATA(J2+1)
OLDSR=TEMPR
OLDSI=TEMPI

```

```

IF (J2-J3) 675,675,670
675  TEMPR=WR*SUMR-OLDSR+DATA (J2)
    TEMPI=WI*SUMI
    WORK(K)=TEMPR-TEMPI
    WORK(KCONJ)=TEMPR+TEMPI
    TEMPR=WR*SUMI-OLDSI+DATA (J2+1)
    TEMPI=WI*SUMR
    WORK(K+1)=TEMPR+TEMPI
    WORK(KCONJ+1)=TEMPR-TEMPI
680  CONTINUE
    IF (KMIN-1) 685,685,686
685  WR=WSTPR+1.
    WI=WSTPI
    GO TO 690
686  TEMPR=WR
    WR=WR*WSTPR-WI*WSTPI+WR
    WI=TEMPR*WSTPI+WI*WSTPR+WI
690  TWOWR=WR+WR
    IF (ICASE-3) 692,691,692
691  IF (IFP1-NP2) 695,692,692
692  K=1
    I2MAX=I3+NP2-NP1
    DO 693 I2=I3,I2MAX,NP1
    DATA (I2)=WORK(K)
    DATA (I2+1)=WORK(K+1)
693  K=K+2
    GO TO 698

```

```

C
C  COMPLETE A REAL TRANSFORM IN THE 1ST DIMENSION, N ODD, BY CON-
C  JUGATE SYMMETRIES AT EACH STAGE.
C

```

```

695  U3MAX=I3+IFP2-NP1
    DO 697 J3=I3,J3MAX,NP1
    U2MAX=J3+NP2-J2STP
    DO 697 J2=J3,J2MAX,J2STP
    U1MAX=J2+J1RG2-IFP2
    U1CNJ=J3+J2MAX+J2STP-J2
    DO 697 J1=J2,J1MAX,IFP2
    K=1+J1-I3
    DATA (J1)=WORK(K)
    DATA (J1+1)=WORK(K+1)
    IF (J1-J2) 697,697,696
696  DATA (J1CNJ)=WORK(K)
    DATA (J1CNJ+1)=-WORK(K+1)
697  U1CNJ=J1CNJ-IFP2
698  CONTINUE
    IF=IF+1
    IFP1=IFP2
    IF (IFP1-NP1) 700,700,610

```

```

C
C  COMPLETE A REAL TRANSFORM IN THE 1ST DIMENSION, N EVEN, BY CON-
C  JUGATE SYMMETRIES.
C

```

```

700  GO TO (900,800,900,701),ICASE
701  NHALF=N
    N=N+N
    THETA=-TWOPI/FLOAT(N)
    IF (ISIGN) 703,702,702
702  THETA=-THETA
703  SINTH=SIN(THETA/2.)
    WSTPR=-2.*SINTH*SINTH
    WSTPI=SIN(THETA)
    WR=WSTPR+1.
    WI=WSTPI
    IMIN=3
    IUNT=2*NHALF-1

```

```

GO TO 725
710 U=JMIN
DO 720 I=IMIN,NTOT,NP2
SUMR=(DATA(I)+DATA(J))/2.
SUMI=(DATA(I+1)+DATA(J+1))/2.
DIFR=(DATA(I)-DATA(J))/2.
DIFI=(DATA(I+1)-DATA(J+1))/2.
TEMPR=WR*SUMI+WI*DIFR
TEMPI=WI*SUMI-WR*DIFR
DATA(I)=SUMR+TEMPR
DATA(I+1)=DIFI+TEMPI
DATA(J)=SUMR-TEMPR
DATA(J+1)=-DIFI+TEMPI
720 U=J+NP2
IMIN=IMIN+2
JMIN=JMIN-2
TEMPR=WR
WR=WR*WSTPR-WI*WSTPI+WR
WI=TEMPR*WSTPI+WI*WSTPR+WI
725 IF(IMIN-JMIN)710,730,740
730 IF(ISIGN)731,740,740
731 DO 735 I=IMIN,NTOT,NP2
735 DATA(I+1)=-DATA(I+1)
740 NP2=NP2+NP2
NTOT=NTOT+NTOT
U=NTOT+1
IMAX=NTOT/2+1
745 IMIN=IMAX-2*NHALF
I=IMIN
GO TO 755
750 DATA(J)=DATA(I)
DATA(J+1)=-DATA(I+1)
755 I=I+2
J=J-2
IF(I-IMAX)750,760,760
760 DATA(J)=DATA(IMIN)-DATA(IMIN+1)
DATA(J+1)=0.
IF(I-J)770,780,780
765 DATA(J)=DATA(I)
DATA(J+1)=DATA(I+1)
770 I=I-2
J=J-2
IF(I-IMIN)775,775,765
775 DATA(J)=DATA(IMIN)+DATA(IMIN+1)
DATA(J+1)=0.
IMAX=IMIN
GO TO 745
780 DATA(1)=DATA(1)+DATA(2)
DATA(2)=0.
GO TO 900

```

```

C
C COMPLETE A REAL TRANSFORM FOR THE 2ND OR 3RD DIMENSION BY
C CONJUGATE SYMMETRIES.
C

```

```

800 IF(I1RNG-NP1)805,900,900
805 DO 860 I3=1,NTOT,NP2
I2MAX=I3+NP2-NP1
DO 860 I2=I3,I2MAX,NP1
IMIN=I2+I1RNG
IMAX=I2+NP1-2
JMAX=2*I3+NP1-IMIN
IF(I2-I3)820,820,810
810 JMAX=JMAX+NP2
820 IF(IDIM-2)850,850,830
830 U=JMAX+NP0
DO 840 I=IMIN,IMAX,2

```

```

DATA(I)=DATA(J)
DATA(I+1)=-DATA(J+1)
840 U=J-2
850 U=JMAX
DO 860 I=IMIN,IMAX,NP0
DATA(I)=DATA(J)
DATA(I+1)=-DATA(J+1)
860 U=J-NP0
C
C END OF LOOP ON EACH DIMENSION
C
900 NP0=NP1
NP1=NP2
910 NPREV=N
920 RETURN
END
SUBROUTINE TRPLT(LETTER,TRPLTR,SENTNUM)
C THIS ROUTINE DOES TRAPEZOID BUILD AND MULTIPLY
INTEGER LETTER(32,32),SENTNUM
DIMENSION TRPLTR(32,32),TRPWDW(32)
1 FORMAT(1H0)
2 FORMAT(1H ,24(" "),/1H ,"" TRAPEZOID PARAMETERS * THIS TRAPEZOID
% WAS APPLIED TO WINDOWS OF SENTENCE NUMBER ",I2,/1H ,24(" "))
3 FORMAT(1H ,/, " LENGTH OF TRAPEZOID PLATEAU = ",I2)
4 FORMAT(1H ,/, " VARIANCE ON NORMALIZED WINDOW = ",F5.2)
5 FORMAT(1H ,/, " TRAPEZOID WINDOW ARRAY IS: ")
6 FORMAT(" ",10F6.3)
7 FORMAT(1H ,/, " THIS IS A NORMAL WINDOW APPLIED TO SENTENCE NUMBER
% ",I2)
IF (PSKIP.EQ."OK") GO TO 45
C DATA NORMALIZED WINDOW SET IPLEN=0 AND TRPHT=VARIANCE
IPLEN = 0 $ TRPHT = 5.
C
C THIS PART BUILDS THE TRAPEZOID
C
INCLN=(32-IPLEN)/2
IF (IPLEN.NE.0) GO TO 9
DO 8 I=1,16
8 TRPWDW(17-I)=TRPWDW(16+I)=EXP(-((I-1)**2)/(2.*((TRPHT)**2)))
WRITE 1
WRITE 7,SENTNUM
GO TO 44
9 IF (INCLN.EQ.0) GO TO 35
DO 10 I=1,INCLN
TRPWDW(I)=TRPWDW(33-I)=(I/(INCLN+1.))*TRPHT
10 CONTINUE
DO 20 I=1,IPLEN
TRPWDW(I+INCLN)=1.
20 CONTINUE
IF((32-(2*INCLN)).EQ.IPLEN) GO TO 43
I0DD=INCLN+1
DO 30 I=1,I0DD
TRPWDW(33-I)=(I/(INCLN+2.))*TRPHT
30 CONTINUE
GO TO 43
35 DO 40 I=1,16
40 TRPWDW(I)=TRPWDW(33-I)=1.
43 CONTINUE
WRITE 1
WRITE 1
WRITE 2,SENTNUM
WRITE 3,IPLEN
44 WRITE 4,TRPHT
WRITE 5
WRITE 6,TRPWDW

```

C THIS PART DOES THE MULTIPLICATION

```
C
45 DO 60 I=1,32
    DO 50 J=1,32
50 TRPLTR(J,I)=TRPHW(I)*(FLOAT(LETTER(J,I)))
60 CONTINUE
   PSKIP="OK"
   RETURN
   END
```

```
*****          ROY00SV  //// END OF LIST ////
```

Appendix F

Data Output For
Various Standard Deviations

RESULTS MATRIX SHOWING THE PROTOTYPE-TO-WINDOW
 DISTANCES FOR PROTOTYPES WHOSE WIDTH VARIATION ENABLES
 THE PROTOTYPE TO BE COMPARED TO WINDOWS OF WIDTH 32 COLUMNS
 MATRIX IS SORTED BY DISTANCE WITHIN THE SENTENCE
 COLUMNS. THE 1 TO 5 CLOSEST IDENTIFYING PROTOTYPES ARE
 SHOWN FOR EACH WINDOW OF WIDTH 32 BEGINNING AND ENDING AT
 THE SENTENCE COLUMN INDICATED BY THE LEFTMOST MATRIX COLUMN.

PROTOTYPES IN THE MATRIX ARE: A1 P1 C1 D1 E1 F1 G1 H1 I1 J1 K1 L1 M1 N1 O1 P1

Window / Prototype Standard Deviations = 2 col MATRIX 1 OF 1

MATRIX OF THE 1 TO 5 CLOSEST IDENTIFYING PROTOTYPES FOR WINDOWS OF WIDTH 32 S

 SENTENCE * GUESS 1 GUESS 2 GUESS 3 GUESS 4 GUESS 5
 COLUMNS * PROTO/DX PROTO/DX PROTO/DX PROTO/DX PROTO/DX

1-	32	* V1-	.439	M1-	.730	O1-	.944	Y1-	.953	I1-	1.004
2-	33	* V1-	.610	O1-	.847	O1-	.957	M1-	.955	U1-	.989
3-	34	* J1-	.729	M1-	.856	O1-	.874	31-	.836	O1-	.952
4-	35	* J1-	.340	M1-	.830	A1-	.895	I1-	.853	K1-	.859
5-	36	* J1-	.330	T1-	.839	K1-	.791	A1-	.715	T1-	.731
6-	37	* T1-	.292	T1-	.649	J1-	.593	A1-	.650	K1-	.602
7-	38	* T1-	.167	T1-	.219	Y1-	.655	A1-	.720	K1-	.756
8-	39	* T1-	.351	T1-	.453	Y1-	.791	F1-	.848	A1-	.853
9-	40	* T1-	.519	F1-	.681	T1-	.726	Y1-	.807	O1-	.900
10-	41	* F1-	.535	P1-	.754	R1-	.795	S1-	.849	O1-	.850
11-	42	* F1-	.423	31-	.464	P1-	.475	E1-	.617	H1-	.657
12-	43	* P1-	.000	E1-	.413	F1-	.431	R1-	.555	F1-	.577
13-	44	* 31-	.374	F1-	.534	F1-	.563	P1-	.671	O1-	.675
14-	45	* 31-	.560	J1-	.609	F1-	.698	E1-	.715	T1-	.731
15-	46	* T1-	.630	J1-	.660	K1-	.693	A1-	.713	T1-	.723
16-	47	* T1-	.455	I1-	.757	K1-	.664	A1-	.719	Y1-	.789
17-	48	* T1-	.450	T1-	.502	Y1-	.739	K1-	.774	A1-	.794
18-	49	* T1-	.615	I1-	.631	Y1-	.751	M1-	.759	V1-	.879
19-	50	* T1-	.778	M1-	.794	Y1-	.809	I1-	.813	Y1-	.874
20-	51	* Y1-	.710	Y1-	.821	T1-	.833	N1-	.851	A1-	.892
21-	52	* X1-	.640	T1-	.719	K1-	.723	T1-	.752	Y1-	.782
22-	53	* K1-	.589	T1-	.643	T1-	.682	X1-	.734	Y1-	.735
23-	54	* T1-	.493	K1-	.655	J1-	.677	Y1-	.745	T1-	.800
24-	55	* T1-	.665	T1-	.661	M1-	.798	Y1-	.821	K1-	.891
25-	56	* T1-	.750	T1-	.790	M1-	.895	V1-	.903	Y1-	.926
26-	57	* O1-	.723	I1-	.806	V1-	.835	T1-	.944	T1-	.994
27-	58	* O1-	.360	G1-	.779	O1-	.847	V1-	.853	I1-	.920
28-	59	* O1-	.000	G1-	.638	O1-	.699	J1-	.735	O1-	.864
29-	60	* O1-	.347	O1-	.564	U1-	.624	G1-	.652	O1-	.729
30-	61	* O1-	.535	D1-	.626	O1-	.706	U1-	.717	O1-	.723
31-	62	* O1-	.508	O1-	.731	J1-	.837	N1-	.839	O1-	.916
32-	63	* O1-	.634	J1-	.736	V1-	.828	O1-	.905	T1-	.976
33-	64	* J1-	.800	O1-	.816	J1-	.822	N1-	.831	T1-	.927
34-	65	* I1-	.724	T1-	.876	G1-	.929	A1-	.935	O1-	.947
35-	66	* I1-	.797	G1-	.810	O1-	.845	O1-	.887	T1-	.916
36-	67	* O1-	.761	O1-	.788	O1-	.818	O1-	.828	I1-	.939
37-	68	* J1-	.677	O1-	.780	O1-	.853	31-	.836	G1-	.920
38-	69	* J1-	.348	T1-	.820	K1-	.879	A1-	.873	O1-	.883
39-	70	* J1-	.765	T1-	.855	K1-	.788	I1-	.717	A1-	.767
40-	71	* T1-	.258	I1-	.480	J1-	.522	K1-	.637	A1-	.696
41-	72	* T1-	.191	T1-	.210	Y1-	.791	A1-	.754	K1-	.750
42-	73	* T1-	.301	T1-	.520	Y1-	.770	F1-	.730	A1-	.891
43-	74	* T1-	.642	F1-	.782	T1-	.813	O1-	.873	Y1-	.922
44-	75	* F1-	.505	O1-	.790	P1-	.869	T1-	.914	O1-	.916

45-77	* O1-	.438	O1-	.456	G1-	.683	L1-	.737	O1-	.805
47-78	* O1-	.890	O1-	.832	O1-	.767	R1-	.852	O1-	.924
48-79	* O1-	.319	O1-	.717	O1-	.845	J1-	.932	O1-	.908
49-80	* O1-	.521	I1-	.771	O1-	.896	G1-	.921	O1-	.986
50-81	* J1-	.622	I1-	.818	O1-	.919	A1-	.938	O1-	.953
51-82	* J1-	.559	I1-	.744	I1-	.808	I1-	.828	K1-	.976
52-83	* Y1-	.576	I1-	.839	I1-	.694	A1-	.739	K1-	.913
53-84	* I1-	.565	I1-	.620	I1-	.751	Y1-	.813	Y1-	.857
54-85	* Y1-	.368	I1-	.679	X1-	.678	Y1-	.739	A1-	.793
55-85	* X1-	.628	I1-	.850	Z1-	.717	Y1-	.721	K1-	.725
55-87	* K1-	.604	I1-	.631	Z1-	.699	A1-	.707	Y1-	.734
57-88	* I1-	.484	K1-	.564	I1-	.638	I1-	.649	J1-	.655
58-89	* I1-	.274	I1-	.432	K1-	.639	A1-	.651	Y1-	.670
59-90	* I1-	.274	I1-	.254	Y1-	.649	A1-	.734	K1-	.780
60-91	* I1-	.387	I1-	.490	Y1-	.709	F1-	.738	V1-	.891
61-92	* I1-	.547	O1-	.651	I1-	.703	Y1-	.830	O1-	.865
62-93	* F1-	.513	O1-	.737	F1-	.825	O1-	.825	O1-	.851
63-94	* F1-	.423	P1-	.469	F1-	.525	E1-	.527	H1-	.630
64-95	* E1-	.661	O1-	.413	I1-	.406	H1-	.541	O1-	.568
65-95	* E1-	.390	O1-	.570	O1-	.607	A1-	.657	F1-	.682
65-97	* E1-	.688	O1-	.679	O1-	.701	Z1-	.713	K1-	.724
67-98	* R1-	.771	I1-	.790	O1-	.800	Z1-	.822	I1-	.825
68-99	* O1-	.714	I1-	.811	O1-	.820	R1-	.851	I1-	.877
69-100	* O1-	.887	O1-	.730	O1-	.765	G1-	.775	O1-	.856
70-101	* O1-	.689	O1-	.689	O1-	.702	O1-	.711	O1-	.869
71-102	* J1-	.520	O1-	.755	O1-	.799	G1-	.814	O1-	.871
72-103	* J1-	.307	I1-	.719	A1-	.814	I1-	.815	K1-	.818
73-104	* J1-	.664	I1-	.464	I1-	.585	K1-	.716	A1-	.717
74-105	* I1-	.233	I1-	.295	Y1-	.706	A1-	.724	J1-	.744
75-106	* I1-	.257	I1-	.384	Y1-	.692	F1-	.801	A1-	.836
75-107	* I1-	.930	F1-	.606	I1-	.677	Y1-	.732	S1-	.840
77-108	* F1-	.350	P1-	.644	S1-	.731	R1-	.741	I1-	.765
78-109	* F1-	.904	P1-	.412	F1-	.577	S1-	.633	F1-	.633
79-110	* F1-	.380	P1-	.501	S1-	.572	R1-	.535	P1-	.525
80-111	* F1-	.572	P1-	.727	S1-	.738	R1-	.758	P1-	.807
81-112	* F1-	.870	O1-	.879	F1-	.879	W1-	.891	K1-	.896
82-113	* A1-	.789	R1-	.912	I1-	.947	J1-	.948	I1-	.967
83-114	* A1-	.683	J1-	.501	F1-	.890	I1-	.940	I1-	.943
84-115	* A1-	.597	I1-	.724	J1-	.738	K1-	.824	I1-	.830
85-115	* I1-	.510	I1-	.889	K1-	.734	A1-	.734	Z1-	.797
85-117	* I1-	.458	I1-	.506	Y1-	.804	K1-	.820	A1-	.855
87-118	* I1-	.610	I1-	.651	Y1-	.925	A1-	.933	V1-	.973
88-119	* I1-	.777	I1-	.510	O1-	.945	V1-	.950	A1-	1.008
89-120	* O1-	.767	I1-	.884	O1-	.948	G1-	.959	I1-	.968
90-121	* O1-	.638	G1-	.492	O1-	.714	O1-	.853	F1-	.951
91-122	* G1-	.775	O1-	.505	O1-	.585	O1-	.732	O1-	.876
92-123	* G1-	.800	O1-	.431	O1-	.638	O1-	.755	O1-	.767
93-124	* G1-	.458	O1-	.505	O1-	.732	L1-	.753	O1-	.804
94-125	* G1-	.871	O1-	.877	O1-	.897	O1-	.847	O1-	.932
95-126	* K1-	.732	I1-	.766	Y1-	.927	I1-	.951	F1-	.966
95-127	* K1-	.560	J1-	.490	I1-	.738	Y1-	.821	I1-	.926
97-128	* I1-	.651	K1-	.719	J1-	.743	I1-	.755	Y1-	.775
98-129	* I1-	.584	I1-	.613	F1-	.715	Y1-	.734	V1-	.812
99-130	* I1-	.596	I1-	.623	F1-	.710	V1-	.732	Y1-	.863
100-131	* I1-	.733	M1-	.799	I1-	.801	V1-	.841	Y1-	.962
101-132	* M1-	.923	V1-	.945	I1-	.948	F1-	.930	I1-	1.021
102-133	* F1-	.951	M1-	1.044	V1-	1.054	P1-	1.073	I1-	1.115

RESULTS MATRIX SHOWING THE PROTOTYPE-TO-WINDOW DISTANCES FOR PROTOTYPES WHOSE WIDTH VARIATION ENABLES THE PROTOTYPE TO BE COMPARED TO WINDOWS OF WIDTH 32 COLUMNS. MATRIX IS SORTED BY DISTANCE WITHIN THE SENTENCE COLUMNS. THE 1 TO 13 CLOSEST IDENTIFYING PROTOTYPES ARE SHOWN FOR EACH WINDOW OF WIDTH 32 BEGINNING AND ENDING AT THE SENTENCE COLUMN INDICATED BY THE LEFTMOST MATRIX COLUMN.

PROTOTYPES IN THE MATRIX ARE: A1 B1 C1 D1 E1 F1 G1 H1 I1 J1 K1 L1 M1 N1 O1 P1

Window / Prototype Standard Deviations = 5 columns MATRIX 1 OF 1

MATRIX OF THE 1 TO 5 CLOSEST IDENTIFYING PROTOTYPES FOR WINDOWS OF WIDTH 32

 SENTENCE * GUESS 1 GUESS 2 GUESS 3 GUESS 4 GUESS 5
 COLUMNS * PROTO/OX PROTO/OX PROTO/OX PROTO/OX PROTO/OX

1-	32	*	V1-	.864	M1-	.755	H1-	.896	R1-	.905	C1-	.923
2-	33	*	B1-	.758	V1-	.709	M1-	.837	H1-	.853	F1-	.881
3-	34	*	M1-	.734	R1-	.705	F1-	.770	J1-	.825	F1-	.920
4-	35	*	J1-	.501	M1-	.800	F1-	.849	A1-	.851	D1-	.909
5-	36	*	J1-	.428	A1-	.785	T1-	.838	K1-	.872	M1-	.954
6-	37	*	T1-	.537	I1-	.649	J1-	.636	A1-	.634	K1-	.915
7-	38	*	I1-	.404	T1-	.406	Y1-	.892	A1-	.949	J1-	1.011
8-	39	*	I1-	.528	T1-	.623	Y1-	.871	F1-	.974	T1-	1.072
9-	40	*	F1-	.775	I1-	.840	T1-	.929	Y1-	.935	S1-	.999
10-	41	*	F1-	.580	P1-	.821	P1-	.830	E1-	.874	V1-	.895
11-	42	*	B1-	.653	P1-	.534	F1-	.555	E1-	.649	M1-	.719
12-	43	*	R1-	.165	R1-	.471	F1-	.479	E1-	.531	M1-	.651
13-	44	*	R1-	.485	R1-	.526	E1-	.690	D1-	.770	C1-	.743
14-	45	*	J1-	.802	R1-	.618	K1-	.839	R1-	.855	A1-	.908
15-	46	*	J1-	.864	A1-	.879	T1-	.903	K1-	.916	I1-	.974
16-	47	*	T1-	.789	I1-	.822	A1-	.946	J1-	1.003	K1-	1.038
17-	48	*	T1-	.826	I1-	.830	T1-	.948	Y1-	.975	M1-	1.016
18-	49	*	T1-	.946	I1-	.959	M1-	.959	Y1-	.950	A1-	.971
19-	50	*	Y1-	.866	V1-	.920	M1-	.934	Y1-	.941	F1-	.961
20-	51	*	X1-	.715	R1-	.843	L1-	.866	F1-	.913	E1-	.916
21-	52	*	Y1-	.687	T1-	.799	K1-	.803	R1-	.876	F1-	.877
22-	53	*	K1-	.571	T1-	.757	Y1-	.824	T1-	.847	Y1-	.927
23-	54	*	K1-	.725	T1-	.828	T1-	.879	I1-	.926	Y1-	.955
24-	55	*	T1-	.866	I1-	.873	K1-	.912	M1-	.959	Y1-	1.010
25-	56	*	I1-	.924	M1-	.969	M1-	.972	T1-	.995	F1-	1.056
26-	57	*	C1-	.828	V1-	.868	F1-	.963	M1-	1.045	T1-	1.052
27-	58	*	C1-	.669	V1-	.835	F1-	.915	D1-	.918	P1-	.961
28-	59	*	C1-	.178	M1-	.769	C1-	.773	T1-	.824	L1-	.840
29-	60	*	C1-	.681	O1-	.820	O1-	.828	O1-	.791	L1-	.799
30-	61	*	O1-	.693	M1-	.765	P1-	.808	C1-	.873	L1-	.905
31-	62	*	O1-	.883	O1-	.903	O1-	1.000	K1-	1.009	M1-	1.045
32-	63	*	O1-	1.012	O1-	1.072	M1-	1.098	K1-	1.112	J1-	1.121
33-	64	*	O1-	1.100	O1-	1.123	C1-	1.136	I1-	1.144	T1-	1.149
34-	65	*	G1-	1.028	O1-	1.081	O1-	1.107	M1-	1.127	M1-	1.172
35-	66	*	G1-	.932	O1-	1.026	P1-	1.049	O1-	1.050	P1-	1.127
36-	67	*	G1-	.926	R1-	.962	F1-	.959	D1-	1.019	C1-	1.025
37-	68	*	J1-	.774	R1-	.856	F1-	.891	A1-	1.011	G1-	1.019
38-	69	*	J1-	.904	R1-	.907	F1-	.910	K1-	.947	P1-	.969
39-	70	*	J1-	.477	T1-	.698	A1-	.856	K1-	.857	T1-	.893
40-	71	*	T1-	.375	I1-	.555	J1-	.731	Y1-	.833	A1-	.895
41-	72	*	I1-	.284	T1-	.301	Y1-	.856	A1-	.933	J1-	1.027
42-	73	*	I1-	.470	T1-	.611	Y1-	.911	F1-	.913	S1-	1.087
43-	74	*	F1-	.703	I1-	.822	T1-	.941	V1-	.975	Y1-	.997
44-	75	*	F1-	.542	C1-	.825	E1-	.903	P1-	.913	V1-	.928

66-77	* D1-	.466	L1-	.499	O1-	.538	H1-	.531	U1-	.597
47-78	* D1-	.132	O1-	.732	H1-	.649	B1-	.715	L1-	.743
48-79	* D1-	.466	O1-	.518	H1-	.838	B1-	.834	L1-	.900
49-80	* D1-	.846	J1-	.961	W1-1.	1.007	R1-1.	1.031	F1-1.	1.034
50-81	* J1-	.658	A1-1.	1.039	W1-1.	1.044	R1-1.	1.121	F1-1.	1.130
51-82	* J1-	.846	A1-	.957	T1-	.939	L1-1.	1.058	W1-1.	1.062
52-83	* T1-	.872	J1-	.905	T1-	.918	I1-	.939	W1-1.	1.100
53-84	* T1-	.845	A1-	.900	Y1-	.912	I1-	.951	J1-	.964
54-85	* X1-	.744	A1-	.871	T1-	.879	W1-	.919	Y1-	.936
55-86	* X1-	.686	A1-	.822	T1-	.825	S1-	.830	M1-	.880
56-87	* X1-	.769	A1-	.772	T1-	.739	J1-	.734	K1-	.810
57-88	* K1-	.733	T1-	.765	A1-	.766	J1-	.776	T1-	.847
58-89	* T1-	.640	T1-	.729	K1-	.730	A1-	.832	Y1-	.845
59-90	* I1-	.603	T1-	.628	Y1-	.659	K1-	.912	A1-	.949
60-91	* I1-	.581	T1-	.774	F1-	.387	V1-	.935	Y1-	.910
61-92	* F1-	.718	V1-	.835	O1-	.871	I1-	.915	E1-	.977
62-93	* F1-	.563	E1-	.750	O1-	.750	P1-	.782	V1-	.813
63-94	* E1-	.434	F1-	.562	F1-	.611	B1-	.615	U1-	.695
64-95	* E1-	.232	B1-	.662	L1-	.649	R1-	.649	F1-	.654
65-96	* E1-	.537	K1-	.660	F1-	.735	H1-	.810	L1-	.810
66-97	* K1-	.797	O1-	.809	F1-	.840	O1-	.832	H1-	.943
67-98	* O1-	.945	O1-1.	1.033	K1-1.	1.020	T1-1.	1.023	H1-1.	1.035
68-99	* O1-	.988	O1-1.	1.043	H1-1.	1.048	O1-1.	1.054	M1-1.	1.089
69-100	* O1-	.974	B1-	.967	O1-	.991	H1-1.	1.023	O1-1.	1.039
70-101	* B1-	.878	R1-	.932	O1-	.961	W1-	.938	O1-	.991
71-102	* J1-	.738	R1-	.860	F1-	.881	A1-	.958	W1-	.981
72-103	* J1-	.567	A1-	.894	K1-	.898	T1-	.911	O1-	.959
73-104	* T1-	.630	J1-	.643	I1-	.776	K1-	.878	A1-	.892
74-105	* T1-	.419	I1-	.519	Y1-	.850	J1-	.370	A1-	.946
75-106	* I1-	.468	T1-	.495	Y1-	.800	F1-	.953	S1-1.	1.050
76-107	* I1-	.684	F1-	.721	T1-	.757	Y1-	.833	S1-	.941
77-108	* F1-	.427	S1-	.847	F1-	.895	E1-	.913	Y1-	.927
78-109	* F1-	.253	P1-	.670	F1-	.737	E1-	.741	S1-	.800
79-110	* F1-	.476	P1-	.858	F1-	.608	E1-	.654	O1-	.685
80-111	* B1-	.628	P1-	.646	F1-	.647	O1-	.673	H1-	.682
81-112	* R1-	.711	O1-	.749	B1-	.751	H1-	.759	F1-	.827
82-113	* R1-	.613	O1-	.895	A1-	.921	H1-	.925	O1-	.927
83-114	* A1-	.694	R1-	.936	J1-	.952	K1-1.	1.012	P1-1.	1.042
84-115	* T1-	.917	J1-	.943	A1-	.947	T1-	.956	K1-	.971
85-116	* T1-	.727	T1-	.917	I1-	.922	Y1-	.946	K1-	.947
86-117	* T1-	.634	I1-	.828	Y1-	.902	T1-	.957	K1-	.932
87-118	* T1-	.718	I1-	.824	Y1-	.954	K1-1.	1.070	F1-1.	1.096
88-119	* T1-	.909	I1-	.913	F1-1.	1.014	V1-1.	1.030	Y1-1.	1.066
89-120	* O1-	.912	F1-	.956	V1-1.	1.008	I1-1.	1.049	T1-1.	1.108
90-121	* O1-	.762	G1-	.824	O1-	.942	F1-	.947	U1-	.954
91-122	* G1-	.448	O1-	.791	O1-	.765	J1-	.832	P1-	.904
92-123	* G1-	.186	O1-	.583	O1-	.681	H1-	.813	U1-	.824
93-124	* G1-	.437	O1-	.717	O1-	.787	R1-	.808	H1-	.811
94-125	* G1-	.807	R1-	.846	O1-	.892	B1-	.939	O1-	.967
95-126	* J1-	.826	K1-	.838	F1-	.969	A1-1.	1.048	Y1-1.	1.051
96-127	* J1-	.752	K1-	.856	T1-	.931	Y1-1.	1.013	A1-1.	1.022
97-128	* T1-	.736	J1-	.839	I1-	.859	K1-	.934	Y1-	.964
98-129	* T1-	.644	I1-	.665	Y1-	.973	J1-1.	1.027	U1-1.	1.064
99-130	* I1-	.567	T1-	.740	V1-1.	1.018	Y1-1.	1.031	H1-1.	1.048
100-131	* I1-	.857	T1-	.950	V1-	.990	F1-1.	1.034	M1-1.	1.089
101-132	* F1-	.859	V1-	.997	O1-1.	1.043	I1-1.	1.103	T1-1.	1.108
102-133	* F1-	.798	P1-	.944	O1-	.950	L1-	.951	F1-	.981

RESULTS MATRIX SHOWING THE PROTOTYPE-TO-WINDOW DISTANCES FOR PROTOTYPES WHOSE WIDTH VARIATION ENABLES THE PROTOTYPE TO BE COMPARED TO WINDOWS OF WIDTH 32 COLUMNS. THE 1 TO 13 CLOSEST IDENTIFYING PROTOTYPES ARE SHOWN FOR EACH WINDOW OF WIDTH 32 BEGINNING AND ENDING AT THE SENTENCE COLUMN INDICATED BY THE LEFTMOST MATRIX COLUMN.

PROTOTYPES IN THE MATRIX ARE: A1 P1 C1 D1 E1 F1 G1 H1 I1 J1 K1 L1 M1 N1 O1 P1

Window / Prototype Standard Deviations = 10 columns MATRIX 1 OF 1
 MATRIX OF THE 1 TO 13 CLOSEST IDENTIFYING PROTOTYPES FOR WINDOWS OF WIDTH 32

SENTENCE	* GUESS	1	GUESS	2	GUESS	3	GUESS	4	GUESS	5
COLUMNS	* PROTO/DX	PROTO/DX	PROTO/DX	PROTO/DX	PROTO/DX	PROTO/DX	PROTO/DX	PROTO/DX	PROTO/DX	PROTO/DX
1- 32	* M1-	.820	V1-	.851	H1-	.883	J1-	.914	F1-	.953
2- 33	* B1-	.822	V1-	.850	H1-	.875	Q1-	.910	F1-	.927
3- 34	* B1-	.834	R1-	.848	M1-	.853	J1-	.852	F1-	.871
4- 35	* J1-	.718	M1-	.902	F1-	.934	A1-	.953	O1-	1.000
5- 36	* J1-	.677	A1-	.957	K1-	1.012	T1-	1.023	M1-	1.039
6- 37	* T1-	.831	I1-	.875	J1-	.879	A1-	1.031	K1-	1.087
7- 38	* I1-	.716	T1-	.756	Y1-	1.116	A1-	1.143	J1-	1.150
8- 39	* I1-	.791	T1-	.988	Y1-	1.076	M1-	1.145	F1-	1.185
9- 40	* F1-	.987	I1-	1.031	M1-	1.069	Y1-	1.116	T1-	1.165
10- 41	* F1-	.794	B1-	.971	F1-	.984	S1-	1.031	V1-	1.046
11- 42	* B1-	.652	P1-	.729	F1-	.731	E1-	.829	F1-	.840
12- 43	* B1-	.498	R1-	.629	F1-	.654	E1-	.635	O1-	.771
13- 44	* B1-	.679	R1-	.694	O1-	.697	E1-	.742	P1-	.838
14- 45	* O1-	.304	K1-	.891	F1-	.944	B1-	.959	O1-	.982
15- 46	* O1-	1.010	K1-	1.045	T1-	1.089	V1-	1.097	A1-	1.166
16- 47	* T1-	1.010	I1-	1.045	M1-	1.073	A1-	1.123	M1-	1.166
17- 48	* T1-	1.050	I1-	1.052	M1-	1.089	A1-	1.130	Y1-	1.175
18- 49	* A1-	1.115	M1-	1.110	Y1-	1.126	N1-	1.131	T1-	1.135
19- 50	* X1-	.971	N1-	1.037	F1-	1.061	S1-	1.055	V1-	1.067
20- 51	* X1-	.832	R1-	.938	F1-	.977	V1-	.977	O1-	.994
21- 52	* Y1-	.795	K1-	.884	T1-	.902	R1-	.907	F1-	.912
22- 53	* K1-	.724	T1-	.850	X1-	.911	E1-	.944	F1-	1.026
23- 54	* K1-	.756	T1-	.953	T1-	.991	E1-	1.055	Y1-	1.073
24- 55	* K1-	.951	T1-	.990	J1-	1.023	H1-	1.071	V1-	1.119
25- 56	* T1-	1.070	V1-	1.073	M1-	1.073	T1-	1.114	O1-	1.159
26- 57	* O1-	.992	V1-	1.031	M1-	1.153	F1-	1.133	O1-	1.166
27- 58	* O1-	.807	V1-	1.033	O1-	1.093	F1-	1.126	T1-	1.138
28- 59	* O1-	.729	O1-	1.003	F1-	1.059	L1-	1.071	F1-	1.083
29- 60	* O1-	.355	O1-	.976	O1-	1.013	L1-	1.023	F1-	1.055
30- 61	* O1-	1.017	H1-	1.044	L1-	1.033	O1-	1.041	F1-	1.130
31- 62	* O1-	1.068	O1-	1.127	H1-	1.153	K1-	1.173	H1-	1.198
32- 63	* O1-	1.128	O1-	1.160	M1-	1.152	H1-	1.170	O1-	1.232
33- 64	* M1-	1.066	H1-	1.159	O1-	1.165	J1-	1.155	O1-	1.195
34- 65	* M1-	1.094	H1-	1.110	O1-	1.125	J1-	1.172	O1-	1.178
35- 66	* G1-	1.066	H1-	1.095	M1-	1.153	O1-	1.155	O1-	1.180
36- 67	* S1-	1.064	B1-	1.071	L1-	1.081	H1-	1.115	Y1-	1.127
37- 68	* J1-	.896	R1-	.932	T1-	1.008	X1-	1.055	S1-	1.065
38- 69	* J1-	.669	R1-	1.014	T1-	1.014	T1-	1.035	O1-	1.057
39- 70	* J1-	.640	T1-	.851	L1-	.996	A1-	.999	K1-	1.004
40- 71	* T1-	.608	I1-	.719	J1-	.843	Y1-	.954	M1-	1.041
41- 72	* T1-	.639	I1-	.751	Y1-	.957	J1-	1.038	A1-	1.113
42- 73	* I1-	.688	T1-	.739	Y1-	1.014	F1-	1.037	V1-	1.176
43- 74	* F1-	.395	T1-	.948	T1-	1.017	Y1-	1.115	V1-	1.165
44- 75	* F1-	.779	O1-	.997	F1-	1.041	V1-	1.054	O1-	1.074

46-77	* D1-	.675	D1-	.775	L1-	.731	H1-	.795	H1-	.807
47-78	* D1-	.724	D1-	.763	H1-	.793	L1-	.845	D1-	.853
48-79	* D1-	.640	D1-	.691	H1-	.802	H1-	1-1.007	F1-	1-1.024
49-80	* D1-	.887	H1-	1-1.040	H1-	1-1.039	D1-	1-1.054	H1-	1-1.124
50-81	* H1-	1-1.057	D1-	1-1.120	H1-	1-1.124	H1-	1-1.172	D1-	1-1.176
51-82	* H1-	1-1.125	H1-	1-1.138	H1-	1-1.140	H1-	1-1.132	A1-	1-1.159
52-83	* F1-	1-1.091	A1-	1-1.127	H1-	1-1.155	H1-	1-1.158	H1-	1-1.168
53-84	* X1-	1-1.022	A1-	1-1.056	H1-	1-1.083	H1-	1-1.111	H1-	1-1.127
54-85	* X1-	.859	A1-	.933	S1-	.993	H1-	1-1.015	J1-	1-1.037
55-86	* X1-	.775	S1-	.869	L1-	.904	Z1-	.915	J1-	.918
56-87	* X1-	.630	J1-	.832	S1-	.851	Z1-	.855	A1-	.859
57-88	* K1-	.824	H1-	.850	L1-	.834	F1-	.835	Z1-	.903
58-89	* F1-	.869	K1-	.860	L1-	.860	Y1-	.940	Z1-	.959
59-90	* L1-	.804	T1-	.820	H1-	1-1.000	J1-	1-1.001	H1-	1-1.011
60-91	* F1-	.693	V1-	.950	H1-	.963	F1-	1-1.142	H1-	1-1.043
61-92	* F1-	.927	V1-	.932	D1-	.934	H1-	1-1.055	F1-	1-1.060
62-93	* F1-	.825	D1-	.840	F1-	.835	P1-	.939	H1-	.942
63-94	* E1-	.721	F1-	.836	F1-	.844	D1-	.847	H1-	.852
64-95	* E1-	.678	L1-	.795	H1-	.830	P1-	.834	D1-	.843
65-96	* E1-	.837	K1-	.860	L1-	.892	H1-	.921	H1-	.926
66-97	* D1-	.915	H1-	.947	D1-	.955	K1-	.932	H1-	1-1.049
67-98	* D1-	.962	H1-	.984	H1-	.984	D1-	1-1.015	G1-	1-1.141
68-99	* H1-	1-1.040	D1-	1-1.045	H1-	1-1.050	D1-	1-1.035	H1-	1-1.153
69-100	* H1-	1-1.111	H1-	1-1.120	D1-	1-1.120	H1-	1-1.133	G1-	1-1.172
70-101	* B1-	1-1.091	H1-	1-1.105	H1-	1-1.121	H1-	1-1.135	S1-	1-1.171
71-102	* J1-	.952	R1-	1-1.050	P1-	1-1.066	A1-	1-1.115	H1-	1-1.126
72-103	* J1-	.791	A1-	1-1.057	K1-	1-1.072	T1-	1-1.035	F1-	1-1.104
73-104	* J1-	.792	T1-	.852	L1-	.852	K1-	1-1.013	A1-	1-1.049
74-105	* F1-	.663	T1-	.741	J1-	.952	Y1-	.957	H1-	1-1.067
75-106	* F1-	.651	T1-	.689	Y1-	.895	F1-	1-1.055	Y1-	1-1.104
76-107	* F1-	.838	T1-	.894	F1-	.874	Y1-	.924	Y1-	1-1.060
77-108	* F1-	.675	S1-	.990	H1-	1-1.014	X1-	1-1.037	F1-	1-1.038
78-109	* F1-	.582	P1-	.637	D1-	.873	E1-	.916	S1-	.956
79-110	* F1-	.689	P1-	.646	H1-	.710	R1-	.775	H1-	.781
80-111	* H1-	.672	R1-	.670	F1-	.681	P1-	.715	D1-	.720
81-112	* H1-	.715	R1-	.739	D1-	.769	R1-	.735	F1-	.873
82-113	* H1-	.890	R1-	.859	D1-	.940	R1-	.935	H1-	1-1.028
83-114	* R1-	1-1.002	H1-	1-1.103	A1-	1-1.113	K1-	1-1.118	D1-	1-1.137
84-115	* F1-	1-1.110	Z1-	1-1.151	Y1-	1-1.153	A1-	1-1.135	K1-	1-1.191
85-116	* F1-	1-1.047	Z1-	1-1.146	Z1-	1-1.155	Y1-	1-1.156	Y1-	1-1.171
86-117	* F1-	1-1.025	T1-	1-1.118	Y1-	1-1.119	Z1-	1-1.125	X1-	1-1.212
87-118	* F1-	1-1.051	T1-	1-1.111	Y1-	1-1.127	F1-	1-1.219	K1-	1-1.245
88-119	* F1-	1-1.119	T1-	1-1.160	F1-	1-1.175	Y1-	1-1.179	V1-	1-1.211
89-120	* A1-	1-1.123	F1-	1-1.137	V1-	1-1.144	D1-	1-1.155	G1-	1-1.190
90-121	* G1-	.889	D1-	.979	D1-	1-1.033	H1-	1-1.040	H1-	1-1.058
91-122	* G1-	.545	D1-	.820	D1-	.858	H1-	.835	H1-	.903
92-123	* G1-	.319	D1-	.628	H1-	.803	H1-	.815	H1-	.855
93-124	* G1-	.514	D1-	.766	D1-	.801	R1-	.844	H1-	.853
94-125	* G1-	.655	R1-	.696	D1-	.914	H1-	.975	D1-	.997
95-126	* J1-	.955	K1-	.995	H1-	1-1.031	D1-	1-1.101	A1-	1-1.106
96-127	* J1-	.919	K1-	1-1.016	H1-	1-1.032	A1-	1-1.101	Y1-	1-1.114
97-128	* F1-	.880	T1-	.971	J1-	.940	Y1-	1-1.078	K1-	1-1.100
98-129	* F1-	.819	T1-	.822	Y1-	1-1.036	J1-	1-1.132	A1-	1-1.105
99-130	* F1-	.616	T1-	.901	Y1-	1-1.131	V1-	1-1.141	F1-	1-1.208
100-131	* F1-	.964	F1-	1-1.059	T1-	1-1.075	V1-	1-1.120	Y1-	1-1.194
101-132	* F1-	.904	V1-	1-1.107	D1-	1-1.110	E1-	1-1.146	Z1-	1-1.151
102-133	* F1-	.842	L1-	.950	F1-	.975	E1-	.931	D1-	1-1.002

Appendix G

Data Output For
Complex and Amplitude
Spectra
Sentences 1 - 4

RESULTS MATRIX SHOWING THE PROTOTYPE-T-WINDOW DISTANCES FOR PROTOTYPES WHOSE WIDTH VARIATION ENABLES THE PROTOTYPE TO BE COMPARED TO WINDOWS OF WIDTH 32 COLUMNS. MATRIX IS SORTED BY DISTANCE WITHIN THE SENTENCE COLUMNS. THE 1 TO 13 CLOSEST IDENTIFYING PROTOTYPES ARE SHOWN FOR EACH WINDOW OF WIDTH 32 BEGINNING AND ENDING AT THE SENTENCE COLUMN INDICATED BY THE LEFTMOST MATRIX COLUMN.

PROTOTYPES IN THE MATRIX ARE: A1 B1 C1 D1 E1 F1 G1 H1 I1 J1 K1 L1 M1 N1 O1 P1 Q

Complex Spectrum : Sentence #1

MATRIX 1 OF 1

MATRIX OF THE 1 TO 5 CLOSEST IDENTIFYING PROTOTYPES FOR WINDOWS OF WIDTH 32

SENTENCE * GUESS 1 GUESS 2 GUESS 3 GUESS 4 GUESS 5
 COLUMNS * PROTO/OX PROTO/OX PROTO/OX PROTO/OX PROTO/OX

1- 32	* V1-	.624	M1-	.732	O1-	.815	R1-	.822	H1-	.847
2- 33	* R1-	.695	R1-	.744	O1-	.789	H1-	.792	V1-	.812
3- 34	* R1-	.730	R1-	.731	V1-	.808	P1-	.879	M1-	.880
4- 35	* J1-	.731	M1-	.770	A1-	.845	R1-	.848	P1-	.913
5- 36	* J1-	.866	M1-	.810	A1-	.810	T1-	.842	I1-	.902
6- 37	* J1-	.849	T1-	.803	I1-	.856	A1-	.849	M1-	.893
7- 38	* I1-	.898	T1-	.808	J1-	.859	Y1-	.920	A1-	.923
8- 39	* I1-	.870	T1-	.856	Y1-	.855	J1-	.912	A1-	.921
9- 40	* I1-	.708	Y1-	.849	F1-	.891	V1-	.835	T1-	.906
10- 41	* F1-	.692	E1-	.710	E1-	.776	L1-	.735	V1-	.828
11- 42	* B1-	.652	E1-	.836	F1-	.573	P1-	.814	F1-	.847
12- 43	* P1-	.293	R1-	.513	F1-	.539	E1-	.553	M1-	.634
13- 44	* B1-	.555	R1-	.632	F1-	.738	O1-	.720	S1-	.734
14- 45	* M1-	.840	S1-	.842	J1-	.653	R1-	.872	F1-	.872
15- 46	* J1-	.774	A1-	.877	M1-	.878	T1-	.837	S1-	.958
16- 47	* T1-	.803	J1-	.867	I1-	.821	A1-	.834	M1-	.928
17- 48	* I1-	.804	T1-	.843	A1-	.882	J1-	.905	M1-	.949
18- 49	* A1-	.863	I1-	.908	V1-	.915	V1-	.922	Y1-	.943
19- 50	* X1-	.822	A1-	.835	M1-	.851	Y1-	.837	V1-	.902
20- 51	* X1-	.678	M1-	.776	A1-	.815	R1-	.843	K1-	.854
21- 52	* X1-	.643	M1-	.736	K1-	.739	Z1-	.810	A1-	.820
22- 53	* K1-	.716	X1-	.756	M1-	.731	T1-	.827	A1-	.857
23- 54	* K1-	.749	T1-	.822	I1-	.824	V1-	.833	Y1-	.911
24- 55	* I1-	.837	T1-	.862	K1-	.923	V1-	.833	Y1-	.966
25- 56	* V1-	.890	I1-	.892	T1-	.935	L1-	.939	L1-	1.009
26- 57	* O1-	.834	V1-	.872	L1-	.894	E1-	.931	F1-	.939
27- 58	* O1-	.512	L1-	.816	O1-	.829	E1-	.842	F1-	.855
28- 59	* O1-	.866	U1-	.713	O1-	.769	E1-	.731	F1-	.809
29- 60	* O1-	.551	G1-	.707	U1-	.723	O1-	.715	O1-	.804
30- 61	* G1-	.779	O1-	.786	O1-	.790	U1-	.851	O1-	.869
31- 62	* O1-	.868	G1-	.917	O1-	.918	R1-	1.015	K1-	1.016
32- 63	* O1-	.945	O1-	.990	O1-	1.022	J1-	1.059	M1-	1.090
33- 64	* O1-	.862	G1-	1.040	O1-	1.046	J1-	1.059	T1-	1.070
34- 65	* O1-	.936	G1-	1.037	O1-	1.041	A1-	1.057	O1-	1.085
35- 66	* O1-	.931	G1-	.969	F1-	.993	O1-	1.009	O1-	1.011
36- 67	* R1-	.903	R1-	.939	O1-	.933	H1-	.934	O1-	.988
37- 68	* J1-	.878	R1-	.878	F1-	.879	A1-	.937	O1-	.940
38- 69	* J1-	.859	A1-	.855	S1-	.915	R1-	.929	T1-	.929
39- 70	* J1-	.809	T1-	.897	I1-	.861	A1-	.839	T1-	.909
40- 71	* T1-	.832	J1-	.877	T1-	.851	Y1-	.853	T1-	.919
41- 72	* I1-	.267	T1-	.374	J1-	.691	Y1-	.826	T1-	.945
42- 73	* I1-	.826	T1-	.815	Y1-	.831	J1-	.844	T1-	.987
43- 74	* I1-	.754	F1-	.850	Y1-	.881	L1-	.832	T1-	.966
44- 75	* I1-	.734	F1-	.812	F1-	.812	F1-	.812	F1-	.812

46-77	* O1-	.523	U1-	.834	F1-	.895	B1-	.812	O1-	.613
47-78	* O1-	.315	O1-	.784	F1-	.639	B1-	.671	U1-	.674
48-79	* O1-	.495	O1-	.730	F1-	.807	R1-	.832	O1-	.832
49-80	* O1-	.501	O1-	.752	F1-	.955	A1-	.816	O1-	.999
50-81	* J1-	.898	W1-1.	1.111	A1-1.	1.017	O1-1.	1.052	F1-1.	1.101
51-82	* J1-	.797	A1-	.951	W1-	.951	F1-	.977	I1-	1.080
52-83	* J1-	.759	T1-	.879	W1-	.832	A1-	.911	T1-	.980
53-84	* J1-	.784	W1-	.825	T1-	.842	A1-	.858	Y1-	.894
54-85	* X1-	.720	W1-	.804	A1-	.805	J1-	.815	T1-	.812
55-86	* X1-	.631	A1-	.746	T1-	.784	W1-	.835	Y1-	.807
56-87	* X1-	.585	A1-	.699	S1-	.752	J1-	.753	Y1-	.779
57-88	* A1-	.699	J1-	.722	T1-	.751	Y1-	.730	S1-	.750
58-89	* T1-	.654	T1-	.657	J1-	.737	A1-	.757	Y1-	.791
59-90	* T1-	.948	T1-	.914	Y1-	.805	J1-	.845	T1-	.849
60-91	* T1-	.527	T1-	.774	Y1-	.836	V1-	.842	F1-	.888
61-92	* F1-	.759	E1-	.891	V1-	.817	L1-	.831	O1-	.844
62-93	* F1-	.557	E1-	.619	L1-	.639	O1-	.724	O1-	.764
63-94	* E1-	.452	F1-	.497	B1-	.601	L1-	.619	F1-	.653
64-95	* E1-	.471	F1-	.581	F1-	.607	B1-	.615	Y1-	.628
65-96	* E1-	.674	K1-	.675	F1-	.736	F1-	.740	O1-	.794
66-97	* K1-	.830	O1-	.876	F1-	.839	O1-	.891	T1-	.895
67-98	* O1-	.873	O1-	.922	A1-	.945	T1-	.955	S1-	.983
68-99	* O1-	.875	O1-	.927	O1-	.933	A1-	.953	O1-	.978
69-100	* O1-	.889	O1-	.936	F1-	.915	O1-	.926	F1-	.941
70-101	* B1-	.330	R1-	.859	S1-	.937	O1-	.913	O1-	.939
71-102	* J1-	.795	R1-	.841	F1-	.887	S1-	.832	A1-	.899
72-103	* J1-	.593	T1-	.882	W1-	.865	A1-	.853	S1-	.899
73-104	* J1-	.484	T1-	.619	I1-	.754	W1-	.875	A1-	.884
74-105	* T1-	.432	I1-	.515	J1-	.574	Y1-	.841	T1-	.920
75-106	* I1-	.436	T1-	.491	Y1-	.764	J1-	.777	T1-	.919
76-107	* I1-	.626	T1-	.716	Y1-	.740	E1-	.832	O1-	.867
77-108	* F1-	.638	E1-	.753	Y1-	.731	S1-	.733	O1-	.879
78-109	* F1-	.476	E1-	.605	F1-	.631	B1-	.717	S1-	.723
79-110	* F1-	.500	P1-	.546	E1-	.580	B1-	.574	F1-	.646
80-111	* P1-	.579	R1-	.606	F1-	.623	O1-	.657	H1-	.677
81-112	* R1-	.698	B1-	.710	O1-	.733	P1-	.733	W1-	.744
82-113	* R1-	.813	A1-	.831	F1-	.851	K1-	.895	H1-	.894
83-114	* A1-	.843	W1-	.912	K1-	.935	R1-	.937	J1-	.964
84-115	* T1-	.892	J1-	.906	T1-	.913	W1-	.923	Y1-	.962
85-116	* T1-	.749	I1-	.889	J1-	.915	Y1-	.923	T1-	.961
86-117	* T1-	.586	I1-	.785	Y1-	.801	J1-	.833	T1-	.892
87-118	* T1-	.757	I1-	.778	Y1-	.857	V1-1.	1.002	U1-1.	1.066
88-119	* I1-	.875	T1-	.917	Y1-	.955	V1-	.953	L1-1.	1.018
89-120	* L1-	.903	O1-	.912	F1-	.953	F1-	.977	V1-	.987
90-121	* O1-	.758	G1-	.805	L1-	.835	O1-	.858	E1-	.885
91-122	* G1-	.594	O1-	.697	U1-	.752	O1-	.770	L1-	.847
92-123	* G1-	.521	O1-	.686	F1-	.781	O1-	.735	U1-	.785
93-124	* G1-	.666	O1-	.773	F1-	.807	A1-	.824	O1-	.826
94-125	* S1-	.801	R1-	.894	F1-	.853	S1-	.819	O1-	.928
95-126	* J1-	.895	S1-	.936	F1-	.955	X1-	.939	A1-1.	1.020
96-127	* J1-	.774	T1-	.936	S1-1.	1.016	A1-1.	1.025	Y1-1.	1.042
97-128	* J1-	.740	T1-	.783	J1-	.859	Y1-	.915	A1-1.	1.046
98-129	* I1-	.703	T1-	.722	J1-	.836	Y1-	.932	W1-1.	1.036
99-130	* T1-	.593	T1-	.803	J1-	.999	Y1-1.	1.015	W1-1.	1.029
100-131	* T1-	.844	T1-	.979	W1-1.	1.035	V1-1.	1.054	T1-1.	1.047
101-132	* L1-	.320	F1-	.934	E1-	.936	Z1-1.	1.044	V1-1.	1.054
102-133	* L1-	.789	F1-	.821	F1-	.864	P1-	.855	U1-	.962

RESULTS MATRIX SHOWING THE PROTOTYPE-10-WINDOW DISTANCES FOR PROTOTYPES WHOSE WIDTH VARIATION ENABLES THE PROTOTYPE TO BE COMPARED TO WINDOWS OF WIDTH 32 COLUMNS. THE MATRIX IS SORTED BY DISTANCE WITHIN THE SENTENCE COLUMNS. THE 1 TO 13 CLOSEST IDENTIFYING PROTOTYPES ARE SHOWN FOR EACH WINDOW OF WIDTH 32 BEGINNING AND ENDING AT THE SENTENCE COLUMN INDICATED BY THE LEFTMOST MATRIX COLUMN.

PROTOTYPES IN THE MATRIX ARE: A1 P1 C1 D1 E1 F1 G1 H1 I1 J1 K1 L1 M1 N1 O1 P1

Complex Spectrum - Sentence # 2

MATRIX 1 OF 1

MATRIX OF THE 1 TO 5 CLOSEST IDENTIFYING PROTOTYPES FOR WINDOWS OF WIDTH 32

SENTENCE * GUESS 1 GUESS 2 GUESS 3 GUESS 4 GUESS 5
COLUMNS * PROTO/DX PROTO/DX PROTO/DX PROTO/DX PROTO/DX

SENTENCE	COLUMNS	* GUESS 1 PROTO/DX	GUESS 2 PROTO/DX	GUESS 3 PROTO/DX	GUESS 4 PROTO/DX	GUESS 5 PROTO/DX
1-	32	* J1-.567	T1-.590	J1-.710	W1-.958	A1-.971
2-	33	* I1-.449	T1-.482	J1-.717	Y1-.951	W1-.982
3-	34	* I1-.316	T1-.662	Y1-.925	J1-.938	W1-1.009
4-	35	* I1-.818	V1-.900	I1-.932	Y1-.932	O1-.941
5-	36	* E1-.730	Q1-.740	F1-.773	S1-.818	O1-.825
6-	37	* R1-.520	E1-.560	F1-.590	F1-.677	O1-.685
7-	38	* R1-.572	R1-.619	E1-.653	K1-.633	O1-.698
8-	39	* J1-.743	K1-.773	F1-.787	W1-.805	A1-.813
9-	40	* T1-.669	J1-.670	I1-.795	W1-.821	A1-.850
10-	41	* I1-.553	T1-.559	J1-.794	W1-.891	A1-.914
11-	42	* I1-.576	T1-.718	Y1-.937	W1-.979	A1-.983
12-	43	* I1-.815	V1-.910	L1-.921	Y1-.927	T1-.939
13-	44	* L1-.774	V1-.790	F1-.855	F1-.887	W1-.929
14-	45	* V1-.741	L1-.745	U1-.797	E1-.820	F1-.845
15-	46	* U1-.661	O1-.769	V1-.771	W1-.792	O1-.792
16-	47	* U1-.711	O1-.715	O1-.727	W1-.737	O1-.785
17-	48	* O1-.778	O1-.795	F1-.800	O1-.818	O1-.823
18-	49	* R1-.880	R1-.875	W1-.913	O1-.913	G1-.947
19-	50	* J1-.808	W1-.907	F1-.917	A1-.935	O1-.974
20-	51	* J1-.568	A1-.884	W1-.917	T1-.971	O1-1.028
21-	52	* J1-.473	T1-.733	I1-.819	A1-.890	W1-.946
22-	53	* I1-.543	T1-.571	J1-.650	A1-.951	W1-.993
23-	54	* I1-.511	T1-.562	J1-.932	Y1-.932	A1-1.024
24-	55	* I1-.786	T1-.921	Y1-.979	W1-1.027	Y1-1.030
25-	56	* E1-.882	F1-.890	V1-.907	L1-.925	O1-.955
26-	57	* R1-.657	E1-.702	F1-.761	F1-.773	W1-.781
27-	58	* R1-.527	R1-.590	P1-.671	E1-.680	W1-.717
28-	59	* R1-.678	Q1-.730	F1-.770	O1-.802	W1-.809
29-	60	* J1-.784	W1-.890	F1-.923	K1-.940	A1-.944
30-	61	* J1-.845	T1-.706	I1-.856	W1-.910	A1-.964
31-	62	* I1-.557	T1-.563	J1-.720	W1-.930	A1-.989
32-	63	* I1-.533	T1-.691	J1-.903	Y1-.942	A1-.952
33-	64	* I1-.785	Y1-.919	T1-.922	S1-.932	A1-.962
34-	65	* S1-.770	X1-.777	F1-.835	F1-.835	P1-.836
35-	65	* X1-.522	E1-.645	S1-.665	F1-.687	O1-.689
36-	67	* K1-.560	X1-.640	F1-.678	R1-.715	S1-.714
37-	68	* K1-.562	X1-.642	F1-.650	E1-.655	O1-.681
38-	69	* T1-.805	K1-.821	Y1-.927	T1-.930	J1-.973
39-	70	* T1-.818	I1-.867	F1-.927	V1-.928	Y1-.964
40-	71	* V1-.876	W1-.911	I1-.953	T1-.937	O1-1.017
41-	72	* O1-.872	O1-.872	V1-.890	O1-.907	G1-.928
42-	73	* O1-.766	R1-.789	G1-.798	O1-.805	O1-.825
43-	74	* B1-.670	R1-.722	G1-.810	W1-.825	O1-.828
44-	75	* R1-.764	Q1-.795	J1-.830	W1-.838	A1-.906

45-77	* J1-	.808	F1-	.836	J1-	.870	A1-	.891	W1-	.935
47-78	* T1-	.810	T1-	.833	J1-	.830	Y1-	.831	F1-	.959
48-79	* I1-	.389	T1-	.405	J1-	.849	Y1-	.857	71-	1.000
49-80	* I1-	.857	T1-	.771	Y1-	.852	V1-	.837	71-	1.000
50-81	* F1-	.801	E1-	.817	V1-	.833	D1-	.879	L1-	.886
51-82	* E1-	.818	F1-	.828	F1-	.837	P1-	.712	C1-	.728
52-83	* P1-	.897	R1-	.807	F1-	.855	E1-	.815	F1-	.841
53-84	* P1-	.879	D1-	.812	F1-	.823	B1-	.718	E1-	.808
54-85	* D1-	.795	P1-	.834	F1-	.839	J1-	.940	R1-	.950
55-86	* J1-	.863	W1-	.951	A1-	.989	T1-	.944	D1-	1.026
56-87	* J1-	.868	A1-	.880	J1-	.934	W1-	.933	T1-	.951
57-88	* A1-	.791	I1-	.910	J1-	.921	W1-	.921	T1-	.961
58-89	* A1-	.754	W1-	.926	M1-	.943	I1-	.950	X1-	.952
59-90	* X1-	.742	N1-	.759	A1-	.813	S1-	.953	W1-	.975
60-91	* X1-	.844	N1-	.860	Y1-	.910	A1-	.941	C1-	.952
61-92	* X1-	.739	N1-	.743	Y1-	.739	A1-	.914	K1-	.836
62-93	* M1-	.870	V1-	.750	Y1-	.759	N1-	.807	X1-	.930
63-94	* M1-	.808	V1-	.647	Y1-	.872	K1-	.978	M1-	1.016
64-95	* M1-	.886	V1-	.708	M1-	.915	R1-	.953	D1-	.995
65-96	* M1-	.818	M1-	.857	W1-	.870	B1-	.874	F1-	.879
66-97	* W1-	.775	B1-	.820	F1-	.839	A1-	.913	71-	.952
67-98	* W1-	.756	A1-	.847	F1-	.890	B1-	.917	J1-	.921
68-99	* J1-	.890	A1-	.790	W1-	.807	T1-	.949	K1-	.902
69-100	* J1-	.888	T1-	.893	I1-	.816	A1-	.825	W1-	.890
70-101	* T1-	.817	I1-	.832	J1-	.854	A1-	.915	Y1-	.941
71-102	* I1-	.472	T1-	.804	Y1-	.856	J1-	.833	A1-	1.007
72-103	* I1-	.718	Y1-	.839	T1-	.851	V1-	.838	F1-	.946
73-104	* F1-	.751	E1-	.776	C1-	.810	V1-	.823	C1-	.863
74-105	* P1-	.820	E1-	.824	F1-	.827	B1-	.831	F1-	.854
75-106	* P1-	.821	R1-	.878	F1-	.819	E1-	.875	F1-	.701
76-107	* P1-	.896	R1-	.760	C1-	.792	B1-	.840	K1-	.857
77-108	* J1-	.913	W1-	.926	F1-	.950	D1-	.975	A1-	.981
78-109	* T1-	.914	J1-	.930	I1-	.936	W1-	.951	A1-	.976
79-110	* I1-	.902	T1-	.935	A1-	.962	W1-	.937	J1-	1.018
80-111	* A1-	.943	I1-	.981	W1-	.995	T1-	1.019	M1-	1.093
81-112	* N1-	.889	A1-	.935	X1-	.975	W1-	.993	S1-	1.066
82-113	* N1-	.841	X1-	.776	S1-	.944	A1-	.953	R1-	.995
83-114	* N1-	.470	X1-	.632	S1-	.851	B1-	.953	Y1-	.965
84-115	* N1-	.858	X1-	.845	S1-	.828	K1-	.911	M1-	.919
85-116	* N1-	.805	X1-	.814	M1-	.842	K1-	.871	C1-	.884
86-117	* M1-	.827	K1-	.910	V1-	.922	R1-	.931	Y1-	.984
87-118	* M1-	.871	V1-	.930	K1-	1.011	W1-	1.513	F1-	1.025
88-119	* V1-	.924	M1-	.945	W1-	.962	A1-	1.030	J1-	1.032
89-120	* W1-	.897	A1-	.960	V1-	.965	J1-	1.024	M1-	1.024
90-121	* W1-	.825	A1-	.850	F1-	.977	B1-	.937	J1-	.999
91-122	* W1-	.769	A1-	.735	F1-	.907	X1-	.943	J1-	.945
92-123	* W1-	.757	X1-	.776	A1-	.785	J1-	.873	C1-	.883
93-124	* X1-	.732	W1-	.802	71-	.806	K1-	.812	J1-	.841
94-125	* T1-	.772	K1-	.804	71-	.817	Y1-	.823	Y1-	.832
95-126	* T1-	.729	I1-	.738	Y1-	.826	K1-	.833	71-	.918
96-127	* I1-	.795	T1-	.816	Y1-	.901	V1-	.921	K1-	.981
97-128	* V1-	.891	I1-	.910	L1-	.950	F1-	.973	F1-	.974
98-129	* C1-	.798	L1-	.837	E1-	.873	F1-	.884	V1-	.931
99-130	* C1-	.891	U1-	.756	L1-	.774	E1-	.737	C1-	.802
100-131	* C1-	.436	G1-	.575	U1-	.837	D1-	.642	C1-	.768
101-132	* D1-	.414	G1-	.590	C1-	.632	D1-	.643	U1-	.576
102-133	* Q1-	.502	D1-	.596	C1-	.663	Q1-	.765	M1-	.820

RESULTS MATRIX SHOWING THE PROTOTYPE-TO-WINDOW DISTANCES FOR PROTOTYPES WHOSE WIDTH VARIATION ENABLES THE PROTOTYPE TO BE COMPARED TO WINDOWS OF WIDTH 32 COLUMNS. THE MATRIX IS SORTED BY DISTANCE WITHIN THE SENTENCE COLUMNS. THE 1 TO 13 CLOSEST IDENTIFYING PROTOTYPES ARE SHOWN FOR EACH WINDOW OF WIDTH 32 BEGINNING AND ENDING AT THE SENTENCE COLUMN INDICATED BY THE LEFTMOST MATRIX COLUMN.

PROTOTYPES IN THE MATRIX ARE: A1 B1 C1 D1 E1 F1 G1 H1 I1 J1 K1 L1 M1 N1 O1 P1 Q1

Complex Spectrum - Sentence #3

MATRIX 1 OF 1

MATRIX OF THE 1 TO 5 CLOSEST IDENTIFYING PROTOTYPES FOR WINDOWS OF WIDTH 32

SENTENCE	* GUESS	1	GUESS	2	GUESS	3	GUESS	4	GUESS	5
COLUMNS	* PROTO/	DX	PROTO/	DX	PROTO/	DX	PROTO/	DX	PROTO/	DX
1- 32	* X1-	.741	A1-	.852	I1-	.857	S1-	.891	T1-	.896
2- 33	* X1-	.824	I1-	.834	Y1-	.847	I1-	.891	S1-	.919
3- 34	* I1-	.837	T1-	.846	Y1-	.852	X1-	.955	S1-	.973
4- 35	* I1-	.863	Y1-	.906	V1-	.908	I1-	.925	L1-	.959
5- 36	* V1-	.834	L1-	.863	C1-	.945	E1-	.943	I1-	.968
6- 37	* C1-	.789	L1-	.806	C1-	.826	V1-	.824	U1-	.813
7- 38	* Q1-	.802	C1-	.702	U1-	.729	S1-	.822	I1-	.839
8- 39	* Q1-	.494	U1-	.731	C1-	.757	G1-	.761	O1-	.801
9- 40	* Q1-	.515	G1-	.819	C1-	.835	O1-	.855	U1-	.871
10- 41	* Q1-	.859	R1-	.921	O1-	.919	S1-	.915	O1-	.967
11- 42	* J1-	.831	R1-	.990	W1-	1.003	P1-	1.052	Q1-	1.087
12- 43	* J1-	.707	W1-	.962	T1-	.966	A1-	1.043	T1-	1.068
13- 44	* J1-	.708	T1-	.946	W1-	.929	I1-	.934	T1-	.953
14- 45	* J1-	.819	T1-	.829	T1-	.859	I1-	.859	A1-	.893
15- 46	* T1-	.817	A1-	.832	X1-	.883	T1-	.919	T1-	.928
16- 47	* A1-	.774	X1-	.789	T1-	.834	S1-	.873	T1-	.878
17- 48	* A1-	.730	R1-	.764	F1-	.767	X1-	.777	S1-	.801
18- 49	* A1-	.746	R1-	.772	J1-	.793	S1-	.811	P1-	.824
19- 50	* J1-	.671	T1-	.752	A1-	.810	W1-	.854	Y1-	.907
20- 51	* T1-	.553	I1-	.651	J1-	.675	Y1-	.645	V1-	.686
21- 52	* I1-	.483	T1-	.513	J1-	.633	Y1-	.650	W1-	.909
22- 53	* I1-	.605	T1-	.719	Y1-	.835	V1-	.940	W1-	.957
23- 54	* F1-	.830	V1-	.876	I1-	.894	E1-	.902	C1-	.927
24- 55	* F1-	.660	M1-	.710	S1-	.729	P1-	.730	C1-	.816
25- 56	* R1-	.899	F1-	.800	I1-	.813	Z1-	.828	P1-	.821
26- 57	* R1-	.582	R1-	.820	F1-	.805	E1-	.839	H1-	.703
27- 58	* R1-	.689	K1-	.761	F1-	.789	S1-	.802	W1-	.826
28- 59	* J1-	.819	K1-	.850	W1-	.855	A1-	.850	L1-	.931
29- 60	* T1-	.747	J1-	.812	J1-	.861	A1-	.916	W1-	.934
30- 61	* T1-	.698	I1-	.702	J1-	.927	A1-	.952	Y1-	1.000
31- 62	* I1-	.706	T1-	.827	V1-	.912	Y1-	.954	M1-	1.010
32- 63	* V1-	.797	I1-	.927	Y1-	.929	W1-	.932	T1-	1.001
33- 64	* V1-	.787	Y1-	.942	F1-	.952	L1-	.955	O1-	.967
34- 65	* V1-	.862	W1-	.895	F1-	.895	R1-	.914	F1-	.906
35- 66	* W1-	.831	P1-	.928	F1-	.943	F1-	.930	P1-	.951
36- 67	* W1-	.822	J1-	.958	T1-	.960	A1-	.934	F1-	1.009
37- 68	* W1-	.808	T1-	.931	J1-	.955	V1-	.952	T1-	.961
38- 69	* W1-	.831	S1-	.886	W1-	.935	X1-	.974	T1-	.975
39- 70	* S1-	.703	N1-	.842	Y1-	.843	R1-	.910	F1-	.939
40- 71	* S1-	.868	X1-	.773	F1-	.835	V1-	.874	F1-	.875
41- 72	* S1-	.548	X1-	.820	I1-	.844	R1-	.859	V1-	.876
42- 73	* S1-	.704	K1-	.860	I1-	.934	E1-	.953	Y1-	.965
43- 74	* S1-	.900	K1-	.909	T1-	.924	J1-	.930	T1-	.942
44- 75	* T1-	.879	I1-	.904	W1-	1.012	J1-	1.037	W1-	1.043

45-75	* I1-	.800	I1-	.800	V1-1.000	V1-1.000	V1-1.000			
46-77	* I1-1.	1.010	C1-1.	1.037	M1-1.	1.062	F1-1.	1.085		
47-78	* C1-	.916	O1-	.937	G1-	.955	J1-	.969	M1-	.982
48-79	* M1-	.796	O1-	.816	F1-	.826	G1-	.827	O1-	.836
49-80	* P1-	.714	M1-	.703	F1-	.732	O1-	.739	G1-	.821
50-81	* B1-	.773	R1-	.790	M1-	.683	O1-	.874	G1-	.944
51-82	* J1-	.851	R1-	.906	A1-	.930	B1-	.933	M1-	.959
52-83	* J1-	.537	T1-	.810	F1-	.923	W1-	.938	T1-	.996
53-84	* I1-	.514	J1-	.560	I1-	.731	V1-	.310	W1-	.945
54-85	* T1-	.329	I1-	.455	J1-	.686	V1-	.814	W1-	.978
55-86	* I1-	.436	T1-	.507	Y1-	.769	J1-	.913	V1-	.998
56-87	* I1-	.701	T1-	.800	Y1-	.805	V1-	.311	F1-	.943
57-88	* F1-	.769	E1-	.843	C1-	.860	V1-	.852	L1-	.890
58-89	* F1-	.666	P1-	.695	F1-	.720	O1-	.721	P1-	.733
59-90	* P1-	.562	R1-	.577	O1-	.600	A1-	.600	P1-	.656
60-91	* O1-	.588	R1-	.610	P1-	.633	R1-	.633	P1-	.641
61-92	* R1-	.744	R1-	.794	O1-	.734	A1-	.817	P1-	.846
62-93	* A1-	.811	J1-	.820	M1-	.875	R1-	.912	K1-	.990
63-94	* J1-	.678	A1-	.791	T1-	.868	W1-	.837	T1-1.	1.015
64-95	* T1-	.657	J1-	.659	I1-	.771	A1-	.837	W1-	.932
65-96	* T1-	.520	I1-	.531	J1-	.783	Y1-	.830	F1-	.927
66-97	* I1-	.465	T1-	.600	Y1-	.732	J1-	.877	Y1-1.	1.100
67-98	* I1-	.851	Y1-	.760	T1-	.819	V1-	.875	L1-	.970
68-99	* L1-	.757	V1-	.763	Y1-	.831	F1-	.855	F1-	.867
69-100	* L1-	.984	U1-	.717	E1-	.727	F1-	.728	C1-	.750
70-101	* U1-	.450	L1-	.510	C1-	.615	E1-	.655	F1-	.685
71-102	* U1-	.342	C1-	.618	L1-	.657	A1-	.670	G1-	.678
72-103	* U1-	.500	O1-	.570	G1-	.680	M1-	.702	O1-	.700
73-104	* O1-	.789	O1-	.811	C1-	.827	O1-	.835	U1-	.874
74-105	* J1-	.967	O1-	.980	C1-1.	1.005	R1-1.	1.021	G1-1.	1.022
75-106	* J1-	.793	T1-	.980	M1-1.	1.075	S1-1.	1.119	T1-1.	1.129
76-107	* J1-	.732	T1-	.820	I1-	.907	W1-1.	1.037	T1-1.	1.123
77-108	* T1-	.783	J1-	.835	I1-	.842	Z1-1.	1.111	W1-1.	1.113
78-109	* T1-	.891	I1-	.893	J1-1.	1.009	T1-1.	1.032	Y1-1.	1.103
79-110	* O1-1.	1.040	P1-1.	1.046	I1-1.	1.037	F1-1.	1.037	T1-1.	1.070
80-111	* P1-	.922	O1-	.971	P1-	.931	O1-	.935	F1-1.	1.007
81-112	* P1-	.878	R1-	.902	O1-	.917	R1-	.949	M1-	.989
82-113	* P1-	.916	P1-	.950	F1-	.971	O1-	.975	A1-	.991
83-114	* A1-	.828	W1-	.937	J1-	.972	B1-1.	1.014	P1-1.	1.047
84-115	* A1-	.697	J1-	.864	W1-	.874	T1-1.	1.031	U1-1.	1.055
85-116	* A1-	.675	J1-	.824	W1-	.835	T1-	.835	U1-	.935
86-117	* A1-	.776	T1-	.860	Y1-	.823	I1-	.834	F1-	.853
87-118	* X1-	.777	I1-	.730	T1-	.730	Y1-	.845	M1-	.854
88-119	* Y1-	.689	I1-	.819	V1-	.830	X1-	.837	T1-	.849
89-120	* V1-	.527	Y1-	.602	F1-	.843	I1-	.933	I1-	.948
90-121	* V1-	.323	Y1-	.687	F1-	.807	F1-	.915	F1-	.949
91-122	* V1-	.490	F1-	.619	Y1-	.847	P1-	.852	F1-	.876
92-123	* F1-	.783	P1-	.736	V1-	.814	W1-	.832	E1-	.843
93-124	* W1-	.733	F1-	.820	F1-	.822	R1-	.857	E1-	.870
94-125	* W1-	.770	T1-	.860	F1-	.912	P1-	.925	M1-	.950
95-126	* W1-	.917	Z1-	.942	J1-1.	1.005	F1-1.	1.010	T1-1.	1.010
96-127	* A1-	.978	I1-1.	1.012	J1-1.	1.049	T1-1.	1.073	F1-1.	1.074
97-128	* A1-	.992	I1-	.990	F1-1.	1.089	C1-1.	1.123	T1-1.	1.125
98-129	* I1-1.	1.057	A1-1.	1.061	F1-1.	1.065	C1-1.	1.071	L1-1.	1.096
99-130	* F1-1.	1.027	L1-1.	1.030	C1-1.	1.039	E1-1.	1.052	P1-1.	1.109
100-131	* L1-	.968	F1-1.	1.007	C1-1.	1.037	E1-1.	1.035	U1-1.	1.065
101-132	* L1-	.991	U1-1.	1.026	F1-1.	1.029	E1-1.	1.055	P1-1.	1.067
102-133	* U1-1.	1.039	L1-1.	1.042	F1-1.	1.091	P1-1.	1.104	O1-1.	1.109

45-77	* Y1-	.699	X1-	.765	T1-	.736	V1-	.675	W1-	.668
47-78	* Y1-	.715	T1-	.762	Y1-	.737	V1-	.613	I1-	.847
48-79	* Y1-	.370	V1-	.676	T1-	.867	M1-	.849	J1-	.594
49-80	* Y1-	.537	V1-	.676	I1-	.679	F1-	.609	W1-	.912
50-81	* V1-	.822	F1-	.844	E1-	.911	W1-	.915	K1-	.933
51-82	* F1-	.836	E1-	.892	L1-	.895	M1-	.916	K1-	.911
52-83	* F1-	.891	L1-	.919	F1-	.920	M1-	.923	O1-	.931
53-84	* O1-	.904	G1-	.955	E1-	.955	L1-	.932	U1-	.994
54-85	* O1-	.898	G1-	.904	O1-	.942	H1-	1.017	O1-	1.023
55-85	* G1-	.890	O1-	.905	O1-	.921	H1-	1.019	O1-	1.035
56-87	* O1-	.922	G1-	.929	O1-	.935	O1-	1.029	M1-	1.038
57-88	* O1-	.993	G1-	1.012	F1-	1.017	Q1-	1.054	M1-	1.090
58-89	* O1-	1.030	O1-	1.060	O1-	1.090	G1-	1.112	B1-	1.128
59-90	* J1-	1.072	O1-	1.082	O1-	1.087	T1-	1.110	M1-	1.158
60-91	* J1-	.973	T1-	1.015	V1-	1.126	I1-	1.132	M1-	1.137
61-92	* J1-	.914	T1-	.959	I1-	1.031	V1-	1.035	M1-	1.115
62-93	* J1-	.985	T1-	.955	I1-	.982	V1-	1.072	M1-	1.086
63-94	* J1-	.936	T1-	.986	J1-	.993	M1-	1.047	V1-	1.066
64-95	* J1-	.965	Z1-	.974	Y1-	.937	M1-	1.003	F1-	1.004
65-96	* Z1-	.686	X1-	.892	F1-	.937	Y1-	.945	M1-	.962
66-97	* Z1-	.431	X1-	.830	F1-	.926	R1-	.927	M1-	.936
67-98	* Z1-	.451	X1-	.869	K1-	.929	M1-	.937	F1-	.937
68-99	* Z1-	.718	K1-	.913	A1-	.931	F1-	.913	M1-	.969
69-100	* A1-	.910	T1-	.928	K1-	.952	I1-	.975	W1-	1.004
70-101	* I1-	.921	A1-	.934	Z1-	.938	K1-	1.025	J1-	1.056
71-102	* I1-	.908	T1-	.989	H1-	1.006	F1-	1.054	L1-	1.070
72-103	* I1-	.914	O1-	.984	L1-	1.030	F1-	1.033	E1-	1.008
73-104	* O1-	.902	L1-	.975	F1-	.991	E1-	1.035	U1-	1.042
74-105	* O1-	.872	L1-	.926	F1-	.954	O1-	.954	E1-	.999
75-106	* L1-	.896	U1-	.915	O1-	.921	F1-	.942	E1-	.984
76-107	* L1-	.909	U1-	.919	F1-	.974	P1-	1.005	E1-	1.010
77-108	* U1-	.984	L1-	.987	O1-	1.033	K1-	1.040	F1-	1.046
78-109	* O1-	1.006	U1-	1.096	K1-	1.034	L1-	1.107	M1-	1.121
79-110	* O1-	1.143	K1-	1.194	U1-	1.210	H1-	1.219	M1-	1.235
80-111	* O1-	1.247	M1-	1.257	M1-	1.307	M1-	1.307	M1-	1.309
81-112	* M1-	1.298	O1-	1.351	M1-	1.354	M1-	1.330	M1-	1.405
82-113	* M1-	1.366	M1-	1.393	O1-	1.437	M1-	1.447	Y1-	1.454
83-114	* M1-	1.424	M1-	1.436	Y1-	1.475	I1-	1.480	X1-	1.481
84-115	* M1-	1.449	X1-	1.486	M1-	1.492	I1-	1.482	Y1-	1.494
85-116	* M1-	1.473	X1-	1.489	E1-	1.493	I1-	1.497	L1-	1.499
86-117	* L1-	1.487	M1-	1.486	X1-	1.491	F1-	1.491	E1-	1.491
87-118	* J1-	1.478	L1-	1.475	I1-	1.479	F1-	1.483	P1-	1.484
88-119	* J1-	1.459	I1-	1.461	T1-	1.461	L1-	1.462	O1-	1.468
89-120	* A1-	1.000	E1-	1.000	J1-	1.000	Z1-	1.000	O1-	1.000
90-121	* A1-	1.000	E1-	1.000	J1-	1.000	Z1-	1.000	O1-	1.000
91-122	* A1-	1.000	E1-	1.000	J1-	1.000	Z1-	1.000	O1-	1.000
92-123	* A1-	1.000	E1-	1.000	J1-	1.000	Z1-	1.000	O1-	1.000
93-124	* A1-	1.000	E1-	1.000	J1-	1.000	Z1-	1.000	O1-	1.000
94-125	* A1-	1.000	E1-	1.000	J1-	1.000	Z1-	1.000	O1-	1.000
95-126	* A1-	1.000	E1-	1.000	J1-	1.000	Z1-	1.000	O1-	1.000
96-127	* A1-	1.000	E1-	1.000	J1-	1.000	Z1-	1.000	O1-	1.000
97-128	* A1-	1.000	E1-	1.000	J1-	1.000	Z1-	1.000	O1-	1.000
98-129	* A1-	1.000	E1-	1.000	J1-	1.000	Z1-	1.000	O1-	1.000
99-130	* A1-	1.000	E1-	1.000	J1-	1.000	Z1-	1.000	O1-	1.000
100-131	* A1-	1.000	E1-	1.000	J1-	1.000	Z1-	1.000	O1-	1.000
101-132	* A1-	1.000	E1-	1.000	J1-	1.000	Z1-	1.000	O1-	1.000
102-133	* A1-	1.000	E1-	1.000	J1-	1.000	Z1-	1.000	O1-	1.000

RESULTS MATRIX SHOWING THE PROTOTYPE-TO-WINDOW DISTANCE FOR PROTOTYPES HAVING WIDTH VARIATION ENABLES THE PROTOTYPE TO BE COMPARED TO WINDOWS OF WIDTH 32 COLUMNS MATRIX IS SORTED BY DISTANCE WITHIN THE SENTENCE COLUMNS. THE 1 TO 13 CLOSEST IDENTIFYING PROTOTYPES ARE SHOWN FOR EACH WINDOW OF WIDTH 32 BEGINNING AND ENDING AT THE SENTENCE COLUMN INDICATED BY THE LEFTMOST MATRIX COLUMN.

PROTOTYPES IN THE MATRIX ARE: A1 B1 C1 D1 E1 F1 G1 H1 I1 J1 K1 L1 M1 N1 O1 P1

Amplitude Spectrum - Sentence # 1

MATRIX 1 OF 1

MATRIX OF THE 1 TO 13 CLOSEST IDENTIFYING PROTOTYPES FOR WINDOWS OF WIDTH 32

SENTENCE COLUMN GUESS 1 GUESS 2 GUESS 3 GUESS 4 GUESS 5
 COLUMNS PROTO/DX PROTO/DX PROTO/DX PROTO/DX PROTO/DX

1- 32	* V1-	.300	P1-	.401	F1-	.450	A1-	.458	P1-	.468
2- 33	* V1-	.307	P1-	.401	F1-	.412	V1-	.415	F1-	.431
3- 34	* R1-	.313	T1-	.362	F1-	.374	R1-	.375	T1-	.432
4- 35	* T1-	.324	P1-	.341	T1-	.350	F1-	.351	H1-	.359
5- 36	* T1-	.300	P1-	.321	T1-	.332	H1-	.337	H1-	.358
6- 37	* T1-	.287	H1-	.288	F1-	.316	H1-	.338	H1-	.342
7- 38	* F1-	.268	H1-	.297	F1-	.319	H1-	.318	F1-	.339
8- 39	* T1-	.230	T1-	.241	F1-	.331	F1-	.318	F1-	.333
9- 40	* R1-	.258	P1-	.301	T1-	.320	F1-	.333	F1-	.343
10- 41	* R1-	.228	P1-	.320	F1-	.347	R1-	.342	T1-	.362
11- 42	* R1-	.211	P1-	.311	F1-	.340	E1-	.353	F1-	.402
12- 43	* R1-	.271	P1-	.301	F1-	.309	E1-	.413	F1-	.491
13- 44	* R1-	.281	P1-	.333	F1-	.437	E1-	.437	K1-	.478
14- 45	* R1-	.320	P1-	.457	F1-	.437	R1-	.457	P1-	.472
15- 46	* O1-	.377	K1-	.480	F1-	.473	P1-	.473	F1-	.492
15- 47	* R1-	.429	K1-	.441	A1-	.489	E1-	.493	F1-	.495
17- 48	* K1-	.432	A1-	.456	F1-	.472	H1-	.476	F1-	.488
18- 49	* K1-	.432	A1-	.440	H1-	.479	H1-	.479	O1-	.522
19- 50	* K1-	.429	A1-	.439	F1-	.473	H1-	.473	H1-	.533
20- 51	* K1-	.417	A1-	.420	H1-	.450	Y1-	.471	V1-	.519
21- 52	* A1-	.402	K1-	.401	H1-	.441	Y1-	.470	V1-	.494
22- 53	* A1-	.390	K1-	.390	H1-	.423	Y1-	.450	V1-	.470
23- 54	* A1-	.391	K1-	.391	F1-	.413	Y1-	.425	V1-	.450
24- 55	* K1-	.384	A1-	.390	Y1-	.411	H1-	.410	O1-	.427
25- 56	* K1-	.392	H1-	.394	Y1-	.400	A1-	.410	H1-	.430
25- 57	* O1-	.375	K1-	.410	Y1-	.420	A1-	.444	V1-	.451
27- 58	* O1-	.312	G1-	.433	K1-	.436	Y1-	.455	O1-	.448
28- 59	* O1-	.378	G1-	.421	O1-	.440	O1-	.440	K1-	.481
29- 60	* O1-	.411	G1-	.430	F1-	.430	O1-	.432	O1-	.478
30- 61	* O1-	.470	G1-	.447	O1-	.453	O1-	.453	O1-	.466
31- 62	* O1-	.430	G1-	.440	O1-	.450	O1-	.457	T1-	.492
32- 63	* O1-	.455	G1-	.475	T1-	.471	O1-	.470	O1-	.511
33- 64	* T1-	.485	O1-	.470	O1-	.470	O1-	.470	H1-	.502
34- 65	* T1-	.380	L1-	.467	O1-	.469	E1-	.497	J1-	.512
35- 66	* T1-	.300	L1-	.330	H1-	.425	J1-	.437	E1-	.472
36- 67	* T1-	.240	L1-	.330	H1-	.360	J1-	.363	F1-	.415
37- 68	* T1-	.210	T1-	.300	H1-	.360	J1-	.360	L1-	.400
38- 69	* T1-	.200	T1-	.270	L1-	.243	J1-	.327	T1-	.383
39- 70	* T1-	.217	H1-	.240	L1-	.240	J1-	.312	H1-	.310
40- 71	* T1-	.211	H1-	.220	L1-	.220	J1-	.311	H1-	.373
41- 72	* T1-	.211	H1-	.220	L1-	.201	J1-	.312	H1-	.362
42- 73	* T1-	.220	H1-	.220	L1-	.200	J1-	.310	H1-	.361
43- 74	* T1-	.210	H1-	.212	L1-	.312	J1-	.310	H1-	.340

row = 165

row = 1.2

45-76	* 01-	.28	T1-	.318	U1-	.329	I1-	.345	C1-	.372
46-77	* 01-	.28	U1-	.321	C1-	.325	S1-	.372	T1-	.393
47-78	* 01-	.27	01-	.328	01-	.328	J1-	.372	01-	.395
48-79	* 01-	.31	01-	.328	01-	.327	01-	.372	U1-	.385
49-80	* 01-	.32	01-	.328	01-	.328	01-	.372	01-	.379
50-81	* 01-	.32	01-	.328	01-	.327	Y1-	.313	Y1-	.386
51-82	* Y1-	.349	A1-	.371	01-	.372	K1-	.329	Y1-	.393
52-83	* A1-	.337	Y1-	.347	M1-	.374	A1-	.316	U1-	.400
53-84	* A1-	.338	Y1-	.349	K1-	.396	A1-	.416	V1-	.426
54-85	* A1-	.333	K1-	.361	Y1-	.396	A1-	.429	V1-	.439
55-86	* K1-	.338	A1-	.339	Y1-	.411	M1-	.413	F1-	.437
56-87	* K1-	.31	A1-	.324	F1-	.402	A1-	.414	Y1-	.431
57-88	* K1-	.297	F1-	.323	A1-	.373	A1-	.416	F1-	.412
58-89	* K1-	.287	F1-	.325	F1-	.395	A1-	.397	01-	.398
59-90	* K1-	.283	F1-	.321	01-	.379	P1-	.379	01-	.393
60-91	* K1-	.289	F1-	.321	F1-	.337	P1-	.356	01-	.387
61-92	* F1-	.308	K1-	.322	01-	.319	P1-	.351	01-	.375
62-93	* F1-	.313	F1-	.317	M1-	.327	01-	.372	F1-	.380
63-94	* F1-	.325	F1-	.325	01-	.324	K1-	.385	01-	.407
64-95	* F1-	.357	F1-	.341	F1-	.327	K1-	.411	01-	.425
65-96	* F1-	.483	01-	.423	01-	.432	F1-	.473	K1-	.473
66-97	* F1-	.483	01-	.424	P1-	.495	F1-	.511	01-	.527
67-98	* F1-	.497	01-	.416	F1-	.542	01-	.545	01-	.550
68-99	* F1-	.477	F1-	.421	01-	.421	T1-	.477	01-	.482
69-100	* F1-	.422	T1-	.451	F1-	.450	01-	.473	01-	.480
70-101	* F1-	.375	T1-	.370	F1-	.422	R1-	.429	01-	.431
71-102	* T1-	.319	01-	.320	01-	.327	R1-	.325	01-	.400
72-103	* T1-	.283	F1-	.323	01-	.330	I1-	.333	01-	.359
73-104	* T1-	.267	F1-	.297	01-	.319	T1-	.327	01-	.336
74-105	* F1-	.266	T1-	.265	01-	.307	I1-	.319	01-	.314
75-106	* F1-	.245	T1-	.284	01-	.310	J1-	.312	01-	.312
76-107	* F1-	.238	F1-	.302	01-	.311	J1-	.327	01-	.332
77-108	* F1-	.249	F1-	.311	01-	.327	P1-	.330	01-	.361
78-109	* F1-	.261	01-	.322	F1-	.341	P1-	.357	01-	.399
79-110	* F1-	.329	01-	.336	01-	.375	P1-	.330	01-	.416
80-111	* 01-	.341	F1-	.347	F1-	.423	01-	.411	01-	.432
81-112	* 01-	.379	01-	.440	F1-	.494	01-	.472	01-	.464
82-113	* 01-	.409	P1-	.454	01-	.477	01-	.476	01-	.481
83-114	* 01-	.422	P1-	.471	01-	.473	R1-	.473	01-	.483
84-115	* 01-	.439	01-	.450	Y1-	.452	01-	.475	01-	.479
85-116	* 01-	.444	01-	.452	Y1-	.451	01-	.473	01-	.475
86-117	* 01-	.441	01-	.459	Y1-	.452	01-	.474	01-	.480
87-118	* 01-	.437	01-	.458	01-	.455	Y1-	.473	01-	.485
88-119	* 01-	.423	01-	.432	01-	.458	T1-	.471	Y1-	.493
89-120	* 01-	.448	01-	.421	01-	.429	01-	.446	01-	.497
90-121	* 01-	.399	01-	.432	01-	.423	01-	.422	01-	.501
91-122	* 01-	.389	01-	.427	01-	.423	01-	.422	01-	.511
92-123	* 01-	.385	01-	.451	01-	.422	01-	.423	01-	.514
93-124	* 01-	.373	01-	.438	01-	.423	01-	.423	01-	.498
94-125	* 01-	.363	01-	.418	01-	.423	01-	.423	01-	.476
95-126	* 01-	.361	01-	.392	01-	.451	01-	.432	01-	.462
96-127	* 01-	.370	01-	.381	01-	.427	F1-	.423	01-	.451
97-128	* 01-	.351	01-	.381	F1-	.424	01-	.423	01-	.452
98-129	* 01-	.386	01-	.410	F1-	.410	01-	.439	01-	.441
99-130	* 01-	.382	F1-	.400	01-	.412	T1-	.431	01-	.432
100-131	* 01-	.387	F1-	.401	01-	.412	01-	.423	01-	.423
101-132	* 01-	.384	F1-	.401	01-	.407	T1-	.422	01-	.432
102-133	* 01-	.397	01-	.395	01-	.407	T1-	.410	01-	.438

RESULTS MATRIX SHOWING THE PROTOTYPE-TO-WINDOW DISTANCES FOR PROTOTYPES WHOSE WIDTH VARIATION ENABLES THE PROTOTYPE TO BE COMPARED TO WINDOWS OF WIDTH 32 COLUMNS. MATRIX IS SORTED BY DISTANCE WITHIN THE SENTENCE COLUMNS. THE 1 TO 17 CLOSEST IDENTIFYING PROTOTYPES ARE SHOWN FOR EACH WINDOW OF WIDTH 32 BEGINNING AND ENDING AT THE SENTENCE COLUMN INDICATED BY THE LEFTMOST MATRIX COLUMN.

PROTOTYPES IN THE MATRIX ARE: A1 G1 O1 D1 F1 F1 G1 H1 I1 J1 K1 L1 M1 N1 O1 P1

Amplitude Spectrum - Sentence #2

MATRIX 1 OF 1

MATRIX OF THE 1 TO 17 CLOSEST IDENTIFYING PROTOTYPES FOR WINDOWS OF WIDTH 32

SENTENCE 1 GUESS 1 GUESS 2 GUESS 3 GUESS 4
 COLUMNS 1 PROTO/DX PROTO/DX PROTO/DX PROTO/DX PROTO/DX

1- 32	* T1-	.309	T1-	.379	T1-	.371	R1-	.307	O1-	.394
2- 33	* T1-	.317	T1-	.320	R1-	.340	R1-	.313	O1-	.399
3- 34	* T1-	.316	T1-	.308	O1-	.368	R1-	.310	O1-	.409
4- 35	* T1-	.339	R1-	.329	T1-	.381	L1-	.417	F1-	.423
5- 36	* T1-	.374	O1-	.370	F1-	.391	F1-	.407	O1-	.452
6- 37	* R1-	.391	T1-	.359	T1-	.409	E1-	.432	O1-	.472
7- 38	* R1-	.398	T1-	.412	L1-	.418	E1-	.413	O1-	.476
8- 39	* T1-	.397	T1-	.390	L1-	.409	F1-	.432	O1-	.461
9- 40	* T1-	.394	R1-	.398	F1-	.411	O1-	.431	O1-	.451
10- 41	* T1-	.323	R1-	.409	L1-	.423	T1-	.433	F1-	.439
11- 42	* T1-	.321	T1-	.421	L1-	.424	M1-	.423	F1-	.444
12- 43	* T1-	.328	L1-	.413	T1-	.421	M1-	.431	O1-	.441
13- 44	* T1-	.341	M1-	.392	L1-	.404	L1-	.417	O1-	.429
14- 45	* O1-	.368	M1-	.394	T1-	.377	O1-	.340	O1-	.396
15- 46	* O1-	.336	M1-	.394	O1-	.371	O1-	.374	O1-	.398
16- 47	* O1-	.338	M1-	.383	O1-	.376	O1-	.373	O1-	.402
17- 48	* O1-	.362	T1-	.354	M1-	.399	O1-	.413	O1-	.407
18- 49	* T1-	.359	O1-	.400	J1-	.403	P1-	.423	M1-	.428
19- 50	* T1-	.370	T1-	.341	T1-	.397	P1-	.410	O1-	.427
20- 51	* T1-	.317	T1-	.374	J1-	.370	P1-	.410	F1-	.417
21- 52	* T1-	.316	T1-	.361	J1-	.393	P1-	.413	O1-	.430
22- 53	* T1-	.326	T1-	.359	J1-	.410	P1-	.410	O1-	.426
23- 54	* T1-	.351	T1-	.391	F1-	.413	P1-	.420	J1-	.440
24- 55	* T1-	.369	R1-	.412	T1-	.421	O1-	.410	T1-	.480
25- 56	* R1-	.393	T1-	.410	F1-	.430	L1-	.458	O1-	.478
26- 57	* T1-	.391	T1-	.400	L1-	.422	P1-	.439	T1-	.514
27- 58	* R1-	.380	O1-	.403	T1-	.428	P1-	.437	T1-	.528
28- 59	* R1-	.391	T1-	.413	T1-	.417	P1-	.431	T1-	.510
29- 60	* R1-	.377	O1-	.421	T1-	.400	T1-	.453	R1-	.474
30- 61	* O1-	.391	R1-	.413	T1-	.410	L1-	.424	F1-	.459
31- 62	* R1-	.391	O1-	.411	T1-	.411	T1-	.412	K1-	.438
32- 63	* R1-	.408	K1-	.410	T1-	.418	R1-	.410	T1-	.422
33- 64	* K1-	.402	R1-	.433	F1-	.437	T1-	.440	F1-	.445
34- 65	* K1-	.402	R1-	.457	O1-	.459	T1-	.451	T1-	.468
35- 66	* K1-	.413	O1-	.472	P1-	.473	F1-	.440	M1-	.464
36- 67	* K1-	.430	M1-	.467	F1-	.481	E1-	.431	O1-	.460
37- 68	* K1-	.408	M1-	.450	T1-	.452	T1-	.417	T1-	.413
38- 69	* M1-	.400	K1-	.454	L1-	.452	R1-	.422	T1-	.425
39- 70	* T1-	.400	M1-	.451	K1-	.451	T1-	.413	T1-	.421
40- 71	* O1-	.400	O1-	.440	T1-	.441	T1-	.421	K1-	.432
41- 72	* T1-	.401	O1-	.452	T1-	.451	F1-	.410	F1-	.419
42- 73	* T1-	.404	O1-	.451	T1-	.450	P1-	.410	F1-	.416
43- 74	* T1-	.404	O1-	.451	T1-	.451	P1-	.410	T1-	.412

44-75	T1-	.297	Y1-	.371	F1-	.394	J1-	.405	M1-	.423
46-77	T1-	.298	I1-	.361	F1-	.391	J1-	.397	P1-	.406
47-78	T1-	.297	I1-	.364	F1-	.394	J1-	.395	T1-	.416
48-79	T1-	.292	Y1-	.370	F1-	.376	J1-	.377	P1-	.411
49-80	T1-	.299	I1-	.367	F1-	.377	J1-	.377	P1-	.411
50-81	T1-	.317	P1-	.355	J1-	.394	K1-	.397	T1-	.401
51-82	T1-	.307	P1-	.372	F1-	.387	K1-	.397	T1-	.436
52-83	T1-	.314	P1-	.357	F1-	.417	T1-	.429	T1-	.447
53-84	T1-	.304	P1-	.371	P1-	.404	K1-	.432	F1-	.491
54-85	T1-	.310	P1-	.401	Y1-	.431	M1-	.432	M1-	.486
55-86	T1-	.416	X1-	.471	M1-	.456	R1-	.463	Y1-	.479
56-87	T1-	.375	K1-	.411	M1-	.427	J1-	.436	Y1-	.452
57-88	T1-	.390	X1-	.412	M1-	.426	J1-	.439	Y1-	.464
58-89	T1-	.364	Y1-	.422	F1-	.425	M1-	.432	Y1-	.445
59-90	T1-	.373	Y1-	.421	Y1-	.425	M1-	.435	Y1-	.449
60-91	T1-	.377	Y1-	.416	Y1-	.433	J1-	.435	Y1-	.454
61-92	T1-	.391	M1-	.393	M1-	.434	X1-	.437	Y1-	.462
62-93	T1-	.373	M1-	.403	M1-	.433	X1-	.439	Y1-	.473
63-94	T1-	.377	M1-	.421	M1-	.437	K1-	.453	Y1-	.489
64-95	T1-	.394	V1-	.413	J1-	.442	X1-	.452	T1-	.495
65-96	T1-	.377	V1-	.421	F1-	.443	J1-	.447	T1-	.483
66-97	T1-	.391	P1-	.401	F1-	.433	K1-	.433	Y1-	.489
67-98	T1-	.416	P1-	.421	F1-	.463	I1-	.458	T1-	.457
68-99	T1-	.401	J1-	.421	T1-	.447	K1-	.455	J1-	.472
69-100	T1-	.382	Y1-	.393	F1-	.399	J1-	.407	T1-	.425
70-101	T1-	.393	Y1-	.371	F1-	.377	J1-	.394	T1-	.412
71-102	T1-	.376	P1-	.366	J1-	.371	J1-	.377	P1-	.396
72-103	T1-	.374	P1-	.352	F1-	.389	I1-	.394	J1-	.387
73-104	T1-	.373	P1-	.364	P1-	.372	K1-	.373	T1-	.403
74-105	R1-	.366	P1-	.371	F1-	.372	T1-	.394	F1-	.424
75-106	T1-	.361	P1-	.335	F1-	.416	K1-	.434	T1-	.457
76-107	T1-	.391	T1-	.443	M1-	.450	P1-	.453	M1-	.490
77-108	T1-	.441	M1-	.453	F1-	.474	X1-	.490	T1-	.501
78-109	T1-	.412	X1-	.475	F1-	.495	K1-	.497	M1-	.552
79-110	T1-	.386	X1-	.486	K1-	.525	R1-	.531	M1-	.561
80-111	T1-	.375	Y1-	.407	M1-	.450	M1-	.476	F1-	.602
81-112	T1-	.374	Y1-	.496	M1-	.492	K1-	.493	Y1-	.639
82-113	T1-	.363	X1-	.489	M1-	.489	K1-	.624	Y1-	.615
83-114	T1-	.353	Y1-	.412	M1-	.619	K1-	.632	Y1-	.664
84-115	T1-	.354	Y1-	.436	M1-	.616	K1-	.627	Y1-	.613
85-116	T1-	.374	Y1-	.471	M1-	.613	K1-	.631	Y1-	.631
86-117	T1-	.373	Y1-	.492	M1-	.636	K1-	.647	T1-	.649
87-118	T1-	.384	X1-	.444	M1-	.446	M1-	.512	T1-	.564
88-119	T1-	.422	Y1-	.461	M1-	.463	K1-	.477	M1-	.536
89-120	T1-	.447	M1-	.471	M1-	.473	X1-	.477	M1-	.497
90-121	T1-	.447	K1-	.465	M1-	.467	J1-	.472	Y1-	.482
91-122	T1-	.404	K1-	.421	M1-	.437	M1-	.453	Y1-	.460
92-123	T1-	.394	M1-	.407	K1-	.426	Y1-	.437	M1-	.449
93-124	T1-	.393	M1-	.406	K1-	.426	Y1-	.426	M1-	.436
94-125	T1-	.391	M1-	.411	Y1-	.417	K1-	.423	M1-	.437
95-126	T1-	.396	Y1-	.401	M1-	.417	J1-	.423	M1-	.424
96-127	T1-	.392	Y1-	.411	M1-	.417	P1-	.415	M1-	.425
97-128	T1-	.394	Y1-	.394	M1-	.421	J1-	.424	T1-	.427
98-129	T1-	.356	O1-	.396	O1-	.436	Y1-	.416	O1-	.423
99-130	T1-	.366	O1-	.349	O1-	.384	G1-	.392	O1-	.424
100-131	T1-	.369	O1-	.357	O1-	.375	O1-	.393	O1-	.426
101-132	T1-	.371	G1-	.355	O1-	.402	O1-	.417	O1-	.442
102-133	T1-	.376	G1-	.376	O1-	.393	O1-	.426	O1-	.435

RESULTS MATRIX SHOWING THE PROTOTYPE-TJ-WINDOW DISTANCES FOR PROTOTYPES WHOSE WIDTH VARIATION ENABLES THE PROTOTYPE TO BE COMPARED TO WINDOWS OF WIDTH 32 COLUMNS. MATRIX IS SORTED BY DISTANCE WITHIN THE SENTENCE COLUMNS. THE 1 TO 13 CLOSEST IDENTIFIABLE PROTOTYPES ARE SHOWN FOR EACH WINDOW OF WIDTH 32 BEGINNING AND ENDING AT THE SENTENCE COLUMN INDICATED BY THE LEFTMOST MATRIX COLUMN.

PROTOTYPES IN THE MATRIX ARE: A1 F1 G1 H1 I1 J1 K1 L1 M1 N1 O1 P1

Amplitude Spectra - Sentence # 3

MATRIX 1 OF 1

MATRIX OF THE 1 TO 13 CLOSEST IDENTIFIABLE PROTOTYPES FOR WINDOWS OF WIDTH 32

SENTENCE GUESS 1 GUESS 2 GUESS 3 GUESS 4 GUESS 5
 COLUMNS PROTO/UX PROTO/UX PROTO/UX PROTO/UX PROTO/UX

1- 32	V1-	.363	Y1-	.351	T1-	.359	H1-	.423	M1-	.433
2- 33	V1-	.375	Y1-	.376	T1-	.399	H1-	.457	X1-	.461
3- 34	V1-	.387	Y1-	.381	T1-	.412	K1-	.430	K1-	.462
4- 35	V1-	.383	Y1-	.383	T1-	.434	O1-	.453	O1-	.462
5- 36	V1-	.378	Y1-	.379	O1-	.415	O1-	.433	M1-	.463
6- 37	O1-	.391	V1-	.404	Y1-	.410	O1-	.431	O1-	.464
7- 38	O1-	.375	O1-	.425	V1-	.471	Y1-	.464	O1-	.469
8- 39	O1-	.376	O1-	.415	G1-	.435	O1-	.457	V1-	.462
9- 40	O1-	.381	O1-	.411	G1-	.432	O1-	.457	V1-	.464
10- 41	O1-	.413	O1-	.413	G1-	.406	O1-	.478	V1-	.489
11- 42	O1-	.409	O1-	.427	O1-	.475	V1-	.477	O1-	.480
12- 43	O1-	.402	O1-	.434	T1-	.489	V1-	.462	Y1-	.471
13- 44	O1-	.410	T1-	.431	K1-	.471	A1-	.452	V1-	.480
14- 45	T1-	.417	K1-	.417	T1-	.432	O1-	.453	T1-	.486
15- 46	V1-	.387	T1-	.406	T1-	.429	F1-	.453	O1-	.466
16- 47	K1-	.388	T1-	.340	F1-	.429	A1-	.435	T1-	.472
17- 48	K1-	.339	T1-	.380	O1-	.415	M1-	.419	F1-	.436
18- 49	K1-	.329	T1-	.380	F1-	.349	F1-	.401	F1-	.417
19- 50	K1-	.320	T1-	.375	T1-	.377	F1-	.391	F1-	.392
20- 51	K1-	.335	T1-	.375	T1-	.372	R1-	.390	F1-	.392
21- 52	R1-	.314	O1-	.346	K1-	.371	T1-	.377	F1-	.383
22- 53	R1-	.319	O1-	.344	O1-	.376	P1-	.370	T1-	.373
23- 54	O1-	.297	R1-	.375	F1-	.370	F1-	.377	V1-	.398
24- 55	R1-	.293	O1-	.351	F1-	.360	K1-	.420	F1-	.420
25- 56	O1-	.286	T1-	.371	T1-	.424	K1-	.439	F1-	.449
26- 57	O1-	.300	O1-	.354	F1-	.437	K1-	.455	T1-	.474
27- 58	O1-	.332	T1-	.417	V1-	.451	P1-	.453	F1-	.469
28- 59	O1-	.367	O1-	.410	V1-	.451	P1-	.479	F1-	.492
29- 60	R1-	.389	R1-	.425	K1-	.432	P1-	.475	V1-	.478
30- 61	O1-	.413	R1-	.420	V1-	.458	H1-	.472	F1-	.483
31- 62	O1-	.431	R1-	.430	T1-	.475	K1-	.471	F1-	.499
32- 63	O1-	.466	R1-	.449	T1-	.469	K1-	.491	O1-	.500
33- 64	R1-	.465	P1-	.468	F1-	.488	K1-	.471	M1-	.510
34- 65	O1-	.492	R1-	.498	O1-	.466	K1-	.463	M1-	.518
35- 66	O1-	.491	O1-	.474	F1-	.472	K1-	.451	T1-	.577
36- 67	O1-	.488	O1-	.484	T1-	.471	K1-	.472	T1-	.474
37- 68	O1-	.481	O1-	.471	F1-	.473	K1-	.477	T1-	.477
38- 69	O1-	.471	V1-	.473	O1-	.437	F1-	.479	O1-	.511
39- 70	O1-	.460	V1-	.473	F1-	.433	M1-	.473	Y1-	.516
40- 71	O1-	.468	F1-	.471	V1-	.441	T1-	.471	T1-	.513
41- 72	O1-	.438	F1-	.474	K1-	.470	O1-	.470	O1-	.576
42- 73	O1-	.434	F1-	.470	V1-	.451	R1-	.440	F1-	.585
43- 74	O1-	.411	F1-	.466	F1-	.497	K1-	.454	F1-	.578

65-75	* 31-	.155	51-	.759	F1-	.315	E1-	.737	F1-	.535
66-77	* 31-	.152	51-	.767	G1-	.411	G1-	.812	F1-	.517
67-78	* 31-	.151	51-	.762	G1-	.404	G1-	.830	F1-	.505
68-79	* 31-	.147	F1-	.751	G1-	.432	G1-	.831	F1-	.462
69-80	* 71-	.354	21-	.277	F1-	.413	L1-	.850	U1-	.464
70-81	* 71-	.333	L1-	.754	F1-	.342	31-	.747	U1-	.399
71-82	* 71-	.284	F1-	.743	F1-	.343	U1-	.770	F1-	.376
72-83	* 71-	.277	F1-	.742	F1-	.336	U1-	.753	F1-	.376
73-84	* 71-	.242	F1-	.271	F1-	.335	U1-	.343	F1-	.313
74-85	* 71-	.233	F1-	.272	J1-	.334	U1-	.745	F1-	.346
75-86	* 71-	.234	F1-	.237	J1-	.325	U1-	.746	F1-	.348
76-87	* 71-	.274	F1-	.721	J1-	.332	P1-	.333	U1-	.351
77-88	* 71-	.310	P1-	.326	J1-	.757	G1-	.337	U1-	.360
78-89	* 31-	.331	F1-	.750	F1-	.751	U1-	.370	F1-	.382
79-90	* 31-	.344	G1-	.373	U1-	.378	31-	.379	F1-	.389
80-91	* 31-	.377	U1-	.743	F1-	.383	G1-	.732	F1-	.415
81-92	* 31-	.377	U1-	.777	P1-	.392	G1-	.732	F1-	.419
82-93	* 31-	.364	P1-	.736	F1-	.402	G1-	.702	F1-	.403
83-94	* 31-	.349	F1-	.777	F1-	.733	G1-	.710	F1-	.412
84-95	* 31-	.341	F1-	.757	F1-	.312	G1-	.711	F1-	.402
85-96	* 31-	.337	F1-	.757	F1-	.750	L1-	.347	F1-	.403
86-97	* 31-	.334	F1-	.770	F1-	.348	L1-	.713	F1-	.403
87-98	* 31-	.326	F1-	.731	F1-	.736	L1-	.752	F1-	.398
88-99	* 31-	.344	F1-	.327	F1-	.334	L1-	.730	F1-	.383
89-100	* 31-	.291	F1-	.731	F1-	.740	L1-	.743	F1-	.363
70-101	* 31-	.272	G1-	.741	F1-	.340	F1-	.351	F1-	.360
71-102	* 31-	.265	G1-	.325	G1-	.354	L1-	.742	G1-	.374
72-103	* 31-	.290	G1-	.332	G1-	.342	G1-	.353	G1-	.387
73-104	* 31-	.342	G1-	.335	G1-	.333	G1-	.403	G1-	.424
74-105	* 31-	.305	G1-	.417	G1-	.419	G1-	.353	G1-	.451
75-106	* 31-	.439	G1-	.405	G1-	.437	G1-	.453	G1-	.465
76-107	* 31-	.445	G1-	.464	G1-	.473	U1-	.730	F1-	.481
77-108	* 31-	.428	G1-	.477	F1-	.430	E1-	.430	F1-	.492
78-109	* 31-	.424	F1-	.473	F1-	.491	E1-	.435	G1-	.495
79-110	* 31-	.443	P1-	.479	V1-	.485	K1-	.496	U1-	.515
80-111	* 31-	.440	P1-	.480	F1-	.486	P1-	.436	V1-	.494
81-112	* 31-	.465	M1-	.484	Y1-	.470	A1-	.478	P1-	.479
82-113	* 31-	.374	M1-	.420	Y1-	.439	A1-	.451	F1-	.463
83-114	* 31-	.347	M1-	.416	Y1-	.417	A1-	.437	M1-	.457
84-115	* 31-	.347	Y1-	.434	F1-	.413	A1-	.451	U1-	.460
85-116	* 31-	.373	Y1-	.421	F1-	.421	A1-	.451	F1-	.470
86-117	* 31-	.335	Y1-	.392	F1-	.421	A1-	.442	U1-	.480
87-118	* 31-	.327	Y1-	.391	M1-	.420	A1-	.432	F1-	.488
88-119	* 31-	.348	Y1-	.371	M1-	.421	A1-	.425	U1-	.493
89-120	* 31-	.270	Y1-	.744	F1-	.345	M1-	.418	U1-	.495
90-121	* 31-	.269	Y1-	.321	F1-	.300	M1-	.417	F1-	.498
91-122	* 31-	.232	Y1-	.711	F1-	.347	M1-	.432	F1-	.479
92-123	* 31-	.244	Y1-	.321	F1-	.349	21-	.453	G1-	.467
93-124	* 31-	.205	Y1-	.354	F1-	.375	G1-	.433	G1-	.495
94-125	* 31-	.308	Y1-	.411	F1-	.422	G1-	.455	G1-	.460
95-126	* 31-	.434	Y1-	.457	G1-	.462	G1-	.454	F1-	.478
96-127	* 31-	.453	G1-	.487	Y1-	.475	V1-	.431	F1-	.514
97-128	* 31-	.450	G1-	.460	Y1-	.430	G1-	.411	V1-	.512
98-129	* 31-	.449	G1-	.473	Y1-	.502	G1-	.505	F1-	.512
99-130	* 31-	.447	G1-	.479	G1-	.503	Y1-	.510	F1-	.515
100-131	* 31-	.447	G1-	.487	G1-	.504	Y1-	.513	F1-	.521
101-132	* 31-	.449	G1-	.491	G1-	.506	Y1-	.525	G1-	.527
102-133	* 31-	.454	G1-	.492	G1-	.510	G1-	.525	Y1-	.535

RESULTS MATRIX SHOWING THE PROTOTYPE-TO-WINDOW DISTANCES FOR PROTOTYPES WHOSE WIDTH VARIATION ENABLES THE PROTOTYPE TO BE COMPARED TO WINDOWS OF WIDTH 32 COLUMNS. MATRIX IS SORTED BY DISTANCE WITHIN THE SENTENCE COLUMNS. THE 1 TO 5 CLOSEST IDENTIFYING PROTOTYPES ARE SHOWN FOR EACH WINDOW OF WIDTH 32 BEGINNING AND ENDING AT THE SENTENCE COLUMN INDICATED BY THE LEFTMOST MATRIX COLUMN.

PROTOTYPES IN THE MATRIX ARE: A1 B1 C1 D1 E1 F1 G1 H1 I1 J1 K1 L1 M1 N1 O1 P1

Amplitude Spectrum - Sentence # 21

MATRIX 1 OF 1

MATRIX OF THE 1 TO 5 CLOSEST IDENTIFYING PROTOTYPES FOR WINDOWS OF WIDTH 32

SENTENCE * GUESS 1 GUESS 2 GUESS 3 GUESS 4 GUESS 5
 COLUMNS * PROTO/DX PROTO/DX PROTO/DX PROTO/DX PROTO/DX

1- 32	* V1-	.319	Y1-	.397	A1-	.428	O1-	.435	O1-	.521
2- 33	* V1-	.311	Y1-	.361	I1-	.341	O1-	.457	M1-	.491
3- 34	* V1-	.259	Y1-	.299	A1-	.345	H1-	.447	O1-	.489
4- 35	* V1-	.240	Y1-	.270	I1-	.330	H1-	.411	O1-	.468
5- 36	* V1-	.241	Y1-	.271	A1-	.336	H1-	.397	O1-	.485
6- 37	* V1-	.255	Y1-	.283	A1-	.316	H1-	.345	H1-	.496
7- 38	* V1-	.270	Y1-	.293	A1-	.338	H1-	.335	Y1-	.501
8- 39	* V1-	.260	Y1-	.304	A1-	.307	M1-	.411	I1-	.497
9- 40	* V1-	.288	Y1-	.307	A1-	.371	H1-	.418	I1-	.488
10- 41	* V1-	.291	Y1-	.306	I1-	.376	H1-	.422	I1-	.474
11- 42	* Y1-	.316	V1-	.326	A1-	.392	H1-	.471	I1-	.467
12- 43	* Y1-	.368	V1-	.373	A1-	.431	O1-	.471	I1-	.473
13- 44	* Y1-	.410	V1-	.416	O1-	.481	I1-	.493	I1-	.508
14- 45	* Y1-	.405	O1-	.420	V1-	.524	O1-	.485	I1-	.547
15- 46	* O1-	.700	Y1-	.521	O1-	.509	O1-	.551	V1-	.569
16- 47	* O1-	.473	Y1-	.521	O1-	.541	O1-	.542	O1-	.545
17- 48	* O1-	.491	Y1-	.506	O1-	.511	O1-	.521	O1-	.536
18- 49	* O1-	.407	Y1-	.421	O1-	.405	O1-	.509	K1-	.510
19- 50	* O1-	.457	X1-	.430	Y1-	.477	V1-	.431	K1-	.490
20- 51	* Y1-	.413	M1-	.431	Y1-	.468	O1-	.473	K1-	.482
21- 52	* Y1-	.305	N1-	.406	Y1-	.464	K1-	.480	I1-	.496
22- 53	* Y1-	.324	H1-	.383	Y1-	.467	K1-	.435	M1-	.487
23- 54	* Y1-	.287	N1-	.371	Y1-	.478	H1-	.435	K1-	.491
24- 55	* Y1-	.208	M1-	.371	I1-	.411	Y1-	.415	V1-	.505
25- 56	* X1-	.257	M1-	.351	A1-	.410	Y1-	.410	V1-	.516
26- 57	* Y1-	.210	N1-	.385	M1-	.513	Y1-	.513	V1-	.524
27- 58	* Y1-	.240	N1-	.331	M1-	.411	Y1-	.417	V1-	.524
28- 59	* X1-	.242	M1-	.340	M1-	.407	Y1-	.413	K1-	.509
29- 60	* Y1-	.278	O1-	.383	K1-	.411	Y1-	.415	I1-	.459
30- 61	* Y1-	.365	M1-	.397	K1-	.410	Y1-	.415	M1-	.500
31- 62	* Y1-	.378	V1-	.421	K1-	.415	O1-	.415	V1-	.507
32- 63	* Y1-	.399	V1-	.430	K1-	.435	O1-	.437	O1-	.526
33- 64	* X1-	.415	O1-	.471	K1-	.511	V1-	.412	O1-	.517
34- 65	* O1-	.478	X1-	.411	O1-	.517	K1-	.433	M1-	.550
35- 66	* O1-	.488	G1-	.421	K1-	.505	X1-	.459	O1-	.569
36- 67	* O1-	.498	G1-	.431	I1-	.537	O1-	.445	K1-	.597
37- 68	* O1-	.517	O1-	.431	O1-	.411	I1-	.413	O1-	.531
38- 69	* O1-	.483	G1-	.420	O1-	.417	Y1-	.415	O1-	.570
39- 70	* Y1-	.488	O1-	.431	V1-	.517	G1-	.410	V1-	.531
40- 71	* Y1-	.474	V1-	.431	I1-	.467	O1-	.417	O1-	.520
41- 72	* Y1-	.377	I1-	.411	O1-	.403	O1-	.415	O1-	.505
42- 73	* Y1-	.377	V1-	.378	I1-	.403	I1-	.417	M1-	.491
43- 74	* Y1-	.327	V1-	.335	I1-	.430	A1-	.411	I1-	.480

75-76	* Y1-	.291	V1-	.729	71-	.423	41-	.632	01-	.467
76-77	* Y1-	.277	V1-	.712	71-	.607	41-	.475	01-	.671
77-78	* Y1-	.243	V1-	.812	71-	.738	41-	.421	01-	.490
78-79	* Y1-	.266	V1-	.703	71-	.778	41-	.413	01-	.694
79-80	* Y1-	.271	V1-	.711	71-	.775	41-	.422	01-	.485
80-81	* Y1-	.291	V1-	.733	71-	.841	41-	.433	01-	.476
81-82	* Y1-	.331	V1-	.833	41-	.414	41-	.653	01-	.489
82-83	* Y1-	.381	V1-	.729	71-	.431	01-	.606	01-	.565
83-84	* Y1-	.408	V1-	.769	71-	.531	01-	.732	01-	.535
84-85	* 01-	.472	Y1-	.573	01-	.576	01-	.735	01-	.591
85-86	* 01-	.523	01-	.424	61-	.675	71-	.633	01-	.645
86-87	* 71-	.619	01-	.641	01-	.650	01-	.573	01-	.676
87-88	* 71-	.706	01-	.647	61-	.649	01-	.635	01-	.694
88-89	* 71-	.644	01-	.676	01-	.674	L1-	.707	01-	.710
89-90	* 71-	.666	01-	.666	01-	.692	L1-	.710	01-	.711
90-91	* 71-	.610	01-	.667	01-	.647	31-	.736	01-	.716
91-92	* 71-	.670	01-	.661	01-	.684	E1-	.701	L1-	.707
92-93	* 71-	.633	01-	.645	01-	.674	E1-	.671	L1-	.708
93-94	* 71-	.612	01-	.645	01-	.671	E1-	.637	01-	.708
94-95	* 71-	.380	01-	.650	01-	.679	E1-	.637	01-	.709
95-96	* 71-	.360	01-	.647	01-	.673	E1-	.640	F1-	.714
96-97	* 71-	.710	K1-	.674	01-	.673	E1-	.633	01-	.710
97-98	* 71-	.369	K1-	.660	01-	.677	E1-	.687	01-	.706
98-99	* 71-	.311	K1-	.660	01-	.663	E1-	.672	L1-	.685
99-100	* 71-	.317	01-	.670	01-	.671	E1-	.672	L1-	.684
100-101	* 71-	.340	01-	.671	01-	.671	E1-	.675	L1-	.641
101-102	* 71-	.340	01-	.671	01-	.672	L1-	.672	01-	.624
102-103	* 71-	.323	01-	.672	01-	.673	L1-	.677	01-	.616
103-104	* 71-	.396	01-	.674	01-	.673	L1-	.678	F1-	.689
104-105	* 71-	.422	01-	.672	01-	.691	L1-	.645	01-	.607
105-106	* 71-	.448	01-	.685	01-	.695	L1-	.677	01-	.692
106-107	* 71-	.474	01-	.672	01-	.699	L1-	.671	01-	.602
107-108	* 71-	.500	01-	.680	01-	.690	L1-	.677	01-	.610
108-109	* 71-	.523	01-	.690	01-	.692	L1-	.645	01-	.610
109-110	* 71-	.543	01-	.691	01-	.695	L1-	.622	01-	.623
110-111	* 71-	.541	01-	.693	01-	.699	31-	.625	01-	.628
111-112	* 71-	.570	01-	.690	01-	.913	31-	.629	01-	.635
112-113	* 71-	.596	01-	.610	01-	.610	31-	.632	01-	.641
113-114	* 71-	.619	01-	.620	01-	.623	31-	.640	01-	.647
114-115	* 71-	.623	01-	.630	01-	.647	61-	.630	01-	.648
115-116	* 71-	.663	01-	.650	01-	.642	L1-	.679	01-	.679
116-117	* 71-	.682	01-	.670	01-	.710	K1-	.710	01-	.710
117-118	* 71-	.720	01-	.733	01-	.730	R1-	.717	01-	.748
118-119	* 01-	.741	61-	.707	01-	.712	K1-	.717	01-	.789
119-120	* 11-1	.900	71-1	.900	01-1	.900	01-1	.900	71-1	.900
120-121	* 11-1	.900	31-1	.900	01-1	.900	01-1	.900	71-1	.900
121-122	* 11-1	.900	31-1	.900	01-1	.900	01-1	.900	71-1	.900
122-123	* 11-1	.900	31-1	.900	01-1	.900	01-1	.900	71-1	.900
123-124	* 11-1	.900	31-1	.900	01-1	.900	01-1	.900	71-1	.900
124-125	* 11-1	.900	31-1	.900	01-1	.900	01-1	.900	71-1	.900
125-126	* 11-1	.900	31-1	.900	01-1	.900	01-1	.900	71-1	.900
126-127	* 11-1	.900	31-1	.900	01-1	.900	01-1	.900	71-1	.900
127-128	* 11-1	.900	31-1	.900	01-1	.900	01-1	.900	71-1	.900
128-129	* 11-1	.900	31-1	.900	01-1	.900	01-1	.900	71-1	.900
129-130	* 11-1	.900	31-1	.900	01-1	.900	01-1	.900	71-1	.900
130-131	* 11-1	.900	31-1	.900	01-1	.900	01-1	.900	71-1	.900
131-132	* 11-1	.900	31-1	.900	01-1	.900	01-1	.900	71-1	.900
132-133	* 11-1	.900	31-1	.900	01-1	.900	01-1	.900	71-1	.900

Vita

Roy Edward Bentkowski was born on 31 August 1955 in Detroit, Michigan. He graduated in 1973 from Grosse Pointe North High School in Michigan. He entered the University of Michigan, in Ann Arbor, where he received the degree of Bachelor of Electrical Engineering in April, 1977. Upon graduation he received a commission in the USAF through the ROTC program. He then entered the School of Engineering, Air Force Institute of Technology, in September 1977.

Permanent Address: 20711 Country Club
Harper Woods, MI 48225

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFIT/GE/EE/79-1	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) SEGMENTATION OF TOUCHING ENGLISH LETTERS		5. TYPE OF REPORT & PERIOD COVERED MS Thesis
7. AUTHOR(s) Roy Edward Bentkowski 2nd Lt.		6. PERFORMING ORG. REPORT NUMBER
9. PERFORMING ORGANIZATION NAME AND ADDRESS Air Force Institute of Technology (AFIT-EN) Wright-Patterson AFB, Ohio 45433		8. CONTRACT OR GRANT NUMBER(s)
11. CONTROLLING OFFICE NAME AND ADDRESS Foreign Technology Division/NIT Wright-Patterson AFB OH 45433		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		12. REPORT DATE March, 1979
		13. NUMBER OF PAGES 133
		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES Approved for public release; IAW AFR 190-17 Joseph P. Hipps, Major, USAF 16 MAY 1979 Director of Information		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Optical Character Recognition (OCR) Pattern Recognition Reading Machine Segmentation Problem		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) This paper examines the problem of building a machine to read uncontrolled type fonts set with essentially no space between letters (within words). The consequence of this type of data, which represents the usual format of printed text, is that the data vectors produced by the optical scanner contain multiple letters and/or fragments of letters that cannot be easily separated. An algorithm based on a variant of running cross-correlation between prototype letters and successively "windowed" fragments of		

ABSTRACT (continued)

tence is employed. The algorithm computes the Euclidean distance between prototypes and the sentence fragment in a filtered domain.

It is shown that appropriate normalization and windowing techniques allow perfect recognition of touching letters within a word. This occurs even when no a priori knowledge of letter location within the word is available, provided that suitable prototypes can be established.

Multiple alphabet prototypes were then built and used to recognize widely differing type fonts. Techniques to set acceptance thresholds were evaluated and the behavior of the resulting recognition system tabulated. A number of false triggers did occur in the test and these were discussed. Recommendations for further improvements in the system are suggested.