NATIONAL INSTITUTE OF OCEANOGRAPHY
WORMLEY, GODALMING, SURREY

N.I.O. Computer Programs 17

N.I.O. Internal Report No. N.17

APRIL 1970

N.I.O. COMPUTER PROGRAMS 17

MATHEMATICAL & STATISTICAL PROGRAMS

N.I.O. Internal Report No. N17

National Institute of Oceanography

N.I.O. PROGRAMS 17

1) PROGRAMS

163	Linear Regression by Least Squares, Y on X	LIREG
166	Single Linkage Cluster Analysis	CLUST
167	Preston Similarity Coefficient	PREST
170	Diversity Indices	DIVER
171	Spectral Analysis	SPAN
174	Bartlett's method for fitting:-	
	a) a linear relationship	
	b) an exponential functional relationship	BARTO
195	Calculation of Similarity Matrices	QMAT
198	Finite Complex Fast Fourier Transformation	FORA

2) SUBPROGRAMS

-30	ARCSINE	ARSIN(x)
-38	ARCCOSINE	ARCOS(x)
-39	TANGENT	TAN(x)
-40	Hyperbolic Cosine	COSH(x)
-41	Hyperbolic Sine	SINH(x)
-59	Double Integer Add and Subtract	IDADD, IDSUB
-60	Double Integer Multiply	IDMUL
-66	Reverse Binary	INVT(IVAL,Q)

ERRATA

Program 198

Under the paragraph entitled "Examples" insert the following:-

N.B. The word length of the input and output files must be defined in the *DEFINE FILE statement, as 192 (allowing 64 real numbers per record)

Subprogram -59

Method

After line 3 add:-

This adds (or subtracts) the double integer in J(1),J(2),by (from) the double integer in I(1),I(2), and places the results in K(1),K(2).

Queries regarding the use or availability of any of the programs in this volume may be made to:-

The Program Librarian,
Data Processing Group,
National Institute of Oceanography,
Wormley, Godalming, Surrey.

from whom a comprehensive list of all current N.I.O. Programs is available.

All the programs in this volume have been compiled and executed on an I.B.M.1800 Computer having the following configuration:-

1802 Processor-Controller with 16,384 words of core storage

2 2310 Disk Drives Model A

1 2401 Magnetic Tape Drive (30 kc/s)(7 Track)

1442 Model 6 Card Read - Punch

1443 Printer, 240 lines/minute

1816 Keyboard-Printer

Facit Paper Tape Reader, 1000 Characters/second

Facit Paper Tape Punch, 150 Characters/second

The operating system was TSX Version 3

Finite Complex Fourier Transform - FORT

This is a subroutine submitted by J. W. Cooley and to be found in the SHARE program library. We have acquired a copy of this subroutine and it is now written on to disk and therefore available for use by anyone requiring it as it stands.

FORT is essentially a fast fourier subroutine, computing the finite complex Fourier transform or its inverse, of a one-dimensional array of N complex numbers, where N must be a power of 2.

The method is dealt with fully in the description supplied with the subroutine, which may be found with one of the programmers in the D.P.G. A brief outline, however, of its method and application will be attempted below.

One must note that two storage cells* are required for each complex number, so 2N cells must be available in core storage space. On our computer there is, at present, the restriction that N, the number of complex terms must be a power of 2 less than or equal to $512 = 2^9$. However, mention must here be made of the FORA suite of linked programs (N.I.O. 198) which is nearing completion. The FORA program will enable the user to Fourier analyse (or synthesise) a very much larger amount of data than FORT alone can handle, by the method of "Doubling the Capacity of FORT" described in its write-up.

The subroutine FORT accepts as input a one-dimensional array of complex Fourier coefficients A(k), or of data X(i). With the complex array A(k), k=0, 1.....N-1 where $N=2^M$ as input, FORT computes

$$X(j) = \sum_{k=0}^{N-1} A(k)w^{jk} \quad \text{for } j=0,\ 1.....N-1 \quad \text{where } W = \exp(2\pi i/N)$$

and replaces the A's by the X's.

The inversion formula for calculating the A's when given the X's is

$$A(k) = \frac{1}{N} \sum_{j} X(j)\, W^{-jk}$$

In the calling (main) program, A must be dimensioned 2N, and S (the sine table), where used, is usually dimensioned N/4, and to compute a Fourier transform the statement

CALL FORT (A,M,S,IFS,IFERR)

must be included. A is the name of the array to be transformed with the real part of A(I), I=0,1.....N-1 stored in the cells with index 2*I+1 and its imaginary part in the cell immediately following.
M determines the length of the array, $N = 2^M$.
S is a vector containing SIN(Jπ/2*N) for J=1, 2....N-1

* N.B. A cell is defined (for a real number) as 3 - 16 bit words in extended precision.

IFS is an integer telling FORT to perform one of the following functions -

IFS = 0 to set up SIN table only

IFS = 1 to set up SIN table and calculate Fourier series

IFS = -1 to set up SIN table and calculate Fourier transform

IFS = 2 to calculate Fourier series only

IFS = -2 to calculate Fourier transform only

IFERR is an error flag set equal to zero if no error occurs.

If an error does occur, IFERR is set to

1 when IFS = $0, \pm 1$ which means that M is less than 1 or greater than 13

1 when IFS = ± 2 which means that the sine table is not large enough or has not been computed

After transformation, the A array will contain the 2N Fourier transforms of the series (or vice versa) and may be printed out with the usual WRITE statement.

Cathy Clayson
4th November 1969

N.I.O. PROGRAM 163

Title Linear regression by least squares, Y on X.

Name LIREG

Machine IBM 1800

Language 1800 Fortran IV

Purpose To fit points to a straight line by least squares.

Control Cards

//bJOB

//b*(Job No./Name/Title)

//bFORbLIREG

etc., followed by the program, then

//bXEQbLIREG

*CCEND

followed by the input data cards.

Input Data

Card No.	Function	Format
1	Number (N) of pairs of values to be fitted	I3
2 et. seq.	X_i, Y_i where i = 1, 2.....N	2F10.4

Any number of groups of cards with the above sequence can follow.

The input data MUST be terminated by a blank card.

Output Data The slope, inverse slope and intercept of the straight line are output as well as the standard errors of these values.

These numbers are followed by the residuals of the points read in, in the order in which the points were input (all in format F10.4, on the lineprinter).

Method The standard statistical equations are used. (e.g. see J. Topping, Errors of Observation and their Treatment, p.105). The standard error of the inverse slope is estimated as $(\frac{\Delta a}{a^2})$ where $y = (a \pm \Delta a)x + (b \pm \Delta b)$. Note that a Y on X fit is calculated, i.e. the values X_1, X_2..... X_N are assumed to be without error.

Programmer R. B. Whitmarsh

N.I.O. PROGRAM 166

Title	Single Linkage Cluster Analysis (Version two)
Name	CLUST
Machine	1800
Language	Fortran
Precision	Standard
Purpose	To carry out a single linkage cluster analysis using data in the form of an upper triangular similarity matrix.

Job Description	//bJOB
	//b*Job No./Name/Title
	//bXEQbCLUST
	*FILES(4,FILE1,1)
	*CCEND
	N.B. The *FILES card will refer to the upper triangular similarity matrix which will have been previously stored on disk by the user (using standard precision).

Data	Immediately follows *CCEND 1) Alphanumeric information describing data (one card only) 2) S1, δS, S2, N formatted 3(E10.3),I4 where;

S1 is the initial similarity value at which the clustering will begin.

δS is the similarity increment which is added algebraically to the similarity level at the end of each clustering cycle.

S2 is the final similarity value. When the similarity level becomes equal to S2 the cluster analysis will cease.

N is the number of entities to be clustered (i.e. the order of the similarity matrix).

It is implied that the similarity level S1 is higher, in terms of similarity than S2. For most distance coefficients this means that S1<S2 and so δS will be a positive quantity. However if the similarity coefficients used is, for example, the correlation coefficient then S1>S2 and δS would be a negative quantity.

3) Entity name card

The entities in the upper triangular similarity matrix can be assigned a four-character name which will then be used in the program output. The card is punched in four-column fields with the order of names the same as the order of entities in the first row of the upper triangular similarity matrix. If there are more than 20 entities further cards can be included.

Output

1) Data description (input item 1)

2) S1, δS, S2, N (input item 2)

At each clustering cycle the program will print the following:

a) Similarity level of clustering cycle.

b) A list of the linkages that occur at that similarity level. Each item of the list contains the names of the two entities that are being linked and the similarity between these two entities.

c) At the end of the cycle the cluster numbers and a list of the entities making up each cluster (at that stage in the cluster analysis) is printed.

d) At the end of the specified number of clustering cycles the program will print out the name of any entities that have not yet been included into any cluster.

Method

When two entities first join to form a cluster the program assigns it a cluster number. This is done sequentially as clusters are formed and the cluster retains this number as it grows larger. When two clusters join together to form a single cluster this new cluster will take on the lowest cluster number of its two constituent clusters.

A more detailed description of the rationale behind the method is given in "The application of a computer to Taxonomy" by P. H. A. Sheath (Journal of Gen. Microbiology Vol. 17, 201-226) and in "The construction of hierarchic and non-hierarchic classifications" by N. Jardine and R. Sibson (Computer Journal Vol. 11, 1968 p. 177-185).

Similarity matrices can be calculated from raw data using N.I.O. PROGRAM 195.

Restrictions

1) The number of entities must be less than or equal to 250. The program will print the message

N GREATER THAN 250

if this restriction is violated.

Execution Time — A matrix of order 60 took approximately 15 minutes to cluster. However execution time will vary for different matrices of the same size.

Programmer — M. Fasham

N.I.O. PROGRAM 167

Title	Preston Similarity Coefficient
Name	PREST
Machine	IBM 1800
Language	Fortran
Purpose	Given a set of samples, the number of species in each, and an upper triangular matrix of species common to each pair of samples, to calculate the Preston similarity coefficient for each sample pair.

Control Cards	//bJOB
	//b*Job No./Name/Title
	//bFORbPREST
	etc., followed by program, then
	//bXEQbPREST
	*CCEND

Data	Immediately follows *CCEND and consists of the following cards:- 1) Integer representing number of samples considered FORMAT: I3 2) YYYY, ZZ, YYYY, ZZ, ... where YYYY is sample number ZZ is number of species in that sample FORMAT: 13(I4,I2) for each card. Subsequent cards same till all samples listed. Last card in this set blank after last sample entry. Then follows upper triangular matrix of number of species common to each pair of samples. FORMAT: 40I2 for each card. Check if N is number of samples, then number of elements in triangular matrix is N(N-1)/2.
Output	Listing of values under following headings:-

Sample 1	Species 1	Sample 2	Species 2	Species in Common	Preston Coefficient
XXXX	XX	XXXX	XX	XX	0·XXX

Restrictions	Maximum number of samples = 100.

Method

The following equation is solved to find the Preston coefficient, Z, for a sample pair:-

$$x^{1/Z} + y^{1/Z} = 1$$

where **x** is the proportion of the joint fauna in one sample,
y is the proportion of the joint fauna in the other sample.

For more detailed discussion, see "The Canonical Distribution of Commonness and Rarity, Part II p.418, by F. W. Preston, (Ecology 43 (3) 1962 pp.410-432).

Modification

If it is required to store the Preston coefficients on disk, the following modifications to the program are necessary:-

1) Remove card 'NC = NC + 1 PRES0400'

2) Replace by instruction to write Preston coefficient to disk:

   'WRITE (5'NC) Z PRES0400'

   using 'NC' as record index which is automatically updated.

3) Add file definition cards:-

   a) 'DEFINE FILE 5 (M,3,U,NC) PRES0030'

   where M is number of sample pairs (M=N(N-1)/2 where N is number of samples).

   and other three parameters fixed as above.

   b) '*FILES(5,file name,drive no.)' just before *CCEND card.

4) Amend job card, to give disk number in col. 15-19, format I5.

5) Add '*IOCS (DISK)' card.

The upper triangular Preston similarity matrix is then stored on disk ready for use (e.g. Cluster Analysis, Multidimensional Scaling).

Programmer

Anne Wilkinson.

N.I.O. PROGRAM 17?

Title — Diversity Indices

Name — DIVER

Machine — IBM 18??

Language — Fortran

Purpose — To compute diversity indices from distribution of individuals among species within a sample.

Control Cards

//bJOB
//b*Job No./Name/Title
//bFORbDIVER
etc., followed by program, then
//bXEQbDIVER
*CCEND

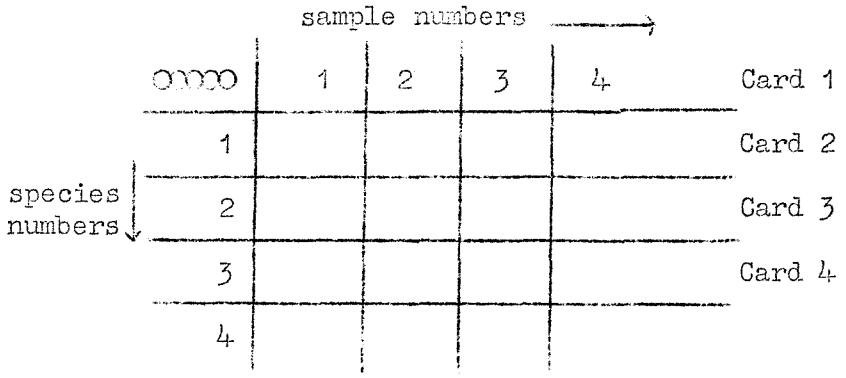
Data

Immediately follows *CCEND and consists of:

Card 1: AAA, BBBB, CCC Format: I3,I4,I3

where AAA is number of sets of data given,
BBBB is total number of samples,
and CCC is total number of species in each set.

Then follow AAA sets of data on cards in the form of a species - individuals matrix, the elements of which are the numbers of individuals in each species within each sample:-



1st card: cols. 1 - 5 0
cols. 6 - 75 14 sample numbers in 14I5 format

2nd card: cols. 1 - 5 species number
etc. cols. 6 - 75 number of individuals in that species in each sample.

In each set there is one card with sample numbers on, followed by one card for each species present in the whole data.

Cols. 76 - 80 are used for card sequencing, and not used by this program.

Subsequent sets of data are similar. Each set of data must contain the same number of cards - i.e. each species in the analysis must have a corresponding card in each set, even if the species does not appear in that particular set of samples.

Format: 16I5 for each card.

Output

Listing of values under the following headings:-

Sample No.	No. Species	No. Individuals	Div.(Simpson'49)
XXXXX	XXX	XXXXX	0.XXX

Div. (Lloyd '64)

XX.XXX

Restrictions

Maximum number of species = 250.

Method

For each sample, two diversity indices are calculated, which depend on the distribution of individuals among species.

a) Simpson (1949)

We subtract the Simpson index from unity so that the value of the index increases with increasing diversity

$$\text{Diversity} = 1 - \sum_{i=1}^{s} \frac{n_i(n_i - 1)}{N(N - 1)}$$

where S = total no. of species
N = total no. of individuals
n_i = no. of individuals in i^{th} species

(Ref. Simpson, E.H. "Measurement of Diversity" Nature (London), 163, 1949, p.688).

b) Shannon-Weaver (1963)

We use the Shannon-Weaver information function as described in

Lloyd and Ghelardi "A table for calculating the equitability component of species diversity", J.Anim.Ecol., 1964, 33, pp.217-225.

$$\text{Diversity} = -\sum_{i=1}^{s} p_i \log_2 p_i \quad \text{where } p_i = \frac{n_i}{N}$$

(For further discussion, see Pielou, E.C., "The Measurement of diversity in different types of biological collections", J.Theor.Biol., 13, 1966, pp.131-144).

Programmer

Anne Wilkinson.

N.I.O. PROGRAM 171

Title Spectral Analysis

Name SPAN

Machine IBM 1800

Language 1800 Fortran IV

Purpose To compute the lagged normalised auto-covariances and auto-spectra of a single time series.

Job Cards

```
//bJOB
//b*Job No./Name/Title
//bXEQbSPAN
*CCEND
```

Data This follows the *CCEND card and consists of cards as shown:-

One parameter card containing the required number of covariance lags (LAGS), the required number of frequencies (FREQ), the sampling interval between successive terms of the series (E), and finally the conversion factor from digital to physical units (CONVF).

Format: 3I6, 4X, F8.5

Then the series itself is punched into subsequent cards, 5 values per card,

Format: A1, 5F10.4

but with a '/' in column 1 of the last card.

Output The first and last terms of the series are output, together with the total number of terms. Then follows the mean and variance of the series. Next, the lagged normalised auto-covariances and auto-spectra are output for the specified number of lags.

Method Let the series of data be x_i where $i = 1(1)N$

Let the conversion factor from digital to physical units be G_x (= CONVF).

Let the mean of the series be represented by X.

Then the variance of the series is given by -

$$V_x = \frac{1}{N+1} \sum_{i=1}^{N+1} (x_i - X)^2 \quad \text{where } X = \frac{1}{N} \sum_{i=1}^{N} x_i$$

and in physical units $V_x' = G_x^2 V_x$

The program uses the IBM supplied subroutine AUTO to compute the lagged normalised auto-covariances and this is given by -

$$Y(r) = \frac{1}{V_x} \cdot \frac{1}{N-r+1} \sum_{i=1}^{N-r+1} (x_i - X)(x_{i+r-1} - X)$$

$$(r = 0(1)LAGS)$$

The auto spectra is calculated from the following equations -

$$EGY(s) = 4.E.V_x.G_x^2 \sum_{r=0}^{L}{}'' \phi(r) \cos\left(\frac{rs\pi}{L}\right)$$

$$(s = 0(1)FREQ)$$

where $\sum''$ means the sum with the first and last terms halved and

$$\phi(r) = Y(r) \cos^2\left(\frac{\pi r}{2L}\right)$$

N.B. This program does not produce exactly the same results as with N.I.O. 92, particularly when N is small, since the equation used by AUTO uses N + 1 whereas N.I.O. 92 uses N.

Restrictions

Provided 10xLAGS ≤ N, then N ≤ 1450 (otherwise see programmer in order to modify program)

LAGS ≥ FREQ

N ≥ LAGS

Execution Time

Approximately (LAGS/6) minutes.

Programmer

Catherine Clayson

N.B. In order to read data from other input sources the 'LUN' number on card 17 has to be altered.

N.I.O. PROGRAM 174

Title Bartlett's method for fitting:-

a) a linear relationship

or b) an exponential functional relationship.

Name BARTO

Machine IBM 1800

Language Fortran

Purpose Given a set of points (x, y) to compute the best values of a and b, by using Bartlett's method, to fit a straight line of the form:-

a) $y = a + bx$ (linear relationship)

or b) $\log y = \log a + b \log x$ (the exponential relationship being $y = ax^b$)

The 70% and 95% confidence limits for the slope b are also found.

Control Cards

//bJOB

//b*Job No./Name/Title

//bFORbBARTO

etc. followed by program, then

//bXEQbBARTO

*CCEND

Job Description and Data

The data immediately follows the *CCEND card and consists of:-

Card 1: One line of title

Card 2: The number of sets of data (K) (I2 format - xx)

For each set of data:

1st card: Either N for linear option or L for exponential option.

2nd card: Number of points (M) in this set of data in I3 format (XXX)

Successive cards: The M points in F9.4 format with 8 numbers to each card. Both co-ordinates must be positive when the exponential relationship is specified.

$x_1 \quad y_1 \quad x_2 \quad y_2 \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots x_8 \quad y_8$

$\vdots$

$x_{M-7} \quad y_{M-7} \quad x_{M-6} \quad y_{M-6} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots x_M \quad y_M$

Output

The data title

For each set of data:

1) The data set number
2) The number of points in the set of data (M)
3) The equation of the line in the form
   a) $y = a + bx$
   or b) $\log y = \log a + b \log x$
4) Two points at the extremes of the line to aid plotting
5) The 70% confidence limits on the slope
6) The 95% confidence limits on the slope

Restrictions

$(x,y) > 0$ for exponential option

$4 \leq N \leq 500$

$(N/3 - 1) \leq IDIV \leq (N/3 + 1)$ and IDIV must be integer.

Failures

The program tests to see that N>3, if it is ≤ 3 then a message is printed and the program skips to the next set of data.

If a negative number is encountered of which the logarithm is required then a message is printed and the next set of data is dealt with.

A test is made on the number under the square root. If this is negative then a message is printed saying

CONFIDENCE LIMITS ARE IMAGINARY

Method

The M data points are divided into 3 groups, the two end groups having the same number, IDIV, of points chosen to be as near M/3 as possible. The three groups must be non-overlapping in the x-direction and if two or more different values of y occur for the same value of x either side of a division, suitable adjustment must be made to the y values.

e.g. (386,10) | (386,12) (386,14)

should be altered to

(386,12) | (386,12) (386,12)

If the exponential function is specified $\log_e x$ and $\log_e y$ are then calculated and hereafter referred to as x and y, otherwise the data remains unchanged and represented by x and y.

The means of the two end groups $(\bar{x}_1, \bar{y}_1)$ and $(\bar{x}_3, \bar{y}_3)$ are first evaluated. The line joining these points gives the value of slope

$$b = (\bar{y}_3 - \bar{y}_1) / (\bar{x}_3 - \bar{x}_1)$$

The functional relation is then a line with this slope passing through the grand mean $(\bar{x}, \bar{y})$

a) $a = \bar{y} - b\bar{x}$

and $y = a + bx$

or b) $a = e^{(\bar{y} - b\bar{x})}$

and $\log y = \log a + b \log x$

The confidence limits of the slope are provided by the solu ions β_1, β_2 of the quadratic equation

$$\tfrac{1}{2}IDIV\ (\bar{x}_3 - \bar{x}_1)^2(b - \beta)^2 = t^2(C_{yy} - 2\beta C_{xy} + \beta^2 C_{xx})/(N-3)$$

using values t_1 and t_2 of t for the 70% and 95% confidence intervals for the M-3 degrees of freedom available within the groups, where

$$C_{yy} = \sum_{1}^{M} (y_i)^2 - \left\{ \left(\sum_{1}^{IDIV} y_i \right)^2 / IDIV + \left(\sum_{IDIV+1}^{M-IDIV} y_i \right)^2 / (M-2IDIV) + \left(\sum_{M-IDIV+1}^{M} y_i \right)^2 / IDIV \right\}$$

C_{xx} is similarly defined

$$C_{xy} = \sum_{1}^{M} (x_i y_i) - \left(\sum_{1}^{IDIV} x_i \sum_{1}^{IDIV} y_i \right)/IDIV - \left(\sum_{IDIV+1}^{M-IDIV} x_i \sum_{IDIV+1}^{M-IDIV} y_i \right)/(M-2IDIV) - \left(\sum_{M-IDIV+1}^{M} y_i \sum_{M-IDIV+1}^{M} x_i \right) / IDIV$$

The confidence limits are then expressed as

$+ (\beta_1 - b)$

$- (b - \beta_2)$

where $\beta_1 > \beta_2$

Finally two points on the line are evaluated.

a) $(x_1, a + bx_1)$ and $(x_M, a + bx_M)$

or b) (e^{x_1}, ae^{bx_1}) and (e^{x_M}, ae^{bx_M})

Notes

The method is described fully in

"Statistical Methods in Research and Production", edited by O. L. Davies, pub. by Oliver and Boyd (1958) p.175.

and

Bartlett, M. S. "Fitting a straight line when both variables are subject to error, Biometrics, Vol. 5, No. 3 (1949) p.207.

Programmer

Maureen Tyler.

N.I.O. PROGRAM 195

Title	Calculation of similarity matrices
Name	QMAT
Machine	1800
Language	Fortran IV
Precision	Standard
Job Description	//bJOB X^{18} //bXEQbQMAT FX^{16}

Purpose

If a set of attributes have been measured for a given number of entities QMAT can be used to calculate a matrix of similarities between entities (called a Q type similarity matrix). QMAT also allows the user to output the calculated matrix in two distinct ways;

1) The matrix is calculated in upper triangular form and written onto disk (data file CLUSF). N.I.O. Program 166 can then be used to carry out a cluster analysis of the matrix.

2) The whole similarity matrix is calculated (Square form) and then punched on to cards. These cards can then be used as input to the factor analysis program that is part of the IBM 1130 statistical package (1130-CA-06X).

The program also possesses the following features,

a) The raw data can be read from disk or cards. If the data is on cards the format can be specified at run time.

b) The attributes can be scaled using two different methods.

c) Three different types of similarity coefficient can be calculated.

d) The similarity matrix can be calculated using a sub set of the total attribute set.

Data

The following cards are required as data

1) Job-Title card
2) Option card
3) Attribute card
4) Entity name cards
5) Variable format cards
6) Data cards

1) Job-Title card

The job-title card allows the user to assign a job number and title information for the job to be processed. This will be printed out by the program

Format:	Column	Meaning
	1 - 4	Job number
	5 - 8	Not used
	9 - 80	Title

2) Option card

Number of entities (cc 1 - 3)

The number of entities must be less than or equal to 30 for square similarity matrices or less than or equal 250 for upper triangular similarity matrices.

Total number of attributes (cc. 4 - 6)

The total number of attributes must be less than, or equal to 100.

Number of attributes to be used in calculation (cc. 7 - 8)

This field specifies the number of attributes that are to be used to calculate the similarity coefficients. This number must be less than or equal to 30.

Data matrix input type (cc. 9 - 10)

This field allows the user to specify the input device (card or disk). The values that can be punched are described below

Value	Meaning
1	Raw data will be read from cards using the format specified by the variable format cards.
2	Raw data will be read from disk

Number of variables on card 1 (cc. 11 - 12)

When a data vector contains more attributes than will fit on one card the user must indicate to the program how many variables are on each card. If there are no variables on a particular card this field must be left blank.

Number of variables on card 2 (cc. 13 - 14)

Same as for cc. 11 - 12.

Number of variables on card 3 (cc. 15 - 16)

Same as for cc. 11 - 12.

Number of variables on card 4 (cc. 17 - 18)

Same as for 11 - 12.

Number of variables on card 5 (cc. 19 - 20)

Same as for cc. 11 - 12.

Number of variables on card 6 (cc. 21 - 22)

Same as for cc. 11 - 12.

Type of Q similarity matrix (cc. 23 - 24)

This field specifies the type of similarity matrix required and also it's mode of output. If "none" is specified the attribute vectors (after transformation if this is specified) are read back to the temporary area on disk and the programs exits. They can then be used as input to the factor analysis program if it is required to carry out an R type factor analysis (i.e. using the similarities between attributes).

The definitions of correlation coefficient, coefficient of proportional similarity and taxonomic distance are given in the formulae section.

Value	Meaning
1	None
2	Square proportional similarity coefficient matrix
3	Square correlation coefficient matrix
4	Upper triangular proportional similarity coefficient matrix
5	Upper triangular correlation coefficient matrix
6	Upper triangular taxonomic distance matrix

Transformation type (cc. 25 - 26)

The attributes can be transformed before calculating the similarity matrices. The definitions of the two transformations are given in the formulae section.

Value	Meaning
1	No Transformation
2	Standardisation
3	Normalisation

Data matrix printout (cc. 27 - 28)

This field allows the user to specify whether the data matrix, the derived data matrix (i.e. the data matrix containing only those attributes specified by the attribute card) and the transformed data matrix are to be printed on the line printer.

Value	Meaning
1	No Print-out
2	Print-out

Similarity matrix output (cc. 29 - 30)

This field specifies whether the matrix is to be printed on the lineprinter or punched on to cards

Value	Meaning
0	No output
1	Print matrix on lineprinter
2	Print and punch
3	Punch only

3) Attribute card

This card allows the user to specify what attributes (out of the total set) are going to be used to calculate the similarity coefficients. The attributes are numbered in the order that they appear on the data card and the numbers of those attributes that are required are typed on to the card using two-column fields starting in column one. For example if the data card contained ten attributes and attributes two and three were required for the calculation then the attribute card would be

4) Entity name card

The user can name the entities to aid in the interpretation of the matrix print-outs. Four character names are assigned to the entities in the same order that they are read in by the program. The entity name card is then punched with these names using 4 column fields. If there are more than 20 entities, further cards may be used.

5) Variable format cards

Up to six variable format cards can be used to specify the format of the data cards. The method of doing this and the formats that can be used are described in the programs descriptions of N.I.O. Subprograms -43 and -45.

6) Data cards

If the data is to be read in from cards up to six cards can be used to specify the attributes for an entity. Obviously the data layout must conform to the format specified by the variable format cards.

Output

The following data can be printed out

1) Raw data matrix
2) Derived data matrix
3) High and low values of the attributes (if normalisation is specified)
4) Mean and standard deviation of the attributes (if standardisation is specified)
5) Transformed data matrix
6) Similarity matrix

Formula

If the input data consists of n entities each of which has m attributes then the data matrix will have n rows and m columns. Thus the element Xij of the matrix will be the jth attribute of the ith entity.

Before calculating Q matrices it is usual to transform the attributes to allow for the fact that they may be all measured in different units (see SOKAL and SNEATH, 1963). The program offers two alternative methods, standardisation and normalisation. Let the mean and standard deviation of attribute j be μj and σj then standardisation is defined by

$$\frac{Xij - \mu j}{\sigma j} \qquad i = 1, 2, \ldots n$$

let the highest and lowest value of attribute j be defined by XHj and XLj then normalisation is defined by

$$\frac{Xij - XLj}{XHj - XLj} \qquad i = 1, 2, \ldots n$$

The element Cjk of the correlation coefficient matrix is given by

$$\frac{\sum_{i=1}^{m} (Zji - Zj.)\,(Zki - Zk.)}{\sqrt{\sum_{i=1}^{m}(Zji - Zj.)^2 \sum_{i=1}^{m} (Zki - Zk.)^2}}$$

where Zji is the (j,i)th element of the transformed data matrix Zj. is the mean of the m attributes for the jth entity. Zki and Zk. are analogously defined.

The element Cjk of the proportional similarity matrix is given by

$$Cjk = \frac{\sum_{i=1}^{m} Zji\ Zki}{\sqrt{\sum_{i=1}^{m} (Zji)^2 \sum_{i=1}^{m} (Zki)^2}}$$

Finally the element Cjk of the Taxonomic distance matrix is defined by

$$Cjk = \sqrt{\frac{\sum_{i=1}^{m} (Zji - Zki)^2}{n}}$$

References

1) SOKAL, R. R. and SNEATH, P.H.A., 1963. Principles of Numerical Taxonomy. W.H. Freeman and Co.

2) HORST, P. 1965. Factor Analysis of Data Matrices. Holt Rinehart and Winston.

3) IMBRIE, J. and VAN ANDEL, T.H. 1964. Vector Analysis of Heavy-Mineral Data. Geol. Soc. Amer. Bull. Volume 75. 1131 - 1155.

Links Used: QMA1

Subroutines Used: DATRD, PRNTB, FMTRD, SBUF, PRNT, FMAT, GMPYX, GDIVX.

Programmer: M. J. Fasham.

N.I.O. Program 198

Title Finite Complex Fast Fourier Transformation

Name FORA (consisting of FORA1, FORA2 and FORA3)

Machine I.B.M. 1800

Language Fortran IV under TSX

Purpose To a) fourier analyse a real or complex set of data
or b) synthesise the coefficients into a real* or complex series
so long as the total number of terms, NTOT, is a power of 2.
(*synthesis into a real series will follow at a later date)

Job Description and Data

```
// JOB          CC 19 X
// *(Project No/Name/Title)
// FOR DATR   User written if required - see Notes.
   followed by program cards for DATR
// FOR DATW   User written - see Notes.
   followed by program cards for DATW
// DUP
*STOREDATAD    CC 19  1 RESLT        CC 30  XXX   Results file
*STOREDATAD      1 FILEN             XXX   Input file to be
                 set up if data is not already in a file,
                 i.e. to be read in by DATR
XXX is a number   ISECT (= no. of sectors of data)
*STORECI        Ø DATR DATR
*FILES (I,RESLT,1), (J,FILEN,1)

*CCEND
*STORECI        Ø DATW DATW  Core load builds DATW and
                             stores on drive 1.
*FILES (I,RESLT,1), (J,FILEN,1)
                      where I and J are logical file nos. in
                      DATR and DATW.
```

// XEQ DATR FX

or

// XEQ FORA1 FX if DATR is required.

Insert the data cards here if they are to be read in DATR, followed by one card with the title of the set of data punched anywhere except in col.1.
Then insert one card containing the following information:-

a) P; determines the total number of numbers on the following data cards so that there are 2^P numbers in the whole series*

b) IFS; set to 2 if fourier synthesis is required.
set to -2 if fourier analysis is required.

c) IFLAG; set to Ø for transform of complex data.
set to 1 for transform of real data.

d) FORAD; the name of the input data file (FILEN in 'Job Descr. and Data') where data must be stored for use with FORA.

e) NAMEF; the name of the output or results file. (RESLT in 'Job Descr. and Data').

f) IDRIV; logical drive number of the disk on which fixed file is situated. (i.e. 1 for the present).

Format: 3I4, 2X, 6A2, I6

(* If IFLAG = 1 then these are 2^P real numbers. If IFLAG = Ø then these are 2^{P-1} complex parts with real and imaginary parts stored in adjacent fields as follows:-

e.g. Ø.5 + Ø.4.i will be stored with Ø.5 followed by Ø.4 in the input data file.

After this data card the following * DELETE cards should be included (to avoid confusion when FORA is used again):-

```
// DUP
* DELETE  CC  21 DATR
* DELETE      DATW
* DELETE  CC  11 RESLT  if no longer required,
                       i.e. if DATW has written results
                       elsewhere or done all work required.
```


Summary

In order that FORA programs may be executed, the following files and programs must be made available:-

1) A named input file to be set up on disk drive 1 (if one does not already exist).
2) A named results file to be set up on disk drive 1.
3) A program (DATR) - user supplied - to read the data and write it to the input file (see d) under Job Descr. and Data) only if data is to be read from cards.
4) A program (DATW) - user supplied - to read data from the results file (see e) under Job Descr. and Data) and manipulate it according to the user's requirements.

Method

The algorithm used for dealing with a large amount of data is that the data is divided into M blocks, each block containing the amount that FORT can handle at one time. Then each block is written to the results file (temporarily) in a position governed by the reverse binary of I, where I = $\emptyset$, 1.....M - 1. This is called a Binary Sort. In FORA2, each block of 512 real data (or less) is transformed using the FORT subroutine and returned to its original position in the file.

FORA3 then combines these separate coefficients produced by FORT, to form the fourier transform for the whole set of data.

The basis for this algorithm is an iterative application of the method 'Doubling the capacity of FORT', described in the write-up of the FORT subroutine.

FORA3 also writes the combined coefficients back to disk (into the results file) ready for manipulation in DATW.

Restrictions

Since there is a restriction on the amount of data a disk may hold and also the results file has to be the same size as the input data file, the maximum amount of data which can be handled at present (until a 3rd. disk drive is acquired) is 36768 = 2^{15} real numbers. This is included in 575 records of 64 real numbers. The original data file and the results file have to be situated on disk drive 1.

Notes and Examples

It was intended that this suite of programs be made as general as possible, i.e. to read and print out data from and to any device and print out any extra results such as phase and the square root of the power spectrum, but such generalisations are impossible since every user would have different requirements.

Therefore it is suggested that, to read and print data, the user write his own programs to input and output data using a particular device. These programs would in fact have to be linked 'core-load' programs; the input program to be attached to the beginning of FORA1 and the output program to be attached to the end of FORA3. If however, the user's data is already stored on disk, it is unneccessary to have an input program since the fixed data file set up in FORA1 may be modified.

Below are examples of the input and output programs to be supplied. Information on how to store the programs in core-image form and also how to set up a 'results' file is given in the paragraph 'Job Description and Data'.

After fourier transformation of a complex set of data the results file will contain the real and imaginary parts (in adjacent positions) of the N complex coefficients:-

$$A(k) = \frac{1}{N} \sum_{j=0}^{N-1} X(j)\, W^{-jk} \quad \text{where } N = NTOT/2$$

in the case of fourier analysis, or

$$X(j) = \sum_{k=0}^{N-1} A(k)\, W^{jk} \quad \text{where } N = NTOT$$

in the case of fourier synthesis.

For the analysis of a real set of data the results file will contain the coefficients of the trigonometric series:-

$$W(j) = \frac{1}{2} a_{\emptyset} + \sum_{k=1}^{N-1} \left(a_k \cos \frac{2\pi jk}{2N} + b_k \sin \frac{2\pi jk}{2N}\right) + \frac{1}{2}(-1)^j a_N$$

with $a_{\emptyset}$ and a_N first, in adjacent positions followed by $a_1, b_1, a_2, b_2 \ldots\ldots\ldots$ up to a_{N-1}, b_{N-1}

Thus one is free to manipulate in DATW, any of this data to produce the required results.

Examples

a) // FOR DATR This name must be used.

```
 * (Fortran control cards)
       INTEGER* P,Q
       DIMENSION X(64)
       COMMON* P,Q,M,N,NTOT,IFS,IFLAG,NAMEF(6),ISECT,IDRIV
       DEFINE FILE 15(64,192,U,IREC)
```

This means that on file 15, there is room for a maximum of $\frac{64 \times 192}{3}$ = 4096 real terms or 2048 complex terms.

```
       IREC = 1
       DO 10 I = 1, 64
       READ (2,200)   X
  200  FORMAT (8I5)
```

Reads 64 records of 64 real numbers or 32 complex pairs from cards.

```
   10  WRITE(15'IREC) X
```

Writes total array to the input file 15

```
       CALL LINK (FORA1)
       END
```

Attaches DATR to FORA1

b) // FOR DATW This name must be used.

```
 *(Fortran Control Cards)
   INTEGER*  P,Q
   DIMENSION  X(64)
   COMMON* P,Q,M,N,NTOT,IFS,IFLAG,NAMEF(6)ISECT,IDRIV
   DEFINE FILE  15(64,192,U,IREC),20(64,192,U,NREC)
```

File 15 must be defined even though not used here.

```
   NREC = 1
   DO  10    J = 1 , 64
   READ (20'NREC) X
   WRITE(3,100) X
```

Reads results from results file 20. and prints them out.

```
100         FORMAT (1X,4(F10.6,2X,F10.6,4X))
10          CONTINUE
            WRITE (3,200)
200         FORMAT (1X,'END OF DATA')
            CALL EXIT
            END
```

(* these statements must exist as shown, to match up with those in FORA)

Execution Time — This is approximately $2.5 \times 10^{-4} \times NTOT \times LOG_2(NTOT)$ minutes, excluding input and output times.

Programmers — J.Crease and C. Clayson.

N.I.O.SUBPROGRAM -30 Classification M

Title	ARCSINE
Name	Function ARSIN(x)
Language	1800 Fortran IV
Machine	IBM 1800
Purpose	To calculate the angle whose sine is x
Inputs	The argument x
Output	The function returns a real result accurate to 9 decimal digits. e.g. To compute $\sin^{-1}A$ and put result in x:- . . . . x = ARSIN(A)
Programmer	William Strudwick

N.I.O.SUBPROGRAM -38 Classification M

Title	ARCCOSINE
Name	Function ARCOS(x)
Language	1800 Fortran IV
Machine	IBM 1800
Purpose	To calculate the angle whose cosine is x
Inputs	The argument x
Output	The function returns a real result accurate to 9 decimal digits. e.g. To compute $\cos^{-1}$A and put result in x :- . . . . x = ARCOS(A)
Programmer	William Strudwick

N.I.O.SUBPROGRAM -39 CLASSIFICATION M

Title	TANGENT
Name	Function TAN(x)
Language	1800 Fortran IV
Machine	IBM 1800
Purpose	To calculate the tangent of an angle
Inputs	The argument x in radians
Output	The function returns a real result accurate to 9 decimal digits. e.g. To compute TAN A and put result in Y:- Y = TAN(A)
Restrictions	No test for A = $\left\| \pi/_2 \right\|$ radians is made Results will be unpredictable.
Programmer	William Strudwick

N.I.O.Program 40 Classification M

Title	Hyperbolic Cosine
Name	Function COSH(x)
Language	1800 Fortran IV
Machine	IBM 1800
Purpose	To calculate the hyperbolic cosine value of x
Inputs	The argument x in radians
Output	The function returns a real result accurate to 9 decimal digits. e.g. To compute COSH x and put result in y:- y = COSH(x)
Programmer	William Strudwick

N.I.O.SUBPROGRAM -41

Title — Hyperbolic sine

Name — Function SINH(x)

Language — 1800 Fortran IV

Machine — IBM 1800

Purpose — To calculate the hyperbolic sine value of x

Inputs — The argument x in radians

Output — The function returns a real result accurate to 9 decimal digits.

e.g. To compute SINH x and put result in y:-

```
      .
      .
      .
      .
y =   SINH(x)
```

Programmer — William Strudwick.

N.I.O. SUBPROGRAM -59

Title	Double Integer Add and Subtract
Name	IDADD, IDSUB
Machine	IBM 1800
Language	1800 Assembler
Purpose	To add or subtract two double length integers and place the result in a double length integer.
Method	The subroutines are called by a fortran CALL IDADD (I(1),J(1),K(1)) or CALL IDSUB (I(1),J(1),K(1)) The arguments are the locations of the most significant part of the double length integers e.g. A HEX number 75FFFA86 would appear as hex 75FF in I(1) and hex FA86 in I(2). The overflow indicator is turned on if the answer lies outside the range -2,147,483,648 to 2,147,483,647. The carry indicator is turned on if a carry condition has occured.
Execution Time	Approx. 0·8 milliseconds.
Programmer	R. Bromley.

N.I.O. SUBPROGRAM -60

Title	Double Integer Multiply
Name	IDMUL
Machine	IBM 1800
Language	1800 Assembler
Purpose	To multiply together two double length integers to produce a quadruple length product.
Method	The subroutine is called by a fortran call statement. CALL IDMUL (I(1),J(1),K(1)) This multiplies the double integer in I(1), I(2) by the double integer in J(1),J(2) and places the product in K(1),K(2),K(3),K(4). K(1) contains the most significant bits of the product and K(4) the least significant bits. IDDIV enters the routine by the entry point called IDMLX. This entry point does not call QZSAV on entry on QZEXT on exiting and references work level area constants only. The overflow and carry indicators are not changed.
Execution Time	Approx. 1·8 milliseconds
Programmer	R. Bromley

N.I.O.SUBPROGRAM-66

Title	Reverse binary
Name	Function INVT(IVAL,Q)
Machine	I.B.M.1800
Language	1800 Assembler
Purpose	To reverse the Q low-order binary digits of a 16-bit word
Parameters	Two parameters are required by the sub-routine namely:- IVAL the word containing the integer. Q the number of bits to be reversed starting from the right hand end of the word.
Output	INVT returns the answer to the accumulator and therefore may be used as a function sub-routine in a Fortran program e.g. J=INVT(4,3) will give J the value 1 but J=INVT(4,6) will give J the value 8
Restrictions	$Q \leq 16$
Programmer	Cathy Clayson