United States

Department of the Interior

Geological Survey

MAGIC

Computer Programs for Paleontologists available on MULTICS

by

Robert A. Spicer, Richard J. Cullip, and Richard Z. Poore

Open-File Report 80-317

This report is preliminary
and has not been reviewed for
conformity with Geological
Survey standards and
nomenclature.

March 14, 1980

Abstract

MAGIC is a self-contained MULTICS "subsystem" consisting of computer programs that have been found useful in a paleontological context. MAGIC contains both simple and sophisticated statistical programs that can be used by a casual computer user with a minimum of instruction. Once in MAGIC information about the available programs and assistance in running them can be obtained.

Introduction

The concept of the MAGIC "subsystem" was developed when it was realized that many people, paleontologists in particular, could make use of MULTICS but either could not afford the time or did not have the wish to become involved in programming. A set of commonly used programs were needed that would require a minimum of computing expertise to use and would accept data in a common format. It was considered desirable that the user need only know how to use a typewriter and that the computer operations be as transparent as possible.

In view of the fact that program requirements were likely to change with time a system had to be provided that made every program compatible with the others but independent in order to facilitate program changes. MAGIC is the result of cooperation between paleontologists of differing disciplines responding to a common need.

Acknowledgments

Thanks are due to the following people who either assisted in assembling MAGIC or acted as "guinea pigs": Robyn Burnham, John Barron, and Sean Murphy-Stone.

Description of the system

MAGIC consists of a series of executive command (ec) segments, program segments, and information segments. A primary ec, magic.ec, is entered when the user types udd Trinity RSpicer magic_dir magic.ec. This primary ec in turn can call, depending on the user's wishes, segments containing information about the MAGIC system or further ec's that run the program segments.

If the user requests information or help in entering data this is provided at the user's terminal after which the option to call any program is presented.

Access to the programs is made through program ec segments which call, and, where necessary, compile, programs. The ec segments also make input and output segments available to the program. In most cases the names of both the input and output segments can be specified by the user.

After completion of a program run, control returns to the primary magic ec from which additional program runs can be performed.

Because MAGIC is constructed of a series of ec segments all program control and data handling can be undertaken outside of MAGIC if so desired. Additional programs may be included in the system, or programs removed, without affecting any other part of MAGIC. The independent nature of the programs also results in reduced central memory requirements.

Input data segments can be created by using any of the MULTICS text editors. Instructions for using QEDX in this context are given in the MAGIC handbook and help segments.

Input/output formats

All programs read input data in stream (v) format and as such the accuracy of the input data is directly controlled by the user.

The construction of input data segments has been standardized as much as possible. For example, most data segments will have a name or title as the first line, the dimensions of the input data matrix (variables, samples) as the second line and the following lines will be the actual data--each line or row representing the variables measured in one sample. There are, however, exceptions and one should always consult the magic handbook before trying a new program.

The output format varies from program to program and is appropriate to the reliability of the results yielded by the various statistical processes. The program WILLI, which allows the user to transform and edit data matrices prior to using them as input for other programs, produces manipulated data segments in the format f10.4 (5 integer and 4 decimal places) for inspection at the terminal, but f20.8 (11 integer and 8 decimal places) as input to subsequent programs.

Program application

Some of the programs in MAGIC should only be used under specific circumstances when data have been collected in a suitable manner. The programs will run on any data that conform to the formats necessary for input, but unless the user is aware of the way the data were collected, what the program is doing to the data, and how the results should be interpreted, the exercise is not only of little value but may also be dangerously misleading. While every effort has been made to ensure that the programs are as "bug-free" as possible (an ongoing process) the authors do not accept responsibility for any errors that may occur.

It is not appropriate that details of the methods of operation and applications of the MAGIC component programs be presented here. Brief descriptions of the programs accompany the instructions for their use in the MAGIC handbook but detailed explanations are available in the following works:

Davis, J. C., 1973, Statistics and data analysis in geology: New York, John Wiley and Sons, 550 p.

Hill, M. O., 1973, Reciprocal averaging: an eigenvector method of ordination: Journal of Ecology, v. 61, p. 237-249.

_____1974, Correspondence analysis: a neglected multivariate method: Applied Statistics, v. 23, p. 340-350.

Jöreskog, K. G., Klovan, J. E., and Reyment, R. A., 1976, Geological factor analysis: Amsterdam, Elsevier, 178 p.

The MAGIC Handbook

MAGIC

MAGIC is a package of computer programs that others have found useful when dealing with paleontologic data. MAGIC is designed so that the casual user can sit down at a terminal and, with a minimum of instruction, be able to use a variety of simple and sophisticated statistical techniques.

New users of Multics are cautioned that the MAGIC handbook assumes that the user has at least rudimentary knowledge of the Multics system. Information on the Multics system may be found in the U.S.G.S. Introductory Multics Handbook which may be obtained at the Reston, Denver, or Menlo Park Computer Center Division offices. For further information, each user is encouraged to obtain and use the Honeywell Multics Programmers' Manuals. The more useful of those manuals are:

Multics Introductory User's Guide	(AL 40)
MPM Commands and Active Functions	(AG 92)
Multics Pocket Guide	(AW 17)

Instructions given in this handbook will allow you to login on the Multics computer, create an area in the computer to store your data, edit and transform your data, and then compute various statistics.

At the present time (September 28, 1979) the following programs are available on MAGIC.

BICOR - calculates covariance and correlation between two sets of variables.

SMOOTH - performs N term smoothing (running average).

WILLI - checks for rows and columns of zeros in a data set and allows various transformations and editing of data segments.

BICLUST - weighted average cluster analysis for binary or semiquantitative data.

CORRAN - correspondence analysis.

PCA - principal components analysis.

CLUSTER - weighted pair group cluster analysis.

TREND - trend surface analysis.

As other programs are added to MAGIC this handbook will be updated. Information on programs in MAGIC can also be called through MAGIC.

OPERATION OF MAGIC

In the following explanation user-generated sequences are preceded by an !, computer-generated sequences are in quotes.

Once you login to the multics system (see next section), MAGIC can be accessed by typing

```
! ec  >udd >Trinity > RSpicer > magic_dir > magic.ec
```

If you intend to use MAGIC on a regular basis you can create a permanent link to MAGIC by typing in

```
! lk  >udd >Trinity > RSpicer > magic_dir > magic.ec magic.ec
```

Once you have linked to MAGIC you can invoke it forever after by typing:

```
! ec magic
```

When you invoke MAGIC you will receive a greeting message.

```
     "MAGIC-USGS multivariate statistical Programs Package 1.0"
"............................................................"
```

The 1.0 refers to the release or version of MAGIC. As alterations are made to the package this number will change.

"Do you require help"

```
     If you need help type   ! yes
     If not           type   ! no
```

For now, assume you do not need help[1] and that you typed in ! no. You will then be given an opportunity to quit MAGIC.

"Do you want to quit MAGIC"

```
     You answer this by typing    ! yes
                   or             ! no
```

If yes, you exit from MAGIC; if no, the routine continues with
"enter program name"

At this point you are ready to type in the name of the program you want to use. NOTE: Read the appropriate writeup in this handbook before trying to use a program.

After your results are printed out, the routine will ask you
"Do you want to quit MAGIC"
AGAIN you answer by typing ! yes
or ! no
If yes, you exit from MAGIC, if no the routine continues with
"enter program name"
and so on and so on--but remember--always quit magic before you logout.

[1] NOW, suppose you responded YES to
"do you require help" you will then be asked
"Do you require a list and description of programs"
AGAIN you answer by typing ! yes
or ! no

If yes, a listing and brief description of programs contained in MAGIC are printed out, then the routine continues with another question. If no, you immediately get the same question which is
"Do you require help inputting data?"
AGAIN answer by typing ! yes
or ! no

If YES, a short information segment concerning input of data will be printed out and then you return to

"Do you wish to quit magic"

If NO, you return immediately return to

"Do you wish to quit magic"

Answer ! yes

or ! no

Good Luck.

LOGIN/LOGOUT

MULTICS is usually accessed with a terminal and a phone line. Multics keeps track of users by a system of Project and Person identifications. To log into multics you must have use of a Person-id registered on the system and know what the Password is for that Person-id. More information on new-user registration may be found in the appropriate Introductory Guide (National Center, Denver, Menlo Park) to Computer Center Division facilities.

After obtaining use of a Person-id, password, terminal, and phone, you are ready to log in.

Most terminals have switches labeled speed and duplex (next to the keyboard). You want the speed switch set to 30 and the duplex switch set to half.

Dial the telephone number of the computer. You will hear the phone ring, and when the computer answers, a steady high-pitched tone will come on. Place the phone into the "coupler" of the terminal and turn the terminal power switch on. Shortly the terminal will type out the two lines identifying itself and telling you the number of current users.

Now you log into the system by typing the command ! login Person-id. The computer will then ask for the Password and blacken in the portion of a line where you type in your password. Now type in password. If done correctly the computer will then provide information about when you last logged in, print any system messages, etc., and finally print out a ready message. When the ready message is printed, you are at command level and ready to enter MAGIC or the editor.

COMMAND sequences for login follow. Computer-generated sequences are in quotes, user-generated sequences are preceded by an !. Do not type the !. This is only to show which lines you type.

```
"multics  mr6.20:  Menlo Park, California"
"Load = 22.0 out of 85.0 units:  users = 22"
! login RPoore
"Password:"
!"*********" (computer first darkens line, then you type password over
it)
"you are protected .........
  ........................."
"R  843  2.12.8  32.853  3Ø2"   This is the ready message.
```

To terminate a session all you need to do is type in the command

```
! logout
```

The computer will then print out a two-line message concerning when you've logged out and usage during the session. Next the computer prints "hangup". You should then disconnect the phone from the terminal, hangup the phone and turn the terminal off.

NOTE:

1. You can substitute l for login so that command is

   ```
   ! l RPoore
   ```

2. Some Person-id's operate a start-up ec. Thus you may obtain a fair amount of printout after you give the correct password. Just wait for the ready message.

3. On the teleterm 1132 terminal, the 1 sometimes generates a comma. If that happens you'll have to give the command a second time.

4. Correcting mistakes: when you type commands, etc., on the terminal they do not register in the system until you provide a newline character (carriage return and line feed). There are two special characters (# and @) that allow you to erase mistakes on a line if you notice them before giving a newline character.

   # - this character goes back one space and erases whatever character is there, or any number of preceding blank spaces, and you can type a new character (correction) in that space. Suppose you mean to type the word DOG, but type DAG by mistake and notice the mistake before going to the next line. You could type # twice and then OG. ! DAG##OG (newline). Multics will only see DOG.

   @ - this character erases all characters preceding it on a line. For example ! See teh dag run @See the dog run (newline), multics will only see "See the dog run".

5. Upper and lower case letters are important in multics in that the system treats them as entirely different characters. In general uppercase letters are used for first two letters of Person-id, first character of Project-id, and in naming data segments. Commands, responses to prompting questions, etc., are lower case.

6. Data segments: In most cases data that you want the programs to operate on will be contained in data segments that you create. It is best to use a short name that incorporates the word data for these data segments. For example, a convenient name for a data segment containing counts of planktic forams in samples from DSDPHole 173 would be something like - PFdata_173. A label containing the word data allows you and, more importantly, others who may work in the same area to identify segments containing original data. Note names of data segments cannot contain blanks. If you want words separated use the underscore symbol (_).

7. The maximum line-length of your terminal should be set to 132 characters, by using the multics command line_length 132. If a 132-column terminal is not available, the output from most programs within MAGIC can be routed to a line printer in the computer center. If available, the option to perform this re-routing is offered near the beginning of each program.

QEDX - MULTICS EDITOR

The QEDX text editor is used to create and edit data segments. Detailed explanation of QEDX is available in section 3 of the Multics Programmers Manual of Commands and Active Functions, and in section 7 of the U.S.G.S. Introductory Multics Handbook. The following is a brief description of how QEDX operates and comments concerning some of the commonly used requests.

QEDX creates a temporary space in the Multics system (called a buffer) in which you can create a new data segment or modify an existing data segment. QEDX operates in two basic modes - (1) input and (2) edit. When in the input mode, data can be added to the buffer. When in the edit mode, data in the buffer can be altered. When in edit mode, and often when in input mode, you will want operations performed on a specific line or group of lines of the buffer. You can specify (address) a line (or lines) in the buffer in a number of ways.

QEDX INPUT MODE REQUESTS

a = enter input mode, append lines typed from terminal after a specified line.

c = enter input mode, replace the specified line or lines with lines typed from the terminal

i = enter input mode, insert lines typed from the terminal before a specified line.

NOTE: once you type in one of the above commands you must type \f to escape input mode. Also you must be in edit mode before giving requests above. And when you invoke QEDX you are in edit mode.

QEDX EDIT MODE REQUESTS

d = delete specified line or lines from buffer

p = print specified line or lines on the terminal

q = exit from the editor

r = read a specified segment into the buffer

w = write current buffer into specified segment

ADDRESSING

Lines in the buffer are automatically numbered (consecutively) from top to bottom by QEDX. Thus if you type in 3 lines, the first line is #1, the second #2, and the third #3, and an imaginary pointer is aimed at line #3. The imaginary pointer is usually aimed at the line that was operated on last and that line is called the current line. If you then insert a line before line #3, the newly inserted line becomes line #3 and the previous line #3 becomes line #4--and the pointer is aimed at new line #3.

For data segments, two useful ways of addressing a specific line or set of lines in a buffer are (1) by absolute line number and (2) by relative line number. Absolute line number refers to the position of a line in terms of the sequence of lines in the complete buffer, whereas relative line number refers to the position of a line relative to the current line (the pointer).

Consider the buffer shown below, with the pointer aimed at line 3.

```
This is line one
This is line two
This is line three
This is line four
```

The absolute address of (This is line two) is 2.

The relative address of (This is line two) is -1.

The relative address of (This is line four) is +1.

Buffer addresses in QEDX precede requests for operations. Addresses can be simple or compound. If you do not specify an address, QEDX automatically assumes you mean the operation is to be performed on the current line.

The following illustrates the operation of QEDX. Computer-generated sequences are in quotes, user-generated sequences are preceded by an !.

You must invoke QEDX by typing the command QX. When invoked QEDX is in edit mode.

```
! qx                      invokes QEDX - it is in edit mode
! a                       enters input mode, and creates new buffer
! This is line one        line one of buffer
! This is line two        line two of buffer
! This is line three      line three of buffer
! This is line four       line four of buffer
! \f                      escape from input mode, now in edit mode
! w Dummy_seg             Writes 4 lines of buffer off to a permanent
                             segment called Dummy_seg
! q                       exit from editor - you are now at command level
"ready message"              and buffer and its contents are destroyed
```

```
! qx                    invokes QEDX - it is in edit mode
! r Dummy_seg           takes contents of Dummy_seg and places them in
                           buffer
! 1 p                   print line 1 (addressing by absolute line #)
"This is line one"      line 1 of the buffer
! +3 p                  print 3rd line down from current line (addressing
                           by relative line #)
"This is line four"     line 4 of the buffer
! d                     delete (since no address given, default is
                           current line [in this case pointer moves to
                           line 3 as line deleted was last line]
! a                     switch to input mode
! This is line 4        entered after current line (becomes new line
                           #4)
! \f                    escape from input mode
! p                     print (current line printed because no address
                           given)
"This is line 4"        line four of buffer
! w Dummy_seg           writes contents of buffer in Dummy_seg - this
                           is an overwrite - it replaces the data that
                           was in Dummy_seg
! q                     exit from editor and return to command level
"ready message"
```

To print, delete, or writeoff a sequence of lines in a buffer you use a compound address to designate the first and last line of the desired sequence.

For example:

! 3,7 d — this deletes the third through the seventh lines of the buffer. The comma separates the two addresses.

! -1,+2 p — this would print the line preceding through the 2nd line following the current line.

NOTE: 1. you must type \f to escape from input mode.

2. you must write the contents of a buffer off to a segment before exiting from the editor. You should specify the segment that the data are going to reside in when you give the write request. When creating a substantial data segment, it is prudent to escape input mode and issue a write request at regular intervals. That way if QEDX is accidently terminated you will only lose the data typed in since the last write request.

3. The name of a data segment cannot contain a blank. If you want to separate words, numbers, whatever, use the underscore character (_).

4. Time and thought put into construction of a data segment is well spent. Any mistake in the data segment will eventually cause trouble in MAGIC.

BICOR (Source - Davis, 1973)

BICOR will calculate the covariance and correlation between two variables. Input for BICOR are pairs of values for observations that you type in during operation of the program. Output from BICOR consists of the following:

1. Number of samples (pairs of observations)
2. Sum of variable 1 (X1)
3. Sum of squares X1
4. Sum of variable 2 (X2)
5. Sum of squares X2
6. Sum of cross products
7. Mean of X1
8. Variance of X1
9. Standard deviation X1
10. Mean of X2
11. Variance of X2
12. Standard deviation X2
13. Covariance between X1 and X2
14. Correlation between X1 and X2.

OPERATION:

BICOR is accessed through MAGIC by typing in bicor in answer to the prompting question - enter program name. You will then be asked to type in the total number of pairs of observations you have (= # of samples). Next you will be asked to enter the pairs of observations and the terminal will also type a ?. You then type in values for the first pair of observations separated by a comma. One pair of variable values should be inputted per line. When you've typed in the first pair, the routine will prompt you with another ? on the next line and wait for you to enter the next pair. The program will continue to prompt you until you enter the total number of pairs of observations you specified previously.

Command sequences for operating BICOR follow. Computer-generated sequences are in quotes, user-generated sequences are preceded by an !.

```
! magic
  etc.
"enter program name"
! bicor
"enter number of pairs of observations"
! 2
"enter pairs of observations"
"?"
! 1,2
"?"
! 3,4
"1        number of samples = 2"
     "                          "
```

SMOOTH (Source = Davis, 1973)

SMOOTH will perform a M-term smoothing operation (running average) on a data sequence. Input for SMOOTH is in the form of a data segment. Output from SMOOTH consists of:

1. list of original values
2. list of smoothed values
3. plot of original values
4. plot of smoothed values

OPERATION:

SMOOTH is accessed through MAGIC by typing in "smooth" in answer to the prompting question - enter program name. You will then be asked to type in the name of your data segment. Next you will be asked to provide a control argument selecting the number of terms to be used in the smoothing operation. You may select any odd number. A 3-term running average is commonly used.

Command sequences for operating SMOOTH follow. Computer-generated sequences are in quotes, user-generated sequences are preceded by an !.

```
! magic
  etc
"enter program name"
! smooth
"enter name of data segment"  ! coarse_fraction_core_8
"enter number of terms to be used in the running average [3 is the most
common]"
! 3
```

"stop"

"Results ...."

"output"

DATA SEGMENT FORMAT FOR SMOOTH

A data segment for SMOOTH is created in an editor (QEDX is recommended).

line 1 title of data segment - you can use up to 80 characters (including blanks)

line 2 number of samples (observations). Maximum # is 350.

line 3 The data, each number (sample, observation) separated by a comma (stream input). Do not put spaces between values. When the end of a line is approached, stop typing at a convenient point (after a comma is best) and start typing the rest of the values at the beginning of the next line. NOTE: You must put a newline character (carriage return and line feed) after your

line N+2 last number.

WILLI (Source - R. A. Spicer)

Willi is a program that allows you to transform and edit a data matrix, and then use the new data matrix as input for statistical programs. Input for WILLI is an M,N data segment. Output from WILLI can include the following depending upon options you select.

1. raw data matrix
2. transformed data matrix
3. edited data matrix

NOTE: WILLI automatically checks the data matrix for columns and rows of zeros and then deletes them. This is a safety feature as a column or row of zeros will cause most of the statistical programs to "Blow up".

Data transformation options include:

(1) = raw data - the contents of data matrix are unchanged but data are checked for columns and rows of zeros.

(2) = percent species transform - raw species counts are converted to percent sample.

(3) = log transform.

(4) = presence/absence (binary)

Edit options include ability to delete up to 50 samples and/or species from the data matrix.

OPERATION

WILLI is accessed through MAGIC by typing WILLI in answer to the prompting question - enter program name. You will then be asked to type in the name of the data segment containing the data to be operated on. Next you will be asked to type in the name of the data segment the results from WILLI are to be written out to.

You next must respond to several control options.

First you must decide if you want a printout of the raw data.

```
type 1 for printout
     2 for no printout
```

Next you respond to transform options

```
type 1 for raw data
     2 percent species transform
     3 log transform
     4 presence/absence transform
```

The transformed data matrix will then be printed out for your inspection. If any rows or columns were all zeros you will receive a warning and these rows or columns will not appear on the transformed matrix.

You are then asked if you want to delete any species. To delete species merely type in the original species # followed by a newline character (carriage return and line feed). You may delete up to 50 species. When you want to terminate the list type a zero (Ø).

For example:

```
"enter species to be omitted"
! 1
! 3
! 7
! Ø
```

You will then be asked which samples you want deleted. Sample deletion is carried out as for species.

For example:

```
"enter samples to be omitted"
! 5
! 7
! Ø
```

If you do not want to delete any species or samples answer Ø (zero) to the appropriate question.

The program will then print out your data matrix with the deletions you requested. If no deletions were required no further printout is given.

COMMAND sequences for operating WILLI follow. Computer-generated sequences are in quotes, user-generated sequences are preceded by an !.

```
! magic
   "etc"
"enter program name"
! willi
"enter name of data segment"  ! test_x
```

```
"enter name of output segment"  ! test_y
"enter 1 for printout of raw data else type Ø"
! Ø
"enter 1 for raw data"
      "2 for percent species transform"
      "3 for log transform"
      "4 for presence/absence"
! 3
" sample                           species   "
                     1         2         3         4          etc.
     1             2.70      2.30      3.13       0.6
                                etc"
"enter species to be omitted"
! Ø
"enter samples to be omitted"
! 1
! 2
! Ø
" sample                                species   "
                     1         2         3         4          etc.
                                etc."
```

In the above sample log transform was requested, no rows or columns of zeros were in data segment and no species and the first two samples were deleted.

DATA SEGMENT FORMAT

Data segment for WILLI is created in an editor (QEDX is recommended).

line 1 - title of data segment - you can use up to 80 characters (including blanks)

line 2 - dimensions of data matrix M,N where M = # of variables (species)
N = # of samples.

line 3 - the data - data for each sample are entered in rows, each number separated by a comma (stream input). If a row of data extends over a line, continue typing on the next line. NOTE: You must put a newline character (carriage return and line feed) after your last number of your last sample.

BICLUST (Sources - Davis, 1973)

Biclust will perform a weighted-average cluster analysis on binary (presence-absence) data or on semiquantitative (abundant-common-rare-absent) data. The program allows the option to use either the Sokal-Michener or the Jaccard binary coefficient as the measure of similarity between samples. Input for BICLUST is in the form of a data segment containing an m,n data segment. Output from BICLUST can include all or part of the following depending upon options you select:

1. input data matrix
2. similarity matrix
3. linkage table
4. dendrogram

OPERATION:

BICLUST is accessed through MAGIC by typing in biclust in answer to the prompting question - enter program name. You will then be asked to type in the name of the data segment containing the data to be clustered.

Next you must provide 4 control arguments. You do this by responding to 4 multiple choice statements.

The first control argument concerns the similarity measurement option.

Ø = Sokal-Michener coefficient

1 = Jaccard coefficient

The second control argument concerns the type of data input.

Ø = binary (presence-absence) data

1 = semiquantitative data

The third control argument allows you to decide if you want the input data matrix printed.

Ø = for printing the data matrix

1 = for not printing the data matrix

The fourth control argument allows you to decide if you want the similarity matrix printed or not.

Ø = for printing similarity matrix

1 = for not printing similarity matrix

Command sequences for operating BICLUST follow. Computer-generated sequences are in quotes, user-generated sequences are preceded by an !.

```
!magic
     etc.
"enter program name"  ! biclust
enter name of data segment  ! plut_coded_data
          "Cluster analysis"
          "This program ...........
          ........................."
"enter Ø for Sokal-Michener coefficient
       1 for Jaccard coefficient"
! Ø
"enter Ø for binary (presence-absence) data
       1 for semiquantitative data"
! 1
"enter Ø if you desire the input data matrix printed
       1 if you don't"
```

```
! Ø
"enter Ø if you desire the similarity matrix printed
       1 if you don't"
! Ø
"stop"
     "Results ........."
```

In the preceding example, samples will be clustered using the Sokal-Michener binary coefficient as the measure of similarity. The variables (species) would be measured semiquantitatively (abundant-common-rare-absent). Both the input data matrix and the similarity matrix will be printed out.

DATA SEGMENT FORMAT FOR BICLUST

A data segment for BICLUST is created in an editor (QEDX is recommended).

line 1 Title of data segment--you can use up to 80 characters (including blanks).

line 2 dimensions of data matrix M, N, where M = number of (species)
variables
N = number of samples

line 3 The data - data for each sample are entered in rows, each number separated by a comma (stream input). If a row of data extends over a line, continue typing on the next line. NOTE: You must put a newline character (carriage return) after the last number of your last sample.

STYLE OF DATA

This program allows two styles of data. In the first style, the variables are coded in a binary state. In this state a variable is either present or absent. The following numbers must be used to indicate presence or absence of variables in a given sample:

2 = present

1 = absent

In the second style, the variables are coded in a four-state manner. The four states are abundant, common, rare, absent.

The following numbers must be used to indicate which category the variables in a given sample fall into:

222 = abundant

221 = common

211 = rare

111 = absent

CORRAN (Source R. Spicer)

CORRAN will perform a correspondence analysis of up to 350 samples and 350 variables. Input for CORRAN is in the form of a data segment containing an M,N data matrix. The input data segment is usually output from Willi. Output from CORRAN includes

1. Position of samples along 3 AXES.
2. Position of species along 3 AXES.
3. Plot of samples in 3-dimensional space (the 3 AXES).
4. Plot of species in 3-dimensional space (the 3 AXES).

NOTE: 3 and 4 are drawn on the digital plotter at the Computer Center.

OPERATION:

CORRAN is accessed through MAGIC by typing in CORRAN in ANSWER to the prompting question -- enter program name. You will then be asked to type in the name of the data segment containing the data you want CORRAN to operate on.

There are no control arguments or options to CORRAN.

COMMAND sequences for operating CORRAN follow. Computer generated sequences are in quotes, user generated sequences are preceded by an !.

```
! magic
   ! "etc."
"Enter Program Name"
! corran
"Enter name of data segment"  ! Core_3_Diatoms
"Correspondence analysis ......"
     "results"
```

"a series of information statements are output documenting the copying of plotter instructions generated by the program to magnetic tape."

DATA SEGMENT FORMAT FOR CORRAN

A data segment for CORRAN is created in an editor (QEDX is recommended).

line 1 title of data segment--you can use up to 80 characters (including blanks)

line 2 dimensions of data matrix M,N - where M = # of variables (species)
N = # of samples.

line 3 the data - the data for each sample are entered in rows, each number separated by a comma (stream input). If a row of data extends over a line, continue typing on the next line. NOTE:

line whatever You must put a newline character (carriage return and line feed) after your last number of your last sample.

PCA (Source - Davis, 1973)

PCA will perform a principal components analysis of up to 350 samples and 350 variables (species). Options are available to use a correlation matrix, variance, covariance matrix or cos θ matrix as a measure of similarity. Input for PCA is in the form of a data segment containing an M,N data matrix. Output for PCA can include the following depending upon options you select.

1. Input matrix.
2. Standardized input matrix.
3. Similarity matrix.
4. Table of eigenvalues.
5. Principal axis matrix.
6. Principal component scores.
7. Two dimensional plots of principal component scores.

OPERATION:

PCA is accessed through MAGIC by typing in PCA in answer to the prompting question - enter program name. You will then be asked to type in the name of the data segment containing the data to be operated on.

Next you must provide control arguments. You do this by responding to multiple choice statements.

The first control argument concerns an option for transposing the data matrix.

1 = no transposition.

2 = transpose.

The next control argument allows you to select the similarity coefficient you want.

1 = correlation matrix.

2 = covariance matrix.

3 = cos θ.

If you select 2 or 3 you will be asked if you want your data row-wise normalized.

1 = for row-wise normalization.

0 = no row-wise normalization.

If you select cos θ you will also be asked if you want each column of data to have zero mean.

1 = zero mean transformation.

0 = no zero mean transformation.

Next you will be given the choice of having results dprinted at computer center.

If you answer yes the results are printed out on a line printer at the computer center.

If you answer no to the dprint option, your input data, results, etc., will be printed out on the terminal.

Note the output from PCA can be extensive and take a long time to print out on the terminal. If you have $>$50 samples or species it is best to dprint.

A scatter plot of individuals plotted against axes 1 and 2, 1 and 3, and 2 and 3 are produced as part of the output.

Command sequences for operating PCA follow. Computer-generated sequences are in quotes, user-generated sequences are preceded by an !.

```
! magic
     "etc."
"enter program name"
! pca
"enter name of data segment"  ! core_3_diatoms
     "PCA"
"enter 1 for no transposition of data matrix"
          "2 for transpose of matrix"
! 1
"enter 1 for correlation matrix"
      "2 for covariance matrix"
      "3 for cos theta transform"
! 1
"Do you wish to dprint the results?"
! No.
     "Title of data segment for core_3_diatoms"
     "input data"
     "similarity matrix"
          etc.
     "results"
```

In the above example correlation matrix was used for PCA analysis of diatom data from core 3. The results printed out at the terminal.

DATA SEGMENT FORMAT FOR PCA

A data segment for PCA is created in an editor (QEDX) is recommended).

line 1 title of data segment - you can use up to 80 characters (including blanks)

line 2 dimensions of data matrix M,N - where M = # of variables (species)

N = # of samples

line 3 the data - data for each sample are entered in rows, each number separated by a comma (stream input). If a row of data extends over a line, continue typing on the next line. NOTE: you must put a newline character (carriage return and line feed) after your last number of your last sample.

CLUSTER (Source - Davis, 1973)

Cluster will perform a weighted-average cluster analysis of up to 350 samples with an option to use either the correlation coefficient or the distance coefficient as a measure of similarity. Input for CLUSTER is in the form of a data segment containing an M,N data matrix. Output from CLUSTER can include all or part of the following depending upon options you select.

1. Input matrix.
2. Similarity matrix.
3. Linkage table.
4. Dendrogram.

OPERATION:

CLUSTER is accessed through MAGIC by typing in CLUSTER in answer to the prompting question - enter program name. You will then be asked to type in the name of the data segment containing the data to be clustered.

Next you must provide 4 control arguments. You do this by responding to 4 multiple choice statements.

The first control argument concerns options available for data input.

Ø = end of job - this is an escape path.

1 = input a data matrix (M,N) where M is number of species and N is number of samples. This option will yield a grouping of variables.

2 = input a data matrix (M x N) and transpose it. This option will yield a grouping of samples.

3 = input a similarity matrix (N,N) or (M,M).

The second control argument allows you to select the correlation coefficient or the distance coefficient as the measure of similarity.

1 = correlation coefficient. Values near +1 indicate a high degree of similarity; values near -1 indicate a very low degree of similarity.

2 = distance coefficient. Low values indicate a high degree of similarity; high values indicate a low degree of similarity.

The third control argument governs whether or not the input data matrix is printed out on the terminal.

Ø = input data matrix is printed out.

1 = input data matrix is not printed out.

If the first option is selected, program will automatically print out source of input data.

We suggest that no printout of the input data matrix be made if either the number of samples or number of variables is >50, unless absolutely necessary.

The fourth control argument governs whether or not the similarity matrix is printed out on the terminal.

Ø = similarity matrix is printed out.

1 = similarity matrix is not printed out.

We suggest that no print out of the similarity matrix be made if >50 observations are involved, unless absolutely necessary.

Command sequences for operating CLUSTER follow. Computer-generated sequences are in quotes, user-generated sequences are preceded by an !.

```
! magic
     etc.
```

```
"enter program name"

! cluster

"enter name of data segment"  ! Core_3_Diatoms

     "Cluster analysis - numerical data"

     "This program ...........

     ........................."

"enter Ø for end of job

       1 to input a data matrix

       2 to input a data matrix and transpose it

       3 to input similarity matrix"

! 1

"enter 1 for correlation coefficient

       1 for distance coefficient"

! 1

"enter Ø if you want input data matrix printed

       1 if you don't"

! 1

"enter Ø if you want similarity matrix printed

       1 if you don't"

!1

"stop"

     "Results ................"
```

In the preceding example, variables (species) will be clustered using the correlation coefficient as a measure of similarity. The input data matrix and the similarity matrix will not be printed out.

DATA SEGMENT FORMAT FOR CLUSTER

A data segment for cluster is created in an editor (QEDX is recommended).

line 1 Title of data segment - you can use up to 80 characters (including blanks).

line 2 Dimensions of data matrix M,N - where N = # of samples
M = # of variables.

line 3 The data - data for each sample are entered in rows, each number separated by a comma (stream input). If a row of data extends over a line, continue typing on the next line. Note: you must put a newline character (carriage return and line

line N+2 feed) after your last sample.

TREND SURFACE (Source - Davis, 1973)

Trend will compute a polynomial trend surface of up to four degrees for a data set of up to 350 samples. Input for TREND is a data segment containing a 3,N matrix, where N = number of samples. Output from TREND can include all or part of the following, depending upon options you select.'

1. value, coordinates and deviation for each dependent variable at each observation.
2. trend surface coefficients.
3. Error measures.
4. contour map of trend surface values.
5. contour map of residual values.

OPERATION

TREND is accessed through MAGIC by typing in TREND in answer to the prompting question - enter program name. You will then be asked to type in the name of the data segment containing the observations you want TREND to operate on. You will then be required to provide 6 control arguments. You do this by responding to 6 multiple choice statements.

The first control argument allows you to select the degree or order of the trend surface used. You can chose the first through the fourth degree with:

first degree trend surface

$$Y \quad b_0 + b_1x_1 + b_2x_2$$

second degree trend surface

$$Y = b_0 + b_1x_1 + b_2x_2 + b_3x_1^2 + b_4x_2^2 + b_5x_1x_2$$

third degree trend surface

$$Y = b_o + b_1x_1 + b_2x_2 + b_3x_1^2 + b_4x_2^2 + b_5x_1x_2 + b_6x_1^3 + b_7x_2^3 +$$
$$b_8x_1^2x_2 + b_9x_1x_2^2$$

fourth degree trend surface

$$Y = b_o + b_1x_1 + b_2x_2 + b_3x_1^2 + b_4x_2^2 + b_5x_1x_2 + b_6x_1^3 + b_7x_2^3$$
$$b_8x_1^2x_2 + b_9x_1x_2^2 + b_{10}x_1^4 + b_{11}x_2^4 + b_{12}x_1^2x_2^2 + b_{13}x_1^3x_2$$
$$b_{14}x_1x_2^3$$

Where

Y = computed value of dependent variable

x_1 = east-west coordinate of specified point

x_2 = north-south coordinate of specified point

The next four questions or control arguments concern the dimensions of the desired map. That is, the dimensions of contour map of trend surface values and/or the contour map of residual values.

The program will ask for the following four values.

1. x_1 value at left edge of map - this is the minimum east-west coordinate value.
2. x_1 value at the right edge of map - this is the maximum east-west coordinate value.
3. X_2 value at bottom of map - this is the minimum north-south coordinate value.
4. X_2 value at top of map - this is the maximum north-south coordinate value.

The lower left-hand corner is the starting point from which all coordinate values are measured. It is convenient to have the coordinates of the starting point equal to (Ø, Ø).

The final control argument you will be asked to provide determines if a contour map of residual values is desired in addition to a contour map of the trend surface values. NOTE: Residual values are determined by subtracting the calculated trend surface values from the observed values for whatever dependent variable is being measured.

Command sequences for operating TREND follow. Computer-generated sequences are in quotes, user-generated sequences are preceded by an !.

! magic

"etc."

"enter program name" ! trend

"enter name of data segment" ! test_trend_data

" TREND SURFACE ANALYSIS"

"enter order of trend surface"

! 1

"enter X_1 value at left edge of map"

! Ø

"enter x_1 value at right edge of map"

! 1ØØ

"enter X_2 value at bottom edge of map"

! Ø

"enter x_2 value at top edge of map"

! 1ØØ

"enter 1 if you want a contour map of residual values, enter Ø if you
just want the trend surface map."

! Ø

"stop"

"Results ...."

The above control arguments will cause a first-degree trend surface to be calculated from the data in segment "test_trend_data" and a contour map of the trend surface will be plotted with an east-west range of Ø to 1ØØ and a north-south range of Ø to 1ØØ. No contour map of the residuals will be plotted.

Caution about map dimensions:

The north-south [x_2 values] range should not be more than 3 times the east-west [x_1 values] range.

DATA SEGMENT FORMAT FOR TREND

A data segment for TREND is created in an editor (QEDX is recommended).

line 1 title of data segment - you can use up to 80 characters (including blanks).

line 2 dimensions of data matrix 3,N where N = number of sample points.

line 3 the data. Each line consists of 3 numbers separated by commas.
1st number is east-west coordinate of sample point.
2nd number is north-south coordinate of sample point.
3rd number is value of dependent variable at the sample point.

line N+2 Remember to place newline character, that is carriage return and line feed, at end of last line.

Program and executive command segment listings

magic.ec

```
&command_line off
ll 132
&print
&print MAGIC - USGS Multivariate Statistical
&print
&print  Programs Package 1.0
&print
&print  =======================================
&print
&if [query  "do you require help?  "]
&then &else &goto nohelp
&if [query "do you require a list and description of programs?  "]
&then pr >udd>Trinity>RSpicer>magic_dir>help.magic.progs
&if [query "do you require help for inputting data? "]
&then pr >udd>Trinity>RSpicer>magic_dir>help.magic.input
&label nohelp
&if [query "do you wish to quit magic? "]
&then &quit
ec >udd>Trinity>RSpicer>magic_dir>[response "enter program name "]
&goto nohelp
```

biclust.ec

```
&command_line off
&print
&print
&print BICLUST
&print
&print
io attach file05 vfile_ [response "enter name of data segment "]
io open file05 si
io attach file06 vfile_ results.biclust
io open file06 so
&print
&print
>udd>Trinity>RSpicer>magic_dir>biclust
io close file05
io close file06
io detach file05
io detach file06
&if [query "Do you wish to dprint the results?" ]
&then dp results.biclust
&else pr results.biclust
&quit
```

biclust.fortran

```
c       cluster analysis
c
c       the program accepts an n by m data matrix where n is the number
c       of observations and m is the number of variables.
c       the program will use a weighted-average clustering
c       method to group samples on the basis of a measure of
c       similarity.  the data is either binary(present-absent)
c       or 4-state coded binary (abundant,common,rare,absent)
c       there is an option between two different measures of
c       similarity.  option 1 is the use of the sokal-michener
c       coefficient and option two is the use of the jaccard
c       coefficient as the measure of similarity between
c       samples.  the program also gives the option of printing
c       the input data matrix and the similarity matrix.
c
c
c       subroutines  required are readm, printm, rcoef, dist, wpga,
c       and dendro
c
c       ---------------------------------------------------------------------
      dimension nx(360,360), ipair(2,360), xlev(360), a(360,360)
      dimension kd(360,360)
      dimension ititle(20)

      common/temp8/nx
      common/temp9/ipair
      common/temp10/xlev
      common/temp11/kd
      common/temp12/a
      md=360
      nd=360
      mm=360
      write(0,1115)
 1115 format("   CLUSTER ANALYSIS"/)
c
c...  read control card
c
      write(0,1001)
 1001 format(/," this program will perform a weighted-average cluster",
     1 1x,"analysis",/," using either the sokal-michener or the",
     2 1x,"jaccard coefficient",/," as the measure of similarity.")
      write(0,1002)
 1002 format(/," enter 0 for the sokal-michener coefficient",/,
     1 "         1 for the jaccard coefficient")
      read(0,1000) koef
      write(0,1003)
      read(0,1000) itype
 1003 format(/," enter 0 for binary (presence-absence) data",/,
     1 "         1 for semi-quantitative data")
 1000 format(v)
c
```

```
c...  read and print input data matrix
c
      call rdata(n,m,ititle,itype)
      write(0,1004)
 1004 format(/," enter 0 if you desire the input data matrix printed",
     1 /,"       1 if you don't.")
      read(0,1000) iwrite
      if(iwrite.ne.0) goto 20
      write(6,2005)
 2005 format(/,"   input data matrix")
      call pdata(n,m,itype)
c
c...  calculate similarity matrix
c
   20 if(itype.eq.1) call expand(n,m)
      call bicoef(n,m,koef)
      write(0,1005)
 1005 format(/, " enter 0 if you desire the similarity matrix printed",
     1 /,"       1 if you don't")
      read(0,1000) iprint
      if(iprint.ne.0) goto 21
c
c     print similarity matrix
c
      if(koef.eq.0) write(6,2003)
      if(koef.eq.1) write(6,2004)
      call psim(n,m)
c
c     calculate and print linkage table
c
   21 continue
      call wpga(n,mm,1)
c
c     print dendrogram
c
      call dendro(n,mm,1)
 2001 format(1h0,4x,'input data matrix -',1x,
     1 'columns = variables,rows = observations')
 2002 format(/,4x,'similarity matrix')
 2003 format(/," similarity matrix using sokal-michener coefficient")
 2004 format(/," similarity matrix using jaccard coefficient")
 3000 stop
      end
c      subroutine to perform weighted pair-group average clustering
c
      subroutine wpga(m,ml,isim)
      dimension x(360,360),ipair(2,360),xlev(360)
      dimension i1(360),i2(360),xsim(360)

      common/temp12/x
      common/temp9/ipair
      common/temp10/xlev
c
c      initialize
```

```
c
      write(6,2004)
      write (6,2001)
      do110 i=1,m
      i1(i)=i
  110 continue
      xxxx=-9.0e+35
      if(isim.ne.1)xxxx=+9.0e+35
      m3=m-1
      ic=0
c
c      for a correlation matrix find the largest similarity in each column
c
c      for a distance matrix find the smallest similarity in each column
c
    1 do100 i=1,m
      if(i1(i).le.0)goto 100
      ix=0
      xx=xxxx
      do101 j=1,m
      if(i.eq.j)goto101
      if(i1(j).le.0)goto101
      goto(11,12),isim
   11 if(x(j,i)-xx)101,101,13
   12 if(x(j,i)-xx)13,101,101
   13 xx=x(j,i)
      ix=j
  101 continue
      i2(i)=ix
      xsim(i)=xx
  100 continue
c
c      for a correlation matrix find mutualy high pairs
c      for a distance matrix find mutualy low pairs
c
      do 102 i=1,m3
      if(i1(i).le.0)goto102
      j=i2(i)
      if(i1(j).le.0) goto102
      if(j.le.i)goto102
      if(i1(j).eq.i)goto14
      if(abs(xsim(i)-xsim(j)).gt.0.00001)goto102
c
c      save parameters for a cluster
c
   14 ic=ic+1
      ipair(1,ic)=i
      ipair (2,ic)=j
      xlev(ic)=xsim(i)
      write(6,2002) i,j,xsim(i)
      i1(i)=j
      i1(j)=0
c
c      average the two columns
```

```
c
      do 103 k=1,m
      x(k,i)=(x(k,i)+x(k,j))/2.0
  103 continue
  102 continue
c
c      average rows that were clustered on this iteration
      do 105 i=1,m3
      if(il(i).le.0)goto105
      if(il(i).eq.i)goto105
      j=il(i)
c
c      average two rows in the new cluster
c
      do106 k=1,m
      if(il(k).le.0)goto106
      x(i,k)=(x(i,k)+x(j,k))/2.0
  106 continue
      il(i)=i
  105 continue
      if(ic.lt.m3)goto1
      write(6,2003)
      return
 2001 format(/)
 2002 format(6x,2i6,f15.5)
 2003 format(/,4x,'columns 1 and 2 -',1x,'observations combined into
     1clusters',/,5x,'column 3 - similarity level of clustering')
 2004 format(//," LINKAGE TABLE : ")
      end
c      program 7.8
c
c      subroutine to print dendrogram
c
      subroutine dendro(m,ml,isim)
      dimension ipair(2,360), xlev(360)
      dimension il(360),i2(360)
      dimension iout(61),xx(13)
      common/temp9/ipair
      common/temp10/xlev
      data iblnk,ici,icp,icm/" ","i",".","-"/
c
c...  determine order that branches will be printed in
c
      write(6,2006)
 2006 format(//," DENDOGRAM : ")
      m2=m-1
      do 100 i=1,m
      il(i)=0
      i2(i)=0
  100 continue
      do 101 i=1,m2
      j=i-1
   11 if (j .le. 0) go to 12
      if (ipair(1,i) .eq. ipair(1,j)) go to 13
      j=j-1
      go to 11
   12 i2 (i)=1
      go to 15
   13 k=i1(j)
      if (k .eq. 0) go to 14
      j=k
      go to 13
   14 i1(j)=i
   15 do 102 j=1,i
      k=j
      if ( ipair (2,i) .eq. ipair(1,j)) goto 16
  102 continue
      go to 101
   16 i2 (k) =0
      i1(i)=k
  101 continue
c
c...  find starting cluster
c
      do 103 i=1,m2
      js=i
      if (i2(i) .ne. 0) go to 20
  103 continue
      goto  3000
   20 node=ipair(1,js)
c
c...  find largest and smallest similarity coef.
c
      xmin=xlev(1)
      xmax=xmin
      do 104 i=1,m2
      if (xlev(i) .lt. xmin) xmin=xlev(i)
      if (xlev(i) .gt. xmax) xmax=xlev(i)
  104 continue
      dx=(xmax-xmin)/25.0
      xmin=xmin-dx
      xmax=xmax+dx
      dx=(xmax-xmin)/60.0
      if (isim .ne. 2) go to 21
      dx=-dx
      xmin=xmax
c
c...  blank out print line array
c
   21 do 105 i=1,61
      iout(i)=iblnk
  105 continue
c
c...  print dendrogram
c
      x=xmin
      do 106 i=1,13
      xx(i)=x
      x=x+dx*5.0
  106 continue
      write (6,2000)
      write (6,2001) (xx(i),i=2,12,2)
      write (6,2002) (xx(i),i=1,13,2)
      write (6,2003)
   22 x=xmin
      if (js .ne. 0) x=xlev(js)
      is=ifix((x-xmin)/dx)+1
      do 110 i=is,61
      iout(i)=icm
  110 continue
      iout(is)=icp
      if (js .ne. 0) write (6,2004) iout,node,x
      if (js .eq. 0) write (6,2004) iout,node
      if (js .eq. 0) go to 31
      do 111 i=is,61
      iout(i)=iblnk
  111 continue
      iout(is)=ici
      write (6,2004) (iout(i), i=1, is)
      node=ipair(2,js)
      js=il(js)
      go to 22
   31 write (6,2003)
      write (6,2002)(xx(i),i=1,13,2)
      write  (6,2001) (xx(i),i=2,12,2)
      write (6,2005)
      return
 2000 format (/)
 2001 format (6x,6f10.4)
 2002 format (1x,7f10.4)
 2003 format (6x,'+',12('----+'))
 2004 format(6x,61a1,3x,i4,f10.4)
 2005 format (/,4x,'dendrogram - ',1x,
     1 'values along x-axis are similarities')
 3000 return
      end
c
c     subroutine to print a matrix having n rows and m columns
c
      subroutine pdata(n,m,itype)
      dimension na(360,360)

      common/temp8/na
c     print matrix out in strips of 20 columns
      do100ib=1,m,20
      ie=ib+19
      if(ie-m)2,2,1
    1 ie=m
cprint heading
    2 write(6,2000)(i,i=ib,ie)
      do101j=1,n
c      print row of matrix
```

```
      write(6,2001) j,(na(j,k),k=ib,ie)
  101 continue
  100 continue
      return
 2000 format(/,10x,20i6)
 2001 format(3x,i4,3x,20i6)
      end
      subroutine rdata(n,m,ititle,itype)
      dimension num(360,360),ititle(20)
      common/temp8/num

c
c     read in title of data segment
      read(5,1001) ititle
c
c     read in dimensions of data matrix - variables,samples
      read(5,1000) m,n
c
c     read in the sample name and the variable values
      do 100 i=1,n
      read(5,1000)(num(i,j),j=1,m)
  100 continue
      return
 1000 format(v)
 1001 format(20a4)
      end

c     subroutine to print a matrix in columns of 20
c
      subroutine psim(n,m)
      dimension a(360,360)
      common/temp12/a

      do 100 ib=1,n,20
      ie=ib+19
      if(ie-n) 2,2,1
    1 ie=n
c     print sample names
    2 write(6,2000) (i,i=ib,ie)
      do 101 j=ib,n
      ji=j
      if(ji.gt.ie) ji=ie
c     print row of matrix
      write(6,2001) j,(a(j,k),k=ib,ji)
  101 continue
  100 continue
      return
 2000 format(9x,20i6)
 2001 format(3x,i4,3x,20f6.3)
      end
      subroutine bicoef(n,m,koef)
      dimension nx(360,360),a(360,360),npres(360),nabs(360)
      dimension cosav(360)
      common/temp12/a
      common/temp8/nx
      kk=0
      nc=m
c
c ... find frequency of 2's and 1's in each sample - used for
c ... calculating expected values of association - see below
      do 71 i=1,n
      npres(i)=0
   71 nabs(i)=0
      do 77 j=1,nc
      do 76 i=1,n
      if(nx(i,j)-1) 76,75,74
   74 npres(i)=npres(i)+1
      goto 76
   75 nabs(i)=nabs(i)+1
   76 continue
   77 continue
c
c ... calculate coefficients of association
      ant=nc
      write(6,13)
      sum=0.0
      avexpt=0.0
      do 170 i=1,n
      do 160 ii=i,n
      anumer=0.0
      denom=0.0
      rel=0.0
      do 130 j=1,nc
      k=nx(i,j)*nx(ii,j)+1
      if(koef.eq.1) goto 90
      goto (100,120,110,120,120), k
   90 goto (100,130,110,120,120), k
  100 rel=rel+1.0
      goto 130
  120 anumer=anumer+1.0
  110 denom=denom+1.0
  130 continue
c
c ... find expected value of association
      expect=npres(i)*npres(ii)
      if(koef.eq.0) expect=expect+float(nabs(i)*nabs(ii))
      if(expect.gt.0.0) expect=expect/(expect+
     1 float(nabs(i)*npres(ii)+npres(i)*nabs(ii)))
      avexpt=(avexpt*sum+expect)/(sum+1.0)
      sum=sum+1.0
      cosav(ii)=0.0
      if(denom.gt.0.0) cosav(ii)=anumer/denom
      rel=rel/ant
      if(rel.gt.0.25) write(6,14) i,ii,cosav(ii),rel
      a(i,ii)=cosav(ii)
  160 continue
  170 continue
```

```
      do 180 i=1,n
      do 180 j=i,n
  180 a(j,i)=a(i,j)
      return
   13 format(/, " the following pairs of samples have a high",
     1 1x,"proportion of uncertain matches",// 10x,"sample",
     2 1x,"pair",5x,"similarity coefficient",5x,"proportion",
     3 1x,"of uncertain matches",//)
   14 format(8x,2i5,13x,f7.4,20x,f7.4)
      end

      subroutine expand(n,m)
      dimension na(360,360),kd(360,360)
      common/temp8/na
      common/temp11/kd
      do 100 i=1,n
      do 100 j=1,m
      ka=na(i,j)/100
      kb=(na(i,j)-ka*100)/10
      kc=na(i,j)-ka*100-kb*10
      k=3*j
      kd(i,(k-2))=ka
      kd(i,(k-1))=kb
      kd(i,k)=kc
  100 continue
      m=m*3
      do 110 i=1,n
      do110 j=1,m
      na(i,j)=kd(i,j)
  110 continue
      return
      end
```

pca.ec

```
&command_line off
&print
&print
&print PCA
&print
&print
io attach file05 vfile_ [response "enter name of data segment "]
io open file05 si
io attach file06 vfile_ results.pca
io open file06 so
&print
&print
>udd>Trinity>RSpicer>magic_dir>pca
io close file05
io close file06
io detach file05
io detach file06
&if [query "Do you wish to dprint the results?" ]
&then dp results.pca
&else pr results.pca
&quit
```

pca.fortran

```
c     principal component analysis
c
c     the program accepts an n by m data matrix where n is the number
c     of observations and m is the number of variables.  if the
c     first option is 1, an m by m matrix of covariances between
c     columns will be computed.  if this option is 2, an n by n
c     matrix of covariances between rows will be computed.  if the
c     option is 0, the program calls exit.
c     if the second is 1, a standardized covariance (correlation) matrix
c     is created.  if this option is 2, a raw covariance matrix
c     is created.
c
c     format of control card
c
c          col 1-3 0 = end of job
c                  1 = do not transpose input data matrix
c                  2 = transpose data matrix
c
c          col 4-6 1 = calculate correlation matrix
c                  2 = calculate covariance matrix
c
c     subroutines required are ready, printm, stand, rcoef, cov,
c     eigenj, and mmult.
c     ==============================================================
c
      dimension x(350,350),a1(350,350),a2(350,350),score(350,350),
     1ititle(20)
      common/temp/a2
      common/temp2/a1
      common/temp3/score
      common/temp4/x
      md=350
      nd=350
      mm=350
c
c. .. read control card
c
      write(0,2010)
 2010 format("   PRINCIPAL COMPONENTS ANALYSIS")
    1 write(0,2007)
 2007 format(/"enter 1 for no transposition of data matrix"/"
     12 for transpose of data matrix")
      read(0,1000)itrans
      if(itrans.lt.1.or.itrans.gt.2)goto1
      write(0,2008)
 2008 format(//"Enter 1 for correlation matrix"/"      2 for covariance
     1 matrix"/"      3 for cos theta transform"/)
      read(0,1000)isim
c
c...  read and print input data matrix
c
```

```
      call readm(n,m,ititle)
      write(6,2009) ititle
      write(6,2001)
      call printm(x,n,m,nd,md)
      if(isim.eq.2.or.isim.eq.3)goto 104
      goto 102
  104 write(0,2011)
 2011 format("for row-wise normalization type 1 else type 0")
      read(0,1000)itag
      if(itag.eq.0)goto103
      call norm(n,m,nd,md)
      write(6,2020)
      call printm(x,n,m,nd,md)
 2020 format(/," normalized input data")
  103 if(isim.ne.3)goto102
      write(0,2012)
 2012 format("for zero mean transformation type 1 else type 0")
      read(0,1000)imeant
      if(imeant.eq.0)goto102
      call zeromn(n,m,nd,md)
      write(6,2021)
 2021 format(/," transformed data matrix")
      call printm(x,n,m,nd,md)
 2009 format(/1x,20a4/)
c
c...  if correlation matrix is to be calculated, standardize
c     input data matrix and print standardized data matrix
c
  102 if (isim .ne. 1) go to 2
      call stand(n,m)
      write(6,2006)
      call printm(x,n,m,nd,md)
c
c...  transpose input data matrix (if required)
c
    2 if (itrans .ne. 2) go to 3
      mt=m
      if (n .gt. m) mt=n
      do 110 i=1,mt
      do 110 j=i,mt
      xs=x(i,j)
      x(i,j)=x(j,i)
      x(j,i)=xs
  110 continue
      mt=m
      m=n
      n=mt
c
c...  calculate and print similarity matrix
c
    3 if (isim .eq. 1) call rcoef(n,m)
      if (isim .eq. 2) call cov  (n,m)
      if(isim .eq.3) call ctheta(n,m)
      write(6,2002)
      call printm(a1,m,m,mm,mm)
c
c...  calculate eigenvalues and eigenvectors
c
      call eigenj(m)
c
c...  move eigenvalues to first column
c     calculate sum of eigenvalues
c
      sume=0.0
      do 100 i=1,m
      a1(i,1)=a1(i,i)
      sume=sume+a1(i,1)
  100 continue
c
c...  calculate percent contribution of each eigenvalue
c
      sumee=0.0
      do 101 i=1,m
      a1(i,2)=a1(i,1)*100.0/sume
      sumee=sumee+a1(i,1)
      a1(i,3)=sumee*100.0/sume
  101 continue
c
c...  print eigenvalues and percent contribution
c
      write(6,2003)
      call printm(a1,m,3,mm,mm)
c
c...  print eigenvectors
c     note... eigenvectors are stored columnwise
c
      write(6,2004)
      call printm(a2,m,m,mm,mm)
c
c...  calculate and print scores
c
      call mmult(n,m,m)
      write(6,2005)
      call printm(score,n,m,nd,md)
      call pcaplot(m,n)
 1000 format (v)
 2001 format (/4x,"input data matrix -",1x,
     1 "columns = variables, rows = observations")
 2002 format (/4x,"similarity matrix")
 2003 format (/4x,"column 1 = eigenvalues",2x,
     1 ",column 2 = percent of trace",/,
     2 5x,"column 3 = cumulative percent of trace")
 2004 format (/4x,"principal axis matrix -",1x,
     1"columns = eigenvectors, rows = variables")
 2005 format (/4x,"principal component scores -",1x,
     1"columns = eigenvectors, rows = observations")
 2006 format (/4x,"standardized input data matrix -",1x,
     1 "columns = variables, rows = observations")
```

```
      stop
      end
c     subroutine to calculate the matrix of cosine theta similarity
c     coefficients between columns of data matrix x
c
      subroutine ctheta(n,m)
      dimension x(350,350),a(350,350)
c
c...  calculate cosine theta between columns i and j
c
      common/temp2/a
      common/temp4/x
      do 100 i=1,m
      do 100 j=i,m
c
c...  zero sums
c
      sx1x1=0.0
      sx2x2=0.0
      sx1x2=0.0
c
c...calculate sums of squares and sum of cross product
c
      do 101 k=1,n
      sx1x1=sx1x1+x(k,i)**2
      sx2x2=sx2x2+x(k,j)**2
      sx1x2=sx1x2+x(k,i)*x(k,j)
  101 continue
c
c...  calculate cosine theta and store in matrix a
c
      a(i,j)=sx1x2/sqrt(sx1x1*sx2x2)
      a(j,i)=a(i,j)
  100 continue
      return
      end
      subroutine norm(n,m,n1,m1)
      dimension x(350,350)
      common/temp4/x
      do 3 i=1,n
      sumsq=0.0
      do 2 j=1,m
    2 sumsq=sumsq+(x(i,j)*x(i,j))
      do 1 j=1,m
      x(i,j)=x(i,j)/(sqrt(sumsq))
    1 continue
    3 continue
      return
      end
      subroutine zeromn(n,m,n1,m1)
      dimension x(350,350)
      common/temp4/x
      do 3 j=1,m
      sumcol=0.0
      do 2 i=1,n
    2 sumcol=sumcol+x(i,j)
      do 1 i=1,n
      x(i,j)=x(i,j)-(sumcol/float(n))
    1 continue
    3 continue
      return
      end
      subroutine pcaplot(m,n)
      dimension result(350,350),xxmax(3),xxmin(3)
      common/temp3/result
      do 100 i=1,3
      xxmax(i)=result(1,i)
      xxmin(i)=result(1,i)
      do 100 j=1,n
      if(xxmax(i).lt.result(j,i)) xxmax(i)=result(j,i)
      if(xxmin(i).gt.result(j,i)) xxmin(i)=result(j,i)
  100 continue
      call ploter(xxmax,xxmin,1,2,m,n)
      call ploter(xxmax,xxmin,1,3,m,n)
      call ploter(xxmax,xxmin,2,3,m,n)
      return
      end
      subroutine ploter(xmax,xmin,ixx,iyy,m,n)
      dimension ssd(350,350),result(350,350)
      dimension isml(11),xmax(3),xmin(3),x(350),y(350),is(350)
      dimension  xc(350),yc(350),iovp(350,5)
      dimension map(110),id(5),ism(14)
      common/temp/ssd/temp3/result
      data ism/" ","*","a","b","c","d","e","f","g","h","+","-","i",
     $ "1000"/
      data isml/1h0,1h1,1h2,1h3,1h4,1h5,1h6,1h7,1h8,1h9,1h /
      data minus/1h-/
      write(6,1004)
      do 120 i=1,n
      x(i)=result(i,ixx)
      y(i)=result(i,iyy)
  120 is(i)=i
      do 121 i=2,n
      if(x(i)-x(i-1)) 121,121,20
   20 j=i
   21 xch=x(j)
      x(j)=x(j-1)
      x(j-1)=xch
      xch=y(j)
      y(j)=y(j-1)
      y(j-1)=xch
      isc=is(j)
      is(j)=is(j-1)
      is(j-1)=isc
      j=j-1
      if(j.eq.1) goto 121
      if(x(j)-x(j-1)) 121,121,21
  121 continue
```

```
      numovp=0
      vink=(xmax(iyy)-xmin(iyy))/59.0
      hink=(xmax(ixx)-xmin(ixx))/99.0
      do 100 iy=1,60
      do 101 i=1,110
  101 map(i)=ism(1)
      do 102 i=1,101,10
  102 map(i)=ism(13)
      do 103 ix=1,n
      iyt=(xmax(iyy)-y(ix))/vink+1.0
      if(iyt.ne.iy) goto 103
      ixt=(x(ix)-xmin(ixx))/hink+1.0
      if(ixt.lt.1.or.ixt.gt.100) goto 103
      id(1)=ism(1)
      id(2)=ism(11)
      id(3)=is(ix)/100
      id(4)=(is(ix)-id(3)*100)/10
      id(5)=is(ix)-id(3)*100-id(4)*10
      nc=5
    1 if(id(3).gt.0) goto 2
      nc=nc-1
      id(3)=id(4)
      id(4)=id(5)
      id(5)=ism(1)
      goto 1
    2 do 242 iwj1=3,nc
      do 242 iwj=1,10
      if(id(iwj1).eq.iwj-1) id(iwj1)=ism1(iwj)
  242 continue
      do 105 i=2,nc
      ip=ixt+i-2
      if(map(ip).ne.ism(1).and.map(ip).ne.ism(13)) goto 3
  105 continue
      do 106 i=2,nc
      ip=ixt+i-2
  106 map(ip)=id(i)
      goto 103
    3 numovp=numovp+1
      xc(numovp)=x(ix)
      yc(numovp)=y(ix)
      do 107 i=1,5
  107 iovp(numovp,i)=id(i)
      if(map(ip).eq.ism(11).or.map(ip).eq.ism(12)) ip=ip-1
    4 if(ip.le.0) goto 103
      i=1
      if(map(ip).eq.ism(13)) goto 5
      do 108 i=1,9
      if(map(ip).eq.ism(i)) goto 5
  108 continue
      goto 103
    5 map(ip)=ism(i+1)
      iovp(numovp,1)=ism(i+1)
  103 continue
      yp=xmax(iyy)-float(iy-1)*vink
      write(6,1000) yp,map
  100 continue
      do 130 i=1,101
  130 map(i)=minus
      do 131 i=1,101,10
  131 map(i)=ism(13)
      write(6,1009) map
      do 132 i=1,11
      x(i)=xmin(ixx)+float(i-1)*10.0*hink
  132 continue
      write(6,1008) (x(i),i=1,11,2),(x(i),i=2,11,2)
      write(6,1006) ixx
      write(6,1007) iyy
      if(numovp.eq.0) return
      write(6,1001)
      write(6,1002)
      do 110 i=1,numovp
      write(6,1003) (iovp(i,j),j=1,5),xc(i),yc(i)
  110 continue
      return
 1000 format(1x,g15.5,2x,110a1)
 1001 format(////,1x,17h overprint values,//)
 1002 format(37h point          xcoord            ycoord   )
 1003 format( 5x,5a1,2g15.5)
 1004 format(///)
 1006 format(///,27h0horizontal eigenvector =   i2   )
 1007 format(27h0  vertical eigenvector =      i2)
 1008 format(3x,6f20.4,/,13x,5f20.4)
 1009 format(18x,110a1)
      end
```

trend.ec

```
&command_line off
&print
&print
&print TREND
&print
&print
io attach file05 vfile_ [response "enter name of data segment"]
io open file05 si
io attach file06 vfile_ results.trend
io open file06 so
&print
&print
>udd>Trinity>RSpicer>magic_dir>trend
io close file05
io close file06
io detach file05
io detach file06
&if [query "Do you wish to dprint the results? "]
&then dp results.trend
&else pr results.trend
&quit
```

trend.fortran

```
c     trend surface analysis
c
c
c     program to compute a polynomial trend surface of degree iord.
c     data are entered using readm and input data matrix is n by 3,
c     where n is the number of observations. the first column
c     contains x1 (east-west or across the map) coordinate, the
c     second column contains x2 (north-south or down the map),
c     and the third column contains the dependent variable.
c
c
c
c     subroutines required are readm,printm, and sle.
c
c      ===========================================================
      dimension a(15,15),b(15),c(15),data(350,5),ititle(20)
      dimension amap(181,101),dist(350)
      width=10.0
      nd=350
      mm=15
      c(1)=1.0
c
c...  read control card
c
      write(0,2015)
 2015 format("   TREND SURFACE ANALYSIS")
      write(0,2014)
 2014 format(/)
      write(0,2016)
 2016 format(/"enter order of trend surface")
      read(0,1000) iord
 1000 format(v)
      write(0,2019)
 2019 format(/"enter x1 value at left edge of map")
      read(0,1001) x1min
      write(0,2020)
 2020 format(/"enter x1 value at right edge of map")
      read(0,1001) x1max
      write(0,2021)
 2021 format(/"enter x2 value at bottom edge of map")
      read(0,1001) x2min
      write(0,2022)
 2022 format(/"enter x2 value at top edge of map")
      read(0,1001) x2max
      write(0,2023)
 2023 format(/"enter 1 if you want a contour map of residual values",1x,
     1 "enter 0 if you just want the trend surface map")
      read(0,1001) itest
 1001 format(v)
      if ((x2max-x2min).le.3.0*(x1max-x1min)) goto 10
      write(0,2024)
 2024 format(//," CAUTION - the north-south map dimension",
     1  " is more than 3 times",/," the east-west map",
     2  " dimension.  The north-south map scale will",/,
     3  " be different than the east-west map scale!",//)
c
c...  read and print input data matrix
c
   10 call readm(data,n,m,nd,5,ititle)
c
c...  calculate number of coefficients
c
      iord2=(iord+1)*(iord+2)/2
      if (n.gt.iord2) goto 15
      write(0,2025) iord,iord2
 2025 format(//" ERROR - a trend surface of order",i3,
     1 " must have",i4," or more data points",///,
     2 " JOB ABORTED!",//)
      stop
c     zero sle matrix
   15 do 100 i=1,iord2
      b(i)=0.0
      do 100 j=1,iord2
      a(i,j)=0.0
  100 continue
c
c...  calculate sle matrix
c
      do 101 i=1,n
      jb=1
      do 102 j=1,iord
      do 103 k=1,j
      jb=jb+1
      kb=jb-j
      c(jb)=c(kb)*data(i,1)
  103 continue
      jb=jb+1
      c(jb)=c(kb)*data(i,2)
  102 continue
      do 104 j=1,iord2
      b(j)=b(j)+c(j)*data(i,3)
      do 104 k=1,iord2
      a(j,k)=a(j,k)+c(j)*c(k)
  104 continue
  101 continue
c
c... solve sle
c
      call sle(a,b,iord2,mm,0.00000001)
c
c...  calculate estimated values and deviation for each observation
c
      do 105 i=1,n
      jb=1
      do 106 j=1,iord
      do 107 k=1,j
      jb=jb+1
      kb=jb-j
      c(jb)=c(kb)*data(i,1)
  107 continue
      jb=jb+1
      c(jb)=c(kb)*data(i,2)
  106 continue
      data(i,4)=0.0
      do 108 j=1,iord2
      data(i,4)=data(i,4)+b(j)*c(j)
  108 continue
      data(i,5)=data(i,3)-data(i,4)
  105 continue
c
c...  print x1, x2, y, estimated y, and deviation
      write(6,2001) ititle
 2001 format(1x,20a4)
      write(6,2002)
      call printm(data,n,5,nd,5)
c
c...  print trend surface coefficients
c
      write(6,2003)
      call printm(b,iord2,1,mm,1)
c
c...  calculate(error measures
c
      sy=0.0
      syy=0.0
      syc=0.0
      syyc=0.0
      do 111 i=1,n
      sy=sy+data(i,3)
      syy=syy+data(i,3)**2
      syc=syc+data(i,4)
      syyc=syyc+data(i,4)**2
  111 continue
      sst=syy-sy*sy/float(n)
      ssr=syyc-syc*syc/float(n)
      ssd=sst-ssr
      ndf1=iord2-1
      amsr=ssr/float(ndf1)
      ndf2=n-iord2
      amsd=ssd/float(ndf2)
      r2=ssr/sst
      r=sqrt(r2)
      f=amsr/amsd
      ndf3=n-1
c
c...  print error messages
c
      write(6,2004)
      write(6,2005) ssr,ndf1,amsr,f
      write(6,2006) ssd,ndf2,amsd
      write(6,2007) sst,ndf3
      write(6,2008) r2,r
c
c...  calculate map size and scale parameters
      iw = 1 + width*10.0
      ih = 1 + min1(180.0,width*6.0*(x2max-x2min)/(x1max-x1min))
      dx1=(x1max-x1min)/float(iw-1)
      dx2=(x2max-x2min)/float(ih-1)
c
c...  calculate and print map one line at a time
c
      write(6,2009)
      x2=x2max
      do 121 i=1,ih
      x1=x1min
      do 122 j=1,iw
      jb=1
      do 124 k=1,iord
      do 125 l=1,k
      jb=jb+1
      kb=jb-k
      c(jb)=c(kb)*x1
  125 continue
      jb=jb+1
      c(jb)=c(kb)*x2
  124 continue
      y=0.0
      do 126 k=1,iord2
      y=y+b(k)*c(k)
  126 continue
      amap(i,j)=y
      x1=x1+dx1
  122 continue
      x2=x2-dx2
  121 continue
      call plot(amap,ih,iw,x1min,dx1,x2max,dx2)
      write(6,2011) iord
c
c...  calculate and plot contour map of residuals
      if(itest.ne.1) stop
      small=(dx1*dx1+dx2*dx2)/1000.0
      x2=x2max
      do 201 i=1,ih
      x1=x1min
      do 202 j=1,iw
c
c...  calculate dist**2 between current grid point and all data points
      do 203 k=1,n
      dist(k)=(x1-data(k,1))**2+(x2-data(k,2))**2
  203 continue
c
c...  fins the 6 nearest data points nad calculate sums
c
```

```
      s1=0.0
      s2=0.0
      do 204 k=1,6
      ic=1
      do 205 l=2,n
      if(dist(l).lt.dist(ic)) ic=l
  205 continue
      if(dist(ic).lt.small) goto 210
      d=sqrt(dist(ic))
      s1=s1+data(ic,5)/d
      s2=s2+1.0/d
      dist(ic)=+9.0e+35
  204 continue
c
c...  calculate grid point and store in matrix
      amap(i,j)=s1/s2
      goto 211
  210 amap(i,j)=data(ic,5)
  211 x1=x1+dx1
  202 continue
      x2=x2-dx2
  201 contioue
c
c...  print grid points on map
      call plot(amap,ih,iw,x1min,dx1,x2max,dx2)
      write(6,2014)
      write(6,2013)
 2002 format(1h0,4x,"column 1 = x1, column 2 = x2,",1x,
     1 "column 3 = y, column 4 = estimated y, column 5 = deviation")
 2003 format(1h0,4x,"trend surface coefficients",3x,
     1 "1 = constant term")
 2004 format(/," source of ",13x,25hsum of  degrees of   mean,/,
     1 10h variation,13x,37hsquares   freedom   squares    f-test,/,
     2 1x,60(1h-))
 2005 format(11h regression,10x,f10.2,i8,2x,f10.2,/,51x,f10.4)
 2006 format(10h deviation,11x,f10.2,i8,2x,f10.2)
 2007 format(16h0total variation,5x,f10.2,i8)
 2008 format(" goodness of fit = ",f10.4,/,
     1 " correlation coefficient = ",f10.4)
 2009 format(/)
 2011 format(1h0,4x,"trend surface map of degree ",i2)
 2013 format(4x,"contour map of residual values")
      stop
      end

c     subroutine to read a matrix
c
      subroutine readm(a,n,m,n1,m1,ititle)
      dimension a(n1,m1),ititle(20)
c     read size of matrix
      read(5,1002)ititle
      read(5,1001)m,n
c     read matrix one row at a time
      do 100 i=1,n
      read(5,1001) (a(i,j),j=1,m)
  100 continue
      return
 1001 format (v)
 1002 format (20a4)
      end

c..   subroutine to print a matrix
c
      subroutine printm(a,n,m,n1,m1)
      dimension a(n1,m1)
c     print matrix out in strips of 10 columns
      do 100 ib=1,m,10
      ie=ib+9
      if (ie-m) 2,2,1
    1 ie=m
c     print heading
    2 write(6,2000) (i,i=ib,ie)
      do 101 j=1,n
c     print row of matrix
      write(6,2001) j,(a(j,k),k=ib,ie)
  101 continue
  100 continue
      return
 2000 format (1x,10i12)
 2001 format (i5,10f12.4)
      end
c
c...  subroutine for solution of n simultaneous equations.
c
      subroutine sle(a,b,n,n1,zero)
      dimension a(n1,n1),b(n1)
      do 100 i=1,n
      div=a(i,i)
      if(abs(div)-zero) 99,99,1
    1 do 101 j=1,n
      a(i,j)=a(i,j)/div
  101 continue
      b(i)=b(i)/div
      do 102 j=1,n
      if (i-j) 2,102,2
    2 ratio=a(j,i)
      do 103 k=1,n
      a(j,k)=a(j,k)-ratio*a(i,k)
  103 continue
      b(j)=b(j)-ratio*b(i)
  102 continue
  100 continue
      return
   99 write(0,1000)
 1000 format(1x,"the mistake is in the sle subroutine")
      end

c     subroutine to plot a contour map from a rectangular matrix
```

```
c
      subroutine plot(y,nr,mc,x1min,dx1,x2max,dx2)
      dimension y(181,101),iout(101),ichar(13)
      data ichar/"3"," ","2"," ","1"," ","$"," ","a"," ","b"," ","c"/
c
c..   find largest and smallest values in map
      ymin=y(1,1)
      ymax=ymin
      do 100 i=1,nr
      do 100 j=1,mc
      yt=y(i,j)
      if(yt.lt.ymin) ymin=yt
      if(yt.gt.ymax) ymax=yt
  100 continue
c
c..   print map one line at a time
c
      write(6,2001)
      write(6,2005) (x1min+(i-1)*10.0*dx1,i=1,11)
      write(6,2006)
      do 101 k=1,nr,6
      k2 = min0(nr,k+5)
      do 101 i=k,k2
      do 102 j=1,mc
      iy=((y(i,j)-ymin)/(ymax-ymin))*13.0+1.0
      if(iy.gt.13) iy = 12
      if (iy.lt.1) iy = 2
      iout(j)=ichar(iy)
  102 continue
      if (k.ne.i) write(6,2002) (iout(j),j=1,mc)
      if (k.eq.i) write(6,2004) (iout(j),j=1,mc),x2max-(k-1)*dx2
  101 continue
      write(6,2006)
      write(6,2005) (x1min+(i-1)*10.0*dx1,i=1,11)
      cint=(ymax-ymin)/13.0
      refc=ymin+7.0*cint
      write(6,2003) refc,cint
      write(6,3000)
      do 200 i=1,13,2
  200 write(6,3001) ichar(i),refc+(i-7)*cint,refc+(i-6)*cint
 3000 format(//,2x,"map",/,2x,"symbol",18x,"value:",/)
 3001 format(5x,a1,"   -   ",f10.2," to ",f10.2)
      return
 2001 format(/)
 2002 format(6x," ",101a1)
 2003 format(/,4x,"reference contour ($) = ",f10.4,3x,
     1 "contour interval = ",f10.4)
 2004 format(6x,"+",101a1," +",f10.3)
 2005 format(11f10.2)
 2006 format(7x,"+",10("         +"))
      end
```

bicor.ec

```
&command_line off
ll 132
&print
&print
&print BICOR
&print
&print
>udd>Trinity>RSpicer>magic_dir>bicor
&quit
```

bicor.fortran

```
c     program to compute
c        a. variance of each variable
c        b. mean of each variable
c        c. covariance between variables
c        d. correlation between variables
c
c     set sums to zero
      sumx1=0.0
      sumx2=0.0
      sx1sq=0.0
      sx2sq=0.0
      sx1x2=0.0
c     read number of samples to be used
      write(0,105)
  105 format("enter number of pairs of observations")
      read(0,1000) ns
      write(0,110)
  110 format("enter pairs of observations")
      do 100 i=1,ns
c     read a sample and add to sum
      write(0,120)
  120 format("?")
      read (0,1000)x1,x2
      sumx1=sumx1+x1
      sumx2=sumx2+x2
      sx1sq=sx1sq+x1*x1
      sx2sq=sx2sq+x2*x2
      sx1x2=sx1x2+x1*x2
  100 continue
c     print sums
      write (0,2000) ns,sumx1,sx1sq,sumx2,sx2sq,sx1x2
c     compute and print mean,variance and standard deviation
c     of variable x1
      amean=sumx1/float(ns)
      var=(float(ns)*sx1sq-sumx1*sumx1)/float(ns*(ns-1))
      stdev1=sqrt(var)
      write(0,2001)amean,var,stdev1
c     compute and print mean, variance and standard deviation
c     of variable x2
      amean=sumx2/float(ns)
      var=(float(ns)*sx2sq-sumx2*sumx2)/float(ns*(ns-1))
      stdev2=sqrt(var)
      write (0,2002) amean,var, stdev2
c     compute and print covariance between x1 and x2
      cov=(float(ns)*sx1x2-sumx1*sumx2)/float(ns*(ns-1))
      write (0,2003) cov
c     compute and print correlation between x1 and x2
      cor=cov/(stdev1*stdev2)
      write (0,2004) cor
 1000 format (v)
 2000 format (1h1,11x,21h number of samples = ,i10,//,
     121x,12hsum of x1 = ,f10.2,//,
     210x,23hsum of squares of x1 = ,f10.2,//,
     321x,12hsum of x2 = ,f10.2,//,
     410x,23hsum of squares of x2 = ,f10.2,//,
     5 9x, 24hsum of cross products = ,f10.2)
2001 format (/,20x,13hmean of x1 = ,f10.2,//,
     116x,17hvariance of x1 = ,f10.2,//,
     2 6x,27hstandard deviation of x1= ,f10.2)
2002 format (/,20x,13hmean of x2 = ,f10.2,//,
     116x,17hvariance of x2 = ,f10.2,//,
     2 6x,27hstandard deviation of x2 = ,f10.2)
2003 format (/,2x,31hcovariance between x1 and x2 = ,f10.2)
2004 format (/,1x,32hcorrelation between x1 and x2 = ,f10.4)
      stop
      end
```

cluster.ec

```
&command_line off
&print
&print
&print CLUSTER
&print
&print
io attach file05 vfile_ [response "enter name of data segment"]
io open file05 si
io attach file06 vfile_ results.cluster
io open file06 so
&print
&print
>udd>Trinity>RSpicer>magic_dir>cluster
io close file05
io close file06
io detach file05
io detach file06
&if [query "Do you wish to dprint the results? "]
&then dp results.cluster
&else pr results.cluster
&quit
```

cluster.fortran

```
c      cluster analysis
c
c      the program accepts an n by m data matrix where n is the number
c      of observations and m is the number of variables.  If the first
c      option on the control card is 1, an m by m matrix of similarities
c      between columns is computed.  if the option is 2, an n by n
c      matrix of similarities between rows is computed.  If the option
c      is 3, an m by m similarity matrix is accepted as input.  if
c      option two is 1, the correlation coefficient will be used in the
c      similarity matrix.  if this option is 2, the distance coefficient
c      will be used.  the program loops back and restarts after
c      completion.
c
c
c      subroutines  required are readm, printm, rcoef, dist, wpga,
c      and dendro
c
c      ----------------------------------------------------------------
      dimension x(350,350), ipair(2,350), xlev(350), a(350,350)
      dimension ititle(20)
      common/temp4/x
      common/temp5/ipair
      common/temp6/xlev
      common/temp2/a
      md=350
      nd=350
      mm=350
      write(0,1115)
      write(6,1115)
      write(6,1116)
 1116 format(/," this program performs a weighted average",
     1 lx,"cluster analysis",/," using either the correlation",
     2 lx,"coefficient or the distance coefficient",/," as",
     3 lx,"the measure of similarity.")
      write(0,1116)
 1115 format("   CLUSTER ANALYSIS - NUMERICAL DATA"/)
c
c...  read control card
c
    1 write (0,1111)
 1111 format (/," enter 0 for end of job", /,
     1 "       1 to input a data matrix",/,
     2  "       2 to input matrix and transpose it",/,
     3 "       3 to input similarity matrix")
      read (0,1112) itype
      if(itype.le.0) goto 3000
 1112 format (v)
      write (0,1113)
 1113 format(/," enter 1 for correlation coefficient",/,
     1 "       2 for distance coefficient")
      read (0,1112) isim
```

```
c
c...  input similarity matrix
c
      if (itype .ne.3) go to 2
      call readm(a,n,m,mm,mm,ititle)
      go to 4
c
c...  read and print input data matrix
c
    2 call readm(x,n,m,nd,md,ititle)
      write(0,1117)
 1117 format(/," enter 0 if you want input data matrix printed",
     1 /,"       1 if you don't.")
      read(0,1112) iprint
      write(6,1118)ititle
 1118 format(/,1x,20a4)
      if(iprint.eq.0) write(6,2001)
      if(iprint.eq.0) call printm(x,n,m,nd,md)
c
c...  transpose data matrix (if possible)
c
      if (itype .ne.2) go to 3
      mt=m
      if (n .gt.m) mt=n
      do 110 i=1,mt
      do 110 j=i,mt
      xs=x(i,j)
      x(i,j)=x(j,i)
      x(j,i)=xs
  110 continue
      mt=m
      m=n
      n=mt
c
c...  calculate similarity matrix
c
    3 if(isim.eq.1) call rcoef(n,m,nd,md,mm)
      if(isim.eq.2)call dist(n,m,nd,md,mm)
c
c     print similarity matrix
c
    4 write(0,1119)
 1119 format(/," enter 0 if you want similarity matrix printed",
     1 /, "       1 if you don't")
      read(0,1112) iwrite
      if(iwrite.ne.0) goto 5
      write(6,2002)
      if(isim.eq.1) write(6,2003)
      if(isim.eq.2) write(6,2004)
      call printm(a,m,m,mm,mm)
c
c     calculate and print linkage table
c
    5 write(6,2005)
```

```
      call wpga(m,mm,isim)
c
c     print dendrogram
c
      write(6,2006)
      call dendro(m,mm,isim)
 2001 format(1h0,4x,'input data matrix -',1x,
     1 'columns = variables,rows = observations')
 2002 format(1h0,4x,'similarity matrix')
 2003 format(1x,"using correlation coefficient")
 2004 format(1x,"using distance coefficient")
 2005 format(/,"     LINKAGE TABLE")
 2006 format(/,"     DENDOGRAM")
 3000 stop
      end
c      subroutine to perform weighted pair-group average clustering
c
      subroutine wpga(m,m1,isim)
      dimension x(350,350),ipair(2,350),xlev(350)
      dimension i1(350),i2(350),xsim(350)
      common/temp2/x
      common/temp5/ipair
      common/temp6/xlev
c
c      initialize
c
      write"(6,2001)
      do110 i=1,m
      i1(i)=i
  110 continue
      xxxx=-9.0e+35
      if(isim.ne.1)xxxx=+9.0e+35
      m3=m-1
      ic=0
c
c      for a correlation matrix find the largest similarity in each column
c
c      for a distance matrix find the smallest similarity in each column
c
    1 do100 i=1,m
      if(i1(i).le.0)goto 100
      ix=0
      xx=xxxx
      do101 j=1,m
      if(i.eq.j)goto101
      if(i1(j).le.0)goto101
      goto(11,12),isim
   11 if(x(j,i)-xx)101,101,13
   12 if(x(j,i)-xx)13,101,101
   13 xx=x(j,i)
      ix=j
  101 continue
      i2(i)=ix
      xsim(i)=xx
  100 continue
c
c      for a correlation matrix find mutualy high pairs
c      for a distance matrix find mutualy low pairs
c
      do 102 i=1,m3
      if(i1(i).le.0)goto102
      j=i2(i)
      if(i1(j).le.0) goto102
      if(j.le.i)goto102
      if(i1(j).eq.i)goto14
      if(abs(xsim(i)-xsim(j)).gt.0.00001)goto102
c
c      save parameters for a cluster
c
   14 ic=ic+1
      ipair(1,ic)=i
      ipair (2,ic)=j
      xlev(ic)=xsim(i)
      write(6,2002)i,j,xsim(i)
      i1(i)=j
      i1(j)=0
c
c      average the two columns
c
      do 103 k=1,m
      x(k,i)=(x(k,i)+x(k,j))/2.0
  103 continue
  102 continue
c
c      average rows that were clustered on this iteration
      do 105 i=1,m3
      if(i1(i).le.0)goto105
      if(i1(i).eq.i)goto105
      j=i1(i)
c
c      average two rows in the new cluster
c
      do106 k=1,m
      if(i1(k).le.0)goto106
      x(i,k)=(x(i,k)+x(j,k))/2.0
  106 continue
      i1(i)=i
  105 continue
      if(ic.lt.m3)goto1
      write(6,2003)
      return
 2001 format(/)
 2002 format(6x,2i5,f15.5)
 2003 format(1h0,4x,'columns 1 and 2 -',1x,'observations combined into
     1clusters',/,5x,'column 3 - similarity level of clustering')
      end
c      program 7.8
c
```

```
c       subroutine to print dendrogram
c
      subroutine dendro(m,ml,isim)
      dimension ipair (2,350), xlev(350)
      dimension il(350),i2(350)
      dimension iout(61),xx(13)
      common/temp5/ipair
      common/temp6/xlev
      data iblnk,ici,icp,icm/" ","i",".","-"/
c
c...  determine order that branches will be printed in
c
      m2=m-1
      do 100 i=1,m
      il(i)=0
      i2(i)=0
  100 continue
      do 101 i=1,m2
      j=i-1
   11 if (j .le. 0) go to 12
      if (ipair(1,i) .eq. ipair(1,j)) go to 13
      j=j-1
      go to 11
   12 i2 (i)=1
      go to 15
   13 k=il(j)
      if (k .eq. 0) go to 14
      j=k
      go to 13
   14 il(j)=i
   15 do 102 j=1,i
      k=j
      if ( ipair (2,i) .eq. ipair(1,j)) goto 16
  102 continue
      go to 101
   16 i2 (k) =0
      il(i)=k
  101 continue
c
c...  find starting cluster
c
      do 103 i=1,m2
      js=i
      if (i2(i) .ne. 0) go to 20
  103 continue
      goto  3000
   20 node=ipair(1,js)
c
c...  find largest and smallest similarity coef.
c
      xmin=xlev(1)
      xmax=xmin
      do 104 i=1,m2
      if (xlev(i) .lt. xmin) xmin=xlev(i)
      if (xlev(i) .gt. xmax) xmax=xlev(i)
  104 continue
      dx=(xmax-xmin)/25.0
      xmin=xmin-dx
      xmax=xmax+dx
      dx=(xmax-xmin)/60.0
      if (isim .ne. 2) go to 21
      dx=-dx
      xmin=xmax

c...  blank out print line array
c
   21 do 105 i=1,61
      iout(i)=iblnk
  105 continue
c
c...  print dendrogram
c
      x=xmin
      do 106 i=1,13
      xx(i)=x
      x=x+dx*5.0
  106 continue
      write (6,2000)
      write (6,2001) (xx(i),i=2,12,2)
      write (6,2002) (xx(i),i=1,13,2)
      write (6,2003)
   22 x=xmin
      if (js .ne. 0) x=xlev(js)
      is=ifix((x-xmin)/dx)+1
      do 110 i=is,61
      iout(i)=icm
  110 continue
      iout(is)=icp
      if (js .ne. 0) write (6,2004) iout,node,x
      if (js .eq. 0) write (6,2004) iout,node
      if (js .eq. 0) go to 31
      do 111 i=is,61
      iout(i)=iblnk
  111 continue
      iout(is)=ici
      write (6,2004) (iout(i), i=1, is)
      node=ipair(2,js)
      js=il(js)
      go to 22
   31 write (6,2003)
      write (6,2002)(xx(i),i=1,13,2)
      write  (6,2001) (xx(i),i=2,12,2)
      write (6,2005)
      return
 2000 format (/)
 2001 format (6x,6f10.4)
 2002 format (1x,7f10.4)
 2003 format (6x,'+',12('----+'))
 2004 format (6x,61a1,1x,i3,f10.4)
 2005 format (1h0,4x,'dendrogram - ',1x,
     1 'values along x-axis are similarities')
 3000 return
      end

c     program 7.9
c
c     subroutine to calculate the matrix of distance coefficients
c     between columns of data matrix x
c
      subroutine dist(n,m,n1,m1,m2)
      dimension x(350,350),a(350,350)
      common/temp4/x
      common/temp2/a
      an=n
c
c...  calculate distance coefficient between columns i and j
c
      do 100 i=1,m
      do 100 j=i,m
c
c...  zero sum and calculate distance
c
      distx=0.0
      do 101 k=1,n
      distx=distx+(x(k,i)-x(k,j))**2
  101 continue
c
c...  calculate distance coefficient and store in matrix a
c
      a(i,j)=sqrt(distx/an)
      a(j,i)=a(i,j)
  100 continue
      return
      end
c
c     subroutine to calculate the matrix of correlations
c     between columns of data matrix x
c
      subroutine rcoef(n,m,n1,m1,m2)
      dimension x(350,350),a(350,350)
      common/temp4/x
      common/temp2/a
      an=n
c
c     calculate correlation coefficient between columns iand j
c
      do100i=1,m
      do100j=i,m
c
c     zero sums
c
      sx1=0.0
      sx2=0.0
      sx1x1=0.0
      sx2x2=0.0
      sx1x2=0.0
c
c     calculate sums, sums of squares and sum of cross-product
c     of columns i and j
c
      do101k=1,n
      sx1=sx1+x(k,i)
      sx2=sx2+x(k,j)
      sx1x1=sx1x1+x(k,i)**2
      sx2x2=sx2x2+x(k,j)**2
      sx1x2=sx1x2+x(k,i)*x(k,j)
  101 continue
c
c     calculate correlation coefficient and store in a
c
      r=(sx1x2-sx1*sx2/an)/
     1sqrt((sx1x1-sx1*sx1/an)*(sx2x2-sx2*sx2/an))
      a(i,j)=r
      a(j,i)=r
  100 continue
      return
      end
c
c     subroutine to read a matrix
c     having n rows and m columns
c
      subroutine readm(a,n,m,n1,m1,ititle)
      dimension a(n1,m1),ititle(20)
      read(5,1004)ititle
 1004 format(20a4)
c     read size of matrix
c
      read (5,1001)m,n
c
c     read matrix one row at a time
c
      do100i=1,n
      read(5,1001)(a(i,j),j=1,m)
  100 continue
 1001 format(v)
      return
      end
c
c     subroutine to print a matrix having n rows and m columns
c
      subroutine printm(a,n,m,n1,m1)
      dimensiona(n1,m1)
c     print matrix out in strips of ten columns
      do100ib=1,m,10
      ie=ib+9
      if(ie-m)2,2,1
```

```
    1 ie=m
cprint heading
    2 write(6,2000)(i,i=ib,ie)
      do101j=1,n
c      print row of matrix
      write(6,2001)j,(a(j,k),k=ib,ie)
  101 continue
  100 continue
      return
 2000 format(/,1x,10i12)
 2001 format(1x,i5,10f12.4)
      end
```

willi.ec

```
&command_line off
&print
&print
&print WILLI
&print
&print
io attach file07 vfile_ [response "enter name of data segment "]
io open file07 si
io attach file08 vfile_ [response "enter name of output segment "]
io open file08 so
>udd>Trinity>RSpicer>magic_dir>willi
io close file07
io close file08
io detach file08
io detach file07
&quit
```

willi.fortran

```
      dimension nj(50),mj(50),ititle(20),x(350,350)
      common/a/ x
      kedflag = 0
      write(0,8)
    8 format("Enter 1 for printout of raw data else type 0")
      read(0,1)krawsig
      read(7,9) ititle
    9 format(20a4)
      read(7,1) m,n
      kj=0
      write(0,201)
  201 format("enter 1 for raw data"/6x,"2 for percent species transform"
     1/6x,"3 for log transform"/6x,"4 for presence-absence")
  205 read(0,1) kflag
      if(kflag.eq.1.or.kflag.eq.2.or.kflag.eq.3.or.kflag.eq.4)goto 203
      write(0,204)
  204 format("please enter 1,2,3 or 4 only")
      goto 205
  203 goto(300,301,302,303) kflag
  300 write(8,310) ititle
  310 format("raw data ",20a4)
      goto 312
  301 write(8,311) ititle
  311 format("percent transform ",20a4)
      goto 312
  302 write(8,313) ititle
  313 format("log transform ",20a4)
      goto 312
  303 write(8,314) ititle
  314 format("presence-absence ",20a4)
  312 do 2 k=1,n
    2 read(7,1)(x(k,j),j=1,m)
      if(krawsig.eq.0)goto 315
      call putout(m,n)
  315 do42k=1,n
      krdflag=0
      jsum=0
      do 40j=1,m
   40 jsum=jsum+x(k,j)
      if(jsum.gt.0)goto42
      write(0,41)k
   41 format("***WARNING***"/"sample ",i3," is barren - it has bee
     1n deleted"/)
      krdflag=1
      call edit(mj,nsig,m,n,krdflag,k,kedflag)
   42 continue
      do 43j=1,m
      krdflag=0
      ksum=0
      do44k=1,n
   44 ksum=ksum+x(k,j)
```

```
      if(ksum.gt.0)goto 43
      write(0,46)j
   46 format("***WARNING***"/"species ",i3," is non existant in
     1 this data matrix - it will be deleted")
      krdflag=1
      call edit(nj,nsig,m,n,krdflag,j,kedflag)
   43 continue
      do 11 k=1,n
      xsum=0.
      if(kflaq.1) goto 11
      do 5 j=1,m
      if(kflag.eq.2) goto 7
      if(kflag.eq.4) goto 36
      x(k,j)=alog(x(k,j)+1)
      goto 5
   36 if(x(k,j).gt.0) x(k,j)=1
      x(k,j)=x(k,j)+1
      goto 5
    7 xsum=xsum+x(k,j)
    5 continue
      if(kflag.ne.2) goto 11
      do 6 j=1,m
    6 x(k,j)=x(k,j)/xsum*100.
   11 continue
      call putout(m,n)
      nsig=0
      krdflag=0
      call edit(nj,nsig,m,n,krdflag,k,kedflag)
      nsig=1
      call edit(mj,nsig,m,n,krdflag,k,kedflag)
   26 write(8,29) m,n
   29 format(2i3)
  412 do 28 k=1,n
      do 28 j=1,m
   28 write(8,30) x(k,j)
   30 format(f20.8)
      if (kedflag.lt.1) goto 414
      goto (400,401,402,403)kflag
  400 write(0,310)ititle
      goto413
  401 write(0,311)ititle
      goto413
  402 write(0,313)ititle
      goto413
  403 write(0,314)ititle
  413 call putout(m,n)
    1 format(v)
  414 continue
      stop
      end

      subroutine putout(m,n)
      dimension x(350,350)
      common/a/x
    write(0,100)
100 format(/"sample                                        species"/)
    i=1
 15 if((m-i).le.9) goto 22
    write(0,31) (j,j=i,(i+9))
 31 format(4x,10(6x,i4))
    do 12 k=1,n
 12 write(0,4) k,(x(k,j),j=i,(i+9))
    write(0,100)
    i=i+10
    goto 15
 22 write(0,31) (j,j=i,m)
    do 13 k=1,n
 13 write(0,4) k,(x(k,j),j=i,m)
  4 format(i4,1x,10f10.4,1x)
    return
    end

    subroutine edit(nj,nsig,m,n,krdflag,kb,kedflag)
    dimension x(350,350)
    common/a/x
    dimension nj(50)
    if(krdflag.lt.1)goto7
    kj=1
    nj(kj)=kb
    goto8
  7 kj=0
    if(nsig.eq.1) goto 1
    write(0,2)
  2 format(/"enter species to fe omitted ")
    goto 3
  1 write(0,4)
  4 format(/"enter samples to be omitted ")
  3 read(0,5) ni
  5 format(v)
    if (ni.gt.0) kedflag = 1
    if(ni.eq.0) goto 24
    kj=kj+1
    nj(kj)=ni
    goto 3
 24 if(kj.eq.0) goto 26
  8 do 25 k=1,n
    lm=0
    do 25 j=1,m
    do 62 l=1,kj
    if(nsig.eq.1) goto 40
    if(j.eq.nj(l)) goto 27
    goto 62
 40 if(k.eq.nj(l)) goto 27
 62 continue
    goto 61
 27 lm=lm+1
    goto 25
 61 if(nsig.eq.1) goto 30
```

```
   x(k,(j-lm))=x(k,j)
   goto 25
30 x((k-lm),j)=x(k,j)
25 continue
   if(nsig.eq.1) goto 43
   m=m-kj
   goto 26
43 n=n-kj
26 return
   end
```

smooth.ec

```
&command_line off
&print
&print
&print SMOOTH
&print
&print
io attach file05 vfile_ [response "enter name of data segment"]
io open file05 si
io attach file06 vfile_ results.smooth
io open file06 so
>udd>Trinity>RSpicer>magic_dir>smooth
io close file05
io close file06
io detach file05
io detach file06
&if [query "Do you wish to dprint the results? "]
&then dp results.smooth
&else pr results.smooth
&quit
```

smooth.fortran

```
c     program smooth
c
c     routine to perform m term smoothing
c
c     xin is data sequence of length n to be smoothed by
c     m-term moving average. length of output is ie=n-m+1.
c     smothed sequence is xout.
c     xout(i) = the smoothed estimate for xin(i+(m-1)/2).
c     m must be an odd number.
c     ====================================================
c
c
      dimension xin(350),xout(350),ititle(20)
c
c     read in the number of terms to be used in the moving average.
c
      write(0,1010)
      read(0,1000) mterm
      nm=1
      write(6,2005) mterm

c
c     read in the data sequence to be smoothed and print it out.
c
      call readm(xin,nn,nm,ititle)
      write(6,2004) ititle
      write(6,2000)
      call printm(xin,nn,1)
      n=nn
      ie=n-mterm+1
      do 100 i=1,ie
      sum=0.0
      do 101 j=1,mterm
      k=i+j-1
      sum=sum+xin(k)
  101 continue
      xout(i)=sum/float(mterm)
  100 continue
c
c     print out the smoothed data sequence.
c
      write(6,2001)
      call printm(xout,ie,1)
      sy=0.0
      syy=0.0
      do 102 i=1,n
      sy=sy+xin(i)
      syy=syy+xin(i)**2
  102 continue
      sys=0.0
      syys=0.0
      syc=0.0
      syyc=0.0
      ssd=0.0
      do 103 i=1,ie
      j=i+m/2
      sys=sys+xin(j)
      syys=syys+xin(j)**2
      syc=syc+xout(i)
      syyc=syyc+xout(i)**2
      ssd=ssd+(xin(j)-xout(i))**2
  103 continue
      sso=syy-sy*sy/float(n)
      ssos=syys-sys*sys/float(ie)
      sss=syyc-syc*syc/float(ie)
      pss=(sss/ssos)*100.0
 2005 format(/," this program performs a",i3,"-term moving",
     1 1x,"average smoothing operation on the data sequence")
 2004 format(/,1x,20a4)
      write(6,1000)
      write(6,1001) sso
      write(6,1002) ssos
      write(6,1003) sss
      write(6,1004) ssd
      write(6,1005) pss
      itype=2
      write(6,2003)
      call tsplot(xin,n,1,itype)
      write(6,2002)
      call tsplot(xout,ie,1,itype)
      stop
 1000 format(v)
 1001 format(/,"  sums of squares of original data = ",f21.8)
 1002 format(1x," sums of squares of truncated data = ",f20.8)
 1003 format(1x," sums of squares of smoothed data = ",f21.8)
 1004 format(1x," sums of squares due to deviation = ",f21.8)
 1005 format(1x," goodness of fit percentage = ",f27.8)
 1010 format(/," enter number of terms to be used in the running",
     c 1x,"average",1h ,"   [3 is the most common]")
 1011 format(/," enter the number of samples")
 2000 format(/," input data sequence")
 2001 format(/," smoothed data sequence")
 2002 format(/," smoothed data curve")
 2003 format(/," input data curve")
      end

c     subroutine to read a matrix
      subroutine readm(a,n,m,ititle)
      dimension a(350,350),ititle(20)
      read(5,1001) ititle
      read(5,1000) n
      read(5,1000) (a(j,1),j=1,n)
 1000 format(v)
 1001 format(20a4)
      return
      end

c     subroutine to print a matrix
      subroutine printm(a,n,m)
      dimension a(350,350)
c     print matrix out in strips of 10 columns
      do 100 ib=1,m,10
      ie=ib+9
      if(ie-m) 2,2,1
    1 ie=m
c     print heading
    2 write(6,2000) (i,i=ib,ie)
      do 101 j=1,n
c     print row of matrix
      write(6,2001) j,(a(j,k),k=ib,ie)
  101 continue
  100 continue
      return
 2000 format(/,1x,10i12)
 2001 format(1x,i5,10f12.4)
      end

c     subroutine to plot one-dimensional data on line printer
c
      subroutine tsplot(a,n,m,itype)
c
c     m is the column in array a to be plotted.
c     n is the number of elements to be plotted.
c
c     if itype=1, the data to be plotted will have range (-1,1).
c     if itype=2, any data may be plotted.
c     if itype=3, log10 of the column m values will be plotted. the
c      original data will not be destroyed.
c      ============================================================
c
      dimension a(n,m),x(250),iout(61),xx(13)
      data ii,istar,iblank/"i","*"," "/
      do 90 i=1,n
      x(i)=a(i,m)
   90 continue
      if(itype.ne.1) goto 11
      xmin=-1.0
      xmax=1.0
      goto 12
   11 xmin=x(1)
      xmax=xmin
      do 100 i=1,n
      if(x(i).lt.xmin) xmin=x(i)
      if(x(i).gt.xmax) xmax=x(i)
  100 continue
      if(itype.ne.3) goto 12
      xmin=alog10(xmin)
      xmax=alog10(xmax)
   12 dx=xmax-xmin
```

```
      xxx=xmin
      do 101 i=1,13
      xx(i)=xxx
      if(itype.eq.3) xx(i)=10.0**xxx
      xxx=xxx+dx/12.0
 101  continue
      write(6,2004)
      write(6,2000) (xx(i),i=2,12,2)
      write(6,2001) (xx(i),i=1,13,2)
      write(6,2002)
      do 102 i=1,n
      do 103 j=1,61
      iout(j)=iblank
 103  continue
      do 104 j=1,61,10
      iout(j)=ii
 104  continue
      xxx=x(i)
      if(itype.eq.3) xxx=alog10(xxx)
      ix=ifix((xxx-xmin)*60.0/dx)+1
      iout(ix)=istar
      write(6,2003) x(i),iout
 102  continue
      write(6,2002)
      write(6,2001) (xx(i),i=1,13,2)
      write(6,2000) (xx(i),i=2,12,2)
      return
2000  format(11x,6f10.4)
2001  format(6x,7f10.4)
2002  format(11x,"+",12("----+"))
2003  format(1x,f10.4,61a1)
2004  format(/)
      end
```

Example of construction and analysis of a test data matrix within the MAGIC environment.

```
qx
a
test data for MAGIC program examples
10,10
50 250 150 50 50 200 100 50 50 50
100 300 170 170 80 80 50 40 10 0
30 60 100 130 250 150 130 80 50 20
75 275 160 110 65 140 75 45 30 25
46 212 140 66 90 190 106 56 50 44
38 136 120 98 170 170 118 68 50 32
83 266 159 142 91 111 68 46 22 12
61 227 146 102 99 154 91 53 38 29
76 242 152 138 108 118 76 50 26 14
39 103 112 126 213 148 119 73 46 21
\f
1,$w test_data
q
r 1034 0.345 5.746 282
```

```
ec magic
MAGIC - USGS Multivariate Statistical

Programs Package 1.0

====================================


do you require help?   no

do you wish to quit magic?   no

enter program name   willi


WILLI



enter name of data segment   test_data

enter name of output segment   test_data_percent
Enter 1 for printout of raw data else type 0
1
enter 1 for raw data
      2 for percent species transform
      3 for log transform
      4 for presence-absence
2

sample                                    species

            1         2         3         4         5         6         7         8         9        10
   1    50.0000  250.0000  150.0000   50.0000   50.0000  200.0000  100.0000   50.0000   50.0000   50.0000
   2   100.0000  300.0000  170.0000  170.0000   80.0000   80.0000   50.0000   40.0000   10.0000    0.0000
   3    30.0000   60.0000  100.0000  130.0000  250.0000  150.0000  130.0000   80.0000   50.0000   20.0000
   4    75.0000  275.0000  160.0000  110.0000   65.0000  140.0000   75.0000   45.0000   30.0000   25.0000
   5    46.0000  212.0000  140.0000   66.0000   90.0000  190.0000  106.0000   56.0000   50.0000   44.0000
   6    38.0000  136.0000  120.0000   98.0000  170.0000  170.0000  118.0000   68.0000   50.0000   32.0000
   7    83.0000  266.0000  159.0000  142.0000   91.0000  111.0000   68.0000   46.0000   22.0000   12.0000
   8    61.0000  227.0000  146.0000  102.0000   99.0000  154.0000   91.0000   53.0000   38.0000   29.0000
   9    76.0000  242.0000  152.0000  138.0000  108.0000  118.0000   76.0000   50.0000   26.0000   14.0000
  10    39.0000  103.0000  112.0000  126.0000  213.0000  148.0000  119.0000   73.0000   46.0000   21.0000

sample                                    species

            1         2         3         4         5         6         7         8         9        10
   1     5.0000   25.0000   15.0000    5.0000    5.0000   20.0000   10.0000    5.0000    5.0000    5.0000
   2    10.0000   30.0000   17.0000   17.0000    8.0000    8.0000    5.0000    4.0000    1.0000    0.0000
   3     3.0000    6.0000   10.0000   13.0000   25.0000   15.0000   13.0000    8.0000    5.0000    2.0000
   4     7.5000   27.5000   16.0000   11.0000    6.5000   14.0000    7.5000    4.5000    3.0000    2.5000
   5     4.6000   21.2000   14.0000    6.6000    9.0000   19.0000   10.6000    5.6000    5.0000    4.4000
   6     3.8000   13.6000   12.0000    9.8000   17.0000   17.0000   11.8000    6.8000    5.0000    3.2000
   7     8.3000   26.6000   15.9000   14.2000    9.1000   11.1000    6.8000    4.6000    2.2000    1.2000
   8     6.1000   22.7000   14.6000   10.2000    9.9000   15.4000    9.1000    5.3000    3.8000    2.9000
   9     7.6000   24.2000   15.2000   13.8000   10.8000   11.8000    7.6000    5.0000    2.6000    1.4000
  10     3.9000   10.3000   11.2000   12.6000   21.3000   14.8000   11.9000    7.3000    4.6000    2.1000

enter species to be omitted
0

enter samples to be omitted
0
```

```
do you wish to quit magic?   no

enter program name   pca


PCA



enter name of data segment   test_data_percent


   PRINCIPAL COMPONENTS ANALYSIS

enter 1 for no transposition of data matrix
           2 for transpose of data matrix
2


Enter 1 for correlation matrix
      2 for covariance  matrix
      3 for cos theta transform

3
for row-wise normalization type 1 else type 0
0
for zero mean transformation type 1 else type 0
0

 enter 0 if you want a plot of the first three principal component axes
       1 if you do not.
1

STOP

Do you wish to dprint the results?   no

                   results.pca        01/03/80  1038.1 pst Thu



percent transform test data for MAGIC program examples


   input data matrix - columns = variables, rows = observations
          1           2           3           4           5           6           7           8           9          10
   1    5.0000     25.0000     15.0000      5.0000      5.0000     20.0000     10.0000      5.0000      5.0000      5.0000
   2   10.0000     30.0000     17.0000     17.0000      8.0000      8.0000      5.0000      4.0000      1.0000      0.0000
   3    3.0000      6.0000     10.0000     13.0000     25.0000     15.0000     13.0000      8.0000      5.0000      2.0000
   4    7.5000     27.5000     16.0000     11.0000      6.5000     14.0000      7.5000      4.5000      3.0000      2.5000
   5    4.6000     21.2000     14.0000      6.6000      9.0000     19.0000     10.6000      5.6000      5.0000      4.4000
   6    3.8000     13.6000     12.0000      9.8000     17.0000     17.0000     11.8000      6.8000      5.0000      3.2000
   7    8.3000     26.6000     15.9000     14.2000      9.1000     11.1000      6.8000      4.6000      2.2000      1.2000
   8    6.1000     22.7000     14.6000     10.2000      9.9000     15.4000      9.1000      5.3000      3.8000      2.9000
   9    7.6000     24.2000     15.2000     13.8000     10.8000     11.8000      7.6000      5.0000      2.6000      1.4000
  10    3.9000     10.3000     11.2000     12.6000     21.3000     14.8000     11.9000      7.3000      4.6000      2.1000

   similarity matrix
          1           2           3           4           5           6           7           8           9          10
   1    1.0000      0.8739      0.6906      0.9654      0.9888      0.8849      0.9244      0.9714      0.9262      0.7878
   2    0.8739      1.0000      0.6480      0.9704      0.8733      0.8020      0.9901      0.9391      0.9779      0.7531
   3    0.6906      0.6480      1.0000      0.6906      0.7908      0.9479      0.7227      0.7941      0.7758      0.9877
   4    0.9654      0.9704      0.6906      1.0000      0.9595      0.8697      0.9902      0.9862      0.9855      0.7952
   5    0.9888      0.8733      0.7908      0.9595      1.0000      0.9445      0.9315      0.9660      0.9457      0.8705
  6       0.8849      0.8020      0.9479      0.8697      0.9445      1.0000      0.8723      0.9390      0.9082      0.9828
  7       0.9244      0.9901      0.7227      0.9902      0.9315      0.8723      1.0000      0.9779      0.9967      0.8200
  8       0.9714      0.9391      0.7941      0.9862      0.9860      0.9390      0.9779      1.0000      0.9867      0.8790
  9       0.9282      0.9779      0.7758      0.9855      0.9457      0.9082      0.9967      0.9867      1.0000      0.8636
 10       0.7878      0.7531      0.9877      0.7952      0.8705      0.9828      0.8200      0.8790      0.8636      1.0000

 column 1 - eigenvalues   ,column 2 - percent of trace
  column 3 - cumulative percent of trace
              1           2           3
  1       9.0517     90.5166     90.5166
  2       0.7420      7.4200     97.9365
  3       0.2063      2.0635    100.0000
  4       0.0000      0.0000    100.0000
  5       0.0000      0.0000    100.0000
  6       0.0000      0.0000    100.0000
  7       0.0000      0.0000    100.0000
  8       0.0000      0.0000    100.0000
  9      -0.0000     -0.0000    100.0000
 10      -0.0000     -0.0000    100.0000

 principal axiamatrix - columns - eigenvectors, rows - variables
              1           2           3           4           5           6           7           8           9          10
  1       0.3159     -0.1951     -0.5749     -0.1053     -0.2310     -0.0034     -0.2657     -0.4870     -0.0452     -0.3967
  2       0.3094     -0.3323      0.4990     -0.3264     -0.0514      0.1344     -0.3275      0.3610      0.0205     -0.4238
  3       0.2798      0.6204      0.1670     -0.3968      0.0820     -0.5397     -0.0434     -0.1331     -0.1775     -0.0494
  4       0.3229     -0.2751     -0.0180     -0.3764      0.2512      0.2831      0.0725     -0.1600     -0.4279      0.5664
  5       0.3251     -0.0372     -0.4521      0.1401      0.1528     -0.2909     -0.3254      0.5935      0.1559      0.2801
  6       0.3193      0.3136     -0.1456     -0.3013     -0.2225      0.4265      0.3901      0.1250      0.5356      0.0411
  7       0.3232     -0.2288      0.2774      0.2559     -0.6762     -0.3263      0.1492     -0.1040      0.0305      0.3210
  8       0.3311     -0.0891     -0.0987      0.2725      0.1957     -0.0824      0.6462      0.2188     -0.3632     -0.3950
  9       0.3279     -0.1374      0.2482      0.2746      0.5489     -0.1000     -0.0082     -0.4022      0.5130      0.0181
 10       0.3044      0.4605      0.1361      0.5077     -0.0660      0.4699     -0.3422     -0.0287     -0.2751      0.0141

 principal component scores - columns - eigenvectors, rows - observations
              1           2           3           4           5           6           7           8           9          10
  1      19.0251     -5.1711      3.9657     -0.4518     -0.1857     -0.0252     -0.4218      0.0831     -0.1469     -0.2292
  2      66.1572    -21.9046      2.0858     -1.4188     -0.6439     -0.0051     -1.4144     -0.0910     -0.6104     -0.6980
  3      44.7126     -5.4012      1.4313     -0.8667     -0.3199     -0.2380     -0.8529     -0.1446     -0.4426     -0.4010
  4      35.6308      0.9901     11.2425     -0.8034     -0.2332     -0.3086     -0.7039      0.1957     -0.2727     -0.3907
  5      37.7468     20.4441      5.7583     -0.5599      0.0173     -0.8188     -0.4827     -0.1837     -0.4103     -0.1999
  6      46.1773      4.7998     -9.8150     -0.6087     -0.1547     -0.5194     -0.6749     -0.6422     -0.5988     -0.2117
  7      29.3045      7.7640     -3.2338     -0.3887     m0.0531     -0.4434     -0.4009     -0.3434     -0.3664     -0.1317
  8      17.6037      4.4943     -0.1577     -0.2643     -0.0416     -0.2598     -0.2573     -0.1415     -0.2024     -0.1000
  9      11.7008      3.2697     -3.0761     -0.1244     -0.0115     -0.1836     -0.1437     -0.2019     -0.1640     -0.0317
 10       7.8450      0.6589     -4.3604     -0.0610     -0.0167     -0.0878     -0.0944     -0.2032     -0.1271     -0.0075


do you wish to quit magic?   no
```

```
enter"program name   cluster

CLUSTER


enter name of data segment   test_data_percent


   CLUSTER ANALYSIS - NUMERICAL DATA


 this program performs a weighted average cluster analysis
 using either the correlation coefficient or the distance coefficient
 as the measure of similarity.

 enter 0 for end of job
       1 to input a data matrix
       2 to input matrix and transpose it
       3 to input aimilarity matrix
2

 enter 1 for correlation coefficient
       2 for distance coefficient
2

 enter 0 if you want input data matrix printed
       1 if you don't.
1

 enter 0 if you want similarity matrix printed
       1 if you don't
1

STOP

                     results   01/03/80  1043.1 pst Thu

  CLUSTER ANALYSIS - NUMERICAL DATA


this program performs a weighted average cluster analysis
using either the correlation coefficient or the distance coefficient
as the measure of similarity.

percent transform test data for MAGIC program examples

    LINKAGE TABLE


          1    5        1.90368
          3   10        1.91050
          7    9        1.06207
          3    6        2.95653
          4    7        1.91454
          4    8        2.19937
          1    4        3.16238
          1    2        5.09753
          1    3        7.49312
    columns 1 and 2 - observations combined into  clusters
    column 3 - similarity level of clustering

    DENDOGRAM

         7.1716    6.0140    4.8564    3.6988    2.5412    1.3836
    7.7504    6.5928    5.4352    4.2776    3.1200    1.9624    0.8048
     +----+----+----+----+----+----+----+----+----+----+----+----+
                                                       .----------   1    1.9037
                                                       i
                                            .---------------------   5    3.1624
                                            i
                                            i          .----------   4    1.9145
                                            i          i
                                            i          i      .---   7    1.0621
                                            i          i      i
                                            i       .-------------   9    2.1994
                                            i       i
                               .--------------------------------------   8    5.0975
                               i
           .----------------------------------------------------------   2    7.4931
           i
           i                                           .----------   3    1.9105
           i                                           i
           i                                      .-------------------  10    2.9565
           i                                      i
          .-----------------------------------------------------------   6
     +----+----+----+----+----+----+----+----+----+----+----+----+
    7.7504    6.5928    5.4352    4.2776    3.1200    1.9624    0.8048
         7.1716    6.0140    4.8564    3.6988    2.5412    1.3836

    dendrogram -  valuea along x-axis are similarities


do you wish to quit magic?   yes
r 1044 5.660 90.020 1519
```