PRENTICE COMPUTER CENTRE

University of Queensland

TECHNICAL MANUAL No. 3

STATISTICAL PACKAGES

MNT-3
August 1981

This manual has been authorized by the Director of the Prentice Computer Centre

TECHNICAL MANUAL NUMBER 3

STATISTICAL PACKAGES

Revised Edition: 1981

C. McGovern
B. Maher

Prentice Computer Centre

University of Queensland

Australia

CONTENTS

CHAPTER 1

PREPARING DATA FOR ANALYSIS

The data preparation phase of any statistical analysis project involving a computer is probably the most important phase, but unfortunately the one which receives least attention and is most likely to suffer first from any time schedule which may be imposed. Time spent in checking data (either manually, or with the computer), and arranging it in a convenient and easily handled form, can save many hours in both human and computer processing time at a later stage in the project.

This section contains hints on data preparation and checking, as well as references to other literature on the subject. Emphasis will be placed on "social science" type analysis, and in particular, the processing of survey data and questionnaires. "Data preparation" in this context refers to the punching of data onto 80 column cards. It is assumed that the actual punching will be carried out by the Computer Centre data preparation service or some other such body.

A "data entry" facility whereby data can be directly entered onto magnetic disk is now available and will be soon be the standard method of entering data onto the system (See **1.3.6** *Magnetic Disk*).

1.1 Questionnaire Design

The specific theory and details of questionnaire design are adequately covered elsewhere (see Moser and Kalton). What will be covered here are the aspects of questionnaire design specifically relating to data preparation and computer processing.

The process of moving data from the point of collection to actual analysis is such that there is a great potential for the introduction of error. Naturally, any transcription process involving human participation increases this potential, and it should be an aim in the design of any questionnaire to minimize the frequency of such human intervention.

It is to this end, that the layout of a questionnaire should be given careful consideration. Naturally, if the questionnaire is to be completed by the respondent and not by an interviewer, then the design of the questionnaire should, as much as possible, facilitate the complete and accurate answering by the respondent. It is possible to achieve this, and still have a layout conducive to data preparation.

If data can be transferred into machine readable form (e.g. punched cards) directly from the source document, then time can be saved, and the risk of human error reduced.

1.1.1 *Pre-coded Answers*

The simplest way to facilitate data preparation is to use *precoded* answers to questions, and there are many ways of achieving this (see Moser and Kalton Ch. 13). e.g.

(A) DO YOU OWN A MOTOR CAR? (Please put X in appropriate box)

YES	☒	1	(26)
NO	☐	2	

(B) IN WHAT STATE OF AUSTRALIA WERE YOU BORN?
(Write appropriate number in box)

1. QUEENSLAND	5. S. AUSTRALIA	
2. NEW SOUTH WALES	6. W. AUSTRALIA	☐ (27)
3. VICTORIA	7. N. TERRITORY	
4. TASMANIA	8. NOT BORN IN AUSTRALIA	

(C) HOW DO YOU NORMALLY GET TO WORK? (OFFICE USE ONLY)
(Please circle)

1. ON FOOT	5. PUBLIC TRANSPORT	
2. BICYCLE	6. TAXI	☐ (28)
3. MOTOR CYCLE	7. NOT APPLICABLE	
4. CAR		

In example (A) the respondent is required to place "X" in the appropriate box. Each box has a number beside it, which is the figure actually punched onto the card.

In example (B), the respondent is required to write the appropriate code for his answer into the box provided.

In example (C), the respondent circles the appropriate answer, the code for which is later transcribed into a box in the right margin.

1.1.2 *Descriptive Answers*

Descriptive answers may not be directly processed by the computer, but may still be essential to the questionnaire. If processing is required, then they should be quantified in some way at a later stage, e.g.

(1) WHAT DO YOU THINK ABOUT LOWERING THE MINIMUM AGE FOR OBTAINING A DRIVING LICENCE FROM 17 YEARS TO 15 YEARS? (OFFICE USE ONLY)

..... ☐ (24)

In this example the respondent is required to express an opinion in the space provided. At a later stage, a scrutineer may interpret this response according to the following categories, and then enter a code into the space provided.

1. Strongly agree
2. Agree
3. Don't care
4. Disagree
5. Strongly Disagree
6. Did not answer.

The respondent, on the other hand, may be asked to rate his own opinion on a ten point scale, e.g.

0 1 2 3 4 5 6 7 8 9 10

STRONGLY DISAGREE — DON'T CARE — STRONGLY AGREE ☐ (24)

1.1.3 *User-Coded Numeric Responses*

Many data preparation errors result from this type of question, and where possible, they should be avoided.

For instance, instead of:

(1) WHAT IS YOUR WEEKLY INCOME IN DOLLARS? ☐☐☐ (24–27)

use

(2) IN WHICH GROUP DOES YOUR INCOME LIE?
- *(a)* UP TO $50
- *(b)* $50 to $100
- *(c)* $100 to $200
- *(d)* OVER $200

☐ (23)

If (1) is used, respondents may try to enter income as dollars and cents, in which case insufficient space is provided; also, they may left justify their numbers in the space provided, rather than right justify, i.e.

[20] instead of [20]

Generally speaking, if classes can be provided with meaningful intervals, such as in the second example, they should be used. If this is not possible, then the actual coding should be left to scrutineers.

1.1.4 *Hints on Questionnaire Design*

1. Where possible, use numeric codes rather than letters, e.g.

   use

   YES 0 ☐ (25)
   NO 1 ☐

   rather than

   YES Y ☐ (25)
   NO N ☐

   This is because computers generally manipulate numbers better than characters, and also because data preparation is quicker for all numerics than for a mixture of alphabetics and numerics.

2. If respondents are to write codes in a box provided ensure there are sufficient positions for the largest code.

3. Indicate the position of each coded response on the punched card (or other recording medium), e.g.

   YES 0 ☐ (64)
   NO 1 ☐

   —the coded answer for this question will be punched in column 64 on the card.

4. If respondents are to complete the questionnaire, ensure that spaces for answers are as close to the right hand side of the page as possible so that the data preparation assistant can scan straight down the page.

5. If actual coding of the answers is to be carried out by a scrutineer on the questionnaire form, ensure that an adequate margin is left on the right-hand side of the page.

6. If the questionnaire is in the form of sheets of paper pinned in the top left corner, print only on one side of the paper as such forms are difficult to handle. Questionnaires in booklet form may be printed on both sides of the paper.

7. Always manually check completed questionnaires before they are submitted for data preparation (See next section).

8. If in doubt, consult the Computer Centre.

9. Ensure with precoded answers that all possible responses are provided for, e.g. it should be possible to answer all questions, even if the answer is "don't know", "not applicable", etc.

1.2 Data Checking

Whether the data is to be punched directly from the source document, or from some intermediate document, the same checking procedures should be observed at all stages. This checking at least in the first case, will have to be done manually by scrutineers. However, as soon as the data is in a machine readable form, the checking may be performed using the computer.

Basically, data should be checked for completeness and accuracy.

1.2.1 *Completeness*

It is important to check first that the data is complete. Incomplete data can be a problem particularly when the respondents themselves actually fill out and code the questionnaire. Incomplete questionnaires should be set aside by scrutineers and some prescribed course of action taken. It is not the job of data preparation assistants to make assumptions about unanswered questions. If necessary, a code for "did not answer" should be provided, as most statistical analysis techniques make allowance for missing values.

1.2.2 *Accuracy*

It is not enough to check that all questions have been answered; as far as possible data should be checked for consistency and accuracy. This type of check is probably best carried out using the computer once the data is in machine readable form. Useful accuracy checks include:

(1) Questions which are to be answered only on the basis of answers to previous questions should be checked carefully, e.g.

(a) DO YOU SMOKE CIGARETTES? YES 0 ☐ (64)
NO 1 ☐

(b) IF YOU ANSWERED "YES" TO QUESTION (a), HOW MANY CIGARETTES DID YOU SMOKE YESTERDAY?
1. NONE
2. LESS THAN 10
3. 10 OR MORE BUT LESS THAN 20 □ (65)
4. 20 OR MORE
5. NOT APPLICABLE

You should check that if the respondent answered "no" to the first question, then he answered "not applicable" to the next; any other answer would be inconsistent, and would warrant further checking.

(2) Range checks should be made on all data items. This will also help to detect any punching errors which may not have been discovered.

Ensure that all precoded answers are within the bounds of the codes provided, e.g.

For yes/no answers, where "yes" = 0 and "no" = 1, check that none of the codes are less than 0 or greater than 1.

For answers which require users to enter amounts which are not precoded (age, income, etc), determine reasonable upper and lower bounds on these amounts, and carefully look at values which do not lie in these bounds, e.g.

If surveying first year students at university, ensure that the age given lies in a range of about 16 years to 40 years. Any answer given outside this range should be checked to ensure its validity.

1.3 Preparing Coding Sheet

Unless questionnaires have been designed with the intention of keypunching directly from them, it is advisable that the results be encoded onto special coding forms. These may be of the standard 80 column type, available from the Computer Centre or they may be designed for a special application. Users wishing to design their own special coding forms should consult with the Computer Centre beforehand.

When preparing coding sheets, the following rules should be observed:

(1) Use ink or biro rather than pencil. If using pencil, use a soft grade (2B).

(2) If a mistake is made in a line, cross out the whole line, rather than trying to make untidy corrections.

(3) Ensure that there is no ambiguity of characters. Observe the following conventions:
O—letter "Oh"
I—letter I
Z—letter Z
0—zero
1—one

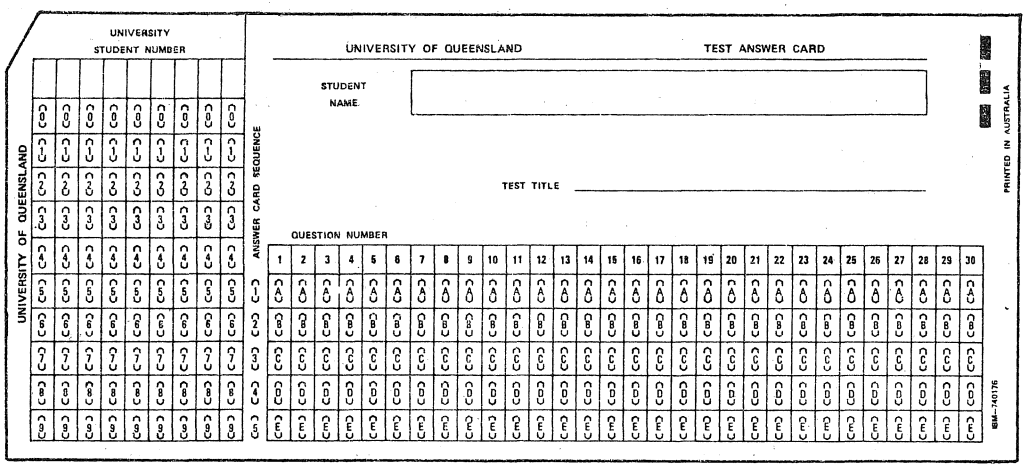
(4) Take care in coding as keypunch operators will punch exactly what appears on the coding sheet and will not make any assumptions or corrections, no matter how obvious any errors may appear.

1.4 Storing Data in Machine-readable Form

There are a number of ways of storing data in a form which can be read directly by the computer. The normal method, which has been mentioned before, is to punch data onto 80 column computer cards. The data may then be transferred onto any one of the other storage media via this stage. A data preparation service which enters data directly onto magnetic disk, is also available.

1.4.1 *Punched Cards*

Punched cards are the traditional form of data storage, and are still widely used. They have 80 columns and 12 rows, and so may store up to 80 characters or digits or combination of both. Each character has a unique code which is seen as a number of punched holes in each column. Cards are read by the computer at the rate of 1000 cards per minute.



Punched cards have a number of advantages;
(1) Can be manipulated before being read by the computer (sorted, etc.)
(2) Easily read and interpreted by humans
(3) Individual cards can be removed, and corrected.
(4) The standard 80 column card is universally accepted, and can be read by most computers.

However there are disadvantages such as bulk, durability and slow transfer rates which make other means of storage more practical. They are still useful as an initial means of transferring data into machine readable form.

1.4.2 *Optical Mark Cards*

These cards are similar in dimension and characteristics to punched cards. However, instead of punching holes to code characters, pencil marks are made in pre-defined fields.

These cards are useful for answering precoded question (e.g. multichoice exams) so that the respondent actually codes the answer directly onto the card. Generally speaking, users will require a special layout for application of these cards, and so should contact the Computer Centre for further information. Cards for the mark sense reader are printed with a special ink which does not interfere with the marks to be read, and users should be careful to ensure that the cards used are of this type. The mark sense card reader will read normal punched cards, provided that they are punched on the special cards mentioned before.

1.4.3 *Magnetic Tape*

Magnetic Tape is a very commonly used storage medium for data on computers, but the actual codes used vary among computer manufacturers. Generally, magnetic tapes have either 7 or 9 tracks, and come in lengths of 1200 feet or 2400 feet. Whilst the Computer Centre does not encourage the use of magnetic tape for the storage of data, it is often a useful way of transferring data from computer to computer.

If users are obtaining data from some other source on magnetic tape then, as much as possible, the tapes should have the following characteristics:

1. Any length up to 2400 feet
2. Must be 9 track (or 7 track)
3. Packing Density of 1600, 800 or 556 bits per inch.
4. Unlabelled
5. EBCDIC, BCD or ASCII code

The Computer Centre has the capability to read most tapes that conform to these characteristics. If there is any doubt as to the type of magnetic tape, it is important to contact the Computer Centre before ordering. Data from magnetic tape will, if size permits, be transferred to disk. Otherwise, it will be converted into a form easily read by the U.Q. machines, and put back onto magnetic tape. A nominal charge will be made for this service.

1.4.4 *Magnetic Tape Cassettes*

Some remote terminals have magnetic tape cassette facilities which enable users to enter data locally and store it on cassettes. This facility is only available to users with terminals of this kind. Unfortunately, there is no uniformity of code between different brands of cassette mechanism, and so they tend to be unsatisfactory for transportation of data between machines.

1.4.5 *Magnetic Disk*

The PDP–10 computer system at the University of Queensland is a disk-based system, providing users with a large but finite space for the storage of data. Magnetic disks provide the fastest, most sophisticated means of data storage yet discussed. It is generally inevitable that, whatever the original form, most data will end up on disk for some length of time during processing.

It is now possible, by means of the recently developed data entry package QDATA, to place data directly on disk and avoid expensive punching charges. QDATA is meant to provide facilities similar to those available on commercial key-to-disk systems. These include several forms of searches to locate specific records for updating, a verify mode to allow checking against previously entered data. The ability exists to create new definitions detailing how data may be entered and checked. While entering a batch, each character is vetted as it is keyed, to ensure that it is allowable in this field and that the field length is not exceeded. Field parameters such as justification, filler characters, and automatic duplication (to name a few) may be nominated. It may also be set up to switch to a different record type on completion of certain types of records.

For more information on the QDATA package, the reader is referred to the the DOC file or the QDATA manual (MNT–4).

1.4.5.1 *On-line Disk Storage*

Data is stored on the PDP–10 disk system as named "files".
(*Note:* Users who are not familiar with the file system on the PDP–10 should read MNT–2 Ch. 4 in detail before proceeding.)

Each user has available an amount of "on-line" disk space in which to store files. This space is directly accessible any time the user logs onto the system (MNT–2 Ch 6). It is limited by a "logged-out quota", which is the total amount of space a user's files may occupy when the user is "logged off" the system. A user is permitted to occupy more space while logged in—up to what is known as a "logged-in quota". This extra space is provided to allow for the generation of temporary storage needed during the execution of a program, but is not available when the user logs off (see MNT–2 Ch 6.2).

It is possible to increase the size of a logged in quota on a temporary basis for very large jobs, e.g. some SPSS jobs. This is explained in Chapter 4.

1.4.5.2 *Off-line Disk Storage*

"Offline" storage differs from "Online" storage in that files stored are not necessarily available to the user immediately after logging onto the system. It is however, cheaper than on line storage, with no real limit on the amount used. Offline disk storage can be effected in two ways.

(a) File Migration System

This system enables a user to request that one or more of his files be "migrated" or transferred from "online" disk storage to a general public "offline" disk storage area. Similarly a user may request that one or more files be transferred from the "offline" area to the "online" area. This is useful for storing infrequently used files, or files which are too large to leave on online storage (MNT–2 Ch 9).

(b) Private Disk Packs

A number of disk drives are available for use with privately-owned disk packs. Users with very large data sets may wish to purchase or rent a disk pack for their exclusive use (see MNT–1). The number of drives available for this purpose is limited, and so it is wise to book a drive ahead of when it will be needed. The amount of storage a user can have on a private disk pack is limited only by the capacity of the pack (30 million characters). The use of private disk packs is discussed in Section **1.4.5.4**.

1.4.5.3 *Getting Data onto Disk*

There are a number of ways of transferring data to disk. However, only two will be mentioned here. It is important that users refer to the appropriate sections in MNT–2.

(a) Creating a file with the EDITOR.

The EDITOR is a program which enables users to create and change files on disk. It is intended for use from a remote terminal (MNT–2 Ch 5) and has the advantage of being fast and convenient when dealing with relatively small files. To *create* a file, a user must first "log" onto the system (MNT–2 Ch 6.1) and run the editor (MNT–2 Ch 6.2.1). A more detailed description of the editor and all its facilities can be found in MNT–6 *A Line Editor for the PDP–10.*

(b) Creating a file from a card deck.

It is desirable that data on punched cards be transferred to disk. This is particularly important where multiple analysis is to be performed on the same data set for, although the card reader can read 1000 cards per minute, this is slow compared to the data transfer rates from magnetic disk. Also, punched cards are not a very durable medium for storage, so as age and number of times read increases, the likelihood of problems in reading them arises.

Using the computer via punched cards is called *batch processing.* This means that, instead of being able to input data directly and get an immediate response as with a remote terminal, users submit decks of cards which are processed in "batches" at some later stage. The user then receives the printout of results. (MNT–2 Ch 7 should be read at this stage). An example of a card deck to put data onto disk is given below.

```
$SEQUENCE
$JOB [124,160]/NAME:NURK/COST:$2.00
$DECK SURVEY.DAT
.
.
.       data cards
.
.
$EOD
$EOJ
```

This copies the data on cards onto disk as a file called SURVEY.DAT. This file may then be manipulated in the same way as a file created from a remote terminal (See MNT–2 Ch 6.2.2 to 6.2.6).

1.4.5.4 *Using Private Disk Packs*

Whilst the public disks are in operation at all times, a user wishing to utilize a private disk pack is required to mount the disk pack before using it (See MNT–2). Each private disk pack has a 4-character "logical" name which is allocated with the disk pack. When using a remote terminal, the MOUNT command is used as follows:

```
.MOUNT EDUA:<cr>
```

This requests that a private disk pack called EDUA be mounted on a drive, and assigned to the user. If the job is to be run through Batch, the user should first consult the Computer Centre for advice on the methods available for securing the mount.

Bibliography

Moser, C.A. and Kalton, G., *Survey Methods in Social Investigation.* Heinemann Educational Books Ltd, London (1971).

CHAPTER 2

BIOMEDICAL STATISTICAL PACKAGE (BMD)

2.1 Introduction

The BMD package is representative of the largest group of statistical packages—the set of "stand alone" programs. It was developed initially as a tool for research at the UCLA Medical Centre and catered as much as possible for the analytic problems of biomedical research. It has undergone a number of changes since it was introduced, including addition of new programs to cover new fields of analysis and refinements of existing programs.

There are at present about 55 individual programs in the BMD package, and these can be classified into six groups;

1. Description and Tabulation
2. Multivariate Analysis
3. Regression Analysis
4. Special Programs (Life and Contingency Tables)
5. Time Series Analysis
6. Variance Analysis

2.2 Available Programs

The Computer Centre has made available a large number of the BMD programs, with a selection from each of the six classifications given before. It is possible to obtain an up-to-date list of the BMD programs which are available by typing the following command on a remote terminal, or placing a card with the same command punched onto it, in a batch run:

```
.DIR STA:BMD???
```

[STA: refers to the particular area of disk storage where the statistical programs are to be found.]

If it is wished to use a BMD program which is not currently available, the Computer Centre must be contacted, and assistance may be given in obtaining the program.

2.3 Using a BMD Program

Before attempting to use any BMD program on the PDP–10, the BMD manual (published by the University of California Press) should be read carefully, particularly the chapter relating to the program to be used.

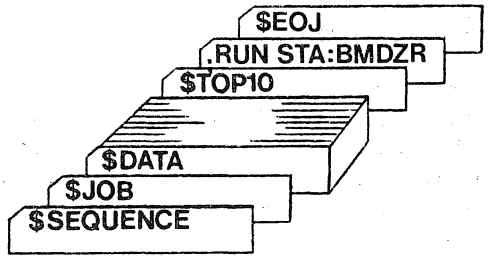
Users should note that some information in the BMD manual applies to run procedures for the particular IBM computer used at UCLA. This information should be ignored as different procedures apply for the PDP–10.

2.3.1 *Running a* BMD *Program through Batch*

2.3.1.1 *Data Stored on Cards*

To run a BMD program through batch with data on cards, the following deck setup should be used:

```
$SEQUENCE
$JOB
$DATA
.
.
.    (BMD control cards and input data as per BMD manual)
.
.
$EOD
$TOPS10
.RUN STA:BMD02R      (or the name of the particular BMD program to be used)
$EOJ
```



2.3.1.2 *Data Stored on Disk*

To run a BMD program through batch with data on disk storage, the following procedure should be adopted:

1. Rename the file to FOR02.DAT, i.e.

   .RENAME FOR02.DAT=MYFILE.DAT

2. Assign logical unit 2 to disk, i.e.

   .ASSIGN DSK:2

3. Rename file back to original name (after running). The following deck set up will apply:

```
$SEQUENCE
$JOB[...,...]/NAME:SMITH/COST:$5.00
$TOPS10
.RENAME FOR02.DAT=MYFILE.DAT
.ASSIGN DSK:2
.RUN STA:BMD05V
.RENAME MYFILE.DAT=FOR02.DAT
$EOJ
```

2.3.2 *Running a* BMD *Program from Terminal*

Due to the volume of output produced from BMD programs, they are not really suited to direct running from a remote terminal. If, however, the output is directed to disk storage, and printed on the high speed line printer, the remote terminal is particularly convenient. The following procedure should be used:

(a) The input data should be in a file called FOR05.DAT. (If not, it is necessary to enter the data through the terminal as the program is running, which is very tedious.)

(b) Assign logical units 5 and 6 to disk.

(c) Run the appropriate BMD program.

(d) The output will go to a file called FOR06.DAT, which may be printed on the high speed printer.

Example

```
.RENAME FOR05.DAT=MYFILE.DAT
Files renamed:
MYFILE.DAT

.ASSIGN DSK:5
DSK assigned

.ASSIGN DSK:6
DSK assigned

.RUN STA:BMD02R

EXECUTION TIME:      0.08 SEC.
TOTAL ELAPSED TIME:      0.52 SEC.
NO EXECUTION ERRORS DETECTED

EXIT

.PRINT FOR06.DAT
Total of 1 block in 1 file in LPT request

.RENAME MYFILE.DAT=FOR05.DAT
Files renamed:
FOR05.DAT
```

2.4 General Hints on Running BMD Programs

(a) The amount of core memory available to individual users during prime shift (8 a.m. to 5 p.m.) is 60K words on the KA and 120P on the KL (1P = 0.5K words). Some of the BMD programs require more than 60Kwords of core memory and so must be run after 5 p.m., when 120Kwords of core is available. When using a remote terminal, the amount of core required can be determined by the following commands:

```
.GET STA:BMD02R
Job setup
```

KL:

```
.CORE
Phys. mem. assigned 39P (Guideline: 512P Max limit:119P)
Swap space left: 4247P
.
```

KA:

```
.CORE
19+=0/60K core
vir. core left=251K
```

The sum of these two figures will give the amount of core memory required (in this case, 19KWORDS). Batch users should specify the amount of core required by their job (if it is more than 60KWORDS) on the $JOB card. This prevents the job from actually being run until the required amount of memory is available, i.e.

```
$JOB [120,131]/NAME:SMITH/COST:$2.00/CORE:64K
```

(b) Users should endeavour to become familiar with the program they are going to use before committing a full set of data. A test run using a small set of data is particularly useful.

(c) Before seeing a consultant about problems in running a BMD program, check that the parameters specified on the PROBLM card are correct. Also, ensure that the data is in the appropriate format and there is the correct number of cards.

CHAPTER 3

SCIENTIFIC SUBROUTINE PACKAGE (SSP)

3.1 Introduction

The Scientific Subroutine Package is a library of subroutines for programs written in FORTRAN IV. These subroutines cover most areas of statistical and numerical analysis, but require a main program written in FORTRAN IV to combine the necessary subroutines for an analysis.

Users should refer to the SSP manual before attempting to use any of the subroutines. A copy of this manual is available for viewing at the Computer Centre.

While SSP is still available, it has been superseded by the the more recent IMSL library of routines which are generally more flexible and efficient. IMSL also offers a wider range of subroutines. Intending users are therefore strongly advised to consider using it in preference to SSP. For further details consult Chapter 4.

3.2 Using SSP

The SSP routines are kept on disk storage as a binary relocatable library file (i.e. a REL file). The SSP library file should be loaded with the user program before execution, i.e.

```
.EXECUTE/REL MYPROG,REL:SSP/SEARCH
```

The SSP routines contain no internal error reporting, and this should be taken care of in the user program.

Note: SSP mathematical routines are also available, and may be located on the REL: directory. Their method of use is the same as for the SSP statistical routines.

CHAPTER 4

INTERNATIONAL MATHEMATICS AND STATISTICS LIBRARY (IMSL)

4.1 Introduction

IMSL is a collection of some 400 subroutines for programs written in the FORTRAN–10 language. The areas covered include:

1. Analysis of experimental design data
2. Basic statistics
3. Categorized data analysis
4. Differential equations; Quadrature; Differentiation
5. Eigenanalysis
6. Forecasting; Econometrics; Time series
7. Generation and testing of random numbers; Goodness of fit
8. Interpolation; Approximation and smoothing
9. Linear algebraic equations
10. Mathematical and statistical special functions
11. Nonparametric statistics
12. Factor analysis, etc.
13. Regression analysis
14. Sampling
15. Utility functions
16. Vector matrix arithmetic
17. Zeroes and extrema; Linear programming

Though the intention of the library is similar to that of SSP it possesses many advantages over it. Notably it offers a much wider range of routines, that are more sophisticated, flexible, and efficient. Importantly all routines were coded specifically for the DEC–10.

4.2 Using IMSL

Like SSP all the IMSL routines are stored on disk in the form of a binary relocatable library file. This file should be searched by the loader, and the required routines extracted prior to execution, i.e.

```
.EXECUTE/F10 MYPROG,STA:IMSL/SEARCH
```

The use of the FORTRAN–10 compiler (indicated via the F10 switch) is neccessary for correct execution. The subroutines themselves contain error detection and reporting facilities as indicated in the manual.

CHAPTER 5

STATISTICAL PACKAGE FOR THE SOCIAL SCIENCES (SPSS)

5.1 Introduction

SPSS was one of the first statistical packages to employ a "total system" concept, whereby the package itself provided all or most of the necessary data handling and file manipulation facilities, as well as the required statistical analysis.

Ideally, such a package should minimize the amount of knowledge a user needs about the particular computer system being used. Unfortunately, this is not really true in all circumstances, and so users who wish to perform analyses on large data sets in particular should endeavour to become familiar with the operation of the actual computer system being used. Such knowledge can often mean a considerable saving in time and expense. This section will discuss the use of SPSS on the PDP–10 both in simple applications, and more complex ones, with an emphasis on making most efficient use of SPSS and the PDP–10.

If any situations arise which are not covered in this manual, users should not hesitate to contact the Computer Centre.

5.2 The PDP–10 Implementation of SPSS

The PDP–10 implementation of SPSS (SPSS–10) was produced at the University of Pittsburgh and is a conversion of the National Opinion Research Centre's SPSSH Version 7.01. SPSS-10 closely follows the documentation in:

> Nie, Hull, Jenkins, Steinbrunner, Bent
> *Statistical Package for the Social Sciences*
> (Second Edition)
> McGraw–Hill Book Company
> (1975)

and

> Hull and Nie
> *SPSS Update*
> *New Procedures and Facilities for Releases 7 and 8*
> McGraw-Hill Book Company
> (1979)

Any differences between the PDP–10 implementation and that described in the McGraw–Hill manual are documented below. A more comprehensive list of differences may be obtained from DOC: by

> .PRINT DOC:SPSS.DOC

and

> .PRINT DOC:SPSS8.DOC

5.2.1 *Running* SPSS–10

SPSS–10 will run in a minimum of 32 KWORDS of core memory. Included in this is a "space" allocation of 3 KWORDS. This default of 3K is sufficient to run small jobs; however, for larger jobs more space is required. The space requirements for a particular job can be calculated using the formulae in the manual. This will give the answer in bytes which is divided by 4 to give the number of PDP–10 words required.

As Chapter 8 of the manual explains, the total *space* allocated for a run, either by default or via the /SPACE: switch, is partitioned into two parts. The first is used by the procedures in SPSS for the storage of the matrices etc. required for the analysis requested. The second, in the jargon known as *transpace*, is used for the storage of the data transformation cards.

The proportional partition of the total space allocation is, by default, in the ratio of 7:1. That is, a given *space* of 3K (or 3072 words) will result in 2688 words going to *workspace* for procedures, and 384 words going to *transpace*. If this default partition ratio is inappropriate for a given job it can be overidden by the use of the ALLOCATE card.

Since the maximum memory allowed for a user during prime shift is 60K, some large SPSS–10 jobs may not be run at this time. Jobs requiring more than 60K will be run after 6 pm.

A typical batch deck for a simple SPSS job would be:

```
$SEQUENCE
$JOB [113,160]/NAME:SMITH/COST:$5.00
$DATA
.
.
.          (SPSS PROGRAM)
.
.
$EOD
$TOPS10
.RUN STA:SPSS
*CDR:
.IF(ERROR).CLOSE
.PRI *.LST
$EOJ
```

To run an SPSS program which is on disk rather than cards, the following deck set up could be used:

```
$SEQUENCE
$JOB [113,160]/NAME:SMITH/COST:$5.00
$TOPS10
.RU STA:SPSS
*LPT:=TEST.SPS
$EOJ
```

The general format is as follows;

```
.RU STA:SPSS
*Destination=Source,Switches
```

"Destination" and "Source" are the standard PDP–10 file specifications, i.e.

```
DEV:FILE.EXT[p,pn]<PROT>
```

where the following conventions will apply:–

	Destination	*Source*
DEV:	Default DSK:	Default DSK:
FILE	Defaults to source filename	No default (*must specify*)
EXT	Defaults to .LST	Defaults to .SPS
[p,pn]	Defaults to user's ppn	Defaults to user's ppn
<prot>	Defaults to system standard	Inappropriate

Example 1

```
.R STA:SPSS
*LPT:=TEST.SPS[160,105]
```

This would look for the SPSS program in a file called TEST.SPS on the [160,105] disk area. The output would be printed on the line printer.

Example 2

```
.R STA:SPSS
*POVA:KRID[20,5]<155>=DSKD:DATA
```

This would look for the SPSS program in a file called DATA.SPS on DSKD:. The output would be written to a file called KRID.LST which would be on the [20,5] area of a disk called POVA:. The file would be given a <155> protection.

Switches

In SPSS–10, "switches" replace the specifications provided by the JCL cards in the IBM implementation (as described in the McGraw–Hill manual). The following switches are presently provided:–

/GET

This switch follows a file specification and overrides any file specification found on the GETFILE control card (in the SPSS program). Although the specification field for a GETFILE card is overridden, use of the /GET switch still requires the occurrence of a GETFILE card in the SPSS program.

/HELP
This switch causes a description of all switches to be typed on the terminal or in the log file.

/INPUT
This switch follows a file specification, and overrides any file specification given on the INPUT MEDIUM control card in an SPSS program. The same conditions apply as for the /GET switch.

/OUTPUT
This switch follows a file specification and overrides any file specification on the RAW OUTPUT UNIT card. The same conditions apply as for the /GET and /INPUT switches.

/SAVE
This switch follows a file specification and overrides any file specification on the SAVE FILE card in an SPSS program. The same conditions apply as for the /GET, /INPUT and /OUTPUT switches.

/SCRATCH
This switch follows a device specification and overrides the default scratch device which is used by SPSS to hold observations between statistical procedures. This file can be very large, and in fact for large SPSS jobs, may exceed the users logged-in disk quota. If there is a risk of this occurring, the user should use a private disk pack or the system scratch pack (DSKB) for scratch purposes.
Note: If only one statistical procedure and no save file is involved, then no scratch space is really necessary. Unfortunately SPSS does not take account of this, and will in fact still produce a large scratch area. To overcome this and significantly reduce run time, use the null device (NUL:) as the scratch device.

/SPACE:n
This switch is designed to allow users to specify memory requirements above the system default. The value of *n* is determined for each statistical procedure on the basis of formulae given below. If $n > 225$ it is assumed to mean "words" of memory. If $n < 225$, it is assumed to mean KWORDS (i.e. 1K=1024 words).

The following is an example of the use of switches:

```
.R STA:SPSS
*LPT:=TEST.SPS,DSKB:/SCRATCH,INPUT.DAT/INPUT/SPACE:5
```

The output would go to the line printer; the SPSS program would be read from DSK:TEST.SPS; the input data (i.e. INPUT MEDIUM card) would come from DSK:INPUT.DAT, irrespective of what file was actually specified on the INPUT MEDIUM card; the scratch file would be written to the public scratch pack DSKB: (ensure that DSKB: is mounted first), and 5140 words of memory would be provided for SPACE.

5.2.2 SPSS *Control Cards*

In SPSS–10 the general control card format is free-field (as opposed to the McGraw–Hill manual which defines control cards in a fixed format), and is interpreted as follows:

1. If column 1 contains neither a blank nor a tab character, then all columns from column 1 up to a tab or two consecutive blanks, or to column 15, are considered the control field.
2. If the card begins with one or more blanks or tabs, then the card is a continuation card, and all characters are part of a specification field.
3. The specification field of a card may contain no more than 65 characters, irrespective of leading blanks.
4. Any tab which is encountered in the specification field is replaced by a single blank character, and the specification field is printed left-justified.
5. If the "numbered" option is specified, then the numbering field must begin in column 73.

The specification fields of some of the SPSS–10 control statements differ from those given in the McGraw–Hill manual. These differences are documented below.

COMPUTING LANGUAGE
SPSS–10 generates F–10 (Fortran–10) warning messages for undefined arithmetic operations such as division by zero, square root of a negative value, etc. The resultant value of these operations is the usual F–10 value. Users may avoid any difficulties by using procedures like the following:

```
COMPUTE                 X = -1.0
IF                      (Y NE 0.0) X = 1.0 / Y
MISSING VALUES          X (-1.0)
```

The following types of cross case transformations have been added to the computing language:

X = ACCUM *e*

Sums across cases. The argument *e* to ACCUM (variable or expression) is evaluated, and then added to the sum of all the previous arguments. The value generated for a particular case is the running sum, up to that point in the file.

X = LAG *e*

To lag time ordered data. The value of LAG for a case is the value of its argument in the previous case. For the first case, if the variable to the left of the equal sign is eligible for ASSIGN MISSING then that value is given. If not, then if the argument has MISSING VALUES, its first missing value is given. If the argument has no missing value then 0.0 is the result. In this case 0.0 should be a missing value for the resultant variable. Multiple case lagging may be accomplished by composition of LAG functions.

X = UNIFORM *n*

UNIFORM generates uniformly distributed random numbers in the range of 0.0 to *n*. See discussion of the SEED command below.

X = NORMAL *sd mean*

NORMAL generates normally distributed random numbers with standard deviation *sd*, mean 0.0 and standard deviation 1.0. For a different mean simply add the value of the desired mean. See discussion of the SEED command below.

X = POISSON *m*

POISSON generates random numbers from the POISSON distribution with mean *m*. See discussion of the SEED command card below.

ALLOCATE TRANSPACE = *value*

In SPSS–10 the space must be specified in words. One word is equivalent to four bytes. On the DEC–10 a given TRANSPACE allocation results in 20 per cent fewer transformations being permitted than in the IBM version.

ASSIGN BLANKS *value*

Many raw-input-data files have been prepared with blank entries when information has not been ascertained. This is particularly true when no missing-value codes were specified at the coding stage. Blanks are normally treated by SPSS–10 as zeroes. However, if there exist in the user's file variables for which both blank and zero have been used as legitimate codes and the user wishes to distinguish between them, the blanks may be given a user-specified value. This is accomplished by placing an ASSIGN BLANKS command before the first procedure request. The value specified on the ASSIGN BLANKS command will be given to *all* blank fields encountered while reading *numeric* data.

BREAKDOWN

Bug: statistics 1 and 2. Coefficients produced are not those indicated in Figure 17.3 on page 259 of the SPSS Manual, 2nd Edition. In particular:

Statistic 1: Eta^2 is not printed. Eta^2 = Between groups sum of squares divided by total sum of squares, and has the same value as the Eta^2 currently produced by Statistic 2.

Statistic 2: Eta^2 and Corr Coeff are printed; the manual indicates that R and R^2 are to be printed. R is the Corr Coeff under a different name, and R^2 is simply the square of R.

Limitation: BREAKDOWN Integer Mode. The maximum number of value labels which can be printed on a single table is 200.

CROSSTABS

Limitation: CROSSTABS Integer Mode. The maximum number of value labels which can be printed on a single table is 200.

DATA LIST

The keyword BINARY when used with DATA LIST must conform to the specification discussed in INPUT FORMAT below.

DISCRIMINANT

There is a limit of 100 variables, groups, or steps (in stepwise solutions). Options 13 through 19 and statistics 7 and 8 are not yet implemented.

FACTOR

No more than 82 variables may be entered on a VARIABLES = list.

FILE NAME *filename, file label*
The FILE NAME card should be considered documentation only. Its specification fields are stored and retrieved with SPSS system save files and they are printed on listings. But they should not be confused with the file specifications which appear on the GET FILE or SAVE FILE cards and are seen in the user's directory listing. In SPSS–10 the FILE NAME card may be used to change a previous file name and file label after a GET FILE and before a SAVE FILE. The information on the FILE NAME card is used by SPSS–10 for creating SPSS system save file names when the user has not explicitly given the information on the SAVE FILE command. This procedure is not recommended.

FINISH
The FINISH card may be absent. SPSS–10 will then generate one. This is useful since INPUT MEDIUM CARD will produce a more useful diagonostic if the number of data cards is short.

GET ARCHIVE
Archiving is not yet implemented.

GET FILE *file specification*
In SPSS–10 the GET FILE card specifies the physical device from which the SPSS system file will be fetched. It takes a standard DECsystem–10 file specification and recognizes the multiple reel file syntax. Dev: defaults to DSK:. File and extension if appropriate do not default. PPN defaults to the user's project-programmer number and Prot is inappropriate. Two buffers are used. The physical blocksize is the installation device default and may be changed with the the .SET BLOCKSIZE monitor command. The file name on the SPSS system file is not verified against the specification field of the GET FILE command.

INPUT FORMAT *FIXED (Format list)* or *FREEFIELD* or *BINARY*
The FIXED format specification may contain the format control characters: A, E, F, G, O, T, and X. If the format specification I*w* is mistakenly used then it will be automatically converted to F*w*.0. For details of writing formats the F–10 Language Manual, Second Edition should be consulted. Variables are automatically given a PRINT FORMAT corresponding to their INPUT FORMAT. If a variable is read with an F*w.d* then it receives a PRINT FORMAT (d). If the variable is read with an A*w* format then it is given PRINT FORMAT (A). A PRINT FORMAT card may be required if the variable is later recoded to F-type. See RECODE below. Five characters may be contained internally in an A-type variable. If fewer are read then they will appear internally left-justified and blank-filled. If more than five are specified then the leftmost characters will be ignored.

The FREEFIELD definition has been changed to correspond to DEC's FOROTS list directed I/O definition. See FORTRAN–10 Language Manual, Second Edition. This means that each case must start on a new card. FREEFIELD may be used in conjunction with N OF CASES UNKNOWN. A control-Z provides the end-of-file when input is from a terminal.

A file specified by keyword BINARY must conform to the FORTRAN–10 unformatted binary I/O specification and may be created by a FORTRAN–10 program. These files will be processed by SPSS–10 almost as efficiently as SPSS system files. The format list specification for BINARY files is not implemented. This means that each case must occupy precisely one logical record and that no variables may be skipped on the record. Although, the user may cause trailing values to be ignored by specifying fewer than the total number of variables in the record on the VARIABLE LIST card.

INPUT MEDIUM *CARD* or *file specification*
The INPUT MEDIUM card takes the standard DECsystem–10 file specification and recognizes the multiple reel syntax. The keyword CARD may also be used as described in the McGraw–Hill manual. The keywords TAPE, DISK and OTHER are not recognized. If they are used then the proper specification should be provided with the /INPUT switch. The default device is DSK:, The file name and extension do not default, the project-programmer number defaults to the user's and protection is inappropriate. The ability to reference an alternate project-programmer number is particularly useful for allowing several students or researchers to reference a common data base. Two buffers are used. The physical blocksize is the installation device default and may be changed with the .SET BLOCKSIZE monitor command (see DECsystem–10 Operating System Command Manual).

MERGE FILES
The MERGE FILE command is not implemented. It probably will not be implemented unless a significant need for it is demonstrated.

MISSING VALUES *variable list (value list) /*
The keyword BLANK may *not* be used to specify missing values. See the above discussion of the ASSIGN BLANKS data-definition control card.

N OF CASES *n* or *UNKNOWN* or *ESTIMATED n*
In SPSS–10 if the number of cases is not known, then the keyword ESTIMATED followed by a reasonable estimate of the number of cases should be specified. This number is used to allocate contiguous blocks of disk storage for scratch and SAVE FILEs. This will increase efficiency and avoid problems with extended ribs. The monitor message "Exceeding quota" may occur during execution, indicating that the user has insufficient disk available for his scratch or SAVE file.

OPTIONS *option number list*
The inclusive TO convention may be used in a number list on the OPTIONS card.

OSIRIS VARS *variable name list*
In SPSS–10 when reading an IBMsystem/370 OSIRIS tape, an INPUT MEDIUM card must precede the OSIRIS VARS card. A .SET BLOCKSIZE dev: *n* monitor command must be issued after the tape mount. OSIRIS dictionary files have a block size of 1600 bytes and would therefore require n to be 400 words. If the data file block size is larger, then *n* should be set to the actual number of bytes divided by four, rounded up to the next full word. The tape must be positioned at the OSIRIS dictionary file except that tape-label files will automatically be skipped. This may be accomplished using PIP. The data file is assumed to follow the dictionary file but may be separated from it by tape-label files.

The OSIRIS file reading routine was written by Mr. Tim Hill of Wesleyan University.

PRINT FORMATS *variable name list (value) / ...*
This card must follow the INPUT FORMAT card. The above discussion of the INPUT FORMAT card should be consulted to understand how SPSS–10 establishes its print format defaults.

PAGESIZE *n* or *NOEJECT*
In SPSS–10 NOEJECT is the default for PAGESIZE. This default may be overridden by a PAGESIZE command. The IBM version of SPSS uses PAGESIZE 54 as the default.

RAW OUTPUT UNIT *CARD* or *file specification*
The RAW OUTPUT UNIT card takes a standard DECsystem–10 file specification or keyword CARD which causes the output specification to be CDP: with file name the same as the source file name and extension .CDP. If cards are requested the user must specify the /CARD: switch on the $JOB card. If RAW OUTPUT UNIT is absent then the default is DSK:FOR09.DAT. When present the defaults are: device defaults to DSK:, file name defaults to the source name, and extension defaults to .DAT. If this file is later queued to the line printer and if it has no carriage control characters then it should be queued with switch /FILE:ASCII. Two buffers are used. The physical blocksize is the installation device default and may be changed with the .SET BLOCKSIZE monitor command.

RECODE *varname list (list = new value) / ...*
and
*RECODE *varname list (list = new value) / ...*
The keyword BLANK is not recognized. See the above discussion of the ASSIGN BLANKS data-definition control card. When using the (CONVERT) procedure, if a variable's PRINT FORMAT is (A), then the variable will be automatically given PRINT FORMAT (0). This default may be later changed with a PRINT FORMATS card.

REGRESSION
Several additions to the REGRESSION procedure are available.
Option 16—Regression through the origin. Option 16 is useful in two ways. First, the SPSS procedure REGRESSION can be used to estimate parameters and test hypotheses for models with a fixed intercept of zero rather than an estimated intercept. Second, more flexibility in using dummy variables is available to the user via option 16. For example, without option 16, if two qroups of observations exist and are represented by the dummy variables (1, 0) and (0, 1), only one of the dummy variables may be entered into the equation. By using option 16, the user may enter both dummy variables into the equation.
In effect, option 16 forces subprogram REGRESSION to do its computing using the uncorrected sums of squares matrix rather than the correlation matrix. Consequently, some statistics printed by the program are not particularly meaningful when option 16 is invoked. The purpose of option 16 is to permit the user to obtain regression and residual sums of squares and mean squares as well as parameter estimates (slopes of b's) for a particular class of models.
Statistic 8 is the computation of a multiple-partial correlation coefficient and an attendant sequential analysis of variance. The sequential analysis of variance performs the hierarchical test described on page 339 of the SPSS Manual (2nd Edition). Statistic 8 is computed only after 2 or more variables are added to an equation in one step unless option 17 is specified.
Option 17 forces computation of statistic 8 after every step regardless of how many variables are entered into the equation in that step.
The extensions to the REGRESSION procedure were provided by Prof. David A. Specht of Iowa State University.

MULT RESPONSE *GROUPS = group name label (var list (value list)) group name ... /*
VARIABLES = var list (value list) var list ... /
FREQUENCIES = item list / and/or
TABLES = item list BY item list item list ... /

The MULT RESPONSE procedure provides a mechanism for the analysis of multiple response items, that is: typically, an item on a survey to which the respondent might legitimately make more than one reply (See Appendix A of DOC:SPSS.DOC; also SPSS UPDATE).

NPAR TESTS *test (parameters) = variable list (parameters) /*
test (parameters) = variable list (parameters) / ...

Subprogram NPAR TESTS performs a large variety of nonparametric statistical tests. All of the tests implemented in this subprogram are described in Siegel's book, *Nonparametric Statistics for the Behavioral Sciences*, New York, 1956. (For more detail see Appendix B of DOC:SPSS.DOC; also SPSS UPDATE).

RELIABILITY *VARIABLES = variable list /*
SCALE (label) = scale list / ... /
VARIABLES = variable list / ... etc.

Subprogram RELIABILITY provides a means for evaluating multiple-item scales through the computation of widely recognized coefficients of reliability. In addition, the program can provide the user with basic summary statistics including item means, standard deviations, inter-item covariance and correlation matrices, scale means, standard deviations, item to scale correlations, and summary statistics of the item means, variances, inter-item correlations and covariances. The program can perform a repeated measurements design analysis of variance a two-way factorial ANOVA with one observation per cell, Tuckey's test for additivity, and Hotelling's T-squared test for equality of means in repeated measurements designs and Freidman's two-way analysis of variance of ranks.

Subprogram RELIABILITY was provided by Prof. David A. Specht of Iowa State University. It is documented in:

David A. Specht
User's Guide to Subprogram Reliability
Department of Sociology
Iowa State University

This document is on file RELIAB.DOC and may be obtained from the local SPSS coordinator or a user consultant. It is also discussed in SPSS UPDATE.

SAMPLE *factor*

See the discussion of the SEED command below and SPSS UPDATE.

SAVE ARCHIVE

Archiving is not yet implemented.

SAVE FILE *file specification*

The specification field of SAVE FILE card specifies the physical device to which the SPSS system file will be written. It takes a standard DECsystem–10 file specification and recognizes the multiple reel file syntax. Dev: defaults to DSK:. File and extension if appropriate do not default. Ppn defaults to the user's project-programmer number and protection defaults to the installation default. Two buffers are used. The physical blocksize is the installation device default and may be changed with the .SET BLOCKSIZE monitor command. In SPSS–10 the file label specification field of a SAVE FILE command may not be present. To change a file name or file label in creating a new SPSS system file use a FILE NAME command following the GET FILE. Note that when saving a file that the scratch file device may not be NUL:.

A maximum of 200 value labels can be saved for any one variable on a system file. However, more than 200 value labels may be defined for use in processing.

SEED *positive odd integer or zero*

In SPSS–10 the random number generator is not initialized from the machine clock. Also each SAMPLE, *SAMPLE, UNIFORM function, etc. has its own independent random-number generator. Therefore, runs which employ random-numbers always yield reproducible results. The SEED command may be used to change the default and produce different streams of random-numbers. Its value should be an odd positive integer or 0.0. If its value is 0.0 then a standard seed will be employed. Multiple SEED cards may occur in a run. When a command card which requires a random-number generator is encountered in a run the last value of SEED is used to initialize its random-number generator. The default seed is 0.0.

SORT CASES *variable list(s), ...*
WORKSPACE = (*number of sort keys* + 1) * *number of cases*

STATISTICS *statistics number list*

The inclusive TO convention may be used in a number list on the STATISTICS card.

TETRACHORIC *VARIABLES = variable list (low value, high value)/ CORRELATIONS = variable list WITH variable list / variable list WITH variable list / ...*

TETRACHORIC is an SPSS procedure which computes tetrachoric correlation coefficients between dichotomous variables.

The use of tetrachoric correlation coefficients is appropriate for variables which have only two observed values, but which may be assumed to be continuous and normally distributed. When these assumptions are met, tetrachoric R will be numerically equivalent to Pearson R and may be considered an approximation to it. Tetrachoric R is widely used in test scoring and item analysis to estimate the Pearson product-moment correlation between dichotomous items when continuity, and not a true dichotomy, is a logical assumption.

Tetrachoric coefficients can only be computed for data which is numeric and dichotomous. If a variable has been defined as alphanumeric or has more than two values, it should be recoded with the RECODE or *RECODE card prior to requesting the TETRACHORIC procedure. For each correlation requested, TETRACHORIC first computes a 2 X 2 table. Tetrachoric R is then computed from the proportion of cases falling into the various subcells of the table.

Two types of information must be specified on the TETRACHORIC card. After the VARIABLES = keyword, all variables for which tetrachoric correlations will be requested must be named, followed by the two values of each variable or variable list. Following the CORRELATIONS = keyword, the variables for which correlation coefficients are desired should be entered. The "var TO var" convention may be used in naming variables and refers to the order of the variables in the VARIABLES = list. Like the PEARSON CORR procedure, the keyword WITH indicates that a coefficient is to be computed for each variable named preceding the WITH paired with each variable named following the WITH. Whenever the keyword WITH is not used to separate variables being correlated, the program computes all possible non-redundant correlations from the variables in the list. Multiple correlation lists may be requested on a single TETRACHORIC card as long as each list is separated from the next by a virgule; only one CORRELATIONS = keyword is permitted.

Statistics available with Subprogram TETRACHORIC:

1. Causes the mean, standard deviation, and relative frequency distribution of each variable referenced in the correlation lists to be computed and printed.

Options available with subprogram TETRACHORIC:

1. Inclusion of missing data. All cases will be included in the analysis regardless of any missing data values which have be defined.
2. Listwise deletion of missing data. A case will be omitted from the calculation of all coefficients requested in a single correlation list if the case has missing data for any variable in the list. The default is pairwise deletion of missing data (a case will be omitted from the computation of a given coefficient if the value of either of the two variables being correlated is missing).
3. Two-tailed test of statistical significance. This option causes a two-tailed test of significance to be computed for each correlation rather than the default one-tailed test.
4. Write correlation matrix on RAW OUTPUT UNIT. This option causes a matrix of coefficients to be written on RAW OUTPUT UNIT for all lists which are specified in matrix form (i.e., the keyword WITH must not be used). All matrices are output as card images with a format of 8F10.7. Each row of the matrix starts on a new card, and the row continues onto as many cards as required.
5. Square matrix print format.
6. Write means and standard deviations on RAW OUTPUT UNIT. This option may only be selected when Option 4 has been requested and causes means and standard deviations to be written on RAW OUTPUT UNIT, preceding each correlation matrix. For each list following the CORRELATIONS = keyword, the means for all variables appearing in the list are written, followed by the standard deviation for all variables in the list. Both means and standard deviations are written in 8F10.4 format.
7. Print subcell proportions and subcell frequencies from which tetrachoric coefficients are computed.

Limitations for Subprogram TETRACHORIC:

1. A maximum of 500 variables may be named in the VARIABLES = list. When the "var TO var" convention is employed, each variable which is implied counts as 1 toward this total.
2. A maximum of 40 individual lists and 500 variable names may appear following the CORRELATIONS = keyword.

3. The maximum number of coefficients which can be requested on a single TETRACHORIC card (and its continuations) varies depending upon whether Statistic 1 has been selected. The amount of WORKSPACE required for TETRACHORIC is:
   SPACE = (NCOR * 4) (MEANS * NVARS * 3) NCORE = the total number of correlations requested
   MEANS = 1 if Statistic 1 has been selected, else 0
   NVARS = number of variables mentioned in the CORRELATIONS = list

WRITE FILEINFO
The WRITE FILEINFO procedure always produces ASCII files. If the CHAR = construction is employed it will be ignored.

5.3 Running SPSS from Terminal

As well as running SPSS–10 from a terminal and using a program in the form of a disk file (as described earlier), it is possible to enter an SPSS program directly.

When the source device is TTY:, SPSS–10 prompts the user with a right angle bracket >. A command or a procedural request with accompanying statistics and options specifications may then be entered in free field format. After the last continuation line (if any), typing an "escape" or "altmode" will cause SPSS to execute the request immediately.

This feature is convenient when used in conjunction with a save file, for applying repeated statistical analysis to a common data set.

5.4 Some Examples of SPSS Runs on the PDP–10

(a) A simple batch run using an SPSS *program with data stored on cards.*

```
$SEQUENCE
$JOB [60,105]/NAME:SMITH/COST:$5.00
$DATA
.
.
.        SPSS program (including data)
.
.
$EOD
$TOPS10
.R STA:SPSS
*CDR:
$EOJ
```

(b) A simple batch run, but with a very large number of data cards.

In this case, it is recommended that the actual input data is first read onto a disk file so that for later runs, all of the cards need not be read in again.

```
$SEQUENCE
$JOB etc.
$DECK INDOT.DAT
.
.
.        (actual input data)
.
.
$EOD
$DATA

INPUT MEDIUM      INDOT.DAT

$EOD
$TOPS10
.R STA:SPSS
*CDR:
$EOJ
```

To use the same data set again in subsequent runs.

```
$SEQUENCE
$JOB
$DATA

INPUT MEDIUM        INDOT.DAT

$EOD
$TOPS10
.R STA:SPSS
*CDR:
$EOJ
```

If the input data is on a private disk pack, then this pack must be available when the job is being run.

5.5 General Hints on Running SPSS–10

(a) Unless otherwise specified, a default time limit of 5 minutes processor time is allocated for each job on the PDP–10. As a general "rule-of-thumb", one minute of processor time should be allowed for each $2.00 of cost limit for a standard priority run (1 minute per $1.00 at low priority).

Thus if a job is expected to cost approximately $20, a time limit of 10 minutes should be allowed (in standard priority), i.e.

```
$JOB [60,105]/NAME:SMITH/COST:$20.00/TIME:20
```

Note: It is more desirable for a job to stop with COST LIMIT EXCEEDED than TIME LIMIT EXCEEDED.

(b) It is difficult to estimate the actual cost of a particular SPSS run, because of the fact that few SPSS users are alike. Also, the cost of an SPSS run does not necessarily increase proportionally with the number of variables, cases, options or statistics etc. Generally, some "order of magnitude" cost estimate can be given. However, if in doubt, the limit should be set high rather than low.

The effects of COST LIMIT EXCEEDED can be minimized in three ways:

(i) The user should try to keep a record of all SPSS runs including information on the number of variables, statistics, options, procedures, priorities, observations etc. This will assist in the making of more accurate estimates.

(ii) As far as possible break large jobs up into small subjobs. For example, if a job required to process all of 80 subfiles, the job should be split up into 4 smaller jobs processing 20 subfiles.

(iii) If the job is being run from a remote terminal, and it stops because of an exceeded cost limit, the user may reset the cost limit and continue, e.g.

```
.R STA:SPSS
*LPT:=TEST.SPS

?COST LIMIT EXCEEDED

EXIT

.SET COST 10.00

.CONTINUE
```

If it is not wished to continue processing, but to check what results have already been obtained, the files should be "CLOSEd" as follows:

```
.R STA:SPSS
*LST:=TEST.SPS

?COST LIMIT EXCEEDED

.SET COST 2.00

.CLOSE

.PRINT TEST.LST
```

Because of the iterative nature of most SPSS jobs, the output obtained up to the point of stopping is useful, and so it may not be necessary to rerun the whole job.

If the job is running through batch, and a COST LIMIT EXCEEDED occurs, the above procedure of closing the files will be undertaken automatically.

(c) When embarking on a new SPSS project, try all procedures to be used, on a small test deck of data. Experience gained in doing this may save considerable time and expense on a later run using a full set of data.

(d) Confusion exists regarding the use of filename extensions in SPSS. Both the input command file and the SPSS system file generated by a SAVE FILE command take .SPS as default extension. Using the same filename (with no extension) for both the command file and the system file can cause the overwriting of the command file. To avoid confusion either use different filenames for command and system files or give explicit extensions in both cases.

(e) By inserting EDIT as the first procedure card, SPSS can be instructed to perform syntax checking of the control cards for mis-spellings, incorrect constructions, etc. In this mode the procedural requests are checked for correctness but are not executed, e.g.

```
.R STA:SPSS
*LPT:=MYSPSS.SPS/EDIT
```

Since the workspace required for such runs is only that needed for TRANSPACE, it is possible, under standard priority, to relatively cheaply check out runs for errors prior to submission of 'live' runs. This can considerably reduce turnaround time on successful analysis. See Page 3 of the documentation on Release 7.01 for further details.

(f) Before seeing a consultant about an SPSS problem, ensure that all files and programs are available, and that an up-to-date listing is obtained. Experience has shown that many SPSS errors are caused by not strictly adhering to the procedures and limitations as documented in the manual, so it is wise to check these closely in relation to the particular statistical procedure being employed.

5.6 VAX 11/780 Implementation of SPSS

SPSS is available on the VAX 11/780 system.

The version available on VAX differs in a number of details from SPSS–10. In broad terms, it follows the SPSS manual more closely than does SPSS–10 so that changes outlined in the document SPSS.DOC on the PDP–10 do not apply to the VAX version.

In particular:

1. Usual default extensions (.SPS and .LST) do not apply.
2. Where data is stored as a disk file, the specification field for the INPUT MEDIUM card must contain just the Keyword DISK (and not a file specification).
3. Unless the ASSIGN command is used (See 4 below), the VAX version of SPSS expects the data-file to be named FOR008.DAT, and will write the output to FOR006.DAT.
4. To use a data-file X.DAT and write output to file A.B, precede the SPSS command with the following *two* ASSIGN commands:

```
ASSIGN X.DAT FOR008
ASSIGN A.B FOR006
```

In accordance with the usual VAX philosophy regarding operation, each required element for a complete command may be entered as a response to a prompt, as follows:

```
$SPSS
Please enter CONTROL file name:
Please enter     bytes of workspace:   (Default=80, i.e. 80K bytes)
Please enter     bytes of labelspace:  (Default=30, i.e. 30K bytes)
```

Alternately, the complete command may be issued in a single line of the following form:

```
$SPSS /output=outputfil.ext workspace labelspace
```

This method removes the need for *one* (only) of the two ASSIGN commands noted in 4 above, viz. ASSIGN A.B FOR006

Further information on running SPSS on the VAX 11/780 can be obtained by consultation with the Centre.

CHAPTER 6

STATPACK

6.1 Introduction

STACKPACK is a statistical analysis program specifically oriented towards working from the terminal.

The coverage of available analysis routines is quite extensive, including:

- data descriptions (means, frequencies, crosstabulation, histograms, etc.)
- correlations
- regression
- ANOVA
- factor analysis
- discriminant analysis
- exponential smoothing
- non-parametric statistics

A variety of utility functions can be performed on data prior to analysis—sorting and transformations are examples. As well, data output from STP may be stored on disk for subsequent re-input.

Although not as powerful as SPSS, it is more suitable than SPSS for those users whose data sets are small and who prefer the convenience of working from the terminal.

The documentation for STP resides on disk and copies of it may be printed via

```
.PRINT DOC:STP.DOC
```

The only deviations from the manual are as follows:

1. At this installation it is accessed from the statistics area (STA:) rather than the system area. That is, to run it use

   ```
   .RUN STA:STP
   ```

2. The subsidiary programs alluded to on page 6 of the manual are not available here.

6.1.1 *Limitations*

Maximum core allowable

nv — number of variables
no — number of observations *no*nvnv*nv*2*max3*nv<8001*
max — larger of *nv* or *no*

See also table of variable–observation combinations.

6.1.2 *Description*

Statpack is an integrated, interactive package written for terminal use. It allows the user to issue simple commands for data analysis and will prompt him for necessary information. When questions of a procedural nature arise, the user may ask for an additional explanation by simply typing HELP. The standard output device is the terminal; however, a command is available to channel output to the line printer, providing the user with the ability to obtain multiple copies.

Data input may be from terminal, disk, magnetic tape, or a structured data bank. Input consists of observations each containing a value for every variable. Variables are defined by a number or an alphabetic name of not more than five characters. Data must be entered before issuing any of the statistical commands. Once data has been entered, the statistical commands will continue analyzing it until the data is modified or replaced. Options exist for evaluating data with missing values. It is also possible to restrict the data to only those observations where a certain set of circumstances occurs.

6.2 Table of Variable–Observation Combinations

The following is a table illustrating the various variable–observation data combinations that can be processed by Statpack. Letting the rows represent the number of observations and the columns represent the number of variables, one can easily determine if a specific variable–observation data combination is possible by simply determining the point where the variable line crosses the observation line. A "yes" indicates that the combination is possible; a blank indicates that Statpack cannot analyze the amount of data necessary for that variable–observation combination.

Number of variables

		1	2	3	4	5	10	15	20	25	30	40	50	75	100
	10	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	
	20	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	
	30	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes		
	40	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes		
	50	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes		
	60	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes		
n	70	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes		
u	80	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes		
m	90	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes		
b	100	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes		
e	125	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes			
r	150	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes				
	200	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes				
o	250	yes	yes	yes	yes	yes	yes	yes	yes	yes					
f	250	yes	yes	yes	yes	yes	yes	yes	yes						
	350	yes	yes	yes	yes	yes	yes	yes							
o	400	yes	yes	yes	yes	yes	yes	yes							
b	450	yes	yes	yes	yes	yes	yes	yes							
s	500	yes	yes	yes	yes	yes	yes								
e	600	yes	yes	yes	yes	yes	yes								
r	700	yes	yes	yes	yes	yes									
v	800	yes	yes	yes	yes	yes									
a	900	yes	yes	yes	yes	yes									
t	1000	yes	yes	yes	yes	yes									
i	1100	yes	yes	yes	yes	yes									
o	1200	yes	yes	yes	yes										
n	1300	yes	yes	yes	yes										
s	1400	yes	yes	yes											
	1500	yes	yes	yes											
	1600	yes	yes												
	1700	yes	yes												
	1800	yes	yes												
	1900	yes	yes												
	2000	yes													

6.3 List of Commands

DATA	—	Data input by terminal
FETCH	—	Read data from disk
FORM	—	Enter special input format
MANIP	—	Manipulate data in core (includes appending)
TRANS	—	Data transformations
STORE	—	Store data on disk
PRINT	—	Print selected variables on line printer
TYPE	—	Type selected variables on terminal
ACBNK	—	Access a stored data bank
MABNK	—	Create a bank from data in STP
SORT	—	Sort data into ascending order
MTA/I	—	Read data from magtape
DESC	—	Description of Data—means, st. dev., var.
BASIC	—	Medians, modes, and ranges

ERANA	—	Std. error of mean, coeff. of skewness, coeff. of var.
ESTAT	—	DESC BASIC ERANA
ZSCOR	—	Z scores
KOLM	—	1 or 2 sample Kolmogorov–Smirnov tests
CORR	—	Correlation matrix
PCORR	—	Partial correlations
KENDL	—	Kendall tau correlations
SRANK	—	Spearman rank correlation
PTBIS	—	Point biserial correlation
TTEST	—	T-test (significance between means)
CORRT	—	Correlated t-tests
MANN	—	Mann–Whitney U test
WILCX	—	Wilcoxon rank
ANOV1	—	Single-factor analysis of variance
ANOV2	—	2-way analysis of variance
1WAYR	—	1-way analysis of variance w/ repeated measures
ANOC1	—	1-way analysis of covariance
REGR	—	Regression
STEPR	—	Stepwise regression
FACTO	—	Factor analysis
PROB	—	Probability associated with t, f, or chi-square
CHISQ	—	Chi-square
CVSMT	—	Exponential curve-smoothing model
PLOT	—	Scatter plot
HIST	—	Histogram
BARGR	—	Bar graph
FREQ	—	Frequency
XTAB	—	Cross tab
XTAB*	—	Cross tab (table form—only if ASSIG is used)
PCENT	—	Percentiles
STOP	—	Restart
HELP	—	For commands
FINI	—	end run (only if ASSIG has been used)
INFO	—	General information
ASSIG	—	Assign output to line printer
DEASS	—	Reinitialize output to terminal
COPYS	—	Indicate more than 1 printer copy (ASSIG and PRINT)
TITLE	—	Label output with identification
NAME	—	Give names to variables
MAKE	—	Make a text to be inserted into lineprinter output

6.4 Program Transfer

Purpose: Initiate the run of another program while in STATPACK.

Description: STATPACK may be used to transfer control to another program (initiate a run to a different program). As the following programs become available, they may be called directly from STATPACK.

BANK

FREQ

TAB

CORL

REGR

When a call to another program is executed, the output file (if one has been created) is queued to the line printer, and the program specified is executed. To run a program type a "/" and the program name in response to "WHICH COMMAND?".

Example:

```
WHICH COMMAND?    /BANK
BANK?
```

CHAPTER 7

MULTIVARIANCE

7.1 Introduction

MULTIVARIANCE is a DEC–10 conversion of a program, originating at the State University of New York at Buffalo, which will perform a variety of linear multivariate analyses.

It will perform univariate and multivariate linear estimation and tests of hypotheses for any crossed and/or nested design, with or without concomitant variables. The number of observations in the subclasses may be equal, proportional, or disproportionate. The latter includes the extreme case of unequal group sizes involving null subclasses such as might arise in the application of incomplete experimental designs. MULTIVARIANCE also performs an exact least-squares analysis.

MULTIVARIANCE is an upgraded version of NYBMUL, incorporating the following additional features:

1. Screening of input data according to range limits or missing value codes.
2. Options to allow recoding of factor levels from value ranges rather than explicitly coding the levels 1,2, ... ,J and to label factor levels.
3. Features to facilitate the analysis of repeated measures data.
4. A table is now provided giving multivariate results for each between-group effect, to parallel the univariate estimates and tests produced by earlier versions of MULTIVARIANCE.
5. A VARIMAX rotation routine has been added for application to subsets of principle components of the correlation/covariance matrix or to the canonical correlation loadings.

The principle changes are:

1. The user may specify how many locations are needed for a run.
2. The factor identificaton,estimation specification and analysis selection cards are different.

The program is documented in:–

MULTIVARIANCE
Version VI, Release 2
International Educational Services.

(Copies of this manual may be purchased from the Centre.)

7.2 MULTIVARIANCE and SPSS

While SPSS contains facilities for performing analysis of variance and covariance, it does not claim to be a general program in this area. Its comprehensive set of procedures for data transportation, recoding, file manipulation, sampling and subgroup selection, as well as its great flexibility of data format, however, make SPSS useful for preparing files for input to MULTIVARIANCE.

For example, MULTIVARIANCE requires that the data be complete variate-wise. To solve this problem, SPSS can be used to either delete missing cases or implement values using subgroup means or estimated values using a regression model. Similarly, the requirement that factor levels be coded 1,2,3 ... could be met via RECODE operations. The output file from SPSS is finally prepared with the WRITE CASES card.

7.3 Using MULTIVARIANCE

To run the program, the user must supply the names of the input and output files and the number of words to allocate for workspace. This last value depends on the design and analysis required, but 4000 should be sufficient for most runs.

```
e.g.    .R STA:MULTIV
        What is the name of the input file?    XXX.DAT
        What is the name of the output file?    YYY.DAT
        Enter the value of param:    4000
```

On completion, YYY.DAT may be printed.

CHAPTER 8

TUTORIAL SYSTEM FOR COMPUTERS (TUSTAT)

8.1 Introduction

TUSTAT is a collection of over 80 interactive statistical programs. All programs within the suite are intended to be easy to use and require minimal knowledge of the computer system used. Data is input in small streams directly from the terminal, on prompting from the program. For cases where more data exists than can conveniently be entered from the terminal, some programs have alternative versions that use data files instead. Most programs provide detailed intermediate results from the algorithm used.

The two main areas of use for the suite are thus:

1. In introductory statistics courses where the emphasis is more on understanding and using the techniques involved, rather than on the details of the algorithm used.
2. For problem solving where the amount of data involved is small.

8.2 Available Programs

The main subject areas covered by the suite are:

1. Analysis of variance and covariance
2. Linear and non-linear regression
3. Distribution sampling
4. Multivariate analysis
5. Non-parametric statistics
6. Miscellaneous subjects (crosstabulation, Markov chains, etc.)

Further details of the programs available can be found in the manual

TUSTAT II
YOUNG O. KOH
SECOND EDITION
1973

8.3 Using TUSTAT Programs

8.3.1 *Program Names*

File names on the PDP–10 can be a maximum of only 6 characters in length. Since some of the programs in TUSTAT have names longer than this (i.e. TRIWAYF) their names have been abbreviated (i.e. to TRIWAF). Other program names have been left as in the manual. Details of the abbreviated names are as follows:

Old Name (as per manual)	*New Name*	*Old Name (as per manual)*	*New Name*
betadis	betads	binomia	binomi
canonic	canoni	canonif	canonf
chisqua	chisqu	cochran	cochra
conting	contin	covone1	cvone1
covone2	cvone2	covtwo1	cvtwo1
covtwo2	cvtwo2	crossta	crosta
discric	discri	discrif	discrf
eigenco	eigenc	eigenfi	eigenf
exponen	expone	factorc	factor
factorf	factof	forwayf	forwaf
friedma	friedm	gammadi	gammad
geometr	geomet	kendalc	kendac
kendall	kendal	kendalp	kendap

kosmone	kosone	kosmtwo	kostwo
krswall	krswal	latinsq	latnsq
lattice	lattic	latticf	lattif
lstcov1	lstcv1	lstcov2	lstcv2
lstmult	lstmlt	lstpoly	lstpol
multipf	mltipf	negativ	negati
nestone	neston	oriregr	orregr
peacorr	peacor	poisson	poisso
polynos	polnos	princif	princf
princip	princi	respons	respon
signtst	sgntst	snedeco	snedec
splotfi	spltfi	splplot	splplt
ssplfil	sspltf	stepfil	stepfi
stepreg	stepre	student	studen
triwayf	triwaf	uniform	unifor
wilcoxo	wilcox		

8.3.2 *Starting the Program*

To run most programs, the procedure is (after logging in)

```
.R BASIC
READY, FOR HELP TYPE HELP
OLD STA:??????
READY
RUN
```

where ?????? is the name of the required program.

For those programs that operate in so called 'file management' mode (i.e. TRIWAF), the method of data entry is to create a file of line numbered (from 04000) DATA statements and, prior to issuing the RUN command, cause this file to be appended to the program by use of the WEAVE command.

Example:

```
.R BASIC
READY, FOR HELP TYPE HELP
OLD STA:??????
READY
WEAVE DATA.DAT
READY
RUN
```

where DATA.DAT is the name of the data file.

8.3.3 *Leaving the* BASIC *System*

After having typed the R BASIC command the users terminal is under control of the BASIC system. One effect of this is that the normal sequence of two Control-C's will not return the user to Monitor command level, but rather to command level of the BASIC system. Thus if the user wishes to get back to Monitor level, he should, after typing the Control-C's and receiving the response READY from the BASIC system, issue the command MONITOR.

Example:

```
©©          where © denotes Control-C
READY       back at BASIC command level
MONITOR
.           Monitor prompt
```

CHAPTER 9

NUMERICAL CLASSIFICATION SUITES

9.1 CLUSTR and TAXAN

CLUSTR and TAXAN are the names given to a suite of programs which are capable of dealing effectively with sets of data which are to be numerically classified. The data represent several entities which are described by relevant attributes.

CLUSTR is suited to ecological and some taxonomic analyses and handles continuous, binary and ordered multistate data. It allows great flexibility in the method by which this data can be analysed.

TAXAN is suited to taxonomic studies and handles each of the above categories of data and also disordered multistate data. It does not, however, offer as large a degree of flexibility in analysis as does CLUSTR.

With CLUSTR the method by which the classification is performed may be controlled in a most flexible manner, by several easily set user options. These options control the following steps in the classification process:

1. A transformation of the raw data may optionally be carried out in one of several ways.
2. A choice of sorting and clustring strategies is available.
3. Output optionally available includes printouts of trellis diagrams, two way tables and summaries of the raw data, and plots of derived dendograms from the sorting strategies.
4. Optional Ordination derived from the methods of Principal Component and/or Principal Co-ordinate Analysis, may be selected.

The program as outlined performs both normal and inverse analyses of two-dimensional raw data in the form of entities versus attributes. Such data are commonly generated in psychological, taxonomic, and ecological studies and also in studies in other social sciences.

CLUSTR can also be used to classify three dimensional data (entity-1 by entity-2 by attribute) as is often required in ecological studies, for example in (sites by times by species) analysis. This extension in no way affects the two-dimensional study of data, and is entirely transparent to users of the latter facility.

The documentation for the suite (USERS MANUAL for CLUSTR) can be obtained from the Computer Centre. It details the format of the options and data cards, as well as the system control cards necessary to run CLUSTR on the PDP–10. Intending users should consult it for further details on the package.

9.2 CLUSTAN

Clustan is a suite of programs developed for the collective study of different cluster analysis methods and their particular application. It handles continous multi-state structures on a sample population of *n* objects.

It is a single overlayed program within which the procedures are subroutines called from the CLUSTAN driver. The latter reads procedure keyword cards which contain the names of the CLUSTAN procedures, punched starting in column 1. Each keyword initialises one CLUSTAN step, and a single CLUSTAN job can comprise any number of different steps.

9.2.1 CLUSTAN *and* SPSS *and* BMD

The following options can be used to link CLUSTAN with other analysis packages, using intermediate data sets for the storage of results:

(i) SPSS COMPUTE, RECODE and IF cards to transform data
(ii) SPSS WRITE CASES with file parameter JCH.
(iii) SPSS correlation matrix output with DISTIN input.
(iv) BMD07M output of canonical variables with FILE input.

The SPSS procedure may also be used to read a data matrix from an SPSS system file created by the SAVE FILE directive in SPSS.

9.2.2 *How to run* CLUSTAN

CLUSTAN may be used from batch or from a terminal. However batch operation is recommended because of the size and processor requirements of CLUSTAN. The commands necessary are:

```
.R STA:CLUSTN
*output filespec=CLUSTAN command
filespec [,data filespec]
  .
  .
  .
  .
```

where filespec is a standard DEC file specification of the form

DEV:filename.EXT[proj,prog,sfd,sfd,..].

DEV: defaults to DSK
.EXT defaults to .DAT
[PPN,path] defaults to your own

The output filespec is the file to which all CLUSTAN output is directed. If omitted this will default to FOR06.DAT.

The CLUSTAN command filespec is the file containing all CLUSTAN parameters and keywords. If this is omitted it will default to FOR05.DAT.

The data filespec is a file containing the data to be processed by CLUSTAN. The specification of a data file is optional and if omitted, data will be read from the CLUSTAN command file, unless parameter JCH in procedure FILE or IUNIT in procedure DISTIN is given a value. If JCH or IUNIT is specified it will be interpreted as a unit number and the file FORxx.DAT (where JCH=xx and cannot be less than 9) will be expected. The specification of a filename will take precedence over the JCH (or IUNIT) parameter and if both filespec and JCH (or IUNIT) are specified JCH (or IUNIT) is ignored.

Examples:

(i)

```
.R STA:CLUSTN
*RUN1=CLUST,OBS.DAT
```

will read CLUSTAN commands from CLUST.DAT, read data from OBS.DAT and produce results in RUN1.DAT,

whereas

```
.R STA:CLUSTN
*RUN2=NEWDAT
```

will expect data in the file NEWDAT.DAT together with the necessary CLUSTAN commands.

(ii)

```
$SEQUENCE
$JOB [10,105]/NAME:SMITH/PRIO:4/COST:
$DECK MYPROG.DAT
  .
  .
  .       (program and data cards)
  .
  .
$EOD
$TOPS10
.R STA:CLUSTN
*LPT: = MYPROG.DAT
$EOJ
```

9.2.3 *Files produced by* CLUSTAN

(a) The binary file CLSAVE.DAT is created and will contain the original data and the last similarity matrix generated, in a form suitable for use by the RESTART procedure. This avoids having to reload the data and recalculate the similarity matrix on every run.
(b) When any of the procedures HIERARCHY, CENTROID or DIVIDE are invoked a dendrogram 'deck' will be written to the file CLPLNK.DAT. Only the last dendrogram 'deck' is stored. CLPLNK.DAT is used subsequently in a run as input to procedure PLINK. It is also saved at the end of each run so the same dendrogram may be plotted at a latter time. It is possible to generate CLPLNK.DAT independently of CLUSTAN and use CLUSTAN only for plotting a dendrogram.

(c) If either of the plotting procedures PLINK or SCATTER are invoked the file CLUSA.PLT is created. This file must be subsequently submitted to the plotter.

(d) Invoking the procedure DUMP will produce the file CLDUMP.DAT. This will contain all classification arrays currently stored. Procedure RELOCATE may also output to CLDUMP.DAT.

(e) Any procedure purporting to punch data onto cards will create the file CLPNCH.DAT, which will contain the 'punched' data set.

(f) Procedure TREE may produce the file CLTREE.DAT if JUNIT is assigned a value.

There are two other files used extensively for workspace. These are CLSDCK.TMP and CLSDAT.TMP and are referred to in the CLUSTAN User's Manual as CLUSDECK and CLUSDATA respectively. Under most circumstances these files are deleted at the end of a run.

9.2.4 SPSS *System Files*

The procedure SPSS allows CLUSTAN to read and process files produced by the SPSS program. Data which has previously been input to SPSS to produce an SPSS System file can be processed by CLUSTAN directly from the system file. There are some limitations placed upon CLUSTAN relating to missing values in the system file and users are referred to the CLUSTAN Manual.

When the CLUSTAN procedure SPSS is invoked CLUSTAN will expect an SPSS System file named CLSPSS.DAT. The specification of an INFILE parameter in procedure SPSS will have no effect.