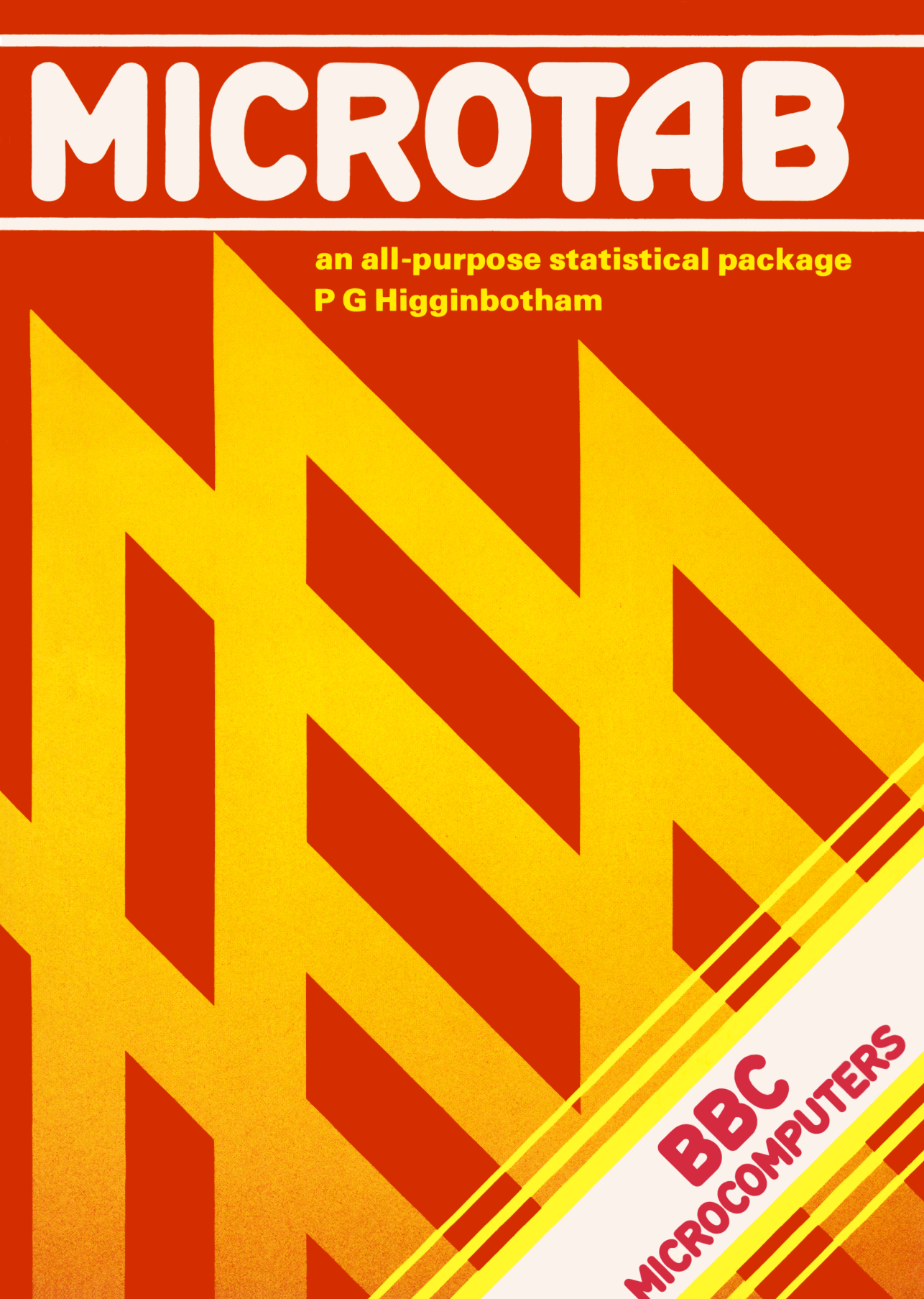
MICROTAB

an all-purpose statistical package

P G Higginbotham



MICROTAB

an all-purpose statistical package

P.G. Higginbotham
Computing Services, Sheffield City Polytechnic

Edward Arnold

Contents

A Brief Introduction to MICROTAB

MICROTAB is an easy-to-use interactive program for entering, editing, transforming, plotting, analysing and filing data on the BBC (Model B, B+ and Master Series) microcomputer. It is particularly aimed at students, scientists and researchers who need to carry out a variety of statistical procedures on small to medium-sized sets of data, although the range of facilities will make it useful to almost anyone who deals with figures.

MICROTAB will provide an ideal accompaniment for those taking an introductory or intermediate course in statistics. The primary aim of this manual, however, is not to teach you statistics but how to use the MICROTAB program.

Unlike some other systems, MICROTAB is an integrated package. Once some numbers have been entered, a whole range of operations can be performed upon them without having to type them in again.

MICROTAB stores data in a 'worksheet' of columns and rows, and its commands operate upon the data stored in the worksheet. The command language is based on natural English words and expressions which are more efficient and flexible than many 'menu' based programs. You can even develop your own shorthand for commands as you become more familiar with the package.

Just to give you the flavour of how the system works, here is a short series of MICROTAB commands, together with the resulting output. The lines of output from MICROTAB are indented:

```
READ HOURS INTO C1 AND RATE INTO C2
38 3.50
 35 4.2
40   3.5
42  3.5
39 3.25
END OF DATA

MULTIPLY  C2 BY C1 AND PUT WAGES INTO C3

PRINT C3

       C3
              133.00   147.00   140.00   147.00
              126.75

DESCRIBE C3

       C3        N=5  MEAN=138.7500   STD=8.8706
```

The first instruction (READ) tells MICROTAB to put the data from subsequent lines into columns 1 and 2 of its worksheet, with the first value on each line going into column 1 (C1), and the second going into column 2 (C2).

Following the READ command are five lines of data which, as you can see, are laid out in a fairly free-and-easy manner. The END command is an optional command to mark the end of the data.

The MULTIPLY command tells MICROTAB to go down columns 1 and 2 row by row, multiply each pair of values, and put the result in the corresponding row of column 3.

The PRINT and DESCRIBE commands tell MICROTAB to display the values in column 3, and to describe column 3 in terms of some simple statistics.

Getting Started with MICROTAB

The MICROTAB program consists of several modules which are loaded in succession. Before you begin, the computer should be running in BASIC. If your machine has extra ROM chips installed which make it start off in some other language, you may first need to type:

```
*BASIC
```

Throughout this manual, it is assumed that the RETURN key is pressed at the end of every command you type.

Users of the B+ and Master series of machines should also select the 'shadow' screen mode, for example by typing:

```
*SHADOW
```

LOADING THE PROGRAM

On a disk system:

1. Insert the disk into drive 0.
2. Either, press SHIFT and BREAK keys together, letting go of BREAK first.

   Or, type: CHAIN"MTAB"
3. Remove the disk and store it back in its sleeve.

On an Econet system:

The procedure may vary slightly depending on which version of Econet you have and also on how your system manager has organised things. All the parts of MICROTAB will probably have been copied to the Econet system's library directory (for example, by the Level 2 File Server program's Convert utility). To load the program you will probably need to type something like:

```
*NET
*I AM X
CHAIN"MTAB"
```

Your system manager will give you exact details of what to do.

THE WORKSHEET

When the program has finished loading and started running, it will begin by displaying a message something like:

```
** MICROTAB 1.0 **

20 columns, 50 rows.
```

This tells you that there is space in the worksheet for up to 20 columns, with up to 50 numbers in each column. The exact number of columns given may be more or less than 20 depending on which model of machine you are using and whether it has disk or Econet interfaces, second processor etc. (see also Appendix 5).

The . character is MICROTAB's prompt to tell you it is now ready for a command.

If you need columns longer than 50, or more columns which are shorter than 50, you can alter the worksheet format using the DIMENSION command to specify the maximum column length you require (up to 255). For example, the command:

```
DIMENSION 25
```

sets the worksheet to have columns of length 25, with more than 20 of them. The maximum number of columns or rows in the worksheet is 255.

Any DIMENSION command should be used BEFORE entering data.

SCREEN MODES

On the standard BBC Model B machine, MICROTAB will only operate in screen mode 7. The B+ and Master machines in 'shadow' mode, and the Model B with suitable extra hardware such as a 6502 second processor, can also use the other screen modes (0 to 6). The screen mode may be changed using the DIMENSION and VDU commands - see the 'Miscellaneous Commands' section on page 73 for more details. The WIDTH command (see page 10) may be used to set the maximum width of the output from MICROTAB.

General Rules for Using MICROTAB

COMMANDS

Every command that you give to MICROTAB should start with one of the recognised command names, and this may be followed by one or more 'arguments' which may be numbers, column names, or occasionally other types of information.

The HELP command will display a list of the names of all the available commands.

Commands should use upper case text (i.e. the CAPS LOCK should be engaged). A command name may be abbreviated to whatever is needed to distinguish it from any other command coming before it in the alphabet, to a minimum of two letters. For example, in the case of the commands MEAN and MEDIAN, ME would be the minimum abbreviation for MEAN and MED would be the minimum for MEDIAN. The first three letters will distinguish most commands but, if in doubt, use the full name.

If you make a mistake in giving a command, MICROTAB will 'bleep' and give you an error message. Appendix 2 gives information on what the different error messages mean.

ENTERING NUMBERS

Numbers entered into MICROTAB may be made up of the digits 0 to 9, the decimal point, the plus and minus signs, and the E character for exponential or 'scientific' notation. Commas or spaces may not appear within numbers. Examples of valid numbers are 99 1984 -22.981 and 1.7E3. Fixed numeric values are also referred to as 'constants'.

STORED CONSTANTS

Stored constants may be used to hold single numeric values and are referred to by the names K1, K2 etc. to K9. The LET command, described in detail on page 34, is used to place a value into a stored constant e.g.

```
LET K2=24.8765
```

Once a value has been assigned to a constant, the constant may be used in a command wherever the actual number would have been used. All the stored constants have an initial value of zero.

SPECIFYING COLUMNS

Columns are most simply referred to by a name of the form Cn where n is an integer number between 1 and the last column that exists in the current worksheet; for example, C1 and C12 are possible column names.

A consecutive series of columns may be referred to using a hyphen. For example, C1 C3-C6 C9 would refer to columns 1, 3, 4, 5, 6 and 9. There may be one or more spaces on either side of the hyphen, and spaces or commas should be used to separate individual column names.

Stored constants can be used to reference columns indirectly by using a name of the form CKn, where Kn is one of the constants K1 to K9. The current value of Kn is used to determine which column is being referred to. For example, if K7 has a value of 3, then referring to CK7 is the same as referring to C3. This form of notation is most useful when used in a file of stored MICROTAB commands - this is described more fully on page 36.

The NAME command (see page 9) may be used to give a column an alternative name of between 1 and 7 characters (letters and/or digits). Such names must always be put inside apostrophes, for example, 'AGE' 'BATCH3' and '1JAN85'. Once a column has been NAMEd, it can be referred to by either its C number, or its NAME. The NAME will be printed in tables, graphs and so on.

The greatest number of columns that may be mentioned in a command line is equal to either the maximum number of rows in the current worksheet or the maximum number of columns in the worksheet, whichever is the larger.

OTHER TEXT IN COMMAND LINES

With certain exceptions, such as the LET command, a command line may contain any other text (i.e. letters, spaces and commas) which may act as a reminder of what the command and its arguments are about. This extra text will be ignored by the program. In the introductory example on page 1, the first command could actually have been entered as RE C1 C2. To clarify the descriptions of commands and examples of their use, extra unnecessary text will often be included.

If a command line contains a # character, then the remainder of the line is ignored by the program. This is one way of adding comments to MICROTAB command lines. See also the NOTE command on page 74.

In the rest of this manual, the following conventions are used when specifying the format of commands:

C	for a column (Cn,CKn or 'NAME' format).
C...C	for a list of one or more column names.
K	for a numeric or stored constant (e.g. 61.7, K3).
E	for an item which can be either a C or a K.
[ ]	to enclose optional parts of commands.

In the formal description of each command, any extra text which is not necessary for its use is printed in lower case. In the examples of commands, such text is in upper case, exactly as it would be entered into the program.

The syntax of some commands contains several C, K or E terms. For example, the formal description of one form of the IRANDOM command (used to generate random integers) is:

IRANDOM K values between K and K and store in column C.

Each of the three K terms is a distinct value which will be supplied when the command is given

e.g. IRANDOM 20 VALUES BETWEEN 1 AND 99 IN C6

Entering Data into MICROTAB

SET the following data into column C

followed by data on one or more successive lines, is used to put values into a single column of the worksheet.

e.g.
```
SET DATA INTO C1
8 5.4
4
6  7
END OF DATA
```

would store the values 8, 5.4, 4, 6 and 7 in column 1, in that order. Values should be separated by spaces and/or commas, and the number of values per line may vary. Explicit numbers or names of stored constants may be entered in any combination. Data is stored until an END or other command is entered, or the number of values exceeds the maximum column length allowed in the current worksheet.

END of data

is an optional command to signify the end of the data. Any other valid command will have the same effect. There may be a slight delay when an END (or other command following data lines) is entered, while MICROTAB processes the data that have just been entered.

READ the following data into C...C

is used to enter data into several columns at once. Each line of data should contain one value for each of the specified columns.

e.g.
```
READ C2 AND C4
8 5.4
4  6
3 5
END OF DATA
```

would store the numbers 8, 4 and 3 in column 2 and the numbers 5.4, 6 and 5 in column 4.

If an incorrect number of values is entered on a data

line, an error message is given and the line is ignored. An END or other command signifies the end of the data.

Once a SET or READ command is given, any previous data in the specified column(s) will be lost, even if no data follows.

The MICROTAB prompt changes to a colon (:) whilst new data is being entered. In the event of an error message, the prompt character (: or .) will indicate whether data is still being accepted.

NAME column C as 'name', column C as 'name', ...

is used to give columns names which will appear in tables, graphs and so on. Each name may consist of between one and seven letters and/or digits and must always be enclosed in apostrophes (SHIFT 7 on the keyboard). Once a column has been NAMEd it may be referred to by either its name or the C-plus-number system. The same name may not be given to more than one column.

e.g. NAME C1 'AGE' C2 'HEIGHT' C3 'WEIGHT'

would name column 1 as 'AGE', column 2 as 'HEIGHT' and column 3 as 'WEIGHT'. The columns and names must be alternated as described. A command such as:

NAME C1-C2 'AGE' AND 'HEIGHT'

would be WRONG.

Displaying Data Values

PRINT the values stored in E...E

displays the values stored in the specified columns and/or stored constants. Any stored constants mentioned are printed first. If a single column is printed, then the values are displayed in a compact layout with several values on each line. If several columns are specified, they are printed side by side in columns. MICROTAB chooses a suitable format for displaying each column.

e.g. PRINT C3

```
          C3
                  23.15     44.20    194.83    347.93
                 446.70    375.00    624.11    787.00
```

PRINT C2-C4, 'SIZES' AND K6

```
          K6=3.14159265

          ROW        C2        C3        C4     SIZES
            1      1129   -0.0033   1.37E7      2.73
            2      3398    0.0650   8.76E6      3.06
            3       621    0.1800   2.98E7      5.19
            4      4046                        7.70
```

Because of their size, the values in column C4 are in scientific notation. The value 1.37E7 in the first row is 1.37 times (10 to the power of 7) or 13700000.

WIDTH of output is K characters [with K characters for columns]

sets the maximum width of the output from commands such as PRINT. Initially, most of the output from MICROTAB is tailored to fit the 40 character wide mode 7 display. You may want to change this if the output is directed to a printer (see page 11) or to a disk file (page 36).

The optional second argument to WIDTH sets the maximum number of characters used for displaying values by PRINT and the column and row statistic commands (pages 30-33). The initial setting is 9, but may range from 9 to 15. Larger settings may display values with more significant digits, or in normal rather than scientific format.

e.g. WIDTH OF OUTPUT IS 40, WITH 12 CHARACTERS FOR COLUMNS

Using a Printer

The output from commands such as PRINT can be directed to a printer as well as the screen. Certain types of printer (e.g. 'serial' printers operated through the computer's RS423 socket) need some preliminary *FX commands before use. These are fully described in Chapter 38 of your BBC Microcomputer User Guide and are probably best given at the start of a MICROTAB session.

On Econet systems, users of a communal printer-server will need to issue some commands (e.g. *PS 235 and *FX5,4) before using the central printer. Your system manager will tell you what you need to do.

After any preliminary setting up, the printer is operated by VDU commands (see page 74) or, more simply, by typing 'control characters' at the keyboard. These are entered by holding down the CTRL key and pressing B or C at the same time.

CTRL B activates the printer so that all further output appears on the printer as well as on the screen - commands as well as results.

CTRL C de-activates the printer - output is just to the screen.

Control characters may be entered anywhere on a command line. For example, if you type a CTRL B at the start of a command line, the whole of the line is sent to the printer, as well as any output resulting from the command. If a CTRL B is typed at the end of a command line, only the output from the command (if any) appears on the printer. A CTRL C is then needed if the next command line is not to go to the printer.

N.B. Once the printer has been activated, it must be switched to its 'on-line' condition, otherwise the computer's printer output buffer will become full and the program will 'hang-up' until the printer is switched on.

By default, information sent to the printer will not exceed the 40 character width of the mode 7 screen. To make full use of an 80 column printer, the WIDTH command (page 10) should be used:

```
e.g.  WIDTH OF OUTPUT IS 80 CHARACTERS
```

Although full use can now be made of an 80 column printer, some output on the screen may 'wrap around', giving a rather strange display until the width is restored to its normal value of 40.

Saving and Retrieving Data on Disk or Tape

The SAVE and RETRIEVE commands, described below, may not be used with the Econet level 1 filing system. Appendix 6 gives an alternative procedure for use with Econet level 1 systems.

SAVE in 'filename' [the data from columns C...C]

stores the specified columns in a file on disk or tape. If no columns are specified, all columns in the current worksheet which contain data are stored, together with the column names. The filename may be up to seven characters long and must be enclosed in apostrophes.

e.g. SAVE 'DATA3'
SAVE 'MYFILE' C7-C11

It is not possible to SAVE the stored constants K1 - K9.

RETRIEVE data from 'filename' [into columns C...C]

retrieves previously SAVEd data from the specified file. If no columns are specified, the whole file will be read back into the original columns from which it was stored. If columns are specified then data will be read from the start of the file into the specified columns. Column names will also be read back as long as they are different from any column names in the current worksheet.

e.g. RETRIEVE FROM 'FILE9' INTO C9 AND C10

reads the first two columns stored in 'FILE9' and puts them in columns 9 and 10 of the current worksheet.

Columns which are read back overwrite any corresponding columns already in the computer's memory. If the data being retrieved will not fit into the current worksheet dimensions, then as much data will be loaded as will fit, and a warning message displayed.

All other file-handling is performed by '*' commands which depend on which filing system is in use e.g.

*CAT displays the names of files on a tape or disk.
*DELETE deletes a specified disk file.

Further information on * commands is given in Appendix 3.

Generating a Series of Values

As well as entering data from the keyboard, it is often useful to be able to generate regular series of values for various purposes. The GENERATE command provides this facility.

GENERATE the first K integers into column C

stores the integers 1, 2, 3, ..., K in the specified column. K must be greater than zero and less than or equal to the maximum column length in the worksheet.

e.g. GENERATE 20 IN C3

puts the integers 1 to 20 inclusive into column 3.

GENERATE the integers from K to K into column C

stores a consecutive series of integers in a column.

e.g. GENERATE -3 TO 2 IN C4

would store the numbers -3, -2, -1, 0, 1, 2 in column 4.

GENERATE values from K, in steps of K, to K and store in C

will produce any series as long as the K values are logical and in the right order. Don't forget to make the step size negative if a descending series is required.

e.g. GENERATE FROM 0 STEP .5 TO 10 IN 'SIZES'

stores the series 0, 0.5, 1.0, 1.5, ..., 9.0, 9.5, 10.0 in the column previously named 'SIZES'.

More complex sequences of values may be generated by use of the LET command which is described in detail on pages 34 and 35.

e.g. LET C3 = 1 + ROW MOD 3

stores the values 1, 2, 3, 1, 2, 3, 1 ... in column 3.

LET C6 = (ROW + 2) DIV 3

stores the values 1, 1, 1, 2, 2, 2, 3, 3 ... in column 6.

Generating Random Values

MICROTAB has several commands which generate sets of 'random' values whose distributions are statistically interesting, such as the normal and binomial distributions.

URANDOM K values and store in column C

generates K random values from a Uniform distribution in the range from 0.0 to 0.99999

e.g. URANDOM 30 IN C6

IRANDOM K values between K and K and store in column C

generates Integer (whole-number) values in the specified range. The third K value should be greater than or equal to the second.

e.g. IRANDOM 30 VALUES FROM 1 TO 99 IN C12

NRANDOM K values with mean of K, stdev of K, store in column C

generates Normally-distributed random values with the specified mean and standard deviation. The command uses the 'Box-Muller' method to generate values. The mean and standard deviation of the resulting column will only approximate to those specified in the command.

e.g. NRANDOM 25 MEAN=100, STD=15 AND STORE IN 'IQ'

generates 25 values in the column previously named 'IQ', with a mean of around 100 and a standard deviation of around 15.

BTRIALS of K Bernoulli trials for p=K and store in column C

generates a set of Bernoulli trials - values which are either zeroes or ones. The probability of a given value being a one is p, where p lies between 0.0 and 1.0 inclusive. With p = 0.5, the command could be used to simulate a 'coin-tossing' experiment - values of zero

might be 'heads' and one might be 'tails'.

e.g. BTRIALS 10 P=0.5 INTO C6

would store 10 values of either 0 or 1 in column 6.

BRANDOM K binomial values with n=K and p=K, store in column C

generates a series of values, each of which is the number of 'ones' from a set of n Bernoulli trials - the probability of a 'one' being p. To put it another way, each value is the sum of the values from a single BTRIALS command.

e.g. BRANDOM 20 BINOMIALS FOR N=10 AND P=0.5 INTO C7

would perform 20 of the 'coin-tossing' experiments from the previous example, each value being between 0 and 10.

BASE value for random-number generator is K

is used to 'seed' the random-number generator used in the above commands to a setting based on K. Repeating an identical IRANDOM command for example, would normally produce a different set of values each time. The same values could be obtained if each IRANDOM command was preceded by a BASE command with a fixed value of K.

e.g. BASE VALUE IS 999

Data Editing Commands

MICROTAB has a variety of commands for copying, correcting, modifying and selecting data values. One of the simplest is the COPY command:

COPY columns C...C into C...C

copies each column in turn from the first list into the corresponding column in the second list.

e.g. COPY C3 INTO C7

copies column 3 to column 7; column 3 is unchanged.

COPY C1 AND C2 INTO C12 AND C9 RESPECTIVELY

copies column 1 into column 12, and column 2 to column 9.

The number of columns mentioned in the command should thus be an even number, although if only one column is given, it is copied to itself.

N.B. Care should be taken with COPY and other commands which can transfer values from one set of columns to another. These commands may not work quite as expected if partially overlapping sets of columns are specified.

e.g. COPY C1-C6 INTO C4-C9

will copy C1 into C4, then C2 into C5, then C3 into C6, then the <u>new</u> C4 into C7 etc.

The next two commands are used for making changes and additions to the numbers in columns, and have several different versions:

SUBSTITUTE the value K in row K of column C

replaces a single existing value with a new value. The specified row should be between 1 and the number of values already stored in the column.

e.g. SUBSTITUTE 2.7 IN ROW 4 OF C5

would store 2.7 in row 4 of column 5, replacing the previous value:

C5		C5
8.0		8.0
3.4	=>	3.4
1.9		1.9
9.3		2.7
5.5		5.5

SUBSTITUTE the value K in row K of column C, store in C

e.g. SUBST 0.2 IN ROW 3 OF C5, STORE IN C8

takes the values from column 5, replaces the value in row 3 with the value 0.2, then stores the amended column in C8. Column 5 is left unchanged.

C5		C5	C8
8.0		8.0	8.0
3.4	=>	3.4	3.4
1.9		1.9	0.2
9.3		9.3	9.3
5.5		5.5	5.5

The previous example could equally well have been:

SUBSTITUTE 2.7 IN ROW 4 OF C5, STORE IN C5

SUBSTITUTE the values K...K in row K of C...C store in C...C

makes several substitutions at once, with the option of storing the amended columns elsewhere in the worksheet.

e.g. SUBST 65 AND 50 IN ROW 3 OF C2 AND C4, STORE IN C8 AND C7

C2	C4		C8	C7
25	59		25	59
44	91	=>	44	91
16	88		65	50
38	40		38	40

In this example, columns C2 and C4 remain unaffected.

If the amended columns are to be stored back in their original positions, the appropriate command would be:

SUBST 65 AND 50 IN ROW 3 OF C2 AND C4, STORE IN C2 AND C4

INSERT the value K in row K of column C

inserts a single extra value into the specified row and column. The existing values are moved to create a gap for the new value. The maximum row number that can be used is the current length of the column + 1, in which case the new value is added to the end of the column.

e.g. INSERT 2.7 IN ROW 4 OF C5

C5		C5
8.0		8.0
3.4	=>	3.4
1.9		1.9
9.3		2.7
5.5		9.3
		5.5

INSERT the value K in row K of column C, store in C

INSERT the values K...K in row K of C...C store in C...C

work like the corresponding versions of the SUBSTITUTE command, except that the new row of values is added into a column rather than replacing existing values.

The next two commands again pick out values by row number, and like INSERT and SUBSTITUTE can operate on one or more columns.

DELETE row K [to row K] from column C

removes the specified row(s) from the column and closes up the gap between the remaining values.

DELETE row K [to row K] from columns C...C
and put the rest in C...C

takes each column from the first set, deletes the specified rows, then puts the remaining rows in the corresponding column from the second set. The total number of columns mentioned should be an even number.

e.g. DELETE ROWS 2 TO 4 FROM C1 AND PUT REST IN C2

would delete rows 2 to 4 inclusive from the values in column C1 and copy the remainder to column 2. In this case, C1 is unchanged afterwards, and C2 will contain 3 fewer values than the original column:

C1		C2
8		8
3	=>	5
1		2
9		
5		
2		

PICK row K [to row K] from column C

PICK row K [to row K] from columns C...C and put into C...C

PICK is the converse of DELETE. It specifies the rows to be kept rather than the rows to be deleted.

e.g. PICK ROWS 2 TO 3 FROM C2

C2		C2
6		9
9	=>	2
2		
7		

In this example, the new values are placed back in the original column.

The OMIT and CHOOSE commands select rows by value rather than position. In addition, the rows selected from the first column are the ones taken from subsequent columns. Look at the examples to make sure you understand this principle, as it is used in other commands such as SAMPLE and SORT described later.

OMIT values of K [through to K] in C [and matching rows from C...C] and put into C [with matching rows into C...C]

e.g. OMIT 2 IN C1 WITH C2 AND C3, INTO C7-C9

C1	C2	C3		C7	C8	C9
1	5	14		1	5	14
2	9	8	=>	1	6	0
1	6	0		1	9	3
1	9	3				
2	7	1				
2	0	2				

This procedure can be of use in separating out sub-groups from a column. In the above example, column 1 might contain a set of code numbers corresponding to the sex (e.g. 1=male, 2=female) of the individuals whose data is

in each row of columns 2 and 3. Following the command, columns 7 to 9 contain the scores for males only - the females having been omitted.

CHOOSE values of K [through to K] in C [and matching rows from C...C] and put into C [with matching rows into C...C]

CHOOSE is the converse of OMIT, specifying the rows to be kept rather than the rows to be dropped.

e.g. CHOOSE 2 IN C1 WITH C2 INTO C3,C4

C1	C2		C3	C4
1	5		2	9
2	9	=>	2	7
1	6		2	0
1	9			
2	7			
2	0			

RECODE values of K [through to K] in C...C into the value of K in C...C

converts values in the specified range to a new value. Other values are just copied to the new column(s).

e.g. RECODE 9 TO 99 FROM C3 INTO 9 IN C6

C3		C6
5		5
23		9
4	=>	4
6		6
41		9
2		2

N.B. It is occasionally possible for the OMIT, CHOOSE and RECODE commands to give unexpected results due to slight 'rounding' errors in the computer's in-built arithmetic when performing complex calculations. For example, the result of a calculation may appear to be 1.5 when actually it is stored as 1.5000001. Such values may fail to be selected when trying to choose values of 1.5. This problem should be rare but if it does occur it can be cured by selecting values from say 1.49 to 1.51.

JOIN E to the bottom of E [to the bottom of E, ..., of E]
and put into C

concatenates the specified columns and/or constants. Each new item gets added to the top of the accumulating column.

e.g. JOIN C1 7 C2 INTO C3

C1	C2		C3
1	5		5
2	9	=>	9
1			7
			1
			2
			1

ERASE the contents of C...C

erases the contents and name of the specified column(s).

e.g. ERASE C10, C14-C16

If unwanted columns are ERASEd they are not stored when the whole worksheet is SAVEd. It is not necessary to ERASE a column before putting new values into it. Commands such as SET, READ, RETRIEVE etc. automatically erase the previous contents of a column before storing new values.

Ranking and Sorting Commands

ORDER the values in C...C and put in C...C

rearranges the values in each of the specified columns into ascending order and stores the sorted values as specified. If only one column is mentioned, the ordered values are stored back in the same column.

e.g. ORDER C1 AND C2 INTO C3-C4

C1	C2		C3	C4
2	5		1	4
4	9	=>	2	5
1	8		3	8
3	4		4	9

It is possible to put the values in a single column into descending order by multiplying the values in the column by -1, ordering the new values, then multiplying again by -1. The MULTIPLY command (page 25) may be used.

e.g.
```
MULTIPLY C1 BY -1 INTO C1
ORDER C1
MULTIPLY C1 BY -1 INTO C1
```

SORT the values in C [with matching rows from C...C] and put into C [with matching rows into C...C]

re-orders the rows of each column but, as with OMIT and CHOOSE (pages 19 and 20), the result from the initial column is used to sort the remaining columns.

e.g. SORT C1-C4

C1	C2		C3	C4
2	5		1	8
4	9	=>	2	5
1	8		3	4
3	4		4	9

RANKS for values in C...C and store in C...C

calculates ranks for each column, with the smallest value being given a rank of 1, next smallest 2, etc. Tied values are given an average rank. If only one column is

mentioned, the ranks are stored back in the column.

e.g. RANKS C1 C2 INTO C3 C4

C1	C2		C3	C4
9	5		4	2.5
7	9	=>	2	4
8	4		3	1
3	5		1	2.5

As with ORDER, the ranks can be reversed (1 for largest etc.) by first multiplying the values by -1. The LET command (page 34) may be used as an alternative to MULTIPLY.

e.g. LET C3 = -C1
RANKS FOR C3

SAMPLE [K] values from C [with matching rows from C...C] and store in C [with matching values in C...C]

chooses rows at random - without replacement - from the first column specified, and takes corresponding rows from other columns. If K is omitted, the whole column is used. The columns being sampled should all be equal in length and contain at least two values. If only one column is mentioned, the new values are placed back in the column.

e.g. SAMPLE C1-C2 C3-C4

C1	C2		C3	C4
1	5		3	8
2	9	=>	1	5
3	8		4	4
4	4		2	9

SAMPLE can thus randomise the rows of a column. Together with the GENERATE command, it may be used to produce randomised lists of values:

e.g. GENERATE 20 C1
SAMPLE C1 C1
PRINT C1

would display a list of the numbers from 1 to 20 in randomised order from column 1.

Column Arithmetic Commands

Various arithmetical operations can be carried out on constants and/or columns on a row by row basis, with the resulting values being placed in a column.

If the columns being operated upon are of unequal length, then the new column will have the length of the shortest of the original columns.

If only constants are being operated upon, the new column will contain the maximum number of rows allowed in the current worksheet.

ADD E to E [to E, ...] and put sum in C

adds the values from the specified columns/constants and stores the sums.

e.g. ADD C3 C8 100, STORE IN C6

C3	C8	100		C6
5	4			109
2	1		=>	103
6	2.5			108.5

SUBTRACT E from E and put differences in C

subtracts the first column/constant from the second, and stores the differences.

e.g. SUBTRACT C2 100 C1

C2		C6
24		76
31	=>	69
15		85
66		34

Note that the minimum abbreviation for SUBTRACT is SUBT (so as to distinguish it from SUBSTITUTE).

MULTIPLY E by E [by E, ...] , and put products in C

e.g. MULT C1-C4 C10

DIVIDE E by E put result in C

An error will result and no values will be stored if any of the values for the divisor are zero.

RAISE E to the power of E and put result in C

raises values to the specified power. An error will result and no values will be stored if, for example, you try to raise negative values to a power of less than one.

ADD and MULTIPLY are of most use where the arguments involve consecutive sets of columns so that the hyphen convention may be used.

More complex column arithmetic operations can be performed using the LET command described on page 34.

Function Commands

A variety of mathematical and trigonometrical functions can be applied to the values in a column. One use of such functions is to transform the values in a column to make them more suitable for use in statistical tests which often make assumptions about normality of distribution, homogeneity of variance and so on.

Function commands all have the format:

FUNCTION of E with resulting values put in C

e.g. SQRT OF C5 INTO C2

would take the square root of each row of C5 and put the result in the corresponding row of C2.

If E is a constant, the whole column is filled with the resulting value.

The functions available are:

ABSOLUTE gives the absolute (positive) value of a number.

e.g. ABSOL C1 INTO C2

C1		C2
3		3
0	=>	0
-5		5

ROUND rounds values to the nearest integer.

e.g. ROUND C1 C2

C1		C2
5.3		5
-5.3	=>	-5
3.7		4

SIGNS gives the signs of values: -1 for negative values, 0 for zero values, and +1 for positive values.

e.g. SIGNS C1 IN C2

C1		C2
5.3		1
-5.3	=>	-1
0		0

SQRT takes the square roots of values. Taking the square root of a negative value will produce an error message, and no result will be stored.

LOG takes logarithms to base 10. Logs of values less than or equal to zero will produce an error message, and no result will be stored.

LN takes logs to base e. The restrictions for LOG apply.

EXPONENT calculates e to the power of the specified E values.

DEGREES converts from radians to degrees.

RADIANS converts from degrees to radians.

COSINE computes the cosine function. The argument must be in radians so use the RADIANS command first if necessary.

SINE computes the sine function - argument is in radians.

TANGENT computes the tangent function - argument is in radians.

ACS computes the inverse cosine function. The result is in radians so use the DEGREES command to convert to degrees if required.

ASN computes the inverse sine function - result is in radians.

ATN computes the inverse tangent function - result is in radians.

The ZSCORES and NSCORES commands are rather like function commands, but are different in that their arguments must only be columns. As with some of the data-editing commands, ZSCORES and NSCORES can operate on several pairs of columns at once.

ZSCORES of C...C store in C...C

calculates the standard or 'z' scores for the values in each column and stores them as specified. The z scores for the values in a column are calculated as:

$$\frac{X - \bar{X}}{s}$$

where $\bar{X}$ is the column mean
s is the column standard deviation
X is an individual value from the column.

Each value in the resulting column of z scores is therefore the number of standard deviations that the original value was from the original column mean. A column of z scores has a mean of 0.0 and a standard deviation of 1.0.

e.g. ZSCORES OF C3 INTO C1

C3		C1
2.4		-0.50214
0.6		-1.11286
8.3	=>	1.49964
5.1		0.41393
3.0		-0.29857

NSCORES of C...C store in C...C

computes 'normal' scores for each column. These are the values that would be expected from sampling a normal population having the same size as the original column, and are based on the ranks rather than the sizes of the values. MICROTAB calculates a normal score as the inverse normal value of

$$\frac{R - 0.375}{N + 0.25}$$

where N is the number of scores in the column
R is the rank of the individual value.

e.g. NSCORES C3 C2

C3		C2
2.4		-0.49446
0.6		-1.17804
8.3	=>	1.17804
5.1		0.49446
3.0		0.00000

In conjunction with other commands, NSCORES could be used to examine whether data values are normally distributed:

e.g.
```
NSCORES OF C1 INTO C2
PLOT C1 AGAINST C2
CORRELATE C1 AGAINST C2
```

The PLOT command (page 42) would produce a scatter-plot of values against their normalised values. A normally-distributed set of scores will give a more or less straight line relationship, whereas non-normal samples will produce a more curved line.

The CORRELATE command (page 58) could be used to give a more precise measure of the straightness of the plot and thus the degree of normality.

Column Statistic Commands

Column statistics compute a single value describing the values stored in a column and print it on the screen. All the commands have the general form:

STATISTIC for columns C...C

where STATISTIC is one of COUNT, AVERAGE, MEAN, STDEV, SUM, SSQ, VARIANCE, MEDIAN, MINIMUM and MAXIMUM.

COUNT gives the number of values stored in the specified column(s).

e.g. COUNT C1 C3

```
C1      N=5
C3      N=12
```

AVERAGE gives the arithmetic mean for each column.
MEAN also gives the arithmetic mean.

SUM gives the sum of the values in each column.

SSQ gives the sum of the squared values in each column.

VARIANCE gives the sample variance for each column. This is calculated using the formula:

$$\frac{\Sigma(X - \overline{X})^2}{(N - 1)}$$

where $\overline{X}$ is the column mean
X is a value in the column
N is the number of values in the column.

STDEV gives the sample standard deviation for each column. This is the square root of the variance defined above.

MEDIAN gives the middle value from a rank-ordered arrangement of the column. With an even number of values, the median is given as the average of the two values either side of the middle position.

MINIMUM gives the smallest value from each column.

MAXIMUM gives the largest value from each column.

DESCRIBE columns C...C

can be used to obtain some of the above statistics in a more compact form. By default, the name, count, mean and sample standard deviation are displayed.

e.g. DESCRIBE C1 C2

```
    C1          N=5 MEAN=183.00  STD=20.70

    SPEED       N=7 MEAN=24.364  STD=4.235
```

NOBRIEF output from commands

produces extended output from the DESCRIBE command.

e.g. NOBRIEF
DESCRIBE C1

```
  C1      N=5
        MEAN=183.00
         STD=20.70
         SUM=915.00
         SSQ=169159.00
         VAR=428.50
        MEDN=188.00
         MAX=207.00
         MIN=151.00
```

BRIEF output from commands

produces reduced output from the DESCRIBE command. This is the initial setting for the output.

INFO on the contents of the worksheet

gives information on the size of the current worksheet, the values of stored constants which are not zero, and a list of the names and counts of all the columns which contain data. INFO can be useful if you have forgotten the names of columns, or which columns are still empty.

Storing Column Statistics

Sometimes, it is useful to be able to store column statistics as data items in themselves. For example, a column could contain the individual sales made by a particular salesman in a week. The mean of each column would therefore represent the average sale in the week by each salesman. If all the column means were put into a new column, a histogram, for example, could then be plotted.

The PUT command has the general form:

PUT STATISTIC for columns C...C into column C

where STATISTIC is one of COUNT, AVERAGE, MEAN, SUM, STDEV, SSQ, VARIANCE, MEDIAN, MINIMUM and MAXIMUM.

e.g. PUT AVERAGE OF C1-C3 IN C10

C1	C2	C3		C10
1	5	3		2.5
2	9	4	=>	5.4
3	8	5		4
4	2	4		
	3			

The number of column statistics to be stored cannot, of course, be greater than the maximum column length in the current worksheet.

Row Statistics

As well as providing summary statistics for columns, MICROTAB provides a corresponding set of operations for the rows of the worksheet.

The general form of the commands is:

RSTATISTIC for columns C...C

where RSTATISTIC is one of RCOUNT, RMEAN, RSUM, RSTDEV, RSSQ, RVARIANCE, RMEDIAN, RMINIMUM and RMAXIMUM.

The result of a row statistic command is always a column of values.

e.g. with the columns:

C1	C2	C3
3	7	2
4	11	3
5		4

RMEAN C1-C3

would display:

ROW 1 MEAN= 4

ROW 2 MEAN= 6

ROW 3 MEAN=4.5

Notice that where columns are of unequal length, the row statistic will be based only on those columns which have a value in the particular row.

As with the column statistics, row statistics may themselves be stored in a column. This is again done using the PUT command:

PUT RSTATISTIC for C...C into C

e.g. PUT RSUM C1-C3 C4

for the above data would store the values 4, 6 and 4.5 in C4.

The LET Command

The LET command is a powerful command for performing more complex operations on columns in a row-by-row manner. The general form of the command is:

```
LET E = expression
```

The word LET and the equals sign must be included.

If the expression evaluates to a single value, E may be a stored constant or a column name. If a stored constant is used, the result is stored and also displayed on the screen. If a column is used, the whole column is filled with the resulting value.

If the expression evaluates to a column of values, then E should be a column name; if E is (wrongly) a stored constant, a value of zero will result.

The expression can consist of any legal combination of column names, numbers, stored constants, arithmetical operators, column functions (excluding NSCORES and ZSCORES), column statistics, PI etc. but NO extra text or comments.

Column names may be in Cn, CKn or 'name' format.

Constants can be numbers in any format, or stored constants.

Arithmetical operators (in decreasing order of precedence) can be:

-	Unary minus
()	Brackets
^	To the power of
DIV	Integer division (as in BBC BASIC)
MOD	Integer remainder (as in BBC BASIC)
*	Multiplication
/	Division
+	Addition
-	Subtraction

Column statistic and function names can be any of those described previously except for NSCORES and ZSCORES, and the usual rules of name abbreviation apply. The argument to a column statistic or function name must be inside brackets which open immediately following the command name. The argument to a column statistic must be a single column name.

e.g.
```
LET K1=98.4
LET C2 = 1/C1
```

```
LET 'DIFFS' = C1-MEAN(C1)
LET K2=STDEV('AGES')
LET K5= (SUM(C1)^2-SSQ(C1))/COUNT(C1)
```

In conjunction with the GENERATE command, compound-interest types of calculation may be performed e.g. the growth in 100 pounds at 5% interest over 20 years:

```
GENERATE 20 INTEGERS IN C1
LET C2=100*(1.05^C1)
```

The right-hand side of a LET expression may reference a value in a particular row and column in the form C(column,row)

e.g. `LET K2=100+C(3,1)`

In addition, the special variable ROW may be used to represent the row number as each row of a result column is evaluated

e.g. `LET C1=ROW + 1`

stores 2, 3, 4, ... in C1 (filling the whole column).

`LET C2=100*(1.05^ROW)`

is analogous to the compound-interest calculation above.

Finally, two special functions - FNG and FNL - calculate the greater and lesser respectively of a pair of values. The values to be compared are placed in brackets separated by a comma.

e.g. `LET K1=FNG(MEAN(C1),100)`

stores in K1 the greater of 100 and the mean of C1.

A series of LET commands could be used to calculate statistics not directly available such as SKEWNESS and KURTOSIS. If the column we want these statistics for is C9, and C1 and C2 may be overwritten, the following series of commands is required:

```
LET C1=C9-MEAN(C9)
LET C2=C1^2
LET K2=MEAN(C2)
LET C2=C1^3
LET K3=MEAN(C2)
LET C2=C1^4
LET K4=MEAN(C2)
NOTE K5 WILL BE SKEWNESS, K6 WILL BE KURTOSIS
LET K5=K3/K2/SQR(K2)
LET K6=K4/K2/K2-3
```

The NOTE command is used to add comments (see page 74).

Storing MICROTAB Commands in a Disk File

A series of MICROTAB commands may be stored in a disk file using the disk filing system *BUILD command, and called up when required using the *EXEC command. For example, to create a file called 'DEMO', type:

```
*BUILD DEMO
ADD C1 TO C2 PUT INTO C3
DESCRIBE C1-C3
ESCAPE (i.e. press the ESCAPE key)
```

Then to retrieve and execute 'DEMO', just type:

```
*EXEC DEMO
```

Such a procedure is also possible using the tape filing system, but is more difficult as no *BUILD command exists (unless you also have the DFS ROM chip fitted). Command files have to be created outside the MICROTAB system. An example of a BASIC program to do this is given in Appendix 3.

Further information about *BUILD, *EXEC etc. can be found in Appendix 3, Chapter 35 of the BBC Microcomputer User Guide, and in manuals for the Disk Filing System.

When commands are stored in a file, it is generally more useful to make them work for any set of columns rather than specific ones as used in the example above. This can be done by referring to columns indirectly with a name of the form CKn, where Kn is one of the stored constants K1 to K9. The current value of Kn determines which column is used.

For example:

```
NOTE EXPRESS COLUMN VALUES AS % OF COLUMN TOTAL
LET K1=4
LET CK1=CK1/SUM(CK1)*100
DESCRIBE CK1

        C4      N=20     MEAN=93.5        S.D.=7.37
```

```
NOTE CALCULATE THE RANGE OF THE VALUES IN A COLUMN
LET K1=3
SORT CK1 C10
LET K2=C(K1,COUNT(CK1))-C(K1,1)

       RESULT=9.5
```

A more general-purpose version of the skewness and kurtosis example given above may now be written. If K1 contains the number of the column to be examined, and K8 and K9 contain the numbers of columns which can be overwritten, the following commands could be stored in a (disk) file called 'SKURT':

```
*BUILD SKURT

LET CK8=CK1-MEAN(CK1)
LET CK9=CK8^2
LET K2=MEAN(CK9)
LET CK9=CK8^3
LET K3=MEAN(CK9)
LET CK9=CK8^4
LET K4=MEAN(CK9)
NOTE K5 WILL BE SKEWNESS
LET K5=K3/K2/SQR(K2)
NOTE K6 WILL BE KURTOSIS
LET K6=K4/K2/K2-3

ESCAPE
```

To calculate the skewness and kurtosis of columns 1 to 3, the following sequence might then be used:

```
LET K8=11
LET K9=12
LET K1=1
*EXEC SKURT
LET K1=2
*EXEC SKURT
LET K1=3
*EXEC SKURT
```

A further use of a stored command file is to perform a series of initialisation commands, such as configuring a printer, dimensioning the worksheet, or programming the user-defined function keys (see the following section).

The Programmable Function Keys

The BBC micro has a set of ten programmable function keys. These can be programmed with any single-line MICROTAB command

e.g. `*KEY 0 "HELP ¦M"`

Henceforth, pressing the f0 key will perform a HELP command. The ¦M sequence performs an automatic RETURN.

Some of the function keys have already been programmed for you. These are:

```
f0  HELP
f1  INFO
f4  WIDTH 40
f8  WIDTH 80
```

You are, of course, free to re-program these keys if you want.

Function keys can be used effectively in conjunction with stored command files. Again using the previous skewness and kurtosis example, you could save some effort by typing:

```
*KEY 5 "*EXEC SKURT ¦M"
*KEY 6 "LET K1=K1+1 ¦M"

LET K8=11
LET K9=12
LET K1=1
f5
f6
f5
f6
f5
```

(f5 and f6 represent pressing function keys f5 and f6 respectively)

In this particular example, the command "LET K1=K1+1" could have been included as the final line of the command file 'SKURT'. K1 could, of course, start with a value other than 1.

Chapter 25 of the BBC Microcomputer User Guide contains further information on the programmable function keys.

Tables and Graphs

MICROTAB has several commands to tabulate data and produce graphs of various kinds. The PRINT command which displays raw data values has already been described

e.g. PRINT C1

```
        C1
             5       3       5       5       5       3       2
             9       4       1       7       7       4       3
```

HISTOGRAM of the values in column C
[with the first mid-point at K [and interval-size K]]

produces a frequency histogram for the column. The first mid-point and the interval-widths may also be specified, although the resulting number of intervals must not be greater than the maximum number of rows in the worksheet.

e.g. HISTOGRAM OF C1

```
          C1
            MID-INT FREQ
                1.5    1 *
                2.5    1 *
                3.5    3 ***
                4.5    2 **
                5.5    4 ****
                6.5    0
                7.5    2 **
                8.5    0
                9.5    1 *
```

The first 'bar' of this histogram includes values that are equal to or greater than 1.0 and less than 2.0. The mid-point of this interval is 1.5. If you wanted the mid-points of the intervals to be 1, 2, 3 etc. rather than 1.5, 2.5, 3.5, the appropriate command would be:

HISTOGRAM OF C1, FIRST MIDPOINT AT 1.0

If the first mid-point is greater than some of the values in the column, then these are omitted from the display.

PUT HISTOGRAM of the values from column C
[first mid-point at K [interval-size K]] in column C

suppresses the display and stores the frequency counts (including zeroes) in the specified column.

TABLE the data in column C

produces a frequency table of values in a column. Also provided are cumulative frequency, percent frequency, and cumulative percentage frequency.

e.g. TABLE OF C1

VALUE	FREQ	CUM-F	%	CUM-%
1	1	1	7.14	7.14
2	1	2	7.14	14.29
3	3	5	21.43	35.71
4	2	7	14.29	50.00
5	4	11	28.57	78.57
7	2	13	14.29	92.86
9	1	14	7.14	100.00

PUT TABLE of the values from column C into column C

suppresses the display and stores the frequency counts in the specified column. Unlike PUT HISTOGRAM, zero values are never stored.

PARSUMS of the values in C...C into C...C

gives the partial sum, for each row of a column, of all the previous values in the column. This can be used to obtain cumulative frequencies or cumulative percentages down a column.

e.g. PARSUM C1 INTO C2

C1		C2
1		1
1		2
3		5
2	=>	7
4		11
2		13
1		14

PARSUMS with sequence length of K of C...C into C...C

produces the partial sum, for each row of a column, of the K values up to and including the current row. This provides a 'moving sum' down the column. A 'moving average' may then be obtained by dividing each value in the new column by the smaller out of its row-number and the sequence length using the FNL function (page 35).

e.g. PARSUMS OF LENGTH 4 FOR C1 INTO C2
LET C3=C2/FNL(ROW,4)
PRINT C1-C3

ROW	C1	C2	C3
1	21	21	21.00
2	17	38	19.00
3	25	63	21.00
4	29	92	23.00
5	26	97	24.25
6	33	113	28.25
7	26	114	28.50
8	29	114	28.50
9	23	111	27.75
10	23	101	25.25
11	17	92	23.00
12	19	82	20.50

The first (K-1) rows of the moving sum and moving average columns are, of course, based upon fewer values than the rest of the column.

PARSUMS using a sequence length of 2, together with a TABLE command, may be used to calculate the number of 'runs' in a column. In the simplest case, the rows in a column may be chosen from two non-zero values, such as '1' for 'heads' and '2' for 'tails' in a series of coin tosses.

e.g. PARSUMS 2 C4 INTO C5

ROW	C4	C5
1	2	2
2	1	3
3	1	2
4	2	3
5	2	4
6	2	4
7	1	3
8	2	3
9	2	4
10	1	3

TABLE C5

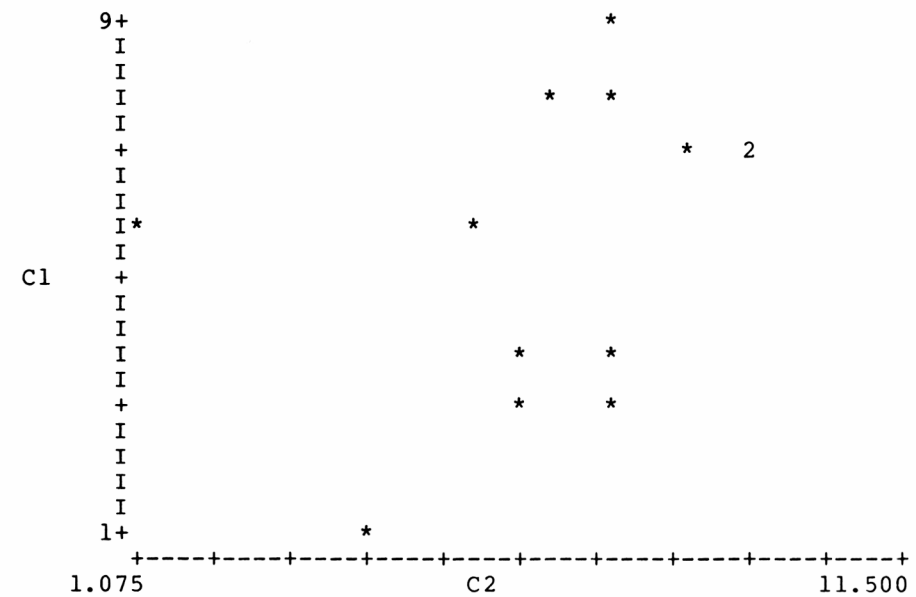
VALUE	FREQ	CUM-F	%	CUM-%
2	2	2	20.00	20.00
3	5	7	50.00	70.00
4	3	10	30.00	100.00

If the two possible values in the original column are P and Q, then the number of runs is 1 plus the frequency of (P+Q) in the PARSUMS column. In the above example where P is 1 and Q is 2, (P+Q) equals 3, so the number of runs in the original column is 1 plus 5 equal to 6.

PLOT y in C against x in C, [y in C against x in C, ...]

produces a scatter-plot of the values, with each point plotted as a '*' character. If between two and nine points fall at the same place, a digit is printed giving the number. Where there are ten or more points at the same place, a '+' character is used. If several pairs of columns are specified, they are all printed on the same axes using the same symbols, but the axes are labelled with the names of just the first two columns.

e.g. PLOT C1 C2



The PLOT command is useful for illustrating any pairwise association between the values in two columns, for example a column of people's heights and a column of their weights. If large values in one column correspond to the large values in the other, and the small values are similarly related, the points will tend to form a line (from the bottom left corner to the top right). If large values in one column correspond to small values in the other, the line will slope in the opposite direction (top left to bottom right). Other types of relationship may be indicated by different curves (e.g. U-shaped). If the columns are less strongly related the points will be more scattered.

A more precise measure of this kind of relationship is the correlation statistic obtained using the CORRELATE command described on page 58.

PLOT y from K to K in C, against x [from K to K] in C

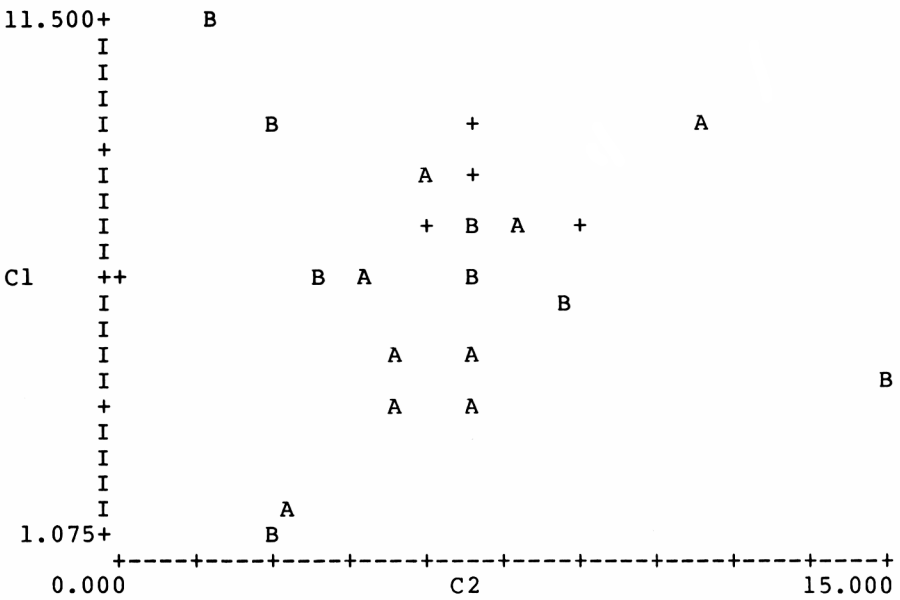
can be used to plot a subset of the values from one or both columns by specifying the lowest and highest values to be included. This can be used to exclude extreme values from the plot.

e.g. PLOT 2.5 TO 7.5 IN C1 AGAINST C2
PLOT 20 TO 50 IN C4, 0.4 TO 0.9 IN C3

MPLOT y in C against x in C [y in C against x in C, ...]

produces a multiple scatter-plot of the values. The output is similar to that from the PLOT command except that each successive pair of columns is plotted with a different character - 'A' for the first pair, 'B' for the second, and so on. Any overlapping points are plotted with a + character.

e.g. MPLOT C1 VS C2, C3 VS C4



MPLOT y from K to K in C, against x [from K to K] in C

may be used to plot a subset of the values, as with PLOT.

TSPLOT the values in C...C [with period K [starting at K]]

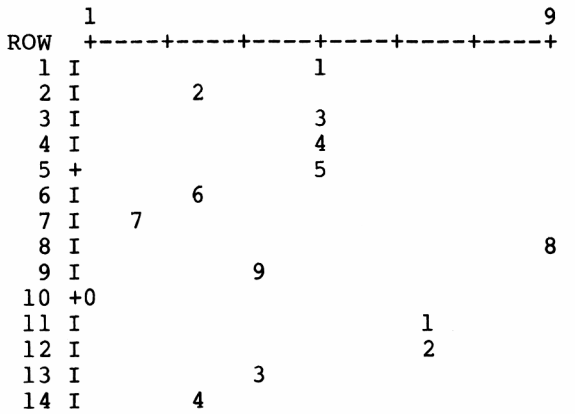
produces a time-series plot of the data - an illustration of how a sequence of values changes over time.

The values in each column are plotted against their row position in the column i.e. the digits 1, 2, 3, ... form the vertical axis. By default, the first point is plotted as a '1', the second as a '2' and so on up to the tenth which is plotted as a '0'. The sequence then repeats.

The period of repetition and starting point of the series may be specified. The period may extend up to 36. For periods of greater than 10, letters are used as plotting symbols - 'A' for 11, 'B' for 12, etc.

If multiple columns are plotted, 'A' is used for the first column, 'B' for the second etc., and overlapping points are plotted as '+'.

e.g. TSPLOT THE VALUES IN C1



If the data were monthly observations, starting in April, a suitable command might be:

TSPLOT C6 WITH PERIOD 12 STARTING AT 4

Time-series statistics (ACF and CCF) are described on pages 69 and 70.

WIDTH of output is K characters

controls the maximum width of the output from the commands PRINT, HISTOGRAM, PLOT, MPLOT, TSPLOT, ACF, CCF and CORRELATE. Although MICROTAB normally uses the 40 column mode 7 of the BBC micro, wider output can be sent to a printer or a file on tape or disk (see pages 11 and 85). The value given in a WIDTH command may be between 30 and 200, and should be a multiple of 5. A WIDTH of more than 40 will make some of the output on the screen 'wrap around' at the ends of lines. A WIDTH of 60 was used prior to producing the PLOT examples above.

HEIGHT of output is K lines

controls the height of the output from the commands PLOT, MPLOT and HISTOGRAM. Values for HEIGHT should be a multiple of 10. The initial value is 20.

Statistical Procedures

The statistical procedures available in MICROTAB include the main parametric and non-parametric tests used in small and medium-sized data analyses. The use of a good statistical text-book in conjunction with the program is recommended, both to provide a full theoretical background to each test and its use, and also for statistical tables to evaluate the results of tests. A short bibliography is given in Appendix 7.

The first group of commands cover procedures used in hypothesis-testing and the comparison of column means. These parametric tests usually make certain assumptions about the data values, for example that they are based on continuous (interval or ratio) scales of measurement, are normally distributed, or come from populations with similar variances.

TINTERVAL [with K percent confidence] for the values in C

calculates a confidence interval for the column based on the distribution of the t statistic. This is the range on either side of the column mean in which K% of such values would be expected to occur by chance.

If K is omitted, a confidence value of 95% is assumed.

If K is less than 1.0 the value is assumed to be a proportion rather than a percentage, and is multiplied by 100 to convert it to a percentage.

e.g. SET C1

```
3.5 3.75 4 2.5 5 6 4.5 4.8
```

TINTERVAL FOR C1

```
        C1          N=8 MEAN=4.2562 STD=1.0642

        95% C.I. = 3.3654 TO 5.1471
```

N.B. MICROTAB uses a numerical approximation method to calculate t intervals, and should not be used where N is less than 5.

TWOSAMPLE t test on data in C and C [with K percent confidence interval]

performs a t test to compare the means of two unrelated samples, which may be of different sizes and different variances. In addition, a confidence interval for the difference between the two means is given. If not specified, the 95% confidence interval is calculated.

e.g. READ DATA INTO C2 AND C3

```
1093  778
918   924
1149  1102
885   901
1056   1022
980   957
1043   1010

TWOSAMPLE C2 VERSUS C3

        C2        N=7 MEAN=1017.71 STD=94.86

        C3        N=7 MEAN=956.29 STD=103.54

        DIFFERENCE=61.43

        95% C.I. = -54.37 TO 177.23

        TWOSAMPLE T=1.157  DF=11
```

The size, mean and standard deviations of the two columns are given, together with the difference between the two means. The confidence interval is the range of values in which 95% (or K% if specified) of such differences would be expected to occur by chance. A difference outside this range implies that the means are significantly different at the specified level of confidence.

The t value is also used to evaluate whether the actual difference between the means is statistically reliable, and its significance may be found from tables. In this example, the critical t value for a two-tailed test with 11 degrees of freedom and a 0.05 probability level is 2.201. The absolute value of the calculated t statistic is less than this, so the two columns would not be considered significantly different - in agreement with the interpretation of the confidence interval above.

The degrees of freedom in the TWOSAMPLE procedure are not usually a whole number but, for convenience, are printed out to the nearest integer value.

POOLED t test on data in C and C

performs a t test for two unrelated samples which have similar variances. It is possibly more common for two samples to have different variances, in which case the TWOSAMPLE command should be used. Where the variances are the same (or quite close), the POOLED test is slightly more powerful.

e.g. POOLED TTEST ON C2 AND C3 WITH 90% CONFIDENCE

```
C2          N=7 MEAN=1017.71 STD=94.86

C3          N=7 MEAN=956.29 STD=103.54

DIFFERENCE=61.29

90% C.I. = -33.14 TO 156.00

POOLED T=1.157  DF=12
```

The output from POOLED is similar in form to that from the TWOSAMPLE command. In this example, which uses the same data, an identical t value is found. However, as the degrees of freedom are larger, the tabled t value, against which the calculated value is tested, is slightly lower (2.179).

TTEST [of mean = K] on values in C [and C]

If only one column is mentioned, TTEST performs a one-sample t test to compare the mean of column C against K. If K is omitted, a value of zero is assumed.

If two columns are given, a t test for matched samples is performed. This is equivalent to a one-sample t test on the difference-scores of the two columns.

e.g.
```
NAME C5 'MATHS' C6 'ENGLISH'
READ 'MATHS','ENGLISH'
44 51
42 49
44 54
43.5 50
44 49.5
41 52
```

```
TTEST ON PAIRED SCORES BETWEEN 'MATHS' AND 'ENGLISH'

        MATHS      N=6 MEAN=43.083 STD=1.281

        ENGLISH    N=6 MEAN=50.917 STD=1.855

        DIFFERENCE=-7.833

        TTEST T=-8.882  DF=5
```

The output is again similar to that from TWOSAMPLE and POOLED except that no confidence interval is calculated. The significance of t may be found from statistical tables. In this example, the critical t value for a two-tailed test with 5 degrees of freedom and a significance level of 0.05, is 2.571. The absolute value of the calculated t statistic is greater than this, so the difference between the means of 'MATHS' and 'ENGLISH' can be accepted as being significantly different from zero.

The Analysis of Variance

The ANOVA (ANalysis Of VAriance) command is a more complicated routine than most, by virtue of the nature of the procedure itself, the various ways it has been adapted for particular applications, and the variety of terminologies used in its description. Because of these problems, the ANOVA command will be given a rather fuller discussion and description than most of the other statistical procedures. This discussion will probably be biased towards the use of ANOVA in psychology and education, with which the author is most familiar, although the general principles will be applicable in most other fields.

In its simplest form, ANOVA might usefully be thought of as an extension of the t test. Whereas the t test can compare the means of two groups measured on some variable, for example test scores in English versus test scores in History, ANOVA can compare more than two groups, for example English, History and Maths. As with the t test the scores can be repeated measurements (e.g. from the same group of students), or scores from independent groups (e.g. students in different classes). The former situation may be known as a 'within-subjects' or (in certain contexts) a 'randomised block' design while the latter is sometimes referred to as a 'between-subjects' or 'completely randomised' design.

In the example just described, the variable being investigated might be called 'Course of Study'. A variable being examined in this way is also known as a 'treatment' or 'factor'. The variations of a factor (e.g. the particular types of 'Course of Study') are often called the 'levels' of treatment. In this example, there are three levels - English, History and Maths.

The Analysis of Variance is able to look at the effects of several factors at once. For example, we could investigate different 'Courses of Study' and also compare results obtained by different 'Methods of Testing' such as continuous assessment versus final examinations. This would be a case of a two-factor design, and each factor could be either a repeated-measure factor or an independent-groups factor. On this basis, there are three sorts of two-factor designs - both independent-groups variables, both repeated-measures, or one of each.

ANOVA with K variables, K repeated-measure variables, K levels of variable A, [K levels of variable B,] on data in C...C

performs an analysis of variance with one or two variables (factors), one or both of which may be repeated-measure variables.

ANOVA only caters for balanced factorial (i.e. symmetrical) designs.

Each 'cell' of the design is represented by a column of values.

Except in the case of a one-variable analysis with independent groups, all the columns used in an ANOVA must be of equal size.

Where there is no repeated-measure variable, a value of zero must be entered as the second argument to the command.

If there is one repeated-measure variable and one non-repeated variable (giving what is also known as a 'mixed' or 'split-plot' design), the repeated-measure variable should be designated variable B. In such cases, the number of levels for the repeated-measure variable is given after the number of levels for the non-repeated variable.

For a one-factor design, the order of the columns is not important. For two-factor designs, the columns should be entered taking each level of factor A in turn, and for that level, giving the columns which comprise the levels of factor B. In other words, A is the 'slower-cycling' factor in the columns list.

The normal test statistic associated with analysis of variance is the 'F' or variance ratio which is calculated by dividing the mean square term for a particular effect by an appropriate 'error' mean square. This calculation is not included in the output from ANOVA since the appropriate error term depends on the nature of the experimental variables and the generalisations that are to be drawn from the results. Experimental variables can be classed as either 'random' (if the particular values used were selected at random from all possible values), or 'fixed' (if values were chosen for convenience or as being of particular interest). On the whole, it is probably more common for variables to be fixed rather than random, and most of the examples below make this assumption, particularly where repeated-measures are involved.

Because of the complexity of the ANOVA command, an example of each type of design will be presented.

ANOVA WITH ONE VARIABLE

The data used in the one-variable examples are given below. The variable in a one-variable design will be referred to as 'A'. The notation a1, a2 etc. is used to designate the columns which comprise the levels of variable A.

	a1	a2	a3
ROW	C1	C2	C3
1	6	3	4
2	20	9	5
3	10	8	9
4	5	0	1
5	7	2	1

```
DESCRIBE C1-C3

        C1          N=5 MEAN=9.60 STD=6.11

        C2          N=5 MEAN=4.40 STD=3.91

        C3          N=5 MEAN=4.00 STD=3.32
```

The first analysis is for the case where the columns are unrelated samples, e.g. test scores in English, History and Maths for 3 separate groups of pupils. This is known as a one-factor independent-groups design, or a simple completely randomised design.

e.g. ANOVA WITH 1 VARIABLE, 0 REPEATED, 3 LEVELS OF A IN C1-C3

SOURCE	SOS	DF	MS
A	97.60000	2	48.80000
ERROR	254.40000	12	21.20000
TOTAL	352.00000	14	

The ANOVA command produces the usual form of summary table, giving the Sums Of Squares (SOS), Degrees of Freedom (DF) and Mean Square (MS) from each contributing source. The F ratio for evaluating the effect of A is calculated by dividing the MS for A by the ERROR MS, which for the above data gives a value of 2.30.

The F ratio is evaluated using statistical tables. In this example, with degrees of freedom of 2 and 12, the calculated value needs to be at least 3.88 to be significant at the 0.05 probability level - thus the

obtained value of 2.30 is non-significant. On the basis of this result, the means of the three columns would not be accepted as being reliably different from one another. In terms of the hypothetical example, we would say that the mean test scores in English, History and Maths are not really different from one another.

Although the example data has columns of equal size, this is the one design where unequal columns could be used.

The alternative one-variable analysis is one where the samples are matched on some basis, e.g. test scores in 3 different courses for the same group of pupils.

ANOVA WITH 1 VAR , 1 REPEATED, WITH 3 LEVELS IN C1-C3

SOURCE	SOS	DF	MS
A	97.60000	2	48.80000
A S	64.40000	8	8.05000
S	190.00000	4	47.50000
TOTAL	352.00000	14	

The F statistic for A is calculated using the MS for the A S interaction as the error term. The resulting value of 6.06, which has 2 and 8 degrees of freedom, exceeds the table value of 4.46 and is therefore significant at the 0.05 probability level.

The F ratio for S (variously known as the 'between-blocks', 'between-replications', or 'between-subjects' effect) may be of interest in certain situations and uses the same error term, giving a value of 5.90. With 4 and 8 degrees of freedom this is also statistically significant. In the example used here, this would suggest that pupils vary fairly substantially in their overall levels of performance - not necessarily an unexpected result.

ANOVA WITH TWO VARIABLES

The example data used in all the two-variable analyses is given below. The second variable is referred to as 'B'. Variable A has two levels, and variable B has four levels. It may again be helpful to make the example more concrete - A could be thought of as 'Course of Study' (e.g. English or History), and B as 'Method of Testing' (e.g. Continuous Assessment, Long Exam Essay, Short Exam Questions, or Multiple Choice Test).

The data is organised as follows:

a1

	b1	b2	b3	b4
ROW	C1	C2	C3	C4
1	12	13	16	16
2	15	14	17	17
3	12	13	16	18
4	12	12	15	17

a2

	b1	b2	b3	b4
ROW	C5	C6	C7	C8
1	10	11	14	19
2	11	12	15	19
3	11	13	14	18
4	11	12	15	20

```
DESCRIBE C1-C8

        C1          N=4 MEAN=12.75 STD=1.50

        C2          N=4 MEAN=13.00 STD=0.82

        C3          N=4 MEAN=16.00 STD=0.82

        C4          N=4 MEAN=17.00 STD=0.82

        C5          N=4 MEAN=10.75 STD=0.50

        C6          N=4 MEAN=12.00 STD=0.82

        C7          N=4 MEAN=14.50 STD=0.58

        C8          N=4 MEAN=19.00 STD=0.82
```

Columns 1 to 4 are the scores in English under the four methods of testing, and columns 5 to 8 are the corresponding scores in History. Although not strictly necessary, it is probably easiest to keep the order of the 'B' conditions the same within each block of the 'A' conditions, e.g. if column 1 here represents 'Continuous Assessment' for English, then column 5 should represent 'Continuous Assessment' for History. If a third course of study were included, it could be put in columns 9 to 12 with the 'Continuous Assessment' scores being in column 9.

In the first example, there are no repeated-measure variables. This would be the case where each group of test scores comes from a separate group of individuals - in our example this would mean a separate group of students for each combination of 'Course of Study' and 'Method of Testing'.

e.g. ANOVA FOR 2 VARS, 0 REPS, 2 LEVELS OF A, 4 OF B, IN C1-C8

SOURCE	SOS	DF	MS
A	3.12500	1	3.12500
B	194.50000	3	64.83333
A B	19.37500	3	6.45833
ERROR	18.50000	24	0.77083
TOTAL	235.50000	31	

As well as the overall effects of the A and B variables, the table includes a term for the interaction of A and B. The interaction term is an indication of how the effect of A varies with different values of the B variable. For example some courses might show bigger differences between test methods than others.

If A and B are both 'fixed' variables, the F ratios for the overall effects of A and B, and the interaction term AB are all calculated using the ERROR MS as the denominator giving values of 4.05, 84.11 and 8.38 respectively. If A is a random variable, then the F ratio for B uses the MS for AB as its denominator. This also applies to A if B is a random variable. The F values are evaluated, as usual, from statistical tables.

In situations where the AB interaction is inappropriate or non-significant, its SOS and DF may be 'pooled' with those for ERROR and a new error MS calculated to provide a new error term for testing the main effects of A and B. This should be done with some caution however.

A second case is where there is one repeated-measure variable, which must be designated 'B'. In terms of our example, this would mean that the scores for each 'Course of Study' were from different groups of students, but that each student was tested by four 'Methods of Testing'.

e.g. ANOVA 2 VARIABLES, 1 REPTD, 2 LEVELS OF A, 4 OF B, C1-C8

SOURCE	SOS	DF	MS
A	3.12500	1	3.12500
BETWN	9.37500	6	1.56250
B	194.50000	3	64.83333
A B	19.37500	3	6.45833
WITHN	9.12500	18	0.50694
TOTAL	235.50000	31	

The F ratio for A uses the BETWN (between groups) MS term as its denominator, while B and AB both use the WITHN (within-groups) MS. The respective F ratios for A, B and AB in this example are 2.0, 127.9 and 12.7 respectively.

Finally, where both variables are repeated-measures - every student does both 'Courses of Study' and is tested by all four 'Methods of Testing' - the appropriate procedure is:

ANOVA 2 VAR 2 REP 2 A 4 B C1-C8

SOURCE	SOS	DF	MS
A	3.12500	1	3.12500
A S	4.12500	3	1.37500
B	194.50000	3	64.83333
B S	3.75000	9	0.41667
A B	19.37500	3	6.45833
A B S	5.37500	9	0.59722
S	5.25000	3	1.75000
TOTAL	235.50000	31	

The F values for A, B and AB may be calculated using the corresponding MS terms of AS, BS and ABS giving F ratios of 2.27, 155.6 and 10.81 respectively. This procedure uses a statistical model which assumes the possibility of interactions between the variables and the individuals.

There is an alternative procedure, where such assumptions are not made, in which the denominator for all the F ratios is calculated as:

$$\frac{SOS(AS) + SOS(BS) + SOS(ABS)}{DF(AS) + DF(BS) + DF(ABS)}$$

In the above example, the new error SOS is

$$\frac{4.125 + 3.75 + 5.375}{3 + 9 + 9}$$

This gives a new common MS error term of 0.63095 which has 21 degrees of freedom associated with it.

The F ratios for A, B and ABS following this procedure are then 4.953, 102.73 and 10.24 respectively.

In all the examples given above, the data were organised so that the list of columns in the ANOVA command was the most straightforward. Other arrangements are possible, as long as the order of columns given is a sensible one for the particular design. In the last example above, the following command would have given exactly the same result:

```
ANOVA  2 2  2 4   C5 C6 C8 C7   C1 C2 C4 C3
```

Correlation and Regression

The correlation statistic is a measure of the similarity of the pattern of the values in a pair of columns, and can range from -1.0 to +1.0. A positive correlation means that large values in one column correspond to large values in the other, and the same for the small values. A negative correlation means that large values in one column correspond to small values in the other, and vice versa.

CORRELATE columns C...C

computes a Pearson product-moment correlation coefficient for each combination of the columns. With more than two columns, the output is printed as the lower triangle of the correlation matrix. The columns must all be equal in size.

e.g. PRINT C1-C4

ROW	ENGLISH	MATHS	HISTORY	SCIENCE
1	47	62	53	85
2	56	56	59	53
3	62	47	65	52
4	58	61	63	60
5	57	51	61	54
6	68	60	75	62
7	59	35	78	42

CORRELATE C1-C4

CORRELATION (N=7)

	ENGLISH	MATHS	HISTORY
MATHS	-0.203		
HISTORY	0.774	-0.578	
SCIENCE	-0.553	0.771	-0.632

To test the significance of the correlation, tables are used to find the 'critical value' of the coefficient for the given N (or degrees of freedom, N-2) at the chosen probability level. For N=7, the critical value for a two-tailed test at the 0.05 probability level is 0.754. Using these criteria, only the correlations between the marks in English and History, and in Maths and Science are statistically reliable. The positive signs of these correlations mean that high scores in English are

associated with high scores in History, and high scores in Maths with high scores in Science.

In some situations (e.g. data with a non-normal distribution) a rank correlation such as Spearman's rho is appropriate. To perform this, each column should first be given RANKS (page 22) and the CORRELATE command used on the resulting columns.

An alternative method of performing correlations, which enables coefficients to be stored, is by direct use of MICROTAB's correlation function in a LET command. The function, called FNC, calculates the correlation between a specified pair of columns and is used in a similar manner to the FNG and FNL functions described on page 35.

e.g. LET K1=FNC(2,7)

stores the correlation between columns 2 and 7 in K1.

The numbers of the columns to be correlated may also be placed in the corresponding rows of two columns.

e.g.
```
READ C5 C6
1  2
2  3
3  1
LET C4=FNC(C5,C6)
```

stores in C4 the correlations of C1 with C2, C2 with C3, and C3 with C1. Columns C4 to C6 may now be SORTed etc.

REGRESSION

Regression analysis is related to correlation and is used to predict one set of values (usually known as the Y values) from one or more sets of other values (X1, X2, etc.). For example, if students' marks in History are a good predictor of their marks in English, a teacher could use a History score to try and assess an English mark for someone who had been ill at the time of the English exam.

REGRESS y in C using K predictors in C...C
[store predicted y values in C [and residuals in C]]

performs a linear regression analysis using one or more predictors. All the examples of the REGRESS command use the data from the previous section on CORRELATE.

e.g. REGRESS Y IN C1 USING 1 PREDICTOR IN C3

```
ENGLISH = 22.01889 +0.55698*HISTORY

COLUMN   COEFFICIENT      STDEV     T
          22.01889     13.32530    1.652
HISTORY    0.55698      0.20384    2.732

S = 4.414  DF=5

R-SQUARED = 59.891%

SOURCE      SOS        DF        MS
REGR      145.45068    1      145.45068
RESID      97.40646    5       19.48129
TOTAL     242.85714    6
```

The regression equation shows how the dependent variable (ENGLISH) is predicted from a constant term together with the contribution from each of the predictor variables - in this case just one (HISTORY). The ENGLISH score for a new individual may thus be calculated by putting his existing HISTORY score into the equation. For example, someone with a score of 40 for HISTORY would be predicted to achieve a score of 44.3 for ENGLISH.

The table which follows the regression equation gives the standard deviation and t value for each term in the equation. The t value, calculated by dividing the coefficient by its standard deviation, may be used to determine whether the coefficient is significantly different from zero (and thus worth including in the regression). The degrees of freedom for this test are (N - K) where N is the number of scores in each column and K is the number of coefficients in the regression equation (including the constant term). In the above example, the critical value for t at the 0.05 probability level, two-tailed, with 5 degrees of freedom, is 2.571. With an obtained value of 2.732, score in History is a reliable predictor of score in English.

S is the standard deviation of Y about the regression line and may be used as an estimate of the error in a predicted Y value.

R-SQUARED, which is the square of the correlation coefficient times 100, measures how well the regression equation fits the data overall.

The analysis of variance table may also be used to examine the overall goodness-of-fit of the regression. As with the ANOVA procedure, the Mean Square (MS) terms are obtained by dividing each Sum of Squares (SOS) by its corresponding Degrees of Freedom (DF). A variance or F ratio may be calculated by dividing the MS for the regression by the residual MS. The F ratio in this

example is 7.99 with 1 and 5 degrees of freedom. Statistical tables show that this value is significant at the 0.05 probability level and again indicates that History scores reliably predict those in English.

The values of S and R-squared are related to the analysis of variance in that the residual MS is equal to S squared, and the regression SOS divided by the total SOS is equal to R-squared.

A fuller form of output from REGRESS is obtained if the command NOBRIEF has previously been given.

e.g. NOBRIEF
REGRESS 'ENGLISH' USING 1 PREDICTOR IN 'SCIENCE'

ENGLISH = 73.4111 -0.26196*SCIENCE

COLUMN	COEFFICIENT	STDEV	T
	73.41110	10.52512	6.975
SCIENCE	-0.26196	0.17661	-1.483

S = 5.808 DF=5

R-SQUARED = 30.556%

SOURCE	SOS	DF	MS
REGR	74.20813	1	74.20813
RESID	168.64901	5	33.72980
TOTAL	242.85714	6	

ROW	SCIENCE	Y ENGLISH	PRED.Y VALUE	RESIDUAL
1	85	47.0	51.1	-4.14491X
2	53	56.0	59.5	-3.52748
3	52	62.0	59.8	2.21057
4	60	58.0	57.7	0.30621
5	54	57.0	59.3	-2.26552
6	62	68.0	57.2	10.83012R
7	42	59.0	62.4	-3.40898

The fuller version of the output gives the X, Y, predicted Y and residual values for each row. If there is more than one X predictor, only the first is given.

At the end of each row, an X code is used to indicate unusual X values (more than 1.96 standard deviations from the X mean). Such values have a large effect on the regression coefficients so they might be checked for accuracy, and DELETEd if thought unreliable. An R code indicates extreme residuals (more than 1.96 standard deviations from the residual mean). In the reduced form of output which resumes following a BRIEF command, only rows with extreme X or residual values are reported.

The following example illustrates a multiple regression with the predicted Y values being stored in a column:

```
e.g.  BRIEF
      REGRESS C1 3 C2-C4, PREDICTED Y VALUES IN C7

        ENGLISH = 11.07323 +0.48098*MATHS
                  +0.58844*HISTORY -0.28576*SCIENCE

        COLUMN  COEFFICIENT      STDEV     T
                   11.07323    23.42469    0.473
        MATHS       0.48098     0.25811    1.863
        HISTORY     0.58844     0.23308    2.525
        SCIENCE    -0.28576     0.19672   -1.453

        S = 3.846  DF=3

        R-SQUARED = 81.726%

        SOURCE       SOS        DF          MS
        REGR       198.47704    3        66.15901
        RESID       44.38010    3        14.79337
        TOTAL      242.85714    6


                              Y     PRED.Y
        ROW    MATHS  ENGLISH     VALUE  RESIDUAL
         1        62     47.0      47.8 -0.79185X
```

More complex forms of regression may be performed by using predictor columns that have been 'transformed'. For example, polynomial regression involves terms that are different powers of the predictor variable. Instead of predicting Y from X, we might want to predict Y from the combined effect of X and X-squared. This just requires the creation of a column containing the extra predictor

```
e.g.  LET C5=C3*C3
      REGRESS C1 USING 2 IN C3 AND C5
```

The absolute maximum number of predictors in a regression is 9, but the actual figure depends on the current worksheet. First of all think of how many rows are allowed in your current worksheet, and how many columns; choose the larger of these two values - call this number 'Max'. For a worksheet of 40 rows by 20 columns, 'Max' would be 40. If the number of predictors you want to use is K, then work out (K+1)*(K+2)/2. For example, for 5 predictors the result is 6*7/2 = 21. If this number is less than or equal to 'Max', the regression may be performed. At least 5 predictors should normally be possible.

Non-parametric Statistics

Procedures such as the t test and analysis of variance described earlier require certain assumptions about the nature of the data being examined, or the characteristics of the populations from which the data are drawn. These assumptions include such things as the values being recorded on at least an interval scale of measurement, being normally distributed, and so on.

Non-parametric statistics or 'distribution-free' statistics require far fewer assumptions, and may be used in a much wider variety of situations, for example when the data is recorded using a nominal or ordinal scale of measurement. Non-parametric tests (like parametric ones) do however make some assumptions, for example that observations are independent.

CHISQUARE test on table in columns C...C

performs the chi square test of association between the columns and rows. The data used in the test should be frequency values - which may have been generated using the TABLE or HISTOGRAM commands.

For example, the people working in three different offices were canvassed to see which of the main four TV channels they watched most. The results of this survey could be stored in three columns (one for each office) with the number of people putting each channel first making up the rows.

e.g. PRINT C1-C3

ROW	C1	C2	C3
1	11	4	4
2	5	3	9
3	9	14	2
4	4	7	5

```
CHISQUARE C1-C3

     C1I      C2I      C3I
-------+--------+--------+-------
   11. I     4. I     4. I   19.
    7.2I     6.9I     4.9I
-------+--------+--------+-------
    5. I     3. I     9. I   17.
    6.4I     6.2I     4.4I
-------+--------+--------+-------
    9. I    14. I     2. I   25.
    9.4I     9.1I     6.5I
-------+--------+--------+-------
    4. I     7. I     5. I   16.
    6.0I     5.8I     4.2I
-------+--------+--------+-------
   29. I    28. I    20. I   77.

CHISQUARE = 17.04  DF=6

 3 EXPECTED FREQS < 5
```

The output consists of a table of the observed and expected frequencies for each 'cell' (the observed is the upper figure in each case), and the chi square value. The chi square calculation for 2 by 2 tables uses the usual correction for continuity. When applicable, there is also a report on the numbers of cells with expected frequencies of less than 5 and less than 1. The chi square test should not be used where 20% or more of cells have expected frequencies of less than 5, or if any cell has an expected frequency of 0. If either of these is the case, it may be possible to combine cells to increase the expected frequencies.

The chi square value may be evaluated from statistical tables. In the above example, the critical value of chi square with 6 degrees of freedom at a probability level of 0.05 is 15.03. The obtained value is greater than this, so in terms of the example, it could be said that there is an association between the office worked in, and the preferred TV channel.

When more than four columns are used, the output will 'wrap around' the end of the 40 column screen.

MANN Whitney test on values in C and C

performs a Mann-Whitney 'U' test for unrelated samples, between the two columns. Although the Mann-Whitney could be thought of as a non-parametric version of the t test,

it really compares the medians of two samples rather than the means.

The columns being compared may be of different sizes, but the total number of values in the two columns combined must not be more than the larger of the number of rows or columns in the current worksheet.

As an example, we might record the number of hours of TV per week watched by people from two groups such as manual workers and office workers.

e.g. PRINT C3,C4

ROW	C3	C4
1	15	8
2	21	10
3	9	7
4	13	6
5	10	9
6	16	9
7	11	4

```
MANN-WHITNEY TEST ON C3 AND C4

        C3          N=7 MEDN=13.00

        C4          N=7 MEDN=8.00

        MANN-WHITNEY U=2.50
```

The smaller of the two possible U values is always given, and the U statistic may be evaluated from tables. In the above example, the tabulated value of U for N1=7 and N2=7 at the 0.05 probability level (two-tailed) is 8. This is larger than the calculated value of 2.5, so the two groups do differ in the amount of TV watched.

Less commonly, there are tables which assume that the larger of the two possible U values has been calculated. If these are used, U may be converted by

```
U = N1*N2 - U
```

which in the above example gives an alternative test value of (49 - 2.5) = 46.5. This needs to be larger than the tabled value for the samples to be different.

When either of the samples contains twenty or more values, a corresponding z value is calculated which again may be evaluated from tables.

WILCOXON test on the values in C and C
WILCOXON test on the difference-scores in C

performs a Wilcoxon signed-ranks test on two related samples, or on a set of difference-scores. The Wilcoxon test is the non-parametric version of the t test for related samples, but instead of comparing means it compares the medians of the two samples.

The data from the Mann-Whitney example above may be used to provide an illustration, if the data is imagined to be taken from the same group of individuals for a week in winter and a week in summer.

e.g. WILCOXON ON C3,C4

```
C3          N=7 MEDN=13.00

C4          N=7 MEDN=8.00

WILCOXON T=0   (TEST N=7)
```

The Wilcoxon T statistic may be evaluated using suitable tables. With a test N (number of non-zero differences) of 7, the critical value of T at the 0.05 probability level (two-tailed) is 2. The calculated value is less than this, so the amount of TV watched in summer and winter would be said to differ in this example.

If twenty or more pairs/difference-scores are used a z statistic is given in addition to T as an alternative means of evaluating the difference between the samples.

When tied pairs of values occur, the T value is corrected appropriately, and the N for testing the value of T will be equal to the number of non-tied pairs.

KRUSKAL Wallis analysis of variance on C...C

performs a non-parametric one-way analysis of variance. The columns may be of different sizes but the total number of values in the analysis must not be more than the larger of the number of columns or rows in the current worksheet.

The Kruskal-Wallis test may be thought of as an extension of the Mann-Whitney test to more than two

groups. This comparison is rather like the relationship of the ANOVA to the t test for uncorrelated samples.

As an example, the sample data for the MANN command above could be extended to represent the amount of TV (in hours per week) watched by three groups of workers, perhaps manual workers, office workers and unemployed people.

e.g. PRINT C3-C5

ROW	C3	C4	C5
1	15	8	6
2	21	10	15
3	9	7	12
4	13	6	14
5	10	9	8
6	16	9	10
7	11	4	13

```
KRUSKAL TEST ON C3-C5

        C3          N=7 MEDN=13.00

        C4          N=7 MEDN=8.00

        C5          N=7 MEDN=12.00

        KRUSKAL VALUE=8.67  DF=2
```

The Kruskal-Wallis H statistic may be evaluated from tables. With three samples of size 7, the critical value of H at the 0.05 probability level is 5.82. The calculated value is larger than this, so the groups are significantly different from one another.

For reasonably large samples, say more than 5 values per column, chi square tables may be used. For the example above with 2 degrees of freedom, the critical value of chi square at the 0.05 probability level is 5.99, so on this basis the groups may again be said to differ.

The Kruskal-Wallis procedure uses a correction for tied values.

FRIEDMAN analysis of variance on C...C

performs a Friedman analysis of variance for related samples. This might be considered as an extension of the

Wilcoxon test to more than two samples, or the non-parametric version of a one-factor repeated-measures ANOVA.

The sample data from the Kruskal-Wallis procedure above will be used as an example. This might now be treated as describing the amount of TV (hours per week) watched by a particular group of individuals at three different times of the year - winter, summer and spring.

e.g. FRIEDMAN TEST ON C3-C5

```
C3          N=7 MEDN=13.00

C4          N=7 MEDN=8.00

C5          N=7 MEDN=12.00

FRIEDMAN VALUE=6.00  DF=2
```

The Friedman statistic may be evaluated from suitable tables. With 3 groups and N=7, the critical value for the Friedman statistic at the 0.05 probability level is 7.14. The calculated value is less than this so the amount of TV watched at the three times of the year is not significantly different.

Time Series Statistics

In addition to the TSPLOT (time-series plot) described earlier, MICROTAB has two commands for examining series of observations made sequentially over time or space.

ACF [for up to K lags] on the values in C

performs an auto-correlation for each lag from 1 to K. If K is not given, a value of 10 plus the square root of the column length is used, unless this exceeds the column length. The column must contain at least 5 values.

The auto-correlation at lag 1 correlates row 1 with row 2, row 2 with row 3, row 3 with row 4, ..., row N-1 with row N. The auto-correlation at lag 2 uses row 1 with row 3, row 2 with row 4, row 3 with row 5 etc.

In addition to the auto-correlations at each lag, a correlogram is printed.

e.g. PRINT C3,C4

ROW	C3	C4
1	31	10
2	50	17
3	101	40
4	10	12
5	13	10
6	26	35
7	53	20
8	96	34
9	13	33
10	15	14
11	30	18
12	46	41
13	96	33
14	12	8
15	14	9
16	23	9
17	53	21
18	101	38
19	14	6
20	31	19
21	46	24
22	95	34
23	9	6
24	22	16

ACF ON C3

```
                -1.0 -0.5  0.0  0.5  1.0
LAG               +----+----+----+----+
  1 -0.084              X
  2 -0.443           XXXXX
  3 -0.334            XXXX
  4  0.168               XX
  5  0.562               XXXXXX
  6 -0.151             XX
  7 -0.297            XXX
  8 -0.200            XXX
  9  0.165              XX
 10  0.363              XXXX
 11 -0.107            XX
 12 -0.217           XXX
 13 -0.122            XX
 14  0.177              XX
```

The fictitious values in this example represent the total numbers of Party Political Broadcasts made each year over a twenty-four year period. The largest positive peak is at a lag of five, which indicates that the number of such broadcasts is fairly cyclical over spans of five years.

For the auto-correlation at a particular lag, an approximate test of significance at the 95% confidence level may be performed by dividing 1.96 by the square root of the column length minus the lag. If the absolute size of the auto-correlation is larger than this value, it is statistically significant. This procedure should only be used with reasonably large columns, say 50 rows or more.

PUT ACF [for up to K lags] on values in C, into C

stores the auto-correlations in the specified column and suppresses the normal display.

e.g. PUT ACF ON C3 IN C6

CCF [with up to K lags] between the values in C and C

performs a cross correlation between two columns of time series data. If no lag is specified, the lag goes from -10 minus the square root of the column length to 10 plus the square root of the column length.

The output is similar in form to that from the ACF

command, although the correlogram has a negative section where the first column lags and the second leads, and a positive section where the first column leads and the second lags. The largest positive peaks occur where the cycle-lengths of the two columns coincide most closely.

e.g. CCF WITH 10 LAGS FOR C3 AND C4

```
                -1.0 -0.5  0.0  0.5  1.0
LAG               +----+----+----+----+
-10   0.325              XXXX
 -9   0.267              XXX
 -8  -0.168             XX
 -7  -0.223            XXX
 -6  -0.132             XX
 -5   0.368              XXXX
 -4   0.220              XXX
 -3  -0.375           XXXX
 -2  -0.360           XXXX
 -1  -0.015              X
  0   0.745              XXXXXXXX
  1  -0.084              X
  2  -0.441          XXXXX
  3  -0.185             XX
  4   0.214              XXX
  5   0.334              XXXX
  6  -0.063              X
  7  -0.245            XXX
  8  -0.217            XXX
  9   0.285              XXX
 10   0.296              XXX
```

The output above uses the data from the ACF example.

The significance of cross-correlation coefficients is tested in a similar manner to Pearson product-moment correlation coefficients.

The cross-correlation coefficients may not be stored.

As is the case with the PLOT procedure, the width of the correlogram output from ACF and CCF may be altered by the WIDTH command. A width of 65 or more produces a wider display.

e.g. WIDTH IS 70

Other Statistical Procedures

In addition to those statistical routines already built into MICROTAB, other procedures and tests may be performed by using various combinations of MICROTAB commands. In the case of statistical tests, it is usually a good idea to try out your sequence of commands with some sample data from a textbook. Sets of commands that are used frequently may be stored in a file on disk (see page 36).

The Sign test, for example, may be used to compare two sets of numbers by examining the sign of the difference between each pair of scores. The number of differences having plus and minus signs is counted, and the smaller of these, together with the total of the two, is used to make the test. If the two sets of values are in C8 and C9, the following commands could be used:

```
LET C12=SIGNS(C9-C8)
OMIT VALUES OF 0 FROM C12
LET K2=COUNT(C12)
PUT TABLE OF C12 IN C12
LET K3=MINIMUM(C12)
```

K3 and K2 are the test values and the result of the test may be evaluated from a statistical table of the binomial distribution.

The Kolmogorov-Smirnov test compares two distributions of values. If the values are in C1 and C2, the following steps might be involved:

```
PUT HISTOGRAM FOR C1 IN C3 FIRST POINT AT K, INTERVAL K
PUT HISTOGRAM FOR C2 IN C4 FIRST POINT AT K, INTERVAL K
PARSUMS FOR C3 INTO C3
LET C3=C3/SUM(C1)
PARSUMS FOR C4 INTO C4
LET C4=C4/SUM(C2)
LET C5=ABS(C4-C3)
MAX C5
```

The values of K should be the same for both HISTOGRAM commands, but will depend on the actual data involved and will affect the final result. It may also be necessary to make the column lengths equal before subtraction (C4-C3) by INSERTing values of 1 at the end of the shorter column.

Miscellaneous Commands

DIMENSION worksheet to columns of K rows

reformats the worksheet to columns of the specified length. The number of columns provided depends on their size, and whether you have disks, second processor etc. The maximum columns or rows in any worksheet is 255.

This command destroys all the data currently in the worksheet so use the SAVE command first if required.

e.g. DIMENSION 40 ROWS

DIMENSION worksheet to K rows, and set screen mode to K

reformats the worksheet, and changes the screen to a mode from 0 to 7. This should only be used on a B+ or Master machine in 'shadow' mode, or on a machine with extra hardware such as a 6502 second processor. If you change to an 80 column mode, a WIDTH command may then be useful.

e.g. DIM 40 ROWS, MODE 3

WIDTH of output is K characters, [with K characters for columns]

sets the maximum width of the output from the commands PRINT, PLOT, CORRELATE, ACF, CCF etc. The value should be a multiple of 5 and not less than 30. The optional second argument sets the maximum number of characters used for displaying column values in the output from PRINT and the column and row statistic commands (MEAN, SUM, RMEAN, RSUM etc.). Between 9 and 15 characters are allowed. The initial setting is 9.

e.g. WIDTH IS 80 CHARACTERS
WIDTH IS 40 CHARACTERS WITH 12 FOR COLUMNS

HEIGHT of output is K lines

affects the height of output from commands PLOT, MPLOT and HISTOGRAM. The value should be a multiple of 10.

e.g. HEIGHT IS 50 LINES

RESTART the program

clears all data from the worksheet and restarts MICROTAB.

STOP

exits from MICROTAB. Don't forget to SAVE your data first if required. You will need to re-load the program from tape or disk if you want to use it again.

HELP

displays the names of all the MICROTAB commands.

NOTE (you can put whatever you like here!)

is used to enter comments in output copied to a printer, disk file etc. See also the # character on page 6.

e.g. NOTE THIS DATA IS FROM GROUP 3 ON 2/3/84

VDU K [K...]

controls various aspects of the screen display in a similar manner to the BBC BASIC VDU statement (chapter 34 of the BBC Microcomputer User Guide). The double-byte convention (using semi-colons) is not supported.

e.g. VDU 12,14

clears the screen and puts it into page mode. Many VDU functions can be performed by entering control characters at the keyboard (see Appendix 4). The VU command is more explicit, for example in stored command files, where it may be used to do such things as inhibit or enable the display of stored commands (VDU 21 and 6 respectively).

MEMORY addresses of start and end of workspace

displays, in hexadecimal notation, the actual memory range occupied by the worksheet. These addresses may be used to *SAVE the worksheet as described in Appendix 6.

e.g MEMORY
5BB4 7193

Appendix 1
Summary of MICROTAB Commands

In the command formats given below:

C represents a column name e.g. C2 C15 'INCOME' CK9
C...C represents a list of column names e.g. C2, C4-C6
K represents a numeric or stored constant e.g. 99 -4.77 K5
E represents an item that may be either a column name or a constant.

Square brackets [] indicate optional parts of a command.

Parentheses () indicate extra comments about a command or its usage.

All text entered into MICROTAB should be in upper case.

A command may be abbreviated to whatever is needed to distinguish it from any other command coming before it in the alphabet, to a minimum of two letters. The first three letters are sufficient to distinguish the majority of commands; those that need four or more letters are RMEDIAN, SUBTRACT and STOP. If in doubt, use the full command name.

Lower case text is used in this summary for information which could be entered as part of a command for clarity or as a memory aid but which MICROTAB will ignore.

Where a command format contains several K or E terms, each represents a distinct value and should be entered in the order described.

* * * *

ABSOLUTE value of E store in C

ACF [for up to K lags] on data in C (Auto-correlation function)

ACS value of E store in C (Inverse cosine function)

ADD E to E, ..., to E and store the result in C

ANOVA with K variables, K repeated-measure variables, K levels of first variable, [and K levels of second variable] on data in C...C

ASN value of E store in C (Inverse sine function)

ATN value of E store in C (Inverse tangent function)

AVERAGE of the values in each of C...C (Same as MEAN)

BASE for random-number generator is K

BRANDOM K binomial values for n=K and probability=K, store in C (Bernoulli trials)

BRIEF output (Reduced output from DESCRIBE and REGRESS)

BTRIALS K binomial trials for p=K, store in column C (0's or 1's; probability of a 1 = p)

CCF [for up to K lags] between data in columns C and C (Cross-Correlation Function)

CHISQUARE test on table in columns C...C

CHOOSE values of K [to K] from column C [with matching rows from C...C] and store in C [with matching rows put into C...C]

COPY C...C into C...C

CORRELATE all combinations of columns C...C

COSINE value of E store in C (Cosine function, E in radians)

COUNT the number of values in each of C...C

DEGREES conversion of radian values in E store in C

DELETE row K [to row K] from column C

DELETE row K [to row K] from C...C and store the remaining rows in C...C

DESCRIBE contents of C...C (Amount of output is set by BRIEF)

DIMENSION workspace to K rows per column (Clears all data)

DIMENSION workspace to K rows, set screen to mode K
(Only use mode 7 on standard BBC model B)

DIVIDE E by E and store result in C

END of data (This is an optional command)

ERASE contents of C...C

EXPONENT of E store in C

FRIEDMAN analysis of variance on data in C...C

GENERATE the first K integers into C

GENERATE the integers from K to K into column C

GENERATE values starting from K, in steps of K, up to K into C

HEIGHT of output is K lines (Affects PLOT, MPLOT etc.)

HELP (Displays the names of all MICROTAB commands)

HISTOGRAM of column C [with first mid-point at K]
[and an interval-width of K] (See also PUT)

INFO on columns and constants in use, and format of workspace

INSERT the value K at row K of column C
(Column is enlarged by one value)

INSERT value K at row K of C, store amended column in C

INSERT values K...K at row K of C...C and store in C...C

IRANDOM K random integers between K and K, and store in C
(Uniformly distributed)

JOIN E to the bottom of E [to bottom of E, ..., to bottom of E]
and store in C

KRUSKAL Wallis analysis of variance on C...C

LET E = expression (The equals sign is required. Expression may contain numeric constants, mathematical operators, trigonometric functions, column names and column statistics)

LN of E store in C (Logarithm to base e)

LOG of E store in C (Logarithm to base 10)

MANN Whitney test on data in C and C (For unrelated samples)

MAXIMUM of the values in each of C...C

MEAN of the values in each of C...C

MEDIAN of the values in each of C...C

MEMORY addresses of start and end of worksheet (In hexadecimal)

MINIMUM of the values in each of C...C

MPLOT y in C vs x in C, and C vs C, ...
(Different symbol for each pair)

MPLOT y from K to K in C vs x [from K to K] in C, and C vs C,...
(Restricted range of values)

MULTIPLY E by E, ..., by E and store result in C

NAME for C is 'NAME', for C is 'NAME', ...

NOBRIEF output (Full output from DESCRIBE and REGRESS cf BRIEF)

NOTE (you can put whatever you like here. You can also use #.)

NRANDOM K values with mean K, standard deviation K, store in C
(Normally distributed)

NSCORES of C...C store in C...C (Normal scores)

OMIT values of K [through to K] from column C [with matching rows from C...C], and store in C [with matching rows put into C...C]

ORDER the values in C...C and store in C...C
(Columns ordered independently)

PARSUMS of values in columns C...C and store in C...C
(Running totals)

PARSUMS of length K for C...C and store in C...C (Moving Sums)

PICK rows K to K of C...C and store in C...C

PLOT y in column C vs x in column C

PLOT y from K to K in column C vs x [from K to K] in column C

PLOT C vs C, and C vs C, and ... (Same symbol for each pair)

POOLED t test on C and C [with a K percent confidence interval]
(Unrelated samples, equal variances)

PRINT contents of [columns C...C] [constants K...K]

PUT ACF [for up to K lags] on data in C into C

PUT COUNT for C...C into C (Instead of COUNT, can also use MEAN, SUM, VARIANCE, STDEV, SSQ, MEDIAN, MINIMUM or MAXIMUM)

```
PUT RCOUNT for C...C into C      (Instead of RCOUNT, can also use
                                  RMEAN, RSUM, RVARIANCE, RSTDEV,
                                  RSSQ,  RMEDIAN,  RMINIMUM  or
                                  RMAXIMUM)

PUT HISTOGRAM of column C [with first mid-point at K]
             [and an interval-width of K] in column C

PUT TABLE of frequencies for data in C into C

RADIANS conversion of degrees value in E store in C

RAISE E to the power of E store in C

RANKS for C...C store in C...C    (Columns ranked independently)

RCOUNT of C...C                                    (Row counts)

READ following data into C...C    (One value for each column per
                                                      data line)

RECODE values of K  [through to K]  from C...C  into the value K
       and store in C...C

REGRESS y in C  using K predictors in C...C   [store predicted y
        values in C [residuals in C] ]    (Maximum 9 predictors)

RESTART the program clearing all data previously entered

RETRIEVE the data from file 'NAME'  [into C...C]

RMAXIMUM of C...C                                  (Row maximum)

RMEAN of C...C                                       (Row mean)

RMEDIAN of C...C                                   (Row median)

RMINIMUM of C...C                                  (Row minimum)
```

ROUND E to the nearest whole number and store in C

RSSQ of C...C (Row uncorrected sum of squares)

RSTDEV of C...C (Row standard deviation)

RSUM of C...C (Row sum)

RVARIANCE of C...C (Row variance)

SAMPLE K values from C, store in C (Sample without replacement)

SAVE the data in file 'NAME' [from C...C]

SET following data into C (Any number of values per data line)

SIGNS of E store in C

SINE of E store in C (Sine function, with E value in radians)

SORT values in C [carrying matching rows from C...C] and store in C [with matching rows into C...C]

SQRT of E store in C (SQuare RooT function)

SSQ of values in each of C...C (Uncorrected sums of squares)

STDEV of values in each of C...C (Sample standard deviation)

STOP the program (SAVE the worksheet first if required)

SUBSTITUTE K for the value in row K of C

SUBSTITUTE value K at row K of C, store amended column in C

SUBSTITUTE values K...K at row K of C...C and store in C...C

SUBTRACT E from E, store result in C (Abbreviation is SUBT)

SUM the values in each of C...C

TABLE of frequencies for data in C (See also PUT)

TANGENT of E store in C (Tangent function, with E in radians)

TINTERVAL with K percent confidence for data in C
(t confidence interval)

TSPLOT of data in C...C (Time-series plot)

TSPLOT with period K [starting at K] for data in C

TTEST [of mean = K] on matched data in C and C

TTEST [of mean = K] on difference-values in C

TWOSAMPLE t test on C and C [with K percent confidence interval]
(Unrelated samples, different variances)

URANDOM K values store in C (Uniformly distributed 0.0 to 1.0)

VARIANCE of the values in each of C...C

VDU character K...K

WIDTH of output is K characters [using K characters for columns]

WILCOXON test on values in C and C (Related samples)

WILCOXON test on difference-values in C

ZSCORES of C...C into C...C (Standard scores)

Appendix 2
MICROTAB Error Messages

Errors in the operation of MICROTAB should produce an audible bleep, and display of a short error message. There are two types of error message - those detected by MICROTAB such as an error in the use of a command (these are followed by an exclamation mark) and those from the computer system itself (for example from BASIC, or from the disk or tape filing systems).

The errors detected by MICROTAB are listed below together with possible causes and/or remedies.

Bad command! The line did not start with a valid MICROTAB command name. It may be mis-spelt, or not upper case, or abbreviated wrongly.

Bad argument(s)! The number, type or order of the arguments following the command is incorrect, for example if a constant is entered when a column name is required or vice versa.

Bad name(s)! The name of a column or file has been entered wrongly. Names of files, or the names given to columns with the 'NAME' command, should consist of up to 7 letters and/or digits, beginning with a letter. Names of this type (as opposed to the C notation for columns) should always be enclosed in apostrophes. This error will also occur if a column name is not recognised, or if you try to give the same name to more than one column.

Bad col(s)! One or more of the columns mentioned in a command line is not allowed, for example a column number less than one, or greater than the number in the current worksheet.

Bad row(s)! One or more rows mentioned in a command line is not allowed, usually because it is greater than the maximum for the current worksheet. This error also occurs in editing and data-generation commands if you try to fill or expand a column beyond its maximum size.

In the case of the MANN or KRUSKAL commands,

the total number of values exceeds the maximum column length of the worksheet. SAVE the data, re-DIMENSION the worksheet, RETRIEVE the data and try again.

Unequal cols! The columns given as arguments to a command did not contain equal numbers of rows.

Not enough data! The column(s) given as argument(s) to a command did not contain sufficient rows or values for the command to be carried out.

Impossible! A command could not be carried out because of the nature of the values involved. For example, some statistical routines cannot be used on a column which has zero variance.

Not found! The file in a RETRIEVE command was not found on the current disk-drive or directory. Try *CAT to check the names of your files.

Worksheet! The current worksheet configuration did not allow the command to be obeyed.

In the case of REGRESS, too many predictors were specified.

In the case of RETRIEVE, not all the specified columns and/or rows were RETRIEVEd from the file. To RETRIEVE any more of the data, the worksheet should be DIMENSIONed to at least that which existed when the file was originally SAVEd. Don't forget that the DIMENSION command will clear the data in memory, so any newly entered data should first be SAVEd in a separate file.

Too big This is a BASIC error message which can occur when commands such as REGRESS are used with very large values. The answer is to change the 'units' of the column e.g. measurements in millimetres may be converted to metres by dividing by 1000. Results then need to be interpreted in terms of the new units.

For the meaning of any other error messages, consult your BBC Microcomputer User Guide, Disk Manual etc.

Appendix 3
Operating System (*) Commands

A MICROTAB command line which begins with an asterisk or 'star' is normally passed on to the operating system or current filing system. Some useful commands of this type are:

*TAPE — Switch to the tape filing system.

*DISK — Switch to the disk filing system.

*CAT or *. — List the names of files on the current disk or tape unit.

*BUILD filename — (Disks only). Copies commands to a file for later execution by *EXEC. Commands are not executed during a *BUILD operation. Pressing ESCAPE terminates the process.

The tape filing system does not provide *BUILD, but it can be emulated by a BASIC program (i.e. outside the MICROTAB system):

```
10 DIM A 20
20 INPUT"Name of file: " F$
30 $A="SPOOL " + F$
40 X%=A MOD 256 : Y%=A DIV 256 : CALL &FFF7
50 ON ERROR GOTO 70
60 REPEAT: INPUT LINE" "A$: UNTIL FALSE
70 *SPOOL
```

When this program is RUN, it will ask for the name of the file to be created, then copy all the text that is typed to the file until the ESCAPE key is pressed.

*EXEC filename — Will read in MICROTAB commands from the specified file until the end of the file is reached. The commands are executed line by line as they are read from the file. Such a command file may be created using *BUILD.

*SPOOL filename — Will copy all succeeding commands and their output to the specified file, until a *SPOOL command (not followed by a filename) is given. This file can be displayed or printed from tape or disk. Use of a WIDTH command beforehand may be used to obtain a version of MICROTAB output which can be output to an 80 column printer, or displayed in an 80 column screen mode when

MICROTAB is not running.

*TYPE filename — Will display the text from a disk file e.g. one created by *SPOOL or *BUILD.

*SAVE filename a1 a2 — Stores the contents of memory in the specified file, beginning at memory address a1 and ending at (a2 minus one). *SAVE can be used to store the worksheet (see Appendix 6) although the MICROTAB SAVE command should be used wherever possible.

*LOAD filename a1 — Loads a file (usually one previously *SAVEd) directly into memory at address a1. If a1 is omitted, the file is loaded back into its original memory location.

*KEY n "text" — is used to program the red programmable function keys (f1 to f9). This is useful where a command is frequently used, or needs to be repeated several times. The CK convention for referring to columns may also be employed to good effect. For example, if t tests are required between C1 and each of C6 to C15, then typing:

```
LET K1=6
*KEY 2 "TTEST C1 VS CK1 |M"
*KEY 3 "LET K1=K1+1 |M"
```

followed by alternate pressing of the f2 and f3 keys could save a lot of time.

Key definitions can be stored in a file using *BUILD and installed using *EXEC.

Certain function keys have already been programmed for you. These are:

```
f0 HELP
f1 INFO
f4 WIDTH 40
f8 WIDTH 80
```

These keys can, of course, be re-programmed if desired.

Certain * commands should not be used from MICROTAB, such as some of the disk filing system commands which overwrite user memory e.g. *COMPACT. Any of these commands which are needed should be carried out before loading and running MICROTAB.

Appendix 4
Control Characters

Control characters are typed by holding down the CTRL key on the keyboard and typing another key - usually a letter - while CTRL is held down.

Control characters may be entered at the start of a command line to control various functions. The most useful ones are:

CTRL B	Copies all subsequent screen output to the printer until a CTRL C combination is typed. Make sure the printer is switched to its 'on-line' setting.
CTRL C	Reverses the effect of CTRL B.
CTRL N	Puts the screen into 'page mode' so that the display halts at the end of each screenful until the SHIFT key is depressed.
CTRL O	Reverses the effect of CTRL N and puts the screen into normal scrolling mode. In scrolling mode, the display can be halted temporarily by pressing CTRL and SHIFT at the same time.
CTRL L	Will clear the computer screen. If the printer has been activated, a Form Feed (new page) may occur.
CTRL U	Will delete the whole of the command line that has been typed so far.

Control characters may also be issued by means of the VDU command (see page 74), although this is more useful where MICROTAB commands are stored in file (page 36).

e.g. VDU 2

is equivalent to pressing CTRL B.

Appendix 5
Increasing Memory Space

The limited memory of the BBC micro Model B (and to a lesser extent the Model B+) restrict the maximum size of the MICROTAB worksheet.

The location in the computer's memory where MICROTAB is loaded is controlled by a special BASIC variable called PAGE. The most usual value of PAGE is &1900. The size of the worksheet may be increased by lowering this value before the program is loaded and run. The new value of PAGE depends on the maximum number of disk files you will have open at the same time. This will probably be between one and three:

- 1 during a SAVE or RETRIEVE command.
- 1 during the use of *SPOOL (e.g. for keeping a record of a session on disk).
- 1 during the use of a stored command file with *EXEC.

The lowest value that PAGE can take is &1200 plus &100 for each file open simultaneously. For example, if you intend keeping a 'session-log' using the *SPOOL facility, and during the session you want to use a SAVE or RETRIEVE command, you could set PAGE as low as (&1200+&200) = &1400. The worst case that should occur in normal operation is a *EXEC command-file performing a SAVE or RETRIEVE whilst the session is being recorded with *SPOOL. To allow for this contingency, PAGE should be set to at least &1500.

If PAGE is changed in this way, MICROTAB should be run by using *EXEC !BOOT or CHAIN"MTAB" rather than SHIFT and BREAK.

These values only apply to the standard Acorn disk filing system (without second processor, Econet or other ROMs which reserve space for themselves) where the initial value of PAGE is &1900. Other systems may require variations to what is described above.

CAUTION should be exercised in following this procedure, since certain operations of the disk filing system can, in principle, overwrite the program and cause loss of data. In particular, the DFS *BUILD command should NOT be used. No responsibility is accepted for faults that arise from using this procedure.

Appendix 6
Alternative Worksheet Storage

The Econet Level 1 filing system does not allow the use of the normal SAVE and RETRIEVE commands for storing data (a 'Bad command' error will result if you try to use them). There is another (although much less preferred) procedure for storing and retrieving the whole of the worksheet which should work on any filing system, including Econet Level 1.

This alternative procedure makes use of the BBC micro's *SAVE and *LOAD commands, which save and load a block of the computer's memory. The general form of the *SAVE command is:

```
*SAVE filename SSSS EEEE
```

where filename is any allowable filename in the current filing system, and SSSS and EEEE are the start and end memory addresses of the data in hexadecimal notation.

Values for SSSS and EEEE may be found using the command MEMORY,

e.g. MEMORY

```
5BB4    7193
```

An example of a command to save the workspace would then be:

```
*SAVE DATA3 5BB4 7193
```

The data-file would be retrieved on a future occasion by:

```
*LOAD DATA3
```

The following rules apply when using *SAVE and *LOAD:

1) The SAVE/RETRIEVE and *SAVE/*LOAD procedures must not be intermixed e.g. you cannot RETRIEVE a file that has been *SAVED.

2) *SAVE stores the whole worksheet, including the empty columns, and so is not efficient when only a relatively small part of the worksheet contains data.

3) *SAVE may only be used with the current Econet Level 1 file-server if the size of the worksheet is not more than 12288 bytes (3000 in hexadecimal notation). The size of the worksheet may be calculated by giving a MEMORY command, then subtracting the second from the first of the displayed values

e.g. MEMORY

```
        5BB4    7193

LET K1 = &7193 - &5BB4

        RESULT=5599
```

4) When *LOADing a file, the size and configuration of the worksheet must be the same as that which existed when the file was originally *SAVED. This can partly be determined by giving a MEMORY command - the values displayed should be the same as those used for the original *SAVE command. Data stored in this way is therefore much less 'portable' between different machines than that stored using the normal SAVE command.

Appendix 7
Bibliography

There is a wide variety of statistics books available, many of which are oriented towards a particular subject area or discipline. Some of those which the author has found useful are:

C.Chatfield — The Analysis of Time Series. Chapman and Hall, 1975.

G.A.Ferguson — Statistical Analysis in Psychology and Education. McGraw-Hill, 1981.

R.E.Kirk — Experimental Design: Procedures for the Behavioral Sciences. Wadsworth, 1968.

J.D. and T.D.Lee — Statistics and Computer Methods in BASIC. Van Nostrand Reinhold, 1982.

C.Robson — Experiment, Design and Statistics in Psychology. Penguin Books, 1973.

S.Siegel — Nonparametric Statistics. McGraw-Hill, 1956.

A more specialised journal article which provides an interesting discussion on the content and accuracy of statistical packages, together with test data, is:

T.G.Greenfield and R.Siday — Statistical Computing for Business and Industry. The Statistician, Vol. 29, 33-35, 1980.

Appendix 8
Enhancements to MICROTAB

Since its original publication, a number of small improvements have been made to the MICROTAB software. Those which apply to version 1.2 are briefly described below.

Function commands (ABS, SQRT etc.) now work with a single argument, e.g.

```
SQRT C1     is equivalent to   SQRT C1 C1
```

Similarly, column arithmetic commands (ADD, MULTIPLY etc.) may now be used with only two arguments, at least one of which must be a column. The result is stored in the last-mentioned column, e.g.

```
ADD 5 C1    is equivalent to   ADD 5 C1 C1
```

*SAVE now saves column names and stored constants, but data *SAVED by different versions of MICROTAB are incompatible.

The READ command with a single column now acts like SET and allows multiple data values to be entered on each line.

ANOVA followed by just a list of columns performs a one-factor independent-group procedure, e.g.

```
ANOVA C1-C3    is equivalent to   ANOVA 1 0 3 C1-C3
```

B+ and Master series machines running BAS128, or systems with the 6502 second processor running HIBASIC, can have columns longer than 255 rows SO LONG AS MICROTAB IS NOT BEING LOADED FROM AN ECONET LEVEL 1 FILE SERVER.

To ensure correct operation, users of BAS128 and HIBASIC with Econet Level 1 MUST keep the SHIFT key held down while MICROTAB is loading.

First published 1985 by
Edward Arnold (Publishers) Ltd, 41 Bedford Square, London WC1B 3DQ.

Reprinted 1986

ISBN 0 7131 3549 2

Printed in Great Britain by
Thomson Litho Ltd, East Kilbride, Scotland

For the relative newcomer to statistics and computing, or those already familiar with both, MICROTAB provides a complete statistical package for use on BBC microcomputers.

A single integrated program allows data to be entered, edited, tabulated, transformed, stored, plotted and statistically analysed. Plotting routines include single and multiple scatter plots, histogram, and time-series plots. Statistical procedures offer descriptive statistics, confidence-intervals, t tests, correlations, multiple regression, one or two-way analysis of variance (including 'repeated-measure' designs) and time-series functions. Non-parametric statistics include chisquare, Mann-Whitney, Wilcoxon, Friedman and Kruskal-Wallis tests. In addition, data may be generated from various 'random' distributions, and their properties explored, and the effects of various transformations demonstrated.

MICROTAB is controlled by over one hundred natural English language commands e.g. PLOT 'HEIGHT' AGAINST 'WEIGHT', although the system allows more experienced users to abbreviate commands on their own convenience.

Output from MICROTAB can also be stored on tape or disk, or sent to any standard printer. It is possible to keep a 'log' of a whole MICROTAB session, which will be of use to those in a teaching environment.

Also published by Edward Arnold

Basic Statistical Computing Software (BBC)
(also available for RML 380Z/480Z, Apple II, PET, Commodore 64)
D Cooke and A H Craven

Edward Arnold