FILE COPY

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Project MAC

Memorandum MAC-M-228-1
COMCOM/Simulation Memo No 23
Noel I. Morris
Thomas H. Van Vleck
Revised 8/15/65

A General Cross-Tabulation System

Introduction

As a research tool, the COMCOM project has developed a highly general survey analysis and cross-tabulation program system. The system is designed to accept survey data in virtually any format, and to produce neat, highly readable output. Included in the system are facilities for doing simple recoding on input data and producing output in a standard format on magnetic tape for further use.

The system is divided into several sections: The Cross-tabulation routines read and process survey input data and write a magnetic tape containing tabular information. The Labeling routines will accept information about the variables used in the input data in order to produce highly readable tables. The Statistics package will compute requested statistical measures on the output from the Cross-tabulation section. The Output routine is designed to produce tabular output in highly readable format, from the tapes produced by the other routines.

The system will handle cross-tabulations of up to six dimensions; that is, six variables may be tabulated against each other simultaneously. Each variable is broken into a number of response levels. The system is limited in the "varmax" (highest value) of a variable only by the capacity of the core storage in the computer.

All system routines are controlled by control cards written in a free, but standard format.

The Cross-Tabs Routines

We wish to provide the survey analyst with tables, which present data in a form generally known as "cross-tabulations". These tables summarize the frequency of responses within the framework of a multi-dimensional attribute space. To read survey data and produce cross-tabulations is the primary objective of these routines.

The most elementary format of an attribute space in multivariate analysis is two-dimensional. The basic model of a cross-tabulation is, therefore, the two variable table. It is the building block for tables summarizing three or more variables. In such a table, two variables which we will designate by the letters C and V, each having c_n and v_n classes respectively, are joined in union to form a two-dimensional space, C X V. When joined, the classes of the two variables form n_c times n_v subdivisions of this attribute space, the cells of the n-fold table. Each cell of such a table represents a unique combination of classes from variable C with a specific class from variable V. The number recorded within each cell of a cross-tabulation gives the frequency with which the two-class combination represented by the cell is observed in the survey under analysis.

A typical example of a two-variable cross-tabulation is the nine-fold attribute space formed by two trichotomies. This is illustrated below:

	C_1	C_2	C_3	
V_1				Mv_1
V_2				Mv_2
V_3				Mv_3
	Mc_1	Mc_2	Mc_3	T

The upper leftmost cell, for example, represents the combination of classes c_1 and v_1 . The upper rightmost cell stands for the combination of classes c_3 and v_1 . Frequencies recorded in each of these cells, as stated above, give the number of occurences of each such two-class combination within the survey sample studied.

With traditional data processing machinery like the IBM Counter-Sorter, a nine-fold table like the one above is produced in a two-step operation. In a first operation, the data cards are classified according to the class definitions of variable C by a sorting pass. This first step, in our example, segregates the data cards into three groups, each representing one of the variable C-classes. At the same time, the operation yields the three marginal frequencies Mc_1, Mc_2, and Mc_3, that is, the number of survey respondents having the attribute c_1, c_2, or c_3. The second step in the production of this nine-fold cross-tabulation consists of a counting operation: Processing one variable C-group at a time, all the cards of such a group are now counted on variable V. The results of this counting operation are entered -- one class C column at a time -into the appropriate table cells.

Once all of the cell frequencies have been entered in the table, the production phase is finished and the analytic one begins. The first step in the analysis of the table consists of choosing a statistical measure. In "rudimentary analysis" -- to return to the phrase quoted earlier -- this may mean the transformation of the raw frequencies into percentages. This requires the classification of the table variables as dependent and independent variables. Whether variable C or V is the independent variable is irrelevant for the machine production of the cross-tabulation but crucial for its analysis. If C is defined as the independent variable, the marginal frequencies Mc_i become the base figures for the computation of percentages. If, on the other hand, V is defined as the independent variable, the percentaging has to be done by table row using the marginal frequencies Mv_j as base figures.

The operations just described -- production of the cross-tabulations and transformation of each table's row or column frequencies into percentages -- are what the Cross-Tab Routines and their associated Output Routines are designed to do. The Cross-Tab Routines are simple to use; they require only two specification cards, though more may be used. Despite their simplicity, however, the routines are much more flexible than most generalized cross-tabulation programs:

1) A given table may contain up to six variables, with no limit on the number

of classes for each variable.

2) The only limit on the number of cards the program will handle or the number of tables it will generate during a run is the capacity of core storage.

Cross-tab Control

The following section describes the cards which are used to control the crosstab routine. There are four functional groups of cards, dealing respectively with data format, values of the input variables, recoding and construction of variables, and table specification. There must be at least one card from the "data format" group and at least one from the "table specification" group in each input deck.

Control cards must all begin with an asterisk in column 1. Columns 2-7 contain the control function, and columns 8-72 contain the control information. If the control information cannot fit on one card, continuation cards may be used. These must contain the asterisk and the same function word in columns 1-7. The last card (and only the last card in a set) must contain an asterisk after the control information.

Comments may be placed anywhere in the control information by bracketing the comment with apostrophes ('). Blanks are ignored anywhere in columns 8-72.

1. Data Format

Survey data may be supplied to the cross-tab routines on cards or on binary tape. The card reading routine will accept variables which occupy more than one card column, but will not be equipped to process multiple punched columns. An editing routine has been written which will accept multiple punched cards and convert them into a binary tape.

a) The binary tape reading routine has been designed to accept virtually any possible configuration for a binary data tape. It will process packed and unpacked tapes with any number of respondents blocked per record. (Packing means putting more than one variable in a computer word. This saves space on the binary tape and reduces tape reading time. Blocking, or putting more than one respondent in a tape record, reduces the number of tape records and also results in a greatly reduced tape reading time. The packing density may be uniform, or it may be variable.) The control card for binary tape data supply is of the following form:

*BINARY A/B/C/D/E/F/G/H/J/K *

A = input survey data tape
B = output tape for generated tables
C = total no. of respondents in survey
D = no. of variables in survey input
E = no. of created variables from filter and boolean routines.
F = greatest level number of variables (varmax)
G = direction of percentaging: "H", "V", or "T"
H = no. of bits per variable, or "N" for non-uniform packing
J = respondents per physical tape record
K = computer words per respondent

If the data on the input tape is not uniformly packed, packing must be specified by an additional card of the following format:

*PACK 1+3-6/n, 2/m, ... *

This card would read "pack variables 1 and 3 thru 6 at n bits per variable, pack variable 2 at m bits, etc." ("-" denotes thru and "+" denotes and). Note that if more than one "*PACK" card (or any other control card) is needed, an asterisk should be used to denote the end of the last card only. If it is required, the "*PACK" card must follow the "*BINARY" card immediately.

b) If the input data is on BCD tape--perhaps in the form of card images--instead of binary tape, the following control card should be used:

*CARDS A/B/C/D/E/F/G *

where "A" through "G" are the same as for the "*BINARY" card.

The "*CARDS" control card must be followed immediately by cards which tell how the variables are laid out on the input deck. This information is conveyed by the "*FORMAT" card:

*FORMAT I3, S4, I1, 1812, ...
*FORMAT ....   *

where the format is in standard MAD notation. The letter "I" designates a field on the survey input deck which is a decimal integer, taking the number of columns specified by the digit after the "I". Similarly, "S" designates a number of columns to be skipped on the input deck; this is especially useful if some questions on a survey have been multiple-punched but do not need to be considered in this cross-tab run. A number preceding a field specification indicates that the specification should be used more than once: thus, "3I3" is the same as "I3, I3, I3" . A slash ("/") in the format specifies that the next card is to be read: this enables the program to consider respondent decks of more than one card. For a more complete description of formats, consult the MAD manual, or Computation Center memo CC-186.

c) If the input is in a special form, containing most of the control information necessary for the use of the input, the following control card may be used: *

*SUPPLY A/B/E/G *

where A, B, E, and G are as on the "*BINARY" card. The "varmax's" of the input variables need not be specified if this option is used, as they must be on the input tape.

2. Values of the Variables

a) If some of the variables used in the survey have different maximum values (varmax), then the "f" entry should be that number of levels that appears most in the survey. Then another control card is necessary to specify the maximum level ("varmax") of only those variables which are different. This card appears as follows:

*VARMAX 1*5-8/n, 4/m, ... *

The interpretation is the same as for the "*PACK" card. Here, however, the maximum level of a variable is defined. Note that a variable with a "varmax" of 10 has 11 levels -- zero thru ten.

b) Variables may be checked as they are read in to insure that they do not exceed their stated "varmax". The appearance of a "*CHECK" card will insure the checking of each respondent's variables. Nothing else need be specified on the card. If a respondent exceeds his "varmax" on one or more variables, the following line will be printed for each such illegal variable:

VARIABLE n OF RESPONDENT m EXCEEDS VMAX OF q WITH RESPONSE OF r.

Then, the response (not the respondent) will be discarded.

3. Recoding and Creating Variables

a) Variables may be recoded by the use of the "*FILTER" card. Filter acts on a single variable in such a way as to combine the levels of that variable. For example, a variable of 10 levels could be changed into a dichotomy. The recombined variable could replace the old variable of 10 levels, or it could become a new variable. The filter control card looks like this:

```
*FILTER   (newv=oldv)0+2/1+4/5-10 ,
*FILTER   (oldv)1-5/6-10 , ...... *
```

In the first case, a new variable is created from an old variable: its eleven levels are combined into three, and the old variable is undisturbed. Note that level 3 is ignored in the recombination and will not be counted during the generation of the tables. In the second case, levels 1 thru 5 are considered equivalent, as are 6 thru 10. Thus the recoded variable has two levels -- 0 and 1.

b) Variables which have a large "varmax" would create too large a recoding table if they were to be "filtered." Therefore in order to avoid running out of storage, it is recommended that variables with maxima greater than 20 not be "filtered." Another operation is possible, which recodes by the use of cutting points. These may be specified on a card of the form: *

```
                1 10 50 100 1000
*SPLIT   (8=2) ~~2/11/51/101/10001~~     *
```

This control card will cause variable 2 to be recoded as variable 8. Responses 0 and 1 will be recoded as 0, responses 2 thru 10 will go to 1, and so forth. Responses greater than 1000 will be ignored.

c) Cards to control the Boolean Generator, described in simulation memo no. 9, should be preceded by a "*BOOL" card, which instructs the cross-tab program to transfer control to the Boolean Generator. The Boolean Generator will then read input cards, setting up its internal control arrays, until it finds a terminating dollar sign. Then it will return control to crosstab. A complete description of this routine and its control cards may be found in COMCOM/ Simulation Memo no. 9.

SPLIT, FILTER, and BOOL may be called upon once each in an input deck. Each time a respondent is read in during the table generation phase, the order of operation is: SPLIT, FILTER, BOOL; any or all of which may be omitted.

Note that if new variables are specified on a "data format" control card, these must be numbered starting with the first number not used by the survey variables. Thus, if 10 new variables are specified in processing a survey of 100 variables, they will be numbered 101 thru 110.

4. Table Generation

The generation of tables is controlled by the "*TABLES" control card. This card specifies which variables are to be cross-tabulated against each other. For example, if a table of variable two against variable one were desired, and also a table of variable two against variable three, the control card would look like this:

```
*TABLES   2/1,   2/3  *
```

This format is somewhat inconvenient, however, when many tables are being specified The same effect on table generation may be had by using the following format:

```
*TABLES  2/1+3  *
```

Hence, more than one table can be specified in one specification. This format can be extended to make possible the generation of many tables:

*TABLES 1-3+5/4+6/17+19, 4/9+11/10/15/17-20 *

The two specifications on this card would cause 16 three dimensional tables to be generated as follows: 1/4/17, 1/4/19, 1/6/17, 1/6/19, 2/4/17, ..., 5/4/19, 5/6/17, 5/6/19. The second specification would cause the generation of 8 five dimensional tables in a similar manner. As many "*TABLES" cards as needed may be used, with an asterisk being used to terminate the last one only.

This completes the entire deck of control cards needed to produce a set of tables. One word of caution should be added, however: With the ease of specifying tables to be generated, one can generate many w.b.f. (waste-baskets-full) of computer output with one "*TABLES" card. The cross-tabs program user should examine his specifications carefully to determine whether he is producing valuable statistical information or just garbage. The routines will only generate as many tables as will fit into core storage; if more tables are specified than can fit, it will ignore the excess. A comment will be printed if storage capacity has been exceeded in the tables specification.

There are two additional cards available for those who have multi-file data tapes which they wish to process efficiently; a "*REWIND" card will rewind the survey input tape at the beginning of the job, and a "*SKIPFL" will skip 1 file on the survey tape. *

?.flag

Be sure to include a "*REWIND" card in the control deck when submitting several runs using the first file on the same survey tape. The operators will <u>not</u> rewind the survey tape between jobs

Example

To give a better idea of how the system can be used, let us concoct an example which might arise in actual practice. Suppose that a survey has been taken which included answers by respondents to questions concerning annual income, years of education, party affiliation and voting record, number of magazines, books and newspapers read per week, and similar survey questions. It is desired to test the effectiveness of a particular index of socio-economic status, defined by the following table, in estimating certain population parameters: we will have three classes, A, B, and C.

	Income, thousands of dollars				
Education	0-5	5-7.5	7.5-10	10-15	15+
0-8 yrs.	C	C	C	B	A
8-12 yrs.	C	C	B	B	A
12-bach.	B	B	B	A	A
some adv.	A	A	A	A	A

Thus, we reduce the 20 different possibilites for combinations of education and income to a trichotomy, which we wish to tabulate against several of the other variables.

If our input were originally on multiple-punched cards, we would first use the recoding program to create a binary tape. We would also punch up the survey questions and from them create a label tape. Assuming that the actual income, shown in dollars, was variable 12, and that we had actual years of education in variable 8, it would be necessary to recode these variables into new levels, using the filter and split routines.

Our input would consist of the binary survey data tape and several control cards for the handling of the input to the system (*BINARY, *PACK), for setting maximum values of variables (*VARMAX, *CHECK), for the operations to be performed on the data (boolean cards and "*SPLIT" and "*FILTER" cards), and for requesting output ("*TABLES").

Suppose that our survey data tape represents the results of a 106-question survey given to 6000 respondents. The data are organized in 10-word blocks, three per tape record; the maximum level of all but variable 12, income, and variable 8, years of education, is 9. We will have non-uniform packing because of the two big variables. It is desired to create the socio-economic status indicator shown above as a new variable and tabulate it against variables 1, 4, 69, 71, 72, 99, and 14 through 25. It will be necessary to create another new variable which will have the five levels of income groupings shown in the headings in the table, and to recombine the years-of-education variable into itself. If the survey tape were on unit 11 and the output to be on unit 12, our input deck for the cross-tabs program would look like this:

```
*BINARY 11/12/6000/106/2/9/H/N/3/10 *
*PACK   12/24 , 8/8   1+7+9-11+13-106/4      *
*REWIND
*VARMAX  12/30000, 8/50 *
*FILTER (8)0-8/9-12/13+16/17-50 'example only' *
*SPLIT  (107=12) 5000/7500/10000/15000/30000  *
*BOOL
    if v8=0 or v8=1 then v108=0
    if v107=3 or v8=2 or v107=2 and v8=1 then v108=1
    if v107=4 or v8=3 or v107=3 and v8=2 then v108=2 $
*TABLES  108/1+4+14-25+69+71+72+99      *
```

OUTPUT CONTROL

The Output Routine reads the binary tape produced by the Cross-Tabulation Routine and prints a listing of the tables generated.

The first variable specified on the "*TABLES" card will specify the vertical cells of a table, the second variable, the horizontal. Higher number variables are represented by the separate blocks of tables. (e.g. A 3 x 3 x 3 table would appear as three 3 x 3 tables.)

Comments may be included in the data deck for the Output Routine. These comments will be printed out above the table specified.

Marginals for the first two (leftmost) variables are automatically produced by the Output Routine. In addition, all cell entries are percentaged. The percentaging specification was given on the "*BINARY," "*SUPPLY," or "*CARDS" card in the Cross-Tabs Routine. This specification is "h", or "v", or "t". The "h" specification implies horizontal percentaging. This means that cell entry percentages will add up to 100% reading across a table row. In this case, the second variable has been made independent. Likewise, "v" implies vertical percentaging, holding the first variable independent. The "t" specification indicates that <u>both</u> the first and second variables should be made independent. Percentaging will be on the total that appears at the junction of the marginal row and column. In a two dimensional table, this will be the total sample size.

A title which will be printed at the head of every table may be specified by submitting a card of the form:

```
*TITLE   ANY TITLE DESIRED, EXTENDING TO COL. 72
```

Variables are normally labeled with the phrase "VARIABLE NUMBER xx" and response levels with "RSPONS y." For more descriptive labeling, it is possible to submit labeling cards which specify an 18-character label for the variable and 6-character labels for up to 15 responses. Only 200 such labels may be used at one time.

The labeling cards must contain a star (*) in column 1, the number of the variable which is being labeled in columns 2-6, and the labeling field in columns 7-72. The labeling field consists of two parts, each of which terminates in a star. The first part is the label for the variable: it must be 18 or fewer characters. The second part is the category label section, which contains labels for each category (level) of the variable, separated by slashes. The category labels may be 6 or fewer characters each. Only one card may be used for each variable; they may appear in any order. It is assumed that the category labels correspond in number and in order to the categories of the variable being labeled.

To replace the labels "VARIABLE NUMBER xxx" with "COLUMN y-xx," which is useful if the input corresponds to successive single columns on one or more cards, a card of the form

*cols

may be supplied. Variable 151 will become column 3-1, and so on.

The following statistics may be requested on separate control cards

Statistic	Cols. 1-6 of control card
gamma	*GAMMA
lambda	*LAMBD
chi square	*CHISQ
phi square	*PHISQ
pearson c	*CCOEF

The statistics will be computed for the bivariate array represented by the first two dimensions of every table. they may be requested in any order, singly or together, but will be printed in the order given above. Only the value of each statistic is computed; probability of occurrence, estimation of the population values of gamma and lambda with confidence limits, and so forth, must be added to the program later.

The data deck for the output routine is organized as follows: the first card in the data deck must contain in cols 1 and 2 the logical tape number of the tape unit on which the table tape is mounted. If a "*TITLE" card is to be used, it comes next, followed by statistics request cards and the "*COLS" card if any of these are desired.

If variables are to be labeled, the label cards are preceded by a "*VARS" card. After the variable-labeling cards, a "*TABS" card may appear to indicated that table comments follow. If one wishes to print a comment before a particular table, he punches the comment on any number of cards and precedes it by a card which has an asterisk in column 1 and the table number in cols. 2-6. After the appearance of such a card, cards are read until the next control card, which contains an asterisk in column 1. (Thus, the comment cards must not have an asterisk in col.

1.) The next card may be another table-designating card for the next comment, or "*END," which must be the last card in the output control deck.

A data deck for a user who wished to include "the works" might appear as follows:

```
12
*TITLE TOLERANCE AND RELIGIOSITY STUDY 1/1/65
*COLS
*GAMMA
*CHISQ
*LAMBDA
*PHISQ
*CCOEF
*VARS
*002  SEX * MALE/FEMALE *
*016  LITERACY * LITRT/ILLIT*
*100  TOL OF DEVIANCE*HIGH/MED / LOW *
*020  AGE   *15-19/20-29/30-49/50 *     *
*102  RELIGIOSITY 1  *HIGH/LOW  *
*99   EDUCATION * HIGH/MED/LOW *
*TABS
*001
SEX VS. LITERACY VS. TOLERANCE OF DEVIANCE
*3
EDUCATION VS. SEX VS. RELIGIOSITY
*5
TOLERANCE OF DEVIANCE VS. INDEX 5
THIS HAS BEEN CONSTRUCTED FROM INDICES 1 AND 3
*END
```

Time estimates

Our tests indicate that the crosstab system is faster than most of the programs developed in the past to perform similar work. Time estimates for a crosstab run are difficult, since the run time is a function of many variables, such as the number of respondents, the number of tables generated, the amount of filtering done, and others less obvious. The following figures have been observed in practice, but an individual run may differ wildly from these guesses.

The advantage of binary input becomes evident when we consider job time. Crosstab can do as well as 30-50 milliseconds/respondent on a simple control deck with binary input; on more complicated runs, the times approach 100-200 ms./respondent. BCD tape is definitely input-output-bound: a fair estimate is 250 ms./card.

Timing on output is somewhat easier to guess, if one has some idea of how many lines of output are expected. Most output runs take about one minute for every 1500 to 2000 lines of print, depending in part on the amount of labeling and statistical computation requested.

Unfinished business

The crosstab routines have been operating for over six months, and most of the major bugs have been painfully removed. Our current effort is directed toward adding a complete set of error diagnostics, to correct the unfortuante tendency of the program to blow up if presented bad specifications.

Crosstab is by no means complete. We hope to add many new features in the coming months, and we welcome suggestions concerning improvement of the system.

For binary decks of the program and a certain amount of sympathy when things go wrong, contact Noel Morris or Tom Van Vleck, T.S. 502.

TABLE NUMBER 16.

THIS IS A SAMPLE OF THE OUTPUT FROM THE TABLE PRINTING ROUTINE. IN THIS SPACE ONE MIGHT IDENTIFY THE VARIABLES BEING TABULATED OR PUT SOME INDICATION OF WHAT THE TABLE WAS MEANT TO SHOW.

VARIABLE NUMBER 107	LEVEL 0			LEVEL 1		
VARIABLE NUMBER 22	LEVEL 0	LEVEL 1		LEVEL 0	LEVEL 1	
VARIABLE NUMBER 53						
LEVEL 0	40 .667	20 .333	60 .600	120 1.000	0 0.	120 .800
LEVEL 1	10 .250	30 .750	40 .400	10 .333	20 .667	30 .200
	50 .500	50 .500	100 1.000	130 .867	20 .133	150 1.000

	LEVEL 2		
	LEVEL 0	LEVEL 1	
LEVEL 2	1 .250	33 .794	16 .480
LEVEL 1	7 .667	27 .930	34 .320
	8 .100	42 .840	50 1.000