A General Program for the Analysis of Surveys

By F. Yates and H. R. Simpson

A program for the analysis of surveys, written for the Elliott 401, is described. In spite of the small size of the machine it has been possible to develop a general program which is capable of performing the very varied operations that are commonly required in the analysis of survey material. This paper was originally presented at the Harrogate Conference of The British Computer Society on 5 July 1960.

Introduction

Surveys are characterized by the collection of information on a number, usually large, of "units." The information on each unit may be of any degree of complexity. In certain opinion and market research surveys, for example, only "yes" or "no" answers to specific questions may be collected, together with simple characteristics, such as age and sex, of the person interviewed; in a farm economic survey full information on the working and economics of the farm may be sought. Certain items of information may be known beforehand and form the basis of selection of the units.

In many surveys there is a hierarchy of units. Thus, in a household survey, the household constitutes one type of unit, and each individual in the household a second type of unit. In such a survey, moreover, we may select a sample of households, and from each selected household a sample of individuals; this is known as *two-stage sampling*. Furthermore, additional items of information may be collected from a sub-sample only of all the units in a survey; this is known as *two-phase sampling*. Multi-stage and multi-phase sampling may be combined in the same survey.

A general program for the analysis of surveys must make provision for the very varied types of analysis that are required in different types of survey. A scheme for such a program on a large computer has been outlined in the third edition of *Sampling Methods for Censuses and Surveys* (Yates, 1960). At first sight it appeared that a program of this generality would not be practicable on a small computer such as the Elliott 401 (a drum machine with a store of 2,944 words) at present available to us at Rothamsted. Further investigation showed, however, that a useful general program could be written. The present paper contains a brief account of this program.

General Scheme of Analysis

The analysis is carried out in two parts. In the first part the information is read in and processed unit by unit. This processing includes any computations which are necessary on the data appertaining to each unit (e.g. grouping, ratios and indices, etc.), and the compilation of all the required tables. The second part consists of the derivation of the final tables from the tables that have been compiled in the first part, and the printing out of these tables. The first part is now complete and in use. The second is being written.

Since the two parts of the program are independent and each is only required at one stage of the analysis, the two parts are not held in the store simultaneously. This considerably economizes storage space, thus making a larger part of the store available for the actual tables. Various other devices for increasing the number of tables that can be constructed simultaneously are discussed below.

Input of Data

The recorded items of information will be termed *original variates*. Variates are referred to by number (1–63). The values appertaining to any particular unit are recorded on one or more punched cards, or in a group on punched paper tape. All variates are treated as integers during the computations. The input routine is designed to read the information from the cards or tape, decode it in the required manner, and store the corresponding values of the variates temporarily in a set of store locations termed the *variate store*. Any required constants may also be held in the variate store. Owing to limitations of the card reader, multiple punching is not admitted (the existence of a single hole in each column read is used as a check), but all 12 positions may be used.

Before transference to the variate store, each variate is tested against pre-assigned upper and lower limits. If these limits are exceeded, the variate is coded as "unknown" ($-1^{10}0^{31}$), and an indication is printed.

The input routine is so written that data can be read from two or more cards, which may have different patterns of fields. The different types of card are distinguished by a single-digit code number punched in some chosen column, and the machine checks that cards are presented to it in the right order. This facility for reading different types of card is particularly important on the 401 because the card reader can only read up to 32 columns simultaneously, the particular columns being selected by plugging. Even with an 80-column reader, however, similar facilities should be provided, since surveys are frequently encountered in which the information on a unit is recorded on two or more cards.

Classification Variates

Variates which are used for classification are termed *classification variates*. The method of table construction requires that each of these variates has a defined (integral) range, and conventionally the lowest value is taken as 0. If a variate not having a lower limit zero is defined as a classification variate, a constant is added to or subtracted from it on input so as to give a lower limit of zero.

Derived Variates

It is frequently necessary to form further variates which are functions of the original variates. Such variates are termed *derived variates*, and are stored alongside the original variates in the variate store.

The formation of the derived variates is controlled by a set of *derived variate instructions* which are variants of the form

$$X_D \quad F(X_A, X_B, K) \quad \text{or} \quad X_D \quad F(X_A, X_B, K', K'')$$

and are written

$$\text{F.D.A.B.K} \quad \text{or} \quad \text{F.D.A.B.K'.K''}$$

Fourteen instructions, denoted by function numbers, are included in the program. These are:

(*a*) Four arithmetical functions, $\times$, , $\times$, and $\div$. In each of these there is provision for a scaling factor, denoted by K, since fixed-point arithmetic is used.

(*b*) Two functions of the logical type, namely:

If X_A, X_B and/or X_K 0, X_D 1, otherwise X_D 0.

(*c*) Four repetitive instructions of the form

$$X_{D\ j} \quad 0, \quad X_{D\ j} \quad X_{A\ j}, \quad X_{D\ j} \quad X_{A\ j} X_B, \quad X_{D\ j} \quad X_{A\ j}/X_B$$

where j 0(1)K' 1, and K'' gives a scaling factor.

(*d*) Two grouping instructions, one giving the group numbers corresponding to assigned grouping limits of some quantitative variate, the other enabling groups to be formed of non-consecutive values of a variate.

(*e*) Two test instructions such that, if X_A X_B or X_A X_B, there is a jump to a non-consecutive instruction, or entry to a specially programmed routine.

Provision is also made for additional instructions, e.g. instructions requiring subroutines, which can be programmed as required for particular surveys.

The instructions are all designed so that they can be packed in single words by the instruction input routine. Up to 63 instructions in all can be stored. These are obeyed consecutively, except as required by (*e*) above. Any variate, once formed, can be used in subsequent instructions, and overwriting of variates is permissible. If A and/or B is coded 0, the value given by the last preceding instruction will be taken for variate A and/or B, and if D is coded 0, the result of the instruction will not be entered in the variate store; this is primarily to save computing time.

As each variate is computed it is tested against assigned limits (unless none are assigned) as for input of the original variates. Any derived variate involving a variate whose value is coded as unknown is also coded unknown.

By a simple modification of the program it can be arranged that, with a single card-reading cycle (see below), all entries in tables are omitted for any unit in which an original or derived variate is coded as unknown, thus permitting the units concerned to be examined and included in the tabulations after correction. Alternatively, and this applies also when there are two card-reading cycles, it can be arranged that the original entries corresponding to units containing errors are later subtracted by a second passage of the original cards of these units.

Formation of Tables

The formation of tables is controlled by a set of *tabulation instructions*. Up to 63 instructions can be used.

The instructions are written in the form

$$V.C$$

where V is the number of the variate to be tabulated, and C is the number of the classification set (see below). If a count is required, V is coded 0 or 1: if V 1 the count is associated with the preceding table and will be omitted if the value of the tabulated variate in this table is unknown or impossible. When an unknown or impossible value of a variate is encountered in a tabulation without an associated count, the address of the cell to which it belongs will be printed out. In either case no entry is made in the tabulation.

If a raised total or count is required, with raising unit by unit, the instruction is written $V.C$. The entry in the table is then gX_V or g, it being arranged that g, the raising factor, is placed in X_{32}. X_{32} may be either an original or a derived variate.

A tabulation or count can be limited to a single value K of a further variate U (with the restriction that 0 K 7). This is effected by writing the instruction in the form

$$V.C.U.K$$

If an instruction is terminated by the entries called for by the instruction will be made in the table given by the last preceding instruction not terminated by a sign. Similarly, if an instruction is terminated by the entries will be made in the table given by the last but one preceding instruction not terminated by a sign. This enables data for individuals recorded in parallel fields on a card to be included in the same table or pair of tables (the last of the pair being an associated count).

A single tabulation instruction, or a pair of instructions for a tabulation and its associated count, may be used

to give the tabulation of a number of consecutively numbered variates, all classified by the same variates (up to three in number). The instruction or pair of instructions is written in the same manner as for a single table, the multiple character of the instruction being indicated by the coding of the associated classification set.

Classification Sets

Up to five-way tables may be constructed. The variates by which the tables are to be classified are defined by *classification sets*. These are written in the form

$$a,b,$$
$$c,d,e,$$

etc., where a, b, c, d, e represent the variate numbers for the classifications required. Up to 31 classification sets may be defined; the classification set 0 gives a total or raised total only.

If the value of one of the classification variates is unknown or impossible, the value of the tabulated variate or count will be entered in an "unknown" cell which is kept separate from the main tabulation.

Storage of Tables

The storage space required for each table, including all marginal totals and the "unknown" cell, is computed during the input of the instructions. The tables are stored consecutively, and the first address A_a of each table (that of the "unknown" cell) is entered in the tabulation instruction words, and is also printed. The number of storage locations available for tables is 1,680 less the number required for grouping limits and any special derived variate instructions.

The spaces for the marginal totals follow the corresponding cells. Thus the address of the cell X_a, X_b, X_c in a three-way table is

$$A_a + [1 + X_a + (A + 1)X_b + (A + 1)(B + 1)X_c]$$

where A, B and C are the ranges of X_a, X_b, X_c. The addresses of the marginal totals are obtained by putting some or all of X_a, X_b, X_c equal to A, B, C. It will be noted that, in consecutive tables having the same classification set, the expression in square brackets is unchanged, and therefore need not be recomputed.

There are three ways in which storage space can be further economized. These are (*a*) packing, (*b*) omission of space for marginal totals, (*c*) analysis by strata. All of these are planned as possible modifications of the present program. The method of packing envisaged is to use each location for two cells of a table. This can be very simply programmed by redefinition of the cell addresses. Provision for the packing of certain tables and not others may also be made. If space for marginal totals is omitted (which will always be done when tables are packed) these spaces will have to be inserted in the second part of the program. This can be done by over-

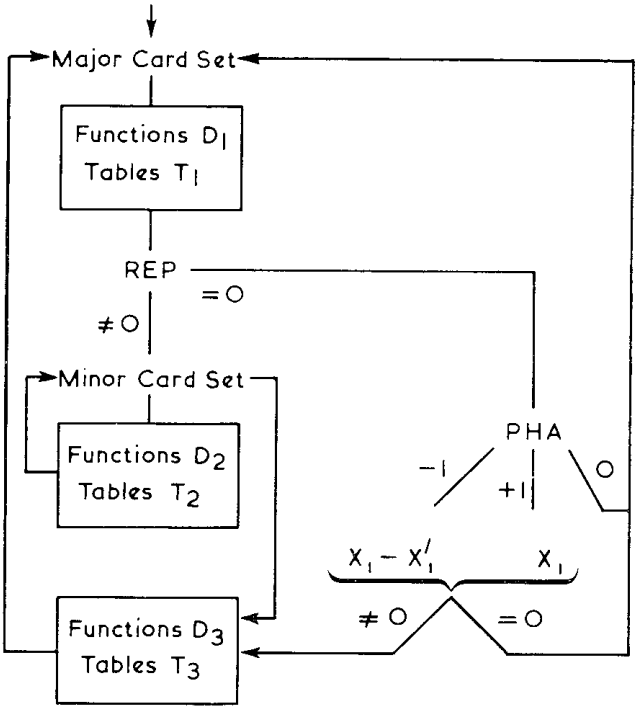

Fig. 1.—Organization of card-reading cycles

writing, working from the end. The tables will then have to be processed in two or more parts in the second part of the program, one or more parts being temporarily stored on tape.

Analysis by strata can be effected by sorting the data according to strata before tabulation. This is particularly useful when each stratum has to be raised by a different factor, and when classification by strata, at least for derived tables (see below), is not required. Tabulation (without raising) is made for the whole of a stratum; the totals are then raised and added into a duplicate set of tables. If unraised totals are also required, a third set of tables is carried.

Card-reading Cycles

To deal with surveys in which group information is recorded on cards different from those relating to the individuals in a group, or in which certain items of information are recorded for a sub-sample only of the units (two-phase sampling), and similar complexities, two card-reading cycles are provided. These are associated with three sets of derived variate and tabulation instructions, in the manner shown in Fig. 1.

The operations to be performed are determined by two control words, REP and PHA. If both are 0 there is only a single card-reading cycle and a single set of derived variate and tabulation instructions. If REP 0 and PHA 1 there is a single card-reading cycle, but after the first set of instructions has been performed the variate X_1 (which serves as the index of second-phase information) is tested; if not 0 a second set of instruc-

tions (D_3 and T_3) required to deal with this information is performed. If REP 0 and PHA 1 the process is similar, but X_1 is compared with X_j, the value of X_1 taken from the first of the following set of cards. This provides for recognition of change of sampling unit in two-stage sampling when there are no cards for the first stage. D_3 and T_3 can then be used for the computation of first-stage sampling errors.

If REP 0 there are two reading cycles. If REP n the machine will check that a number n of minor sets of cards is presented after each major set. If REP 1 an indefinite number (possibly zero) of minor sets will be read. Before the first minor card set, instructions D_1 and T_1 are performed; after each minor set, instructions D_2 and T_2; and after the last minor set, instructions D_3 and T_3. This covers a number of commonly occurring contingencies, e.g. group information with individuals on separate cards, two-phase sampling in which the second-phase information is on separate cards, sampling on successive occasions, trailers, etc.

In order to be able to test for change of card set and compare indices on the current and following card, the actual reading of the cards is always one card ahead of the associated operations, the information on the following card being stored temporarily in the original coded form, until it is required for processing.

Intermediate Recording of the Tabulations

Provision is made for punching the whole of the tabular results at any desired points, in a form suitable both for printing and for reading back into the machine. This provides a safeguard, in long tabulations, against breakdown of the machine, failure due to faulty cards, etc., and also serves temporarily as a primitive form of output for the first part of the program, pending the completion of the second part.

Processing of Tables

When the whole of the information has been read in and the basic tables constructed, further computations are required to provide the final tables which are to be printed. It is intended to make provision in the second part of the program for the following operations:

(*a*) Formation of marginal totals. This will be done at the outset for all tables without any special instructions.

(*b*) Formation of a new table by arithmetic operations on the corresponding cells (including margins) of two tables with the same classification variates. A simple example is the provision of a table of means from a table of totals and the corresponding table of counts: each cell of the table of totals requires to be divided by the corresponding cell of the table of counts.

(*c*) Similar operations to (*b*) when one of the tables is classified by fewer variates than the other. A simple example is the raising of a two-way table classified by strata and another variate: the raising factors for the different strata constitute a one-way table, and each cell of the two-way table must be multiplied by the corresponding cell of the one-way table; new marginal strata totals will be required for the raised table. In this case the new table has to be classified according to the variates of the larger table. Other cases arise in which the new table only requires to be classified by the variates of the smaller table (or even fewer variates).

(*d*) Division by designated margins to give percentages.

(*e*) Amalgamation of certain classes in a table.

(*f*) Omission of certain classes from a table. New marginal totals will be required in this case.

The required operations will be controlled by *table derivation instructions* similar to the derived variate instructions in the first part of the program. Their exact form has yet to be decided. Provision will be made for overwriting tables (in the same way as variates can be overwritten) in order to save storage space. Each table will be identified by a number which will correspond with or supplement the numbering of the tabulation instructions in the first part of the program.

Printing of Tables

With the address system adopted, printing out the store locations of a table in order, with a new line after each marginal mean of the first variate, gives an acceptable layout. In addition, an extra space may be introduced before these marginal means, and an extra carriage return and line feed before and after each row of marginal means of the second variate. The number of the table, the number of the variate tabulated (if relevant), and the numbers of the classification variates will be printed at the head of each table. No provision will be made in the general program for printing marginal headings, as this would require too much storage space, and unduly elaborate the program.

Provision will be made for the insertion of decimal points and for rescaling.

Calculation of Sampling Errors

No special provision has been made for the calculation of sampling errors, but such errors as are required can be computed by the use of suitable instructions. As a simple example we may take the computation of the variance per unit of a variate x in a random sample. This is given by

$$s^2 \quad \frac{S(x \quad \bar{x})^2}{n \quad 1} \quad \frac{Sx^2 \quad (Sx)^2/n}{n \quad 1}$$

where S denotes summation over the sample, and n is the number of units. x^2 can be formed as a derived variate, and Sx^2 by tabulating this (total only). Sx and n will be available as the totals of other tables, and s^2

can then be formed by a series of table derivation instructions of types (*b*) and (*c*).

It may be asked why computations of this type are not included in standard form as part of the general program. We have indeed incorporated the computation of sampling errors in a program which deals with stratified random samples, but to deal in a general manner with all the complexities that arise in the many, often complicated, types of sampling that are met with in practice would unduly complicate the program, and would make it too unwieldy for a machine the size of the 401. Moreover, we have found that, in practice, evaluation of sampling errors is not very frequently required, except when considering the appropriate size of sample in future surveys, and the efficiency of different sampling methods: in the latter case the sampling errors of different sampling methods have to be evaluated. In the interpretation of the results of a single survey an adequate assessment of the sampling errors can usually be made from internal evidence of the tabular material (its regularity, or lack of regularity, over secondary classifications).

An Autocode Form of the Instructions

Douglas and Mitchell (1960) have recently described a program for the analysis of surveys of the market research type, which was written for Pegasus. In it they have adopted an autocode form of language. A somewhat similar language has been adopted in the scheme outlined in *Sampling Methods for Censuses and Surveys*, and could be used in the 401 program: all that would be required would be a preliminary interpretive routine which would translate the autocode instructions into the coded instructions of the 401 program. Storage space is not a major problem, since the translation has to be performed before the actual analysis is begun. The writing of such interpretive routines is, however, a lengthy and involved task, and it was decided that the more direct instruction code adopted in our program would be adequate for our needs.

References

Douglas, A. S., and Mitchell, A. J. (1960). "AUTOSTAT: a Language for Statistical Data Processing," *The Computer Journal*, Vol. 3, p. 61.

Yates, F. (1960). *Sampling Methods for Censuses and Surveys* (3rd edn.), London: Griffin.

Market Surveys with a Small Computer

By R. L. Cook

This paper, which was presented at the Harrogate Conference of The British Computer Society on 5 July 1960, describes a general-purpose Market Survey program for use on an 803 computer which has been developed from a specification jointly prepared by Social Surveys (Gallup Poll) Limited and Elliott Brothers (London) Limited.

Introduction

The 803 is a medium-speed transistorized computer with a magnetic-core store of 4,096 words of 39 bits. Input is by paper tape or punched card. The Elliott card reader is used, and is arranged to read each column both as a 12-bit number, when reading cards punched in the manner described in this paper, and as a 6-bit character for use with data punched in a conventional punched-card code.

The requirements of the market survey program are as follows:

1. To read data from punched cards without requiring the user to conform to any particular punching convention.
2. To produce a set of double-entry frequency tables from the input data.
3. To produce results in a form directly available for reproduction.
4. To retain simplicity of programming and operation, but also to allow sufficient flexibility to deal with the widest variety of problems possible.
5. To provide a facility for checking the data cards for consistency.

Data Recording

The results of a market survey are assumed to be punched on cards. More than one card may be required to hold the results of one interview, and the word *Block* is used here to mean the one or more cards which contain the answers to one interview.

Questions posed in a survey are of two types: the question may require a single integer answer, or may request the interviewee to indicate which of one or more attributes on a selected list he possesses. An example of the first type is "how old are you," and of the second type "to which of the following age groups do you