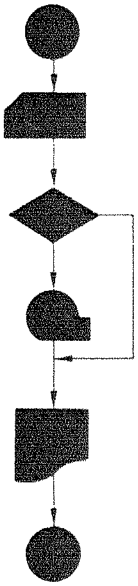
THE NORTHEASTERN FOREST-INVENTORY DATA-PROCESSING SYSTEM. V. DESCRIPTION OF SUBSYSTEM TABLE.



by
Robert W. Wilson Jr.
and Robert C. Peters

U. S. FOREST SERVICE RESEARCH PAPER NE-73
1967

NORTHEASTERN FOREST EXPERIMENT STATION, UPPER DARBY, PA.
FOREST SERVICE, U.S. DEPARTMENT OF AGRICULTURE
RICHARD D. LANE, DIRECTOR

About the Authors

ROBERT W. WILSON, JR. took his Bachelor's degree at The Pennsylvania State University in 1947 and his Master's and Ph.D. degrees at Yale University in 1948 and 1965, respectively. He joined the U. S. Forest Service in 1948 and has worked in various research capacities for the Northeastern Forest Experiment Station. From 1961 to 1965 he was in charge of the Station's biometrics unit at New Haven, Conn. He is assigned at present to the Forest Insect and Disease Laboratory at West Haven, Conn.

ROBERT C. PETERS obtained his Bachelor's degree from the University of California in 1960 and his Master's at Yale University in 1961. He joined the Forest Service in 1961 as a research forester, and was assigned to the Station's biometrics unit from 1961 until 1965, when the unit was discontinued. Mr. Peters played a key role in the development of the data-processing system reported here.

THE NORTHEASTERN FOREST-INVENTORY DATA-PROCESSING SYSTEM. V. DESCRIPTION OF SUBSYSTEM TABLE.

CONTENTS

PREFACE

THIS paper is the fifth in a series of ten papers prepared to describe the forest-inventory data-processing system of the Northeastern Forest Experiment Station. This system was devised for using modern, large-scale, high-speed computers in processing forest-inventory data. The series will comprise the following papers:

I. Introduction.
II. Description of subsystem EDIT.
III. Operation of subsystem EDIT.
IV. Information for programmers — subsystem EDIT.
V. Description of subsystem TABLE.
VI. Operation of subsystem TABLE.
VII. Information for programmers—subsystem TABLE.
VIII. Description of subsystem OUTPUT.
IX. Operation of subsystem OUTPUT.
X. Information for programmers —
subsystem OUTPUT.

V-A. INTRODUCTION

ONE of the major projects of the U.S. Forest Service is a nationwide forest survey, which is designed to obtain useful and timely information about the timber resources of the United States. In the course of the surveys, which are made mainly on a state-by-state basis, great masses of detailed data are collected about timber volumes, growth, timber cut, and other characteristics of the timber resource.

In recent years the volume of information obtained from forest-survey field plots has increased greatly. The task of compiling and analyzing this mass of data with mechanical computing machines was both cumbersome and time-consuming.

A solution to this problem was seen in the development of the high-speed electronic computers. The Northeastern Forest Experiment Station, which was responsible for conducting the forest survey of the heavily forested Northeastern States, investigated the possibilities and devised the Northeastern Forest-Inventory Data-Processing System.

This paper describes a part of the system, subsystem TABLE, that is designed specifically to reduce large amounts of sample data to tables of statistics for the samples. In turn, sample summary output, is designed for use as input to program OUTPUT (see part VIII of this series) to produce tables of statistics for the sampled populations.

The principal value of the program lies in its versatility. While applying a standard, straightforward procedure to the reduction of data, it provides a very great measure of freedom in fitting both inputs and outputs to the requirements of particular data-reduction problems. Within the context of the program, the origin of the input data is immaterial; a data set that is to be processed may be sample data, but it need not be. The data set is simply a collection of values for one or more attributes of one or more objects (sampling units) that are to be reduced to tables of statistics that characterize the set. Similarly, the output tables may summarize any aspect of the data set. Several different tables, each representing all or a selected part of the input data, may be used to form these summaries. And, in order that both the inputs and the outputs may be variable, the details of the procedures by which one is converted to the other are also variable. Consequently, the program can be applied to a wide variety of data-reduction problems.

The program is written in the standard FORTRAN IV language, and is operative at the Yale University Computer Center on an IBM 7094/7040 Direct Coupled System under the IBSYS operating system with IBJOB processor.[1] It will operate with little or no modification on other comparable systems.

Part VII in this series contains a selection of programming information that will be useful if the standard version of the program must be modified for any reason. Detailed instructions for setting up and executing jobs with the standard version are given in part VI. Copies of these publications and information on the FORTRAN IV source decks for the program can be obtained from the Northeastern Forest Experiment Station, 6816 Market Street, Upper Darby, Pennsylvania 19082.

V-B. PROGRAM OUTPUTS

The primary outputs from the program are sets of statistical tables. One set of output tables is produced for each set of sample input data. An output set may consist of up to 40 two-dimensional tables. No table in the set may have more than 50 rows and 50 columns, including the row and column totals that are formed automatically for every table.[2]

The content of the tables in an output set depends entirely upon the demands of the particular application. Any attribute of the sampling units in a data set (see part V-C) can be summarized by categories determined from other sampling-unit attributes (see part V-D).

[1] Mention of a particular product should not be construed as an endorsement by the Forest Service or the U. S. Department of Agriculture.

[2] There is also an overall restriction on the total number of cells in a set of output tables which is explained in part VII-C.

In addition, there is a choice of statistics to be provided for every cell of each table in the output set. The choices are:[3]

1. Simple sums over sampling units.
2. Means over sampling units.
3. Means and their variances over sampling units.
4. Means, their variances, and their covariances with the grand mean over the sampling units (for use when the data set summaries are to be used to make ratio estimates).

All output sets from a program run are written in a single magnetic tape file of binary records for rapid transmission to program OUTPUT,[4] in which the tables are labeled and printed as population statistics after appropriate weighting and summing. For the purpose of debugging the job control deck, the alternative of printing the tables in block form as BCD records is also available. No binary tape is written in this option.

During execution, the program prints a job summary consisting of messages of three types: those that identify errors in the job control deck that have halted execution, those identifying errors in input records (or the job control deck) that cause the record to be deleted but processing to continue, and those that signal successful reading of the job control deck and identify the data sets and numbers of sampling units that have been processed.

V-C. DATA INPUTS

The data input to the program consists of a single magnetic tape file of ordered unit records. Each record in the file must have exactly the same format as every other record. The file contains the data from sets of observed sampling units. Just how the sampling units are represented by the unit records depends upon the characteristics of the particular problem. The key is the way in which the field observations were made.

If all attributes were actually observed on the sampling unit as a whole (a plot, for example), the sampling unit observation

[3] These options, and the computations carried out for each, are explained in detail in part VII-F.

[4] The format of the output sets is described in detail in part VII-E.

would consist of a single value for each attribute. The sampling unit observation could then be represented by a single unit record.

If, on the other hand, all attributes were observed on subdivisions of the sampling unit (on trees in the plot, for example), then there would be several values for each attribute. In this case, one unit record would be required to represent each observed value of the set of attributes, so the sampling unit as a whole would be represented by a set of unit records equal in number to the number of subdivisions on which observations were made.

Generally, attributes of both kinds are observed on sampling units. In this case, there must still be a unit record to represent each subdivision observed, but the unit record format must also provide for entry of the single-valued attributes observed on the sampling unit as a whole. These values are repeated in every unit record of the set representing the sampling unit.

An additional possibility arises if attributes are actually observed on the sampling unit as a whole, on subunits of the sampling unit, and on subdivisions of the subunits. This is simply an extension of the previous case. The unit-record format must provide for recording the values of all three kinds of attributes. There will be one unit record representing each observed subdivision of the subunits. The values of attributes observed on the subunits will be repeated in each record of the subset representing the subunit; and, as before, the values of attributes observed on the sampling unit as a whole will be repeated in each record of the whole set representing the sampling unit.

In summary then, the individual unit record always represents the least subdivision of a sampling unit that has been directly observed, whatever that may be. The standard version of the program allows up to 160 unit-records in the sets representing sampling unit sets. The unit records must have a common format, and appropriate values for each of up to three kinds of attributes actually observed in a sampling unit must be recorded in every unit record. There is no provision for header cards of any kind.

The order of the unit records in the input file also depends upon the characteristics of the problem. The unit records must always be ordered by subunits, if any, within sampling units. In turn, the sampling unit sets must always be ordered into data sets. The significance of the data set is that it contains all the data required to make one set of output tables. What the data set represents in terms of the population that has been sampled depends upon the methods of sampling and of compilation that are employed. In stratified sampling, it will represent a sampling stratum. In other types of sampling, it will generally represent any kind of geographical or other unit for which output tables of statistics are required. Any number of data sets may be contained in the input file.

If required by the problem, the data sets themselves may be ordered into groups; and these groups, in turn, into still larger groups. For example, if stratified sampling has been employed in each of several survey units covering the population sampled, then the data sets must be ordered by survey unit. However, it must be remembered that all data sets processed in a single production run are subject to the same set of processing rules.

V-D. PROGRAM LOGIC AND PROCEDURES

The program consists of seven principal phases or steps connected in a simple and straightforward manner (fig. 1). The first step simply reads and stores in the computer all of the control information contained in the job control deck (part VI-C).

The second step reads a set of unit records representing a sampling unit and stores all of the data in blocks, according to the kind of attribute represented in each data field (see part V-C and part VI-C, sec. 330). Consequently, all input data for a sampling unit is available throughout the processing of that sampling unit.

The third step executes the CALCUL subroutine (part VII-B). This step is provided expressly to permit the calculation of data field values from information in more than one unit record of the sampling unit set, since this kind of operation cannot be performed in a unit-record editing process.

The fourth step produces the facsimile output tables at the sampling-unit level. This key step in the compilation process will be discussed in some detail below.

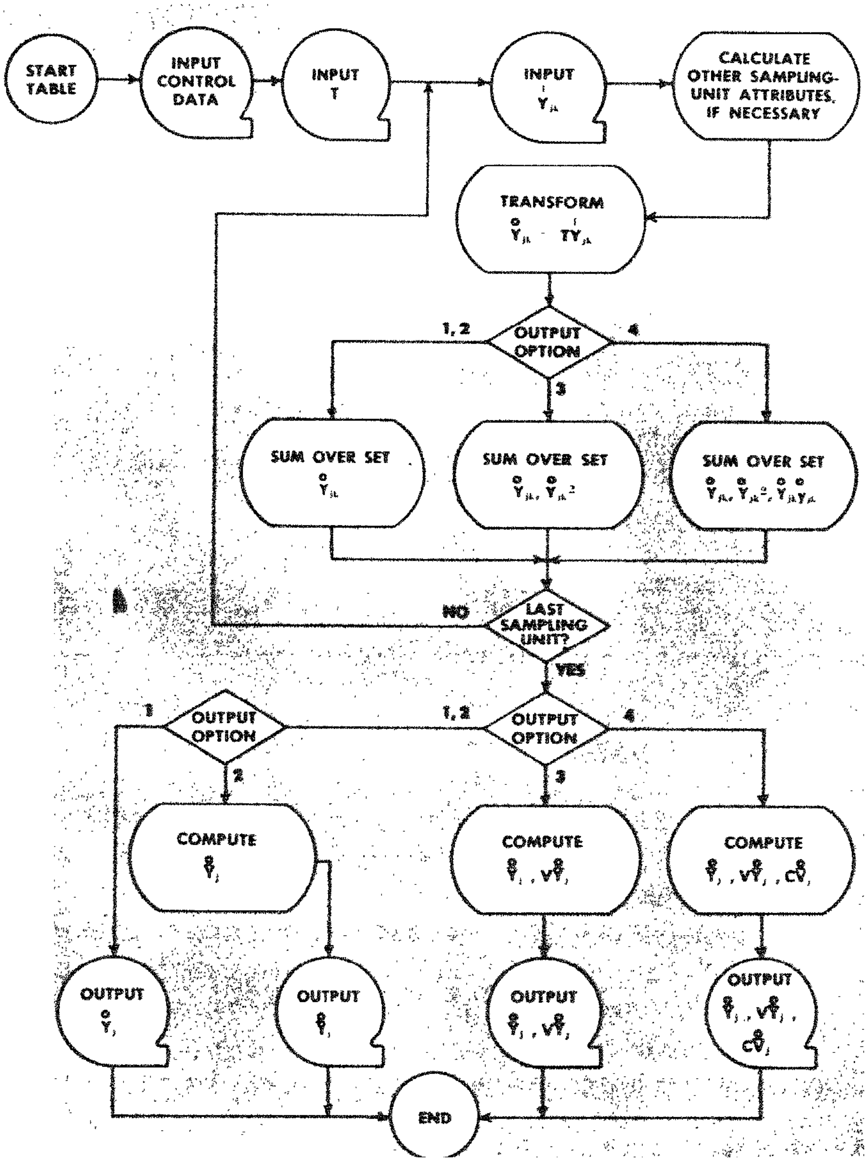

START TABLE
INPUT CONTROL DATA
INPUT T
INPUT Y'_{jk}
CALCULATE OTHER SAMPLING-UNIT ATTRIBUTES, IF NECESSARY
TRANSFORM $\overset{\circ}{Y}_{jk} = TY'_{jk}$
OUTPUT OPTION
1, 2
4
3
SUM OVER SET $\overset{\circ}{Y}_{jk}$
SUM OVER SET $\overset{\circ}{Y}_{jk}, \overset{\circ}{Y}_{jk}^2$
SUM OVER SET $\overset{\circ}{Y}_{jk}, \overset{\circ}{Y}_{jk}^2, \overset{\circ}{Y}_{jk}\overset{\circ}{y}_{jk}$
NO
LAST SAMPLING UNIT?
YES
1
OUTPUT OPTION
2
1, 2
OUTPUT OPTION
3
4
COMPUTE $\hat{\overset{\circ}{Y}}_j$
COMPUTE $\hat{\overset{\circ}{Y}}_j, v\hat{\overset{\circ}{Y}}_j$
COMPUTE $\hat{\overset{\circ}{Y}}_j, v\hat{\overset{\circ}{Y}}_j, c\hat{\overset{\circ}{V}}_j$
OUTPUT $\overset{\circ}{Y}_j$
OUTPUT $\hat{\overset{\circ}{Y}}_j$
OUTPUT $\hat{\overset{\circ}{Y}}_j, v\hat{\overset{\circ}{Y}}_j$
OUTPUT $\hat{\overset{\circ}{Y}}_j, v\hat{\overset{\circ}{Y}}_j, c\hat{\overset{\circ}{V}}_j$
END

Figure 1. — Generalized flow chart of TABLE.

The fifth step adds the completed sampling-unit tables to the output tables being accumulated for the data set; and if required, also adds tables of squares and of cross-products (of cells with totals) to special data set tables used to compute variances and covariances.

Steps two to five are repeated until every sampling unit in the data set has been processed. Then, the sixth step computes the required statistics for each output table from the sums that have been accumulated. The tabulated statistics are written on magnetic tape in binary mode (or printed, if the debugging output option has been taken) in the seventh and final step.

If there are additional data sets to be processed, the program then returns to step two and the cycle is repeated for each successive set.

The formation of final output tables begins with the formation of facsimile output tables for each input sampling unit, in sequence. The formation of these tables is governed entirely by a general table-making procedure provided in the program; in conjunction with information about the relationships between sampling unit data and output tables in a given application that is conveyed to the program in the job control deck.[5]

The general procedure (fig. 2) provides that each observed value of a given attribute[6] is summed (entered) into a given fac-

[5] The process of forming the facsimile output tables for the sampling units is similar in concept to a matrix transformation:

$$\overset{O}{Y} = T\,\overset{I}{Y}$$

where

$\overset{O}{Y}$ = a given two-dimension facsimile output table in which an input attribute is tabulated according to the values of two other attributes;

T = the transform that controls the process, consisting of the general table-making procedure and the control information for a given job; and

$\overset{I}{Y}$ = a two-dimension array of sampling unit input in which the data are tabulated according to observation (rows) and attribute (columns).

[6] The number of entries of a given attribute per sampling unit depends upon the kind of attribute and the number of subdivisions of the sampling unit on which it was observed (see part V C). If the attribute was observed on the sampling unit as a whole, it has only one value, so only one entry is made in the appropriate facsimile tables. If the attribute was observed on a subdivision of a sampling unit, there will be as many values (and as many entries in the table) as there were subdivisions of the sampling unit. The general procedure automatically enters each observed value of an attribute, using information about the attribute classification furnished with the input record description in the control deck (part VI, sec. 330).

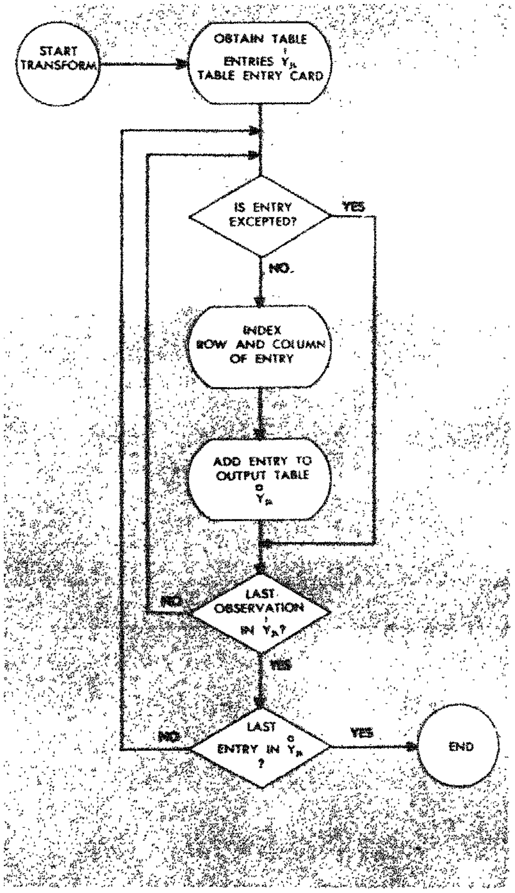

START TRANSFORM
OBTAIN TABLE ENTRIES Y_{jk} TABLE ENTRY CARD
IS ENTRY EXCEPTED?
YES
NO.
INDEX ROW AND COLUMN OF ENTRY
ADD ENTRY TO OUTPUT TABLE $\overset{o}{Y}_{jk}$
LAST OBSERVATION IN Y_{jk}?
NO
YES
LAST ENTRY IN $\overset{o}{Y}_{jk}$?
NO
YES
END

Figure 2. — Flow chart of the transform operations in TABLE.

simile table in a particular row and column (location), unless conditions are specified under which certain values of the attribute are not to be entered. The procedure also provides that up to 4 attributes may be entered in a given table. The general procedure can form almost any kind of tabulation of sampling unit data.

The information provided in the control deck (part VI-C, sec. 320) defines the particular set of output tables required. For each output table, the following information is given:

1. A short, unique name by which the output table can be identified.
2. The dimensions (number of rows and number of columns) of the table.

3. The attributes in the sampling unit data that are to be summed into the table. The values of these attributes must be expressed as floating-point numbers.

4. The attributes in the sampling unit data that determine the row and the column (location) in the output table into which each value of the entry attribute is to be summed. The values of these attributes must be expressed as fixed-point numbers.

5. The operations (and input tables, if any) by which values of the location attributes are converted to row and column indexes; chosen from among the five operations that are available in the program:

   (a) LIST, the operation that defines an index as the position in a list occupied by a given value of a location attribute. An input table contains the list of all values the given attribute is permitted to assume. For example, assume the following relationships between the values of a location attribute and the rows of an output table into which an entry is to be made: when the value is 1, the entry is in the second row; when the value is 2, the entry is in the third row; when the value is 4, the entry is in the fourth row; and, when the value is 6, the entry is in the first row. Because there are four possible values of the location attribute in one-to-one correspondence with a row index, there must be four entries in the input table: 0006, 0001, 0002, and 0004.

   (b) RANGE, the operation that defines an index position in a list occupied by a range of values that contains a given value of a location attribute. An input table contains the list of ranges that cover all possible values for the given location attribute.

   For example, assume the following relationships between the values of a location attribute and the columns of an output table into which an entry is to be made: when the value of the location attribute is less than 50, the entry is to be made in column 1; when the value lies between 50 and 73, inclusive, the entry is to be made in column 2;

and, when the value lies between 74 and 99, inclusive, the entry is in the third row. There must be three entries in the input table:

00000049, 00500073 and 00740099

(c) LOOKUP, the operation that defines an index as the value in one list of values whose position corresponds to the position in another list occupied by a given value of a location attribute. An input table provides both lists. For example, assume the following relationships between the values of a given location attribute and the rows of an output table in which an entry is to be made: when the value is 1 or 7, the entry is in the first row; when the value is 2, 3, or 4, the entry is in the second row; and, when the value is 5 or 6, the entry is in the third row. Since seven values of the location attribute are permitted, there must be seven entries in the input table:

00010001, 00020002, 00030002, 00040002,
00050003, 00060003, and 00070001.

(d) EQUATE, the operation that defines an index as equal to a given value of a location attribute. No input table is required.

(e) CONST, the operation that defines an index as equal to a given constant. No input table is required but the constant must be specified along with the operation name.

6. The conditions under which a value of an entry attribute is not to be summed into a given facsimile output table. These conditions are referred to as exceptions. Each condition requires the specification of an attribute, a constant, and a relational operator. The exception (no entry) occurs whenever a value of the attribute bears the specified relation to the constant. Any number of conditions may be applied to any entry attribute providing that the total number specified in the control deck does not exceed 150.

I-E. CONCLUSION

The foregoing chapters have described how program TABLE is designed to carry out a general data-reduction process and to incorporate a great many variations in detail automatically. This design makes the program applicable to many different data-reduction problems; but, its very flexibility means also that it cannot provide a fully automatic solution to any problem.

The user always has the responsibility for preparation of the job control deck in which the particulars of a given problem are specified. The description of the deck in part VI can be used as a check list in assembling the minimal information required; but successful application of the program demands, in addition, a thorough knowledge of the problem including the end-use of the results and the origins of the data.

It will be found that the preparation and checking of the job control deck is not easy. The deck contains a great deal of detailed information about the problem, not all of which can be checked by the program prior to test runs. Consequently, while the program offers an efficient means to solve a variety of problems involving large amounts of data or extensive tabulations of data, some other means will generally be better for simpler problems.

■