SESSION XI SYMPOSIUM: BUILDING INSTRUCTIONAL DATA-GATHERING MODELS WITHOUT A PROGRAMMER

DANA B. MAIN, *University of Michigan, Presider*

MODEL R—Model building and model modification for instruction

GERALD H. SHURE and KIRK BRAINERD
University of California, Los Angeles, California 90024

An integrated set of three computer programs that enable the implementation and student investigation of computer models of behavioral phenomena is described. These models are intended as tools for aiding in the instruction of undergraduate social science courses in research methods. Each program provides for natural language interaction with one of three classes of users: A "Modeler," who implements a model of some behavioral phenomenon; an "Instructor," who tailors his own or others' models for his course of instruction; and a "Student Experimenter," who applies an experimental design to the model and receives synthetic data in return. None of these users need have any prior computer expertise nor do they depend on external documentation on how to use the programs.

BACKGROUND

This paper describes an integrated set of three computer programs that aids in the design, reconfiguration, and student investigation of computer models. These models represent phenomena typically studied in undergraduate social science courses in research methods. The range of models that may be built need not be limited to these disciplines. The models are intended as tools for aiding in the instruction of such courses and address some of the problems frequently encountered.

These courses usually involve a laboratory experience activity for the students that has three phases: first, the design of an experiment; then the running of the experiment; and finally, the analysis and interpretation of the resulting data.

It has traditionally been a weakness of such courses that the second phase—the actual running of experiments—has taken most of the time allotted for laboratory experience. The activity in this phase involves getting equipment to work, arranging schedules with volunteer student subjects, etc. Much of this work may

have little relevance to the interrelated issues of hypothesis testing and the experimental design process other than producing data that contains information useful to evaluation of the designs. (We, of course, are not referring to those courses in which the development or evaluation of the procedural tools are themselves the focus of interest.)

As a consequence of these demands of instrumentation, these courses frequently fall short of their intended purposes. More broadly conceived objectives, such as the study and use of experimental designs to carry out a series of coordinated experiments in which the results of earlier studies are used to plan the design of subsequent ones, are abandoned as unrealistic. More typically, the student finishes in a mad scramble, trying to collect the necessary data for a single study conducted under limitations imposed by time restraints and scarcity of subjects and laboratory space.

One solution to these problems has been to use computer models to simulate quantitative data as these might be generated in one or more actual experiments. Such models have been used to allow students to explore possible determinants of schizophrenia, imprinting, motivational factors in routine task performance, and drugs (Main & Head, 1971), as well as the variables influencing verbal reinforcement (Johnson, Note 1). While such models have usually been simple ones, considerably more complex and empirically valid models

An earlier version of this paper was presented at the National Conference on the Use of On-Line Computers in Psychology, Boston, Massachusetts, November 20, 1974. Research reported herein was conducted under Contract F30602-74-C-0016 with the Advanced Research Projects Agency, Department of Defense.

can be built for instructional purposes. One such model by Shure, Malamuth, and Johnston (1974) allows students to explore the role of 19 variables that may influence a shift in the degree of risk taking when decisions are made by groups rather than by individuals and demonstrates that such models need not be trivial or unrealistic.

A general feature of almost all of these models is the focus on students who use them to conduct experiments within a particular problem area. For each experiment, the student selects a few variables from within a larger set in order to test some hypotheses using a particular research design. The selected variables and design are input to a computer program that then outputs values of the dependent variable(s). The student then attempts to learn about the problem area by pursuing hypotheses through a series of experiments, each of which deals with only a small subset of the available manipulable variables and yields data appropriate to these manipulations. These data are then analyzed and interpreted by the students. To the extent that the model is a reasonable approximation of what is known about the behavior of the variables in the problem area, the student has an opportunity to "discover" relationships and test hypotheses with his data and to learn to conduct his research through efficient experimental designs. Because the actual procedures of experimentation are avoided, the students spend more time on research design issues and in interpreting and integrating the results obtained. The procedure, then, is formally similar to conducting experiments in the real world although it is not a substitute for the laboratory experiment. For the student, the use of these models has advantages noted above including that of using the feedback of one study in planning or modifying a series of experiments; for the instructor, the use of these models permits him to give students more precise feedback because the nature of the simulated world is known to him.

COMPUTER SYSTEMS TO ASSIST IN MODEL CONSTRUCTION

Some 3 years ago, Stout (Note 2) and Main and Head (1971) reported on the development of the EXPERSIM system with a supervisor to facilitate the development and use of such models in instruction. Their system was designed to allow instructors with a limited knowledge of FORTRAN programming to create models of modest complexity and to allow students without any programming knowledge and with a minimum of procedural details to specify experiments at a computer terminal or through batch processing procedures and then receive a hard copy output of the simulated data appropriately labeled for each condition.

For the model designer, the Michigan system assumed the routine burdens of input, error checking, and output and provided a number of subroutines for producing a

background of random variation such that observations are treated as samples from appropriately defined distributions and for performing a number of other necessary common programming tasks. The system reduced the programming tasks by as much as 60%. In spite of such support and the widely acclaimed value of the method, scarcely more than a half-dozen models have been developed in the few years in which the supervisor has been available, and these models are generally simple in structure and involve only a handful of independent variables. While a number of factors may account for this state of affairs, the requirement for some programming skill appeared to be a major consideration in inhibiting the development of new models. Furthermore, the limited use of the early models by instructors at locations other than those of the model designer was undoubtedly influenced by the fact that these models were not readily modified to suit the purposes of other course formats or substantive emphases.

MODEL-R—A DATA GENERATOR SYSTEM

It is against this background of both perceived usefulness and barriers to use that the current data generator system was designed and implemented. EXPERSIM I freed the student wanting to generate data from computerized models from having to be, or deal directly with, a programmer. The current effort goes further by similarly freeing the model designer and the instructor wishing to adapt another's model to his own purposes and teaching style.

The data generation system consists of three related parts. Each respectively provides for interaction with one of three classes of users: A "Modeler," who implements a model of some behavioral phenomenon; an "Instructor," who tailors his own or others' models for his course of instruction; and a "Student Experimenter," who applies an experimental design to the model and receives synthetic data in return. These three programs, called MODEL-R, INSTRC, and EXPMTR, respectively, achieve natural language interaction between the processor and all three classes of users and minimize their dependence on external documentation. The three basic stages in the use of the data generator package are shown in Figure 1.

MODEL-R Dialog

At Stage A, the model designer starts with the MODEL-R program and a clear slate and produces a model of behavioral phenomenon—relationships among all relevant variables to the phenomena of interest.

At Stage A in Figure 1, we see schematically how the human-to-computer dialog produces the basic model.

Each subsection of the dialog allows specification of some aspect of the model being defined. The items to be specified, in the order that the dialog requests them from the user, are:

Figure 1.

Naming of elements of the model. Here the program elicits descriptive titles for the independent variables and their levels, and for the dependent variables. The current data generator system allows up to 20 independent variables, each having up to 20 discrete levels (unique values) and up to three dependent variables. (These are currently presented as integer scores.)

Delineation of relationships among variables. For each dependent variable of a model, the user specifies a base mean and standard deviation. These values are assumed to represent the dependent variable score for a population where no information about the independent variables is given. For each level of each independent variable, a value is specified for each dependent variable to be added algebraically to its base mean to represent the effect of that level of that independent variable. For example, if IQ were the dependent variable with a base mean of 100, and social class was an independent variable with three values—upper, middle, and lower class—then, for each social class, depending on its association with IQ, some points would be added or subtracted from the base mean to represent the effect of social class on IQ. Similarly, a value is specified that will represent the effect of an independent variable level on the base standard deviation of a dependent variable, except that these values will be multipliers of, rather than be added to, the base standard deviation.

In addition to the main effects of independent variables upon dependent variables mentioned above, models may contain specifications for first-order interaction effects on means and standard deviations of dependent variables.

In all cases, for a given cell of a student's experimental design, the mean and standard deviation effects of the independent variable levels and interactions specified by the student will each be summed and then applied additively or multiplicatively to the respective base parameters, for each dependent variable, to determine the parameters of the distribution for that cell. Correlations between pairs of dependent variables are also specified at this time. This last feature allows for the representation of effects associated with intrasubject designs or multiple assessments.

Next, *logically incompatible or nonexistent combinations* of levels for pairs of independent variables are indicated to keep the model from generating data on these. For instance, in an experiment using sex (male, female) and state of pregnancy (yes, no) as independent variables, the generally nonexistent category, "pregnant males," may be eliminated by the model designer.

Delineation of the costs. Finally, the model designer may designate any levels of independent variables to assign costs in units, such as subjects or dollars,

associated with running these conditions. These will then be reported to the student at Stage C as costs per experimental replication.

INSTRUCTOR Dialog

At Stage B, the instructor user starts with the EXPMTR program and the basic model from Stage A and produces a model configured for a particular pedagogical purpose.

While the basic model might contain all of the variables thought to be relevant to a particular behavioral phenomenon, an instructor, at a particular point in a course, may wish to defer student consideration of certain experimental manipulations or levels until later in the course because of their complexity or for other reasons. Independent variables and certain levels of IVs may be hidden to make them inaccessible to the student-experimenter. Furthermore, an instructor might wish to make sure that certain variables are controlled in the same way in all experiments in a particular set. For example, in a series of studies, he may require that all subjects be male or all use one of a set of procedures. It is important that these options be available independently from the specification of relationships among variables in the basic model at Step "A," because it is anticipated that different configurations will be made of each basic model to suit the varied pedagogical purposes of different instructors or of the same instructor at different stages in the class. Finally, default levels are specified to cover the case in which the student's experimental design fails to assign values to all of the variables in the model.

The instructor dialog allows direct specification of all these aspects of the model chosen for configuration. It may exclude some independent variables or levels as options for student consideration; it may fix the levels of "hidden" variables or set default values for some in case the student fails to specify them. The output of this is given a name by the instructor who configured the model.

STUDENT-EXPERIMENTER Dialog

At Stage C, the student-experimenter uses the EXPMTR program and a configured model from Stage B to produce an experimental design and resulting data. The human-to-computer dialog to specify an experiment is implemented as the last part of the EXPMTR program but will usually be run under another name chosen by the instructor who configured the model at Stage B.

The sequence of events for a student-experimenter interacting with this dialog covers the following: (1) A list of all variables, made available for the student-experimenter, is displayed for his choice. (2) Following each variable he chooses to manipulate (to assign different values), a list of levels available for that variable is displayed, allowing the choices of levels to be made. (3) For each variable the student-experimenter

wishes to control (hold constant for the entire experiment), a list of levels available is similarly displayed, allowing one choice to be made for each variable. (4) The student-experimenter is then advised of the cost per replication of the experiment design just specified; then he indicates the number of replications of the design he wants to run. (5) An option is presented allowing immediate hard copy to be requested. (6) At this point, another student may identify himself and a new experiment may be initiated or the process may be terminated.

Looking at the relationships among the components of the model, we see that the first two stages may be used independently as many times as desired. This allows many student experiments to be run from a single model configuration and also allows a single basic model to be configured differently for differing pedagogical purposes without having to go back and respecify the basic relationships among variables. At all three stages, use of the programs does not depend on external documentation.

Computations for Data Generation

After a student-experimenter has specified an experimental design to be run, the data generator synthesizes data according to the following procedure: (1) For each cell of the experiment design and for each dependent variable, the program computes the dependent variable population parameters (mean and standard deviation) by summing the main and first-order interaction effects on the mean of the dependent variable over all independent variables controlled in the experiment design and adding this sum to the base mean of the dependent variable. Effects on the standard deviation of each dependent variable are similarly combined and applied to the base standard deviation but in a multiplicative rather than an additive fashion. (2) Next, for the first dependent variable, the model generates the required number of sample scores (up to 100) for each cell of the experimental design. Given the cell mean (M_x) and standard deviation (SD_x) from the preceding step and a randomly generated standard score from a normal distribution (e_{x_i}), the program generates the cell scores (X_i) for each replication (i):

$$X_i = M_x + SD_x \cdot e_{x_i}. \quad (1)$$

The scores for a second dependent variable (Y_i) are similarly generated from the given group mean (M_y) and standard deviation (SD_y) values, but, in addition, the formula introduces any given correlation term (r_{xy}) between the first and second dependent variables scores (Y_i) for each replication (i):

$$Y_i = M_y + SD_y(r_{xy}e_{x_i} + \sqrt{1 - r_{xy}^2} \cdot e_{y_i}). \quad (2)$$

As required, an extension of these same procedures is used for generating scores for a third dependent variable.

```

EXPERIMENT SPECIFICATIONS: SEX AND AGGRESSION
CONDITIONS THAT HOLD FOR ALL CELLS:
SEX OF SUBJECT
    MALE
CONDITIONS THAT ARE MANIPULATED IN THE EXPERIMENT:
EXPOSURE OF SUBJECT TO READING MATERIAL
    EROTIC READING MATERIAL
    MYSTERY READING MATERIAL
    SCIENCE FICTION READING MATERIAL
SEX OF EXPERIMENTER
    MALE
    FEMALE
INSTRUCTIONAL LABELING OF BEHAVIOR ASKED OF SUBJECT
    SUBJECT ASKED TO ADMINISTER "PUNISHMENT"
    SUBJECT ASKED TO ADMINISTER "FEEDBACK"
    SUBJECT ASKED TO ADMINISTER "STIMULATION"
    SUBJECT ASKED TO ADMINISTER "SEVERE PUNISHMENT"

CONDITIONS FOR CELL 1:
EXPOSURE OF SUBJECT TO READING MATERIAL
    EROTIC READING MATERIAL
SEX OF EXPERIMENTER
    MALE
INSTRUCTIONAL LABELING OF BEHAVIOR ASKED OF SUBJECT
    SUBJECT ASKED TO ADMINISTER "PUNISHMENT"

DEPENDENT VARIABLES:
(1) SELF REPORT OF SEXUAL AROUSAL
(2) BLOOD PRESSURE MEASURED FOLLOWING EXPOSURE
(3) AGGRESSION MEASURED BY SUBJECT'S ADMINISTRATION OF SHOCK

DEPENDENT VARIABLE DATA:

```

REPLICATION	DV: 1	SCORE	DV: 2	SCORE	DV: 3	SCORE
1		5		74		3
2		7		75		4
3		4		77		5
4		4		81		4
5		7		87		4
6		7		91		6
7		6		88		5
8		6		83		5
9		8		87		4
10		10		87		5
MEANS		6.40		83.00		4.50
STANDARD DEVIATIONS		1.838		5.981		.085

Figure 2.

(3) Finally, the model computes summary statistics for the score in each cell and costs for running the entire experimental design. (4) The results are then formatted for output. Figure 2 illustrates a sample formatted output for one cell of an experimental design.

Implementation of the MODEL-R System

The data generator system is comprised of two basic parts. The first carries on conversation with its user to gather specifications from which the second generates data. The former conversationally elicits specifications from a designer or instructor for a model, then from a student for an experiment design, and finally transfers control to the latter to carry out required computations and produce data for the student.

The conversational portion of the system is conducted by a package called GENDIA (for GENeral DIALOG processor) (Brainerd, 1973). Design of the GENDIA package is a direct outgrowth of the Laboratory Implementation System (LIS) (Meeker, Shure, and Cooperband, 1971) developed for the UCLA Center for Computer-Based Behavioral Studies (CCBS), and carries this development forward with increased generality and flexibility. Some of GENDIA's features are of interest here.

GENDIA and LIS are both table driven. That is to say that they both interpretively execute dialog specifications that are written and stored in tabular form that is human readable as well as machine readable. The advantage of this approach is that all of the functions that all human-to-computer dialogs have in common (such as the ability to present a display to the user's terminal, to receive input from the terminal, or to decide what information to display next) are written into the program once and for all; and all functions that change from one dialog to another (such as the actual contents of displays, the actual sequencing of displays depending on the user's inputs, and just what inputs from the user will be recognized as meaningful) can easily be specified each time a new or modified dialog is needed. This separation of functions yields tremendous savings in development time for interactive applications.

An additional feature of GENDIA is a powerful and flexible table-editing capability that can be called upon by the specifications for any dialog. This allows the model-building dialog to produce tables as its output which describe the model that the instructor has specified. The fact that the model description is in tabular form facilitates its inclusion in displays presented to the student who designs an experiment to run on the model. The table-editing capability then allows the student's experiment-designing dialog to produce experimental specifications in table form, which the computation module reads in order to generate data.

Both GENDIA and the computation module are written in FORTRAN for exportability and ease in updating. They operate in the CCBS PDP-10 environment.

REFERENCE NOTES

1. Johnson, R. R. The use of simulation games to teach decision processes and strategy in scientific research. Unpublished report to the ESSO Education Foundation, New York, 1971.
2. Stout, R. L. Specifications for a simulation program to control computer simulations used for behavioral science instruction. Mimeographed material, Department of Psychology, University of Michigan, 1971.

REFERENCES

- Brainerd, K. Introduction to the general dialogue processor (GENDIA). CCBS Document CCBS-P-31, University of California, Los Angeles, 1973.
- Main, D. B., & Head, S. Computer simulation in the elementary psychology laboratory. *Proceedings of the Conference on Computers in the Undergraduate Curricula*, 1971, 424-428.
- Meeker, R. J., Shure, G. H., & Cooperband, A. S. An implementation system for designing computer-based experiments. *Proceedings of the 1971 Annual Conference of the Association for Computing Machinery*, 1971, 169-178.
- Shure, G. H., Malamuth, N. M., & Johnston, S. A. A computer simulation of group risky shift for teaching undergraduate research methods. Paper presented at the American Psychological Association, New Orleans, Louisiana, 1974. Also CCBS Document CCBS-P-16, University of California, Los Angeles, 1974.