From computer to student: flexibility in program design  JOHN C. HURD

Computer-assisted instruction needs no apology. With the price of teachers going up and the price of computing going down it clearly will play an increasing role in education in the days to come. But CAI is not just a matter of hardware and correct course material. Let me raise with you some psychological issues which may ultimately determine whether or not CAI blossoms smoothly and naturally or only painfully and at length.

Range of Students

The first point is the incredible range of the students who confront us. The admissions process has selected them according to their academic ability, but their technological experience ranges from everything to nothing. In a given class we may have those taking a degree in computer science, who tend to be impatient with explanations, documentation, and features designed for naive users. These students may, for example, attempt to breach the security of the system in order to play Star Trek. To hold their interest the system should appear terse and efficient.

At the other end of the scale are those students who have an antipathy toward typewriters, much less any experience with a machine that talks back to them. These students are apprehensive about the whole idea of CAI. They are sure that no good can come of it, at least as far as they themselves are concerned. The inexperienced user needs all possible help. In addition to a written manual he needs on-line documentation which is displayed automatically as he begins new lessons, or tries new features of the system, or fumbles in his attempts to use features he has used before.

Perversely, however, the naive student is a temporary phenomenon. The system which provides him with all appropriate help is the system with which he is impatient two weeks later. A CAI system should therefore automatically throttle back on the amount of unsolicited help it offers as a student’s experience grows.
The Microcomputer as Geography Tutor

JOHN MIRON

In May 1980, the Advisory Committee on Educational Development agreed to fund my project entitled 'Interactive Tutorial Programs in Geographic Methods for a Microcomputer'. The purpose of this project has been to develop a set of interactive computer programs as a self teaching aid for students in GGRB02Y (Introductory Quantitative Methods in Geography) at Scarborough College. The ED grant was used to hire a student assistant to create these programs and to cover incidental expenditures. In total, 25 computer programs were successfully developed. These programs are currently being used by students in GGRB02Y.

In this article, a number of topics are discussed. First, the objectives of, and rationale for, the project are discussed. Secondly, the nature and extent of the work completed is reviewed. Finally, some initial impressions of the usefulness of these computer programs in practice are described. The remainder of this paper is devoted to these three topics in sequence. In an appendix, the computer programs are described in more detail.

Objectives and Rationale

There has been a substantial proliferation of 'canned' computer programs in the last decade or two; programs developed for users which do not require of them any special programming knowledge. Many of these programs operationalize methods in statistics or quantitative analysis such as are discussed in GGRB02Y. A first question of any critic of the present project must therefore be 'Why develop still another set of canned computer programs?'

I respond to this kind of question as follows. It is my belief that students in any introductory quantitative methods course need a great deal of assistance. For many programs such as Geography, this kind of course is mandatory. Thus, many students whose primary interest is in Geography, Sociology, Psychology or some other substantive area find themselves 'faced' into a course which, although helpful in the long-run, is perhaps confusing and likely foreign to their perceived interests. In the past, I have used teaching assistants and a substantial amount of my own out-of-classroom time to nurture students in GGRB02Y. However, I increasingly have felt that still more must be done to help students along.

An opportunity to do more arose when Geography at Scarborough College acquired two small (one an 8,000 byte and the other a 16,000 byte memory) Commodore microcomputers which are now housed in our Quantitative Lab. These machines are easy to operate, durable, quiet, and inexpensive stand-alone computers. They can be programmed in BASIC, and are interactive. Computer programs can be typed in or loaded in from a cassette. In the 1979–1980 session, students in GGRB02Y were taught to write and run their own simple data analysis programs in BASIC. During that year, I also began to write more advanced 'canned' programs which the students could also use. Also during that year, a set of 72 canned statistical programs were obtained from the Department of Geography at the University of Liverpool. Thus, by the end of the 1979–1980 session, we had a basic operational system of hardware and software and a class of students capable of using it.

Although this system was useful as it stood, its possible application as a teaching aid had barely begun. The canned programs which we had were primarily of the standard problem-solving type. The student typically would select an appropriate program, enter data when requested, and be presented with a set of final calculations as a result. While useful as an operational tool, such a program does not typically show how the calculations were carried out or the assumptions under which a particular test or procedure is valid. Further, these programs typically do not encourage the students to learn or become more confident about the statistical method being employed. They provide answers but not explanations.

The objective of this project has been to develop a different kind of canned program which does explain a solution. I have called such programs 'Tutorial Programs' in the sense that they should act as a patient tutor. They should (1) explain how a problem is solved as well as actually solving it, (2) make clear the 'hidden' assumptions associated with a method, and (3) prompt the student to think about problems by asking questions and posing quizzes. The 25 programs developed in this project were designed with these specific objectives in mind.
Work Completed

Brief descriptions of the 25 programs are included in the appendix. These programs have been numbered to correspond to chapters from the course text, *Statistics: Probability, Inference and Decision* by R. Winkler and W. Hays. For example, programs 7.1, 7.2, and 7.3 are drawn from Chapter 7 on hypothesis testing. Note that these programs could equally well be used with any standard statistics textbook. These programs are basically of three broad, overlapping types:

1. **Show-Me Programs**: These are programs which emphasize how to solve a particular statistical problem. They detail all intermediate steps and calculations. Programs of this type include 2.1, 2.2, 3.1, 3.2, 4.3, 5.1, 6.1, 7.1, 7.2, 7.3, 10.1, 11.1, and 12.1.

2. **Quiz Programs**: These are programs designed to test the student's knowledge of a particular topic. Answers usually are of a multiple choice form. The programs are quite forgiving in that the user is always given a second try when a mistake is made. All Quiz programs give test scores on completion. Some also have extensive explanations of intermediate calculations. Programs of this type include 1.1, 1.2, 2.1, 3.1, 3.2, 4.1, 7.1, and 10.3.

3. **Analyzer Programs**: These programs are more like conventional canned programs with an emphasis on solution rather than detailed explanations. Some programs, (1.3 and 10.2) help a user to interpret a problem graphically on an interactive basis. Others (2.3, 5.1, and 5.4) solve fairly difficult problems for which intermediate steps would be cumbersome to display. Another (4.4) undertakes a Monte Carlo simulation to illustrate graphically how a Poisson distribution is obtained. Finally, others (5.2 and 5.3) illustrate alternative algorithms for solving the same problem.

Many of these programs have built-in numerical examples. This includes almost all of the 'Show-Me' and 'Quiz' programs. A few of these programs randomly generate any number of different numerical problems. The rest have between 3 and 20 built-in case studies or examples. Thus, the user can try a variety of problems of a similar type until he or she feels confident about that technique.

Usefulness of these Programs

I have now had an opportunity to see these programs in use for an academic year. Student reaction to them has been generally favourable. Some programs have been clearly more popular than others. Those making heavy use of interactive graphics, for instance, have been especially well-received. In part, this is because they are fun to watch and simple to operate. They are also popular because they induce group learning. Typically, while one user is running such a program (e.g., 1.1) others are watching over his or her shoulder and offering unsolicited but often informative commentary. Other programs whose outputs are primarily numerical require much more concentrated attention, are not as conducive to group learning, and thus are not as popular.

Of course, popularity is not the only criterion by which to judge the success of these programs. Some programs, such as 10.1, 11.1, and 12.1 have also been 'successes' even though they make little use of interactive graphics. These programs were designed to illustrate how one would systematically apply a particular statistical method. Thus, in addition to solving a numerical problem, these programs also illustrate how one would go about efficiently solving the same problem by hand. This is very valuable in a practical sense. It is my impression that students in the course are now better able to conceptualize and develop efficient solutions to such problems than were their predecessors. In this important sense, some of the programs have been very successful.

From my experiences in other years, I have come to realize that canned programs will not be successful, let alone used, unless they are made necessary in the week-to-week development of the course. In GGRBOY, lectures are complemented by weekly tutorials in which assignments are given and discussed and sample problems are solved. In these tutorials, students are given brief introductions to the programs. Further, the weekly assignments often necessi-
tate the use of these programs. Thus, students are continually encouraged or prodded into using these programs as appropriate throughout the academic year.

I have also observed students using these programs as study aids at the end of each term. The programs cover almost all of the important concepts and techniques used in the course. With their emphasis on assumptions and on systematic solution development, they are valuable for reviewing ideas and approaches. This is another indication of the usefulness of these programs.

Conclusion
The use of microcomputers as teaching aids is relatively recent. As anyone who has used computers in a course can attest, the typical large university computer system has many drawbacks. It is typically awkward to access while maintaining faculty-student contact. Computer terminals are often crowded, untidy, and noisy with occasionally long queues and machines easily subject to shutdown or failure.

By contrast, these microcomputers rarely break down because they have few moving parts. When a microcomputer is turned on, it is instantly ready for student use whereas conventional computers have to be readied for use each time they are turned on. They are clean (no paper input or output), quiet, and easily maintained. They are also very inexpensive. Because they are free-standing (i.e., are not connected to any large computer) and need only a standard 115 volt power supply, they can be installed wherever needed at a minimal cost. Our two machines for example are in the same room where the tutorials are held. Thus, the machines can be placed wherever the professor and students are likely to interact.

To be sure, microcomputers do have disadvantages. They have small memories and a limited repertoire of canned programs and cannot easily make use of datasets stored on other computer systems. However, these limitations are not usually of much importance in an introductory quantitative methods course with the possible exception of limited software. The grant from OED has helped to overcome this particular problem by funding the development of the above-described tutorial programs.

In this project, a set of 25 tutorial programs have been developed which cover the range of topics considered in GGRB02Y. These programs are specifically designed to be something different from conventional programs. They broadly try to simulate a patient tutor. Some of the programs pose quizzes, others draw diagrams to illustrate solutions, and still others detail solution procedures showing all intermediate calculations. The objective has been to get the student to understand how and why a particular technique works rather than to generate answers to numerical problems.

My casual impression is that these programs have been successful on several counts. Students have found several of the programs to be quite enjoyable. Others they have found to be very useful for laying systematic solutions to common kinds of problems. There has also been a widespread use of these programs as review study aids near exam time indicating the value attached to them by the students.

Such programs have become an important part of an overall teaching strategy in GGRB02Y. They are not intended to be a self-teaching system. They are specifically designed to complement lecture and tutorial material and match the textbook. Further, students are motivated to use these programs through the design of the weekly assignments. The various instructional components of the course, lectures, text, tutorials, assignments, and tutorial programs, are thus linked.

Appendix: Program Descriptions

1.1 Venn Quiz
A Venn diagram with 3 interlocking events is displayed. Twenty set expressions are given. For each expression, user indicates areas of diagram that are included in the expression. Correct areas are shaded in. User’s score is given at the end of the quiz.

1.2 Event Quiz
Program employs 30-day Boston weather record. User inputs 2 or 3 event definitions. Twenty set expressions are presented and user must give the number of days of 30 which satisfy the expression. Correct areas are shaded in. User’s score is given at the end of the quiz.

1.3 Venn Analyzer
Program analyzes a 3-set Venn Diagram to indicate areas which satisfy a user-supplied set expression. A Venn Diagram is displayed and user enters expression. The areas that satisfy the expression are shaded in.

2.1 Sequences and Combinations Quiz
Five problems are posed. User is asked to identify whether problem is one of sequences or of combinations. User is then asked about appropriate probability distributions.
2.2 Equiprobability Examples
Program gives a number of problems and solutions involving calculations of probabilities in equiprobability experiments. Program poses questions for which user computes and enters answers. Detailed solution steps are then described so that user can trace source of any errors.

2.3 Runs Calculator
Two samples, A and B, are pooled in a Wald-Wolfowitz procedure in which the number of runs of As and Bs are counted. Program calculates the probability of any given number of runs when the two samples are drawn from the same parent population. Program can be used to calculate Wald-Wolfowitz probabilities or to test significance of two inputted samples.

3.1 Random Variable 1 Quiz
This program generates a probability distribution for a discrete random variable. User is asked to calculate mean, median, mode, variance, and standard deviation of the distribution. Program provides detailed numerical solutions and gives user score at end.

3.2 Random Variable 2 Quiz
Program generates a joint density function for two discrete random variables. Joint probabilities and a graph of the function are displayed. Questions about marginal and conditional probability, expected values, covariance and correlation are asked. Solutions are given. User's score is given at end of quiz.

4.1 Process Quiz
Program asks user to identify different types of probability processes and distributions. Ten sample problems are given. The quiz involves multiple choice questions and a score is given.

4.2 Probability Distribution Analyzer
Program calculates probabilities for discrete and continuous random variables. There are 11 different distributions: Binomial, Pascal, Poisson, Multinomial, Hypergeometric, Exponential, Uniform, Normal, Student's T, Chi-Square, and F Distributions. Program asks for parameter and random variable values or ranges and then calculates probabilities.

4.3 Probability Distribution Examples
Program gives examples of problems involving probability distributions. Multiple choice questions are asked about type of distribution and the probability. Correct answers and solutions are given.

4.4 Map Simulator
Program performs Monte Carlo simulation of a uniform spatial process. A map with 400 zones is presented and hit 100 times at random. Number of hits in each zone is recorded. After 100 hits, program totals number of zones hit 0, 1, 2, 3, or more times and calculates frequency.

5.1 Sample Analyzer
Program calculates a variety of sample statistics and presents a histogram for a set of data. Program calculates: Average, Median, Minimum, Lower Hinge, Upper Hinge, Maximum, Biased and Unbiased Standard Deviation, H Spread, Range, Mean Absolute Deviation, Far-out and Outside Values, and Sample Size.

5.2 Heap Sort Algorithm
Observations are inputted by user. Program sorts the numbers and prints out the ordered sample and time taken to sort data.

5.3 Quick Sort Algorithm
Observations are inputted by user. Program sorts the observations and prints out the sorted values. The time taken to sort the data is also given.

5.4 Curve Fit
Program fits 6 different curves to a set of data inputted by user. The 6 possible functions are:
1) \( Y = A + BX \)
2) \( Y = Ae^{BX} \)
3) \( Y = AX^B \)
4) \( Y = A + B/X \)
5) \( Y = 1/(A + BX) \)
6) \( Y = X/(AB + B) \)
Program calculates best fitting values for A and B. \( R^2 \) is also calculated. The Y estimate is calculated as well as the mean percentage error.

6.1 Confidence interval Calculator
Program calculates confidence intervals for parameter picked by user. Two sets of sample data are provided with which the user must calculate intervals. Five types of intervals and 3 levels of confidence are given. Questions about assumptions, distribution used and the confidence interval estimates are asked.

7.1 Hypothesis Testing Quiz
Program gives examples of hypothesis testing problems. Seven multiple choice questions are posed to user. A detailed solution to each question is presented after choice. Questions cover type I and type II errors, decision rules, and hypothesis testing for means and variances.

7.2 TYPE I and TYPE II Error Problems
Program gives an example of a problem involving hypothesis testing. The problem illustrates how probability of TYPE I and TYPE II errors may be used to find a decision rule.

7.3 Paired Sample Analyzer
Program permits user to input data for paired samples. User chooses confidence level, type of decision rule, and expected difference. User then inputs own sample data. Program solves for mean sample differ-