Evaluation and development techniques for computer assisted instruction programs*

by M. TARTER, T. S. HAUSER, and R. L. HOLCOMB

Department of Biostatistics
The University of Michigan
Ann Arbor, Michigan

INTRODUCTION

Computer-Assisted Instruction presents the individual conducting educational research with the potential for more controlled and reliable experimental design and data collection. Certain features of programmed instruction also present new challenges to the researcher. Chief among these is a problem related to the individualization of programmed instruction. The authors have developed automated experimental design routines which are used to partially overcome these problems. This procedure will be referred to as “Computer-Assisted Pairing” or CAP.

An individual student’s response to instruction is greatly affected by his background. Although it is possible to use such sophisticated statistical techniques as Analysis of Covariance to adjust for background differences, most educational studies rely on the simple statistical device of “paired-comparisons.”

To accomplish this pairing, a pretest is usually administered to a group of students and, on the basis of the pretest results, the students are allocated to pairs. After N/2 pairs are formed a student within each pair is randomly assigned to the experimental group and his counterpart is assigned to the control group. At this stage the treatment, e.g., instruction, is administered to the proper group of students and the “placebo” to the counterpart group. A post-test is then administered to all students, and the differences \(d_i\) between the \(i\)th pair of treated and untreated students’ scores are recorded. The results can be analyzed by means of a simple Student-\(t\) test or a more sophisticated analysis can be conducted. (For example, if two separate post-tests are administered, a Hotelling T-test can be used to handle the resulting multivariate data.)

The above requires a rather idealized experimental environment. Such standard obstacles to straightforward analysis as missing or incorrectly recorded observations are more than likely to occur, especially in classroom teaching situations. However, aside from problems of data-collection, there is one major obstacle to the experimental design that is likely to interfere with the progress of the paired-comparison experiments in CAI research. It may not be possible to assemble and pretest all \(N\) students before administering the instruction sequence to the treatment pair members.

This problem is much more likely to occur in CAI studies than in research involving conventional classroom instruction. Since one of the primary advantages of programmed instruction is that the student can choose his own time of instruction, it would be artificial to require all \(N\) students to have completed a pretest before any student is allowed to proceed to the treatment, i.e., instructional sequence. In fact, an ideal experimental situation would keep information about the separation of pretest and treatment instruction from the student. In this way a single-blind, or even double-blind experiment could be conducted where the student need not know whether he is being given “treatment” or “placebo” instruction and the researcher need know only after the experiment has been completed.

Naturally the branching feature available in any of today’s CAI languages could be used to administer such a double-blind study. However, the problem remains: How can at least a part of the desirable features of pairing be retained in the process of conducting a double-blind experiment, with treatment or placebo administered immediately after pretest completion. In other words, how can the pretest score of the \(i\)th student be used immediately (before the remaining \(N-1\) students are tested) to branch the student to a treatment or placebo sequence.

*Research sponsored by the Public Health Services Biostatistics Training Grant NIH 5 T01 GM 00045-11, NIH GRS 67 SPH Project 78, and Hill Rhodes No. 9.
Of course any technique which pairs in "real time" rather than after the entire sample has been examined will be less efficient than the conventional method. In a sense, the convenience and advantage of using a double-blind experiment will be purchased at the cost of greater variance of the differences \{d_i\}. The best method of real time pairing, therefore, will increase the variance of the \{d_i\} least. A considerable amount of theoretical research has yet to be done on the method proposed in this paper, in terms of variance reduction. However, CAP seems to be a reasonable approach and there is evidence to show that it is far superior to the one alternative method which has been applied.

The alternative pairing method

The alternative method is implemented as follows: The pretest is administered to each of the first N/2 (N even) students and each of these students is assigned at random to either the treatment or placebo group. The \((N/2) + 1\)st student is then given the pretest and paired with the previously tested student whose pretest score is most similar; say the Ith student. If the Ith student was assigned to the treatment group, the \((N/2) + 1\)st student is assigned to the control group and vice versa. The pretest is then administered to the \((N/2) + 2\)nd student. The scores of all previously tested students, with the exception of students \((N/2) + 1\) and I, are compared to the \((N/2) + 2\)nd student’s score. The student whose score is most similar to that of the \((N/2) + 2\)nd student is then paired and assigned to the opposite group.

The disadvantage of this simple procedure is that the decision to pair is left to the last possible point, i.e., when half of the group is already allocated to the experimental or control group. Thus, there is no opportunity for two students among the first \((N/2)\) to be paired with each other. It might also be noted that this design is particularly sensitive to non-random ordering of the students. For example, consider the case where students who present themselves for instruction early in the semester tend to be better students than those who use the teaching technique later (after conventional course exam grades are returned). In this situation, the simple sequential pairing method will tend to allocate a good and a poor student to the same pair which defeats the very purpose of pairing.

In the remainder of this paper, the group of students who have been tested and allocated to the treatment or control group (but not paired) will be referred to as the "pool." In the previous example, the "pool" is filled to the largest possible extent before being emptied, i.e., its members paired. The sequential pairing method proposed in this paper allows students to be removed from the pool at each stage rather than waiting until the pool is filled to capacity and then emptying it.

The CAP algorithm

The new pairing method will be referred to as the CAP algorithm because the implementation almost certainly requires high speed digital computation. Since in CAI investigations the pretest will probably be administered by console presentation, the same program which administers the pretest can be used to store previous students’ pretest scores and implement the CAP algorithm.

The CAP algorithm can be divided into two parts. First, the main routine uses the Ith student’s transformed score and decides whether he should be paired or entered into the pool. Second, a sub-program determines the appropriate transformation and applies it to the Ith student’s score and the score of all other students within the pool.

The reason why "transformed scores" are necessary will become clear when the statistics underlying the CAP algorithm are discussed. Temporarily, the assumption will be made that the distribution of pretest scores is uniform on the interval \([0,1]\). This can be represented mathematically by the statement that the distribution \(f(x)\) of the pretest scores equals:

\[
f(x) = I_{[0,1]}(x)
\]

where \(I_{[a,b]}(x)\) is the indicator function

\[
I_{[a,b]}(x) = \begin{cases} 
1 & \text{if } a \leq x \leq b \\
0 & \text{otherwise}
\end{cases}
\]

(If \(I(a)\) is to be 1 and \(I(b)\) zero this function is written \(I_{[a,b]}(x)\).)

The CAP main program performs the following operations: At the Ith insertion, i.e., immediately after the Ith student’s pretest score \(Y[1]\), is available, a pairing criterion \(DEL\) is computed. The score within the pool that is closest to \(Y[1]\) is determined. Let the score closest to \(Y[1]\) be denoted by \(Y[MIN]\). The expression \(Z = |Y[MIN] - Y[1]|\) is calculated and if \(Z \leq DEL\), then the MINth and Ith students are paired. If the MINth student had been assigned to the treatment group, the Ith student is assigned to placebo group and vice versa. If \(Z > DEL\), then the Ith student is randomly assigned to either the treatment or control group and added to the pool. Naturally, if the pool size equals the number of students still untested at the Ith insertion then the \((I + 1)\)st through Nth students are all paired. This is equivalent to setting \(DEL = 1\) whenever the pool size equals \(N-1\).
The most important part of the main program is naturally the computation of DEL. When only early scores are available, DEL should be small since there are at this stage many possible future scores, \( Y[J] \), which might satisfy the inequality \( |Y[\text{MIN}] - Y[I]| \) \( \geq |Y[J] - Y[I]| \). Conversely when almost all students have been pretested, DEL should be large since there are fewer scores that can occur between \( Y[\text{MIN}] \) and \( Y[I] \).

The "probability of mis-pairing," which will be symbolized by \( A \), was used as a criterion for the determination of DEL. Two scores can be said to be mispaired at Stage I if one of the \( N-I \) "future" scores occurs between \( Y[I] \) and the \( \text{MIN} \)th pool member paired with \( Y[I] \). Now after the 1st insertion, what is the probability that one of the \( N-I \) remaining scores will occur in the interval beginning with \( Y[I] \) and ending with \( Y[\text{MIN}] \) (or beginning with \( Y[\text{MIN}] \) and ending with \( Y[I] \))? If the remaining \( N-I \) scores can be assumed to be independent and uniformly distributed on the interval \([0,1]\) then the probability of mis-pairing is exactly

\[
A = 1 - (1 - |Y(I) - Y(MIN)|)^{N-I}.
\]

By solving the above for \( |Y(I) - Y(MIN)| \) one can determine that DEL is related to \( A \) by the function

\[
\text{DEL} = 1 - (1 - A)^{(N-I)}.
\]

If an allowance is made for a high probability of mis-pairing, large \( A \), then DEL will be close to one. On the other hand, if one chooses a small probability \( A \) then DEL will be small. However, if \( A \) and consequently DEL are assigned small values then most students will remain unpaired until the pool size equals \( N-I \), at which point DEL must suddenly become equal to one. Therefore, the efficient implementation of the CAP main program depends upon some reasonable determination of \( A \) and hence DEL. To accomplish this the probability \( A \) is considered as a parameter, related only to the sample size \( N \) and constant throughout the pairing process. Under this assumption, the pairing criterion DEL is related to \( I \) through the formula

\[
\text{DEL} = 1 - (1 - A)^{(N-I)}
\]

where now both \( N \) and \( A \) are considered as parameters.

By use of the function given in the preceding paragraph the problem of finding an efficient pairing criterion has been reduced to the problem of estimating the parameter \( A \). An overall criterion of efficient pairing therefore, must be introduced and the parameter \( A \) estimated on the basis of this criterion. The total pair separation was chosen as this criterion. Separation in the case of the score \( Y[I] \) and its eventual mate \( Y[\text{MIN}] \) is defined as simply the distance

\[
|Y[I] - Y[\text{MIN}]|.
\]

Clearly the best sequential pairing method is the one which yields the total pair separation closest to the pair separation possible for the ordinary pairing situation. In the ordinary pairing situation, complete information—in the form of knowledge of all \( N \) scores, is available before any student is to be paired. The most efficient sequential pairing algorithm would be the one which best used the limited information available at the 1th stage, i.e., that obtained from the \( I \) previously measured scores.

The estimation of the parameter \( A \) is made in the following way. For a given sample size \( N \), the estimate \( A \) is chosen which minimizes the total pair separation. Estimates of \( A \) have been obtained for various sample sizes and used in the CAP program.

A program was constructed which simulated the above pairing process, and, therefore, could be used to estimate \( A \). The total pair separation was measured for repeated samples of size 50, 100, 200, and 250 where \( A \) was chosen as \( .05 * K, K = 0, \ldots 18 \). For all samples the optimal value of \( A \) was found to be surprisingly large. Since the alternative method of sequential pairing, which was described earlier, is a special case of the CAP procedure where \( \text{DEL} = 0 \) for all \( I \) from 1 to \( N/2 \) and \( \text{DEL} = 1 \) for all \( I \) from \( N/2 + 1 \) to \( N \), the observation that \( A \) and hence DEL should be large for small values of \( I \) tends to show that the CAP procedure greatly improves upon the alternative sequential pairing method. In fact for all sample sizes, the CAP procedure tends to reduce the total pair separation by a factor of at least two. In Table I the estimated optimal values for \( A \) are given, as well as two other estimates which are of interest. These are: First, the size of the pool \( N_I \), when \( N_I = N - I \), i.e., at the time when DEL is set equal to one and all subsequent scores are paired. Second, the number of times \( N_I = 0 \), i.e., the pool is depleted during the pairing process. Obviously, if the pool is depleted immediately before any stage \( I \) other than \( N \) then the \( I \)th score must be entered into the pool.

Theoretical and simulation work has shown that the CAP main program provides a substantial improvement over the alternative simple sequential pairing method. Actual trials with real data are currently being conducted to check the implementation of the CAP technique.

One of the reasons trials with actual data are needed to test the efficiency of CAP is that the CAP main program requires the assumption that the pretest scores are uniformly distributed. Since this is obvious-
ly a very restrictive assumption, a CAP subroutine is used to preprocess the Ith and all scores within the pool as soon as the Ith pretest score is available. The remainder of this section will deal with the theory and implementation of this transform subroutine.

The transform subroutine

Let the sequence of N random variates, identically distributed with density function f(x), be represented by X[1], ..., X[N]. Let the function F(x) represent the cumulative distribution function associated with density f(x). The random variables, F(X[1]), F(X[2]), ..., F(X[N]) are uniformly distributed with density function \( f(x) \), be represented by \( X[1], \ldots, X[N] \). Let the function \( F(x) \) represent the cumulative distribution function estimation.

The sample cumulative or step function \( F^*(x) \) would in ordinary circumstances be considered a good estimate of \( F(x) \). However, for the purposes of CAP preprocessing, it is a poor estimate. The step function \( F^*(x) \) equals

\[
F^*(x) = \frac{1}{n} \sum_{i=1}^{n} [I_{X[i]}(x) + (1/2) I_{X[i-1]}(x)]
\]

where \( X_i \) represents an arbitrary sample point. \( F^*(x) \) is not a smooth or differentiable function. Also, since \( 2nF^*(x) \) must be an integer for every value of \( x \), \( F^*(x) \) would distort the "local spacing" of the transformed values. Since in the CAP main program the spacing of consecutive points is particularly important, it is obvious that \( F^*(x) \) is not a suitable transformation to uniformity.

What is required is a smoother estimate of \( F(x) \) which can be updated easily as new pretest scores are obtained. An estimate of \( F(x) \) which not only fulfills the above two requirements, but also is more efficient than \( F^*(x) \) has been investigated by two of the authors1, 2 and a few of its properties will be reviewed in this paper. This estimate will be represented as \( F_m(x) \) where

\[
F_m(x) = \frac{x - a}{b - a} + \sum_{k=1}^{m} \frac{(b - a) \tilde{c}_k}{\pi k} \sin \pi k \left( \frac{x - a}{b - a} \right)
\]

and \( n \) represents the number of data points \( X_1, X_2, \ldots, X_n \), and "a" and "b" are two predetermined constants, preferably such that for most \( X_i \) the inequality \( a < X_i < b \) will be satisfied. It is shown2 that as \( m \) approaches infinity \( F_m(x) \rightarrow F^*(x) \). Also for all densities with bounded variation, e.g., all continuous distributions commonly encountered in statistical research (the Normal, Cauchy, Laplace, Gamma, and Logistic) \( F_m(x) \) is a more efficient estimate than \( F^*(x) \).

Here efficiency is measured in terms of Mean Integrated Square error \( J(F_m) \) where

\[
J(F_m) = E \int_a^b [F(x) - F_m(x)]^2 \, dx.
\]

In Ref. 2 it is also shown that the constant \( F_m(x) \) associated with the most efficient estimator of the form \( F_m(x) \) is usually less than 10. Consequently \( F_m(x) \) provides both an easily computed and a smooth estimate of \( F(x) \) and \( F_m(x) \) is actually more efficient than \( F^*(x) \). In Ref. 1 and 2, a rule for determining the optimal value of \( m \) is given. This stopping rule is based on an unbiased estimator of Mean Integrated Square Error \( J(F_m) \). Also4 a computation scheme is given which allows the constants \( \tilde{c}_k \) of \( F_m(x) \) to be computed recursively. Since the estimator of \( F(x) \) should be revised after each new pretest score becomes available, the recursive computation of \( \tilde{c}_k \) and hence \( F_m(x) \) represents a considerable saving in terms of computer time.

Implementation of CAP

The following is a brief outline of the implementation of CAP:

A. Since it is likely that no \( a \) \( \text{priori} \) information about the form of \( F(x) \) will be available, the first 20 students are added to the pool and assigned at random to the treatment or placebo group.

B. Using the previous 20 as well as the 21st pretest score the transformation \( F_m(x) \) is determined. The procedures used to compute \( \tilde{c}_k \) and the stopping rule for determination of \( m \) are given in Ref. 1.

<table>
<thead>
<tr>
<th>Sample Size</th>
<th>Optimal A</th>
<th>Expected Pool Size When Pool Must Be Emptied</th>
<th>Expected No. of Times Pool is Emptied</th>
<th>Total Separation Using CAP with Optimal A</th>
<th>Total Separation Using Alternative Pairing Method (A = 0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>.55</td>
<td>4.55</td>
<td>.200</td>
<td>1.35</td>
<td>2.64</td>
</tr>
<tr>
<td>100</td>
<td>.60</td>
<td>3.80</td>
<td>.250</td>
<td>1.54</td>
<td>4.02</td>
</tr>
<tr>
<td>150</td>
<td>.65</td>
<td>2.95</td>
<td>.550</td>
<td>1.68</td>
<td>5.01</td>
</tr>
<tr>
<td>200</td>
<td>.75</td>
<td>1.55</td>
<td>1.00</td>
<td>1.92</td>
<td>5.01</td>
</tr>
</tbody>
</table>

\( \tilde{c}_k = \frac{2}{(b - a)n} \sum_{i=1}^{n} \cos \left( \frac{\pi k (X_i - a)}{b - a} \right) I_{[a,b]}(X_i) \)
C. The original 21 pretest scores
   \[X[1], X[2], \ldots, X[21]\]
   are transformed by means of \(F_m(x)\) to
   \[Y[1], Y[2], \ldots, Y[21].\]
D. The score \(Y[\text{MIN}]\) is determined where
   for all \(J\) from 1 to 20.
E. An estimate of \(A\) has been read into the program
   to suit the eventual sample size \(N\) of this particular
   CAI experiment. The pairing criterion \(\text{DEL}\) is computed
   \[\text{DEL} = 1 - (1 - A)^{\frac{1}{(N - 21)}}.\]
F. If
   \[|Y[\text{MIN}] - Y[21]| \leq \text{DEL},\]
   the 21st and the MINth students are paired and the
   21st student is assigned to the treatment group if the MINth student is assigned to the
   placebo group or vice versa.
G. If
   \[|Y[\text{MIN}] - Y[21]| > \text{DEL},\]
   the pool size is increased to 21 and the 21st student
   is randomly assigned to either the treatment
   or placebo group.
H. This process is repeated as the 22nd through
   Nth student’s pretest scores become available.
   However,
   1. For each new score, the constants \(c_k\) are
      updated and a new value of the transform
      \(Y[1]\) calculated for each student in the
      pool.
   2. If at any time before the last pair is formed,
      the pool is emptied, the next student is en-
      tered into the pool. (Equivalently, \(\text{DEL}\) is
      set equal to 0.)
   3. When the pool size equals the number of
      students still to be tested all subsequent
      student are paired with their closest coun-
      terparts within the pool. (Equivalently
      \(\text{DEL}\) is set equal to 1.)

By following steps A through H, each student with-
in every pair has been randomly assigned to the treat-
ment or placebo group. Also each student, i.e., the
Ith, is available for the treatment even though the pre-
test has yet to be administered to N-I students.

The construction of CAI programs

Up to this point a method for conducting a CAI ex-
periment has been described, but no comment has
been made concerning the source of data for such an
experiment. In this section a brief description will
be given of a particular CAI project and the process
of program development rather than evaluation will
be emphasized.

For the past two years the School of Public Health
at The University of Michigan has been conducting
an extensive experiment on the effect of Computer-
Assisted Instruction within a section of a large univer-
sity. This project has already generated ten programs
of more than intermediate size, although much data
has yet to be gathered before any final conclusions
can be announced. Programs have been written in
such diverse areas as Biostatistics, Epidemiology,
Environmental Health, Public Health Dentistry, Public
Health Education, and Industrial Toxicology. Several
different procedures have been used to con-
struct CAI programs and, therefore, our observations
about these procedures may be of value to other work-
ers in this field. The construction of CAI programs
is an expensive process at this early stage of hardware
development and our observations may suggest short-
cuts and point out pitfalls.

The best place to start when discussing CAI pro-
gram construction procedures is with the personnel
who actually participate in the construction process.
Four categories can be listed.
A. The person who will actually be responsible for
   the implementation of the completed programs
   and who initiates the construction process. At
   the University this will usually be the professor
   who wishes to use the material in one of his
   classes.
B. Subject matter oriented staff working under the
   supervision of the person in category A. Here
   the term “subject matter oriented” is used to
distinguish this type of person from CAI pro-
grammers.
C. CAI programmers—with experience and train-
ing in education or psychology, but with little
background in the specific subject matter areas
   to be programmed.
D. Teaching assistants, research fellows, trainees,
   and others, who might be called the transient
   work force.

A brief description of the process itself follows. The
list given below is, of course, a highly idealized one.
However, like any other form of computer program-
ming there are definite clear-cut stages, e.g., flow-
charting, coding, debugging, which must be carried
out before a satisfactory working program is obtained.

1. An initial decision about the subject matter to
   be taught must be made.
2. The level of sophistication of the students who
   will take this program should be determined.
3. Should remedial sections be provided?
4. If yes to No. 3, how elementary should the
   material in the remedial sections be?
5. Should advanced sections be provided for the
   brighter students?
6. If yes to No. 5, how advanced should these sections be?
7. A list of the specific concepts that must be presented in a teaching program on this subject should be constructed.
8. The sequential order in which these concepts should be presented must be determined.
9. At least one question for each concept or fact that tests the understanding of the students must be written.
10. Information that must be presented to the student along with each concept that is presented must be determined.
11. A list of typical misconceptions by students should be made.
12. Constructive responses that would correct these misconceptions should be listed.
13. At least two general constructive comments to be presented to the students who respond with an answer that was not anticipated should be written.
14. Pictures or graphs that may be helpful to the students in the understanding of the concepts or facts presented should be obtained.
15. Appropriate use of slides, tapes and typeouts should be determined.
16. The general flow the program is going to follow should be decided upon.
17. The prepared materials in the computer language used must be programmed, i.e., coded.
18. The program must be entered into the computer.
19. The coding should be debugged.
20. The program should be tested (using students) to determine if it actually teaches.
21. Appropriate content revisions indicated by early student testing should be made.
22. Observations should be made of the perfor-
mance of at least ten students as they take the program and their comments and questions should be noted.

23. Further changes as indicated from students' reactions to the program should be made.

A number of methods have been used in writing programs in the School of Public Health. All of the methods involved the 23 steps described above.

The most successful of the alternatives we have tried involves the professor (A), a graduate student or staff member (B or D), and the programmer (C). The professor and his assistant (A and B, or D) completes steps 1-8 (the general outline of content materials to be used with a specification of the desired upper and lower limits of the materials). The student carries through steps 9-15 and discusses step 16 (general flow of the program with the possibilities of branching) with the programmer (C). The programmer handles steps 17-19. Then the responsibility is again assumed by the student and the professor for steps 20-23. Throughout this period, channels of communication have been established between the three people involved. When the work is distributed in this manner, all concerned seem to find the time and maintain interest in the program. The programmer is available for consultation in regard to possible uses of the computer, and the graduate student can solve the majority of the content problems on his own.

One problem we have found in implementing this method has been a lack of interest on the part of some graduate students. They have felt the typical pressures for research as opposed to teaching as a prerequisite for advancement within their own fields. The attitude of graduate students in general toward teaching was often a negative one. However, we have found that certain students who plan to teach in the future take this opportunity to develop teaching materials very seriously. This is also noted in Reference 3. They learn how important it is to break topics into small sections and sequence them in a logical order. They are often more strongly motivated after they observe students taking their programs. Frequently advanced graduate students (D) or professional staff (B) do not realize how a misplaced fact or lack of information can lead to misconceptions on the part of the learner who is not familiar with a subject.

Our experiences lead us to believe that CAI programs must be written with full cooperation and communication between the professor (who devotes as much time as possible) and the programmer. To save the professor's time a student or staff assistant who is familiar with the subject matter and the programmer, carries through several of the 23 steps. Very few of the professors at the University of Michigan, School of Public Health have had the time needed for optimal participation in a project of this type. Therefore, the professor-student or staff assistant-programmer combination is usually the most feasible. This method also enables him to participate with a number of professor-student pairs. The quantity and quality of programs written is this way in our opinion represents a great improvement over other methods attempted. This statement, of course, is now being rigorously verified using CAP in conjunction with other evaluation procedures.

CONCLUSIONS

Computer-Assisted Pairing is a dynamic design for paired comparison evaluation of Computer-Assisted-Instruction sequences. It appears to be of considerable practical value since the pretest-treatment-posttest sequence can be made invisible to the subject. Simulation studies have shown CAP to be substantially superior to previously considered alternatives.

The application of the technique is certainly not limited to Computer-Assisted-Instruction. CAP can be applied fruitfully to almost any experimental situation where paired comparisons are needed. It is especially useful when task initiation and evaluation can be done by the computer.

ACKNOWLEDGMENT

The authors gratefully acknowledge the advice and support of Drs. Karl Zinn and Stanford Ericksen of the Center for Research on Learning and Teaching, University of Michigan and Associate Dean John Romani of the University of Michigan School of Public Health. Also we would like to thank Miss Wendy Hiller for help in checking the final manuscript of this paper.

REFERENCES

1 M TARTER R HOLCOMB R KRONMAL
A description of new computer methods for estimating of the density of stored data
2 R KRONMAL M TARTER
The estimation of probability densities and cumulatives by Fourier series methods
Accepted for publication Jour Am Stat Assoc
3 M TARTER
Programmed instruction in statistics from the professor's point of view
The American Statistician 21:28-31 1967
4 V CLARK M TARTER
Preparation for basic statistics automated text
McGraw Hill 1967
5 J E COULSON
Programmed learning and computer-based instruction
Wiley 1962
6  W  FEURZEIG
   *A conversational teaching machine*
   Datamation 10:6 1964
7  K  L  ZINN

Survey of materials prepared for instruction or research
on instruction in on-line computer systems
Automated Education Handbook E Goodman (Ed) Detroit
Automated Education Center 1965