HOW LITTLE COMPUTING NEED WE TEACH TO STATISTICS MAJORS?[1/]

BU-483-M September 1973

S. R. Searle

Biometrics Unit, Cornell University, Ithaca, N.Y.

Abstract

It is more important for statistics majors to learn statistics than computing. The dire need is for people who really know what to compute more than for those who know how to compute. There are already enough people who are expert in efficiently computing nonsense.

1. Introduction

Barely seventeen years ago many universities were taking delivery of their first stored program computer, the IBM 650. In its original form this machine's sole device was a drum of 2000 words, each of 10 decimal digits plus sign, with no decimal point. There was no core storage (60 words of high-speed core became available around 1958), there were no magnetic tapes, not even any on-line printers, all I/O being by means of cards. (Often, the tabulator for printing one's output cards was not even in the same room!) There were no canned programs and no languages such as we know them today. BLIS (Bell Labs Interpretive System) was followed by SOAP (Symbolic Optimum Assembly Programming) and by 1959 the forerunner of today's FORTRAN became available, in the form of a 3-pass procedure through the computer. Otherwise, programming was in assembly language using 2-address instructions in which, for example,

---

[1/] Paper invited for Workshop 4 "Education and Training at the Interface", at the Seventh Annual Symposium on the Interface of Computer Science and Statistics, Iowa State University, Ames, Iowa, October 18-19, 1973.

0001 70 1951 002

was an instruction that one chose to store in location 0001 on the drum, an instruction to read 80 columns into the eight 10-digit storage locations 1951 through 1958. Clearly, in order to write a program that would successfully carry out analysis of variance calculations, for example, one had to know just exactly what formulae one wanted.

Let us look at the position today. Computers, by standards of the 1950's, have become giant-sized with high-speed core storage measured only in units of 2000 locations, or (2^n)K as we are now so familiar with. Endless storage devices such as drums, tapes, discs, slow core and an ever increasing array of I/O devices can be "hung on the main frame", as today's jargon would have it. But not only are the machines themselves vast: there are also voluminous libraries of canned programs, ready-made programs that can do all manner of marvellous things. Insofar as statistics is concerned, these things consist mainly of high-speed consumption of data and arithmetic massaging thereof.

2. Computers Demand Knowing Statistics

What are the effects of all this on statistics? They are many and varied, and only some of those that affect the consulting aspect of the statistical profession will be considered here. In this context, the largest effect of computers on statistics has probably been that analyses of large, nay very large, sets of data are now perfectly feasible both in regard to time and money, whereas prior to today's goliath machines they were not. We need think only of the arithmetic involved in such things as large-scale multivariate analysis, factor analysis, variance components estimation, and iterative procedures for non-linear estimation problems to be immediately aware of what can be achieved today that was completely impractical

yesterday. But the big need still remains: to be a good statistician one must know what it is that has to be calculated. If anything, this need is even more urgent than it was before the advent of computers. Prior to their omnipresence a statistician had to know what calculations he required of his (or his client's) data, because he either had to do them himself or personally instruct someone in the miniscule details of what did have to be done. Nowadays, with program libraries at hand, the statistician who just wants to calculate "answers" might know, in a rows-by-columns analysis of variance situation, for example, that he requires a row sum of squares and a column sum of squares. Locating a program whose documentation announces that its output RSS is the row sum of squares and CSS is the column sum of squares, our "answer"-seeking statistician might then do nothing more than push his data through that program and out pops the "answers" he so eagerly awaits. But the salient questions are something like this. Are the calculations that the program has carried out really the ones that the statistician wants? Does he know exactly what he wants? Can he understand the documentation and accurately interpret it in terms of the analysis it performs? If he uses a second program purporting to do the same calculations and gets different "answers" from the same data, does he have sufficient knowledge of theory to figure out why the answers differ and what each represents?

Placing reliance on library programs to do the computing for us puts, if anything, a greater burden on a statistician's knowledge than when he wrote his own programs. This is so because in writing one's own program one saw to it that it did carry out the calculations required, and one knew what these were. But nowadays the statistician has to have all his options open and to know all of the varied meanings that different people use for the same descriptive words, e.g., for "row sum of squares". When he wrote his own program he knew his own meaning for that phrase. But a program documentation may mean something different, without clearly saying so, and the statistician therefore needs to be thoroughly familiar with all possible meanings, not just the one he customarily uses. Only then can he be effective in

ascertaining what it is that different programs are calculating, and hence be of help to his client in explaining their differences.

This burden of knowledge on the statistician is not to suggest that he should go back to writing his own programs. Far from it, for with today's amazing computer power, with the high degree of expertise that goes into most library programs and with the very large data sets that they can handle, using library programs is clearly the efficient procedure to follow. But in doing so we must become vividly aware of the potential pitfalls and in our teaching strive to produce people that will avoid them. On this count, the need seems to me to be one of training statisticians who know what to compute, more than training them to know how to compute. There is an abundance of people who are experts in knowing how to compute, but they need direction insofar as statistics is concerned, because some of them have wasted their expertise computing nonsense.

3. An Example: The 2-way Classification

Consider the familiar rows-and-columns linear model, the 2-way cross classification, represented by

$$E(y_{ijk}) = \mu + r_i + c_j + rc_{ij} \qquad (1)$$

where $E(y_{ijk})$ is the usual expected value of the k'th observation in the i'th row and j'th column, μ being a general mean, r_i the effect of the i'th row, c_j the effect of the j'th column and rc_{ij} the corresponding interaction effect. In general let $i = 1,2,\cdots,a$, $j = 1, 2, \cdots, b$ and for data in which every cell has n observations (balanced data), $k = 1, 2, \cdots, n$. When the cells have differing numbers of observations, n_{ij} in the i,j'th cell, $k = 1, 2, \cdots, n_{ij}$ including the possibility that $n_{ij} = 0$ for some cells; i.e. they contain no data. This is the case of unbalanced data.

Suppose our "answer"-seeking statistician had unbalanced data to analyze according to this model. If a computer program had output that included a term described as "row sum of squares" it would be essential to know whether it was the sum of squares for rows after fitting the mean, or the sum of squares for rows after fitting the mean and the columns. A statistician in this situation must not only recognize the possible different meanings (as well as recognizing that they occur only for unbalanced data and not for balanced data), he must also be able to verify his interpretation of the output, by trying the program on some data whose sums of squares he knows.

Just how confusing this can be and indeed is, right at the present time, is illustrated by Francis [1973], who analyzes a set of data for the model (1) for just 2 rows and 5 columns with unequal numbers of observations in the 10 cells. We will not be concerned here with the numerical results, numerical accuracy, nor the expense of using different programs on the same small set of data. These are discussed in some detail in Francis [1973]. Instead, we devote attention to the actual expressions computed as sums of squares by the 4 widely available statistical program packages that Francis used.

Denote by $R(\mu,r,c,rc)$ the reduction in sums of squares due to fitting the model (1). Similarly, the sum of squares for fitting $E(y_{ijk}) = \mu + r_i + c_j$ will be $R(\mu,r,c)$, that for fitting $E(y_{ijk}) = \mu + r_i$ will be $R(\mu,r)$, and so on. We also define differences such as

$$R(r|\mu) = R(\mu,r) - R(\mu) ,$$

described as the sum of squares due to fitting rows after fitting the mean. In this way we have two possible ways of partitioning the total sums of squares, as shown in Table 1.

(Show Table 1)

These partitionings are well-known and extensive discussion of them is given, for example, in Searle [1971, Chapter 7].

The 4 analysis of variance routines used by Francis [1973] are as follows. (i) Program ANOVA from the SAS (Statistical Analysis System) package — see reference [1]. (ii) Program BMDX64, a general linear hypothesis program from the Biomedical Computer Programs, Dixon [1970]. (iii) CAROLINA, a multivariate analysis of variance program — see reference [2]. (iv) Program MANOVA, a multiple analysis of variance program in the OSIRIS package — see reference [6].

The documentations of all four of these programs lead the reader to believe that they can each carry out the calculations for an analysis of variance of unbalanced data. The form of the calculations they performed on the Francis data are shown in Table 2, in most cases in terms of the sums of squares shown in Table 1.

(Show Table 2)

The object of Table 2 is <u>not</u> that of discussing which program computes the correct thing. The message in the table, insofar as the statistics-computer interface is concerned, is that clearly these 4 different programs mean quite different things for the otherwise innocent-sounding phrase "row sum of squares", for example. The appropriateness of the different meanings is certainly open to discussion, but that is really not the question here. The point is, that with only numerical results in front of him, and with no indication as to the different algebra behind each result, many a statistician would be easily perplexed by this state of affairs. And without a good training in statistics he would be unable to sleuth out why the results are different. And note: that sleuthing requires no knowledge of computing whatever. It <u>does</u> require a thorough statistical training. It is this kind of situation which urges me to soft-pedal the teaching of computing to statistics majors, in order to give them just as much training in statistics as they can possibly take.

4. Other Examples

There is an abundance of other examples of a statistician's needing to rely more on his statistical training than on his computer training in order to be an effective statistician. One that comes to mind concerns estimating variance components for the model (1). A widely used manual for this purpose is Harvey [1960], which suggests (on pages 67, 68 and 76) that for one method of estimation the expressions $R(\mu,r,c,rc) - R(\mu,r,rc)$ and $R(\mu,r,c,rc) - R(\mu,c,rc)$ be calculated. Through erroneous computing procedures for $R(\mu,r,rc)$ and $R(\mu,c,rc)$, non-zero values are sometimes obtained for these differences. Detailed explanation of the wrong computing procedures is available in Searle [1972], but the fact that the differences <u>are</u> identically zero, i.e.,

$$R(\mu,r,c,rc) - R(\mu,r,rc) \equiv 0 \ , \tag{2}$$

is readily seen. First, $R(\mu,r,c,rc)$ is the reduction due to fitting the model (1), a reduction whose value is well known to be

$$R(\mu,r,c,rc) = \sum_{i=1}^{a} \sum_{j=1}^{b} \bar{y}_{ij\cdot}^2 \Big/ n_{ij} \ . \tag{3}$$

(e.g., Searle [1971, p. 292].) Second, by definition of the R()-notation for reductions in sums of squares, $R(\mu,r,rc)$ is the reduction due to fitting the model

$$E(y_{ijk}) = \mu + r_i + (rc)_{ij} \ .$$

This is indistinguishable from the model for a 2-way nested classification with the factor (rc) nested within the r-factor. And the reduction in sum of squares for fitting such a model is well known (e.g., Searle [1971, p. 252]) to be $\sum_{i=1}^{a} \sum_{j=1}^{b_i} \bar{y}_{ij\cdot}^2 \Big/ n_{ij}$, where in the case being considered here $b_i = b$. Hence

$$R(\mu,r,rc) = \sum_{i=1}^{a} \sum_{j=1}^{b} \bar{y}_{ij}^2/n_{ij} \ , \qquad (4)$$

and so together with (3) we have (2). Any calculation of $R(\mu,r,rc)$ which leads to $R(\mu,r,c,rc) - R(\mu,r,rc)$ being other than zero is therefore wrong. Again, the point here is not so much what is right or wrong as it is that a statistician using output from a canned program often needs to know much more about statistics than about computing in order to make correct use of that program output.

Multivariate analysis is a fertile source of computer output often being misunderstood due to the user's inadequate training in statistics. At the same time, multivariate analysis is an area where computers have suddenly taken many kinds of analyses from the realm of almost practical impossibility to one of reasonable feasability. Yet how often do we come across someone who has run a factor analysis program, probably several times, without really having the slightest idea of what the output means — not in terms of their own data, but in terms of the underlying theory and objectives of factor analysis as a statistical method. Computer programs that can do all the necessary arithmetic are now so easily available whether the would-be user knows the background theory or not, that erroneous uses of such programs are continually on the increase. Only when, with interactive computing, we are asked by the program "What does a correlation of 0.7 mean?" will this situation improve. For then, if we give no response or a wrong one, the terminal can type the message "read Snedecor and come back tomorrow" and simultaneously with a spark, a flash of flame and a smell of burning destroy the output before our eyes.

With the easy availability of carrying out long and complex arithmetic, surely then it is our duty as teachers of statistics to see that our students learn the fundamental meaning of the answers, the computer output, that they get. This means teaching them as much statistics, not computing, as they can take. They need to know

what to compute far more than they need to know how to compute it. There are already enough computing experts who can and do very efficiently compute nonsense. We need no more of them.

5. Computer Appreciation

The recommendation here is not to teach no computing, but it is very definitely to teach only a little computing. I have in mind what could be called a computer appreciation course, of 3 hours credit for one semester. Its basis would be 3-5 lectures on a simple language (10-statement FORTRAN, PL/C or BASIC, or some such) followed by having the students use the language to write and make operational 8-10 programs illustrative of the varied uses to which a computer can be put. Some of these uses are arithmetic, counting, classifying and editing of data, ranking, sorting, alphabetic sorting, looking up tables, and simulation, including the generation of pseudo-random numbers. The object would be not to write programs that the student might keep as his own personal library (although he could if he wished), but to teach the student, through doing, what is involved in program writing and what a computer can be made to do. The course would also include introduction to elementary computer hardware and machine logic, binary and other number systems, and the existence of other languages, library programs, subroutine writing and so on.

A course such as this has been taught in one form or another in the N. Y. State College of Agriculture and Life Sciences at Cornell for 8 years. Its yearly enrolment is now approximately 150. Students are far from being only statistics majors — they are graduates and undergraduates, from many and varied disciplines, all interested in learning some of the rudiments of using and understanding the use of a computer. The general background of the audience means that the course is not oriented solely towards statistics. But for statistics students that does not worry me. The

need is to get some acquaintance with computers, how they get used and what they are capable of doing. The result should be that when a statistician comes to the point of having to use a computer, he need not be ignorant of what is involved and therefore not fearful nor untrusting. Either, with a little effort backed up by a general background and understanding of computers and programming, he will be able to write his own program; or, better still, he will be able to talk intelligently to a programmer about what he wants. Either way, he need not be in any sense a computer expert — he need only have a sympathetic appreciation of the skills involved in using computers.

6. Numerical Analysis

One aspect of computing that statisticians need in their general appreciation of computers is, surely, a modest introduction to numerical analysis, especially to such topics as rounding error, iterative techniques, and the solution of non-linear equations. Without citing details we all know of the difficulties that can arise solely from rounding error and the problem of having to define zero. As statisticians, most of us probably need to know a little more about these topics than we do now.

Computing a matrix inverse, or solving linear equations, can serve as illustration. In a course on matrix algebra, I have for some years taught nothing about the numerical methods of inverting a matrix. But the example from Townsend [1966] that

$$\begin{bmatrix} 2.04 & 2.49 \\ 2.49 & 3.04 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} .45 \\ .55 \end{bmatrix} \text{ has solution } \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} -1.0 \\ 1.0 \end{bmatrix}$$

whereas

$$\begin{bmatrix} 2.04 & 2.49 \\ 2.49 & 3.04 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} .451 \\ .55 \end{bmatrix} \text{ has solution } \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} 1.20\ddot{6} \\ -.66 \end{bmatrix}$$

is a striking illustration of the power of small numbers and hence of the effect that rounding error can have. This is accentuated by further noting that if either equation be put into a computer that uses only 2-digit numbers the two problems, after rounding, would both become

$$\begin{bmatrix} 2.0 & 2.5 \\ 2.5 & 3.0 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} .45 \\ .55 \end{bmatrix} \text{ with solution } \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} .1 \\ .1 \end{bmatrix} .$$

The widely differing solutions to these sets of apparently very similar equations vividly illustrate the need to appreciate the apparent tricks that numbers can play, and this need is all the more acute when the underlying numerical methods are being left to the computer, as is appropriate today. These illustrations serve as no more than warnings and highlight the need for enough formal training in numerical analysis to have a better appreciation of it.

At all times, the idea is that statisticians, to be effective, need not be highly skilled in <u>how</u> to compute but certainly need to be very well versed in <u>what</u> to compute.

References

[1] Barr, A. J. and J. H. Goodnight, [1971] Statistical Analysis System, Student Supply Store, North Carolina State University, Raleigh.

[2] CAROLINA: Multivariate Analysis of Variance Program, Psychometric Laboratory, University of North Carolina.

[3] Dixon, W. J. ed., [1970] Biomedical Computer Programs, X-Series Supplement, U.C.L.A.: University of California Press.

[4] Francis, Ivor, [1973] A comparison of several analysis of variance programs. J. Am. Stat. Assoc., 68 (in press).

[5] Harvey, W. R., [1960] Least-squares analysis of data with unequal subclass numbers. ARS-20-8, Agricultural Research Service, U.S.D.A., Beltsville, Md.

[6] OSIRIS II, OS Users Manual, [1970] Inter-University Consortium for Political Research, The Institute for Social Research, Ann Arbor: University of Michigan.

[7] Searle, S. R., [1971] Linear Models, Wiley, New York.

[8] Searle, S. R., [1972] Using the R()-notation for reductions in sums of squares when fitting linear models. Paper BU-417-M in the Biometrics Unit, Cornell University, Ithaca, N. Y.

[9] Townsend, E. C., [1966] On the problem of obtaining numerical solutions to least squares equations. Paper BU-222-M in the Biometrics Unit, Cornell University, Ithaca, N. Y.

TABLE 1. PARTITIONING SUMS OF SQUARES

Source of Variation	Degrees of Freedom*	Sum of Squares
A. Fitting rows before columns		
Mean	1	$R(\mu)$
Rows, after mean	a-1	$R(r\|\mu) = R(\mu,r) - R(\mu)$
Columns, after mean and rows	b-1	$R(c\|\mu,r) = R(\mu,r,c) - R(\mu,r)$
Interaction, after mean, rows and columns	s-a-b+1	$R(rc\|\mu,r,c) = R(\mu,r,c,rc) - R(\mu,r,c)$
Residual	$n_{..}$-s	$SSE = \Sigma\Sigma\Sigma y^2_{ijk} - R(\mu,r,c,rc)$
Total	$n_{..}$	$\Sigma\Sigma\Sigma y^2_{ijk}$
B. Fitting columns before rows		
Mean	1	$R(\mu)$
Columns, after mean	b-1	$R(c\|\mu) = R(\mu,c) - R(\mu)$
Rows, after mean and columns	a-1	$R(r\|\mu,c) = R(\mu,r,c) - R(\mu,c)$
Interaction, after mean, rows and columns	s-a-b+1	$R(rc\|\mu,r,c) = R(\mu,r,c,rc) - R(\mu,r,c)$
Residual	$n_{..}$-s	$SSE = \Sigma\Sigma\Sigma y^2_{ijk} - R(\mu,r,c,rc)$
Total	$n_{..}$	$\Sigma\Sigma\Sigma y^2_{ijk}$

*a rows, b columns, s filled cells, $n_{..}$ observations

TABLE 2. CALCULATIONS PERFORMED BY 4 COMPUTER PROGRAMS ON UNBALANCED DATA OF A 2-WAY CROSSED CLASSIFICATION OF 2 ROWS AND 5 COLUMNS WITH DATA IN EVERY CELL

($a = 2$, $b = 5$, $s = 10$, $n_{..} = 1310$) See Francis [1973]

Source of variation (as described in program output)	Degrees of Freedom	Sum of squares as calculated and, where appropriate, description used in Table 1	
ANOVA program from SAS			
Rows	1	$R(r\|\mu)$	Rows after mean
Columns	4	$R(c\|\mu)$	Columns after mean
Rows × columns	4	$R(\mu,r,c,rc) - R(\mu,r) - R(\mu,c) + R(\mu)$	
Residual	1300	SSE	Residual
Corrected total	1309	$\Sigma\Sigma\Sigma y_{ijk}^2 - R(\mu)$	
BMDX64 program			
		Numerator sums of squares for F-statistic for testing	
Mean	1	H_1: $\mu + \frac{1}{2}\Sigma r_i + .2\Sigma c_j + .1\Sigma\Sigma(rc)_{ij} = 0$	
Rows	1	H_2: $\frac{1}{2}(r_1 - r_2) + .1[\Sigma_j(rc)_{1j} - \Sigma_j(rc)_{2j}] = 0$	
Columns	4	H_3: $c_j - .2\Sigma c_j + \frac{1}{2}\Sigma_{i=1}^{2}(rc)_{ij} - .1\Sigma\Sigma(rc)_{ij}$ for $j=1\cdots4$	
Interaction	4	$R(rc\|\mu,r,c)$	Interaction, after mean, rows and columns
Error	1300	SSE	Residual
CAROLINA			
Rows	1	$R(r\|\mu)$	Rows after mean
Columns	4	$R(c\|\mu,r)$	Columns after mean and rows
Rows × columns	4	$R(rc\|\mu,r,c)$	Interaction, after mean, rows and columns
Within cells	1300	SSE	Residual
MANOVA program from OSIRIS			
Grand mean	1	$R(\mu)$	Mean
Rows	1	$R(r\|\mu,c)$	Rows, after mean and columns
Columns	4	$R(c\|\mu)$	Columns, after mean
Rows × columns	4	$R(rc\|\mu,r,c)$	Interaction, after mean, rows and columns
ANOVA error	1300	SSE	Residual