Courses in Statistical Computing and Computational Statistics

James E. Gentle

Statisticians spend a large portion of their working days using the computer. In addition to the standard things that almost everyone does, like E-mail and text processing, and the standard uses teachers make of computers in the classroom, statisticians' use of computers includes data analysis with prepackaged software, development of algorithms and software to implement new statistical methods, Monte Carlo simulation to study the performance of statistical procedures, and mathematical analysis using symbolic processing software. Although the use of the computer for things like E-mail and Web surfing or for classroom demonstrations of the central limit theorem are important, the following comments do not address those activities. I should also make a disclaimer about any specific software package I mention in the following. There are many good software packages in each of the relevant areas of application and of the general types of software; my mention of specific packages is not meant to imply that those packages are any better or any worse than other packages. (Also, some names are trademarked, and my use of the name without a mark designation does not imply an acceptable generic use of the name.)

KEY WORDS: Computer arithmetic; Monte Carlo simulation; Programming.

1. HOW MUCH DO STATISTICIANS NEED TO KNOW ABOUT COMPUTING?

Obviously, statisticians' required levels of expertise in computing depend on many things. Some differences in the required/expected levels are related to the educational level of the statistician, whether it is an undergraduate degree, master's degree, or PhD. The area of statistics also influences the extent and the nature of the needed computing expertise. A BS- or MS-level statistician doing data analysis probably can perform the work adequately with only a good working knowledge of the use of a statistical software package such as Minitab, R, SAS, S-Plus, or SPSS. Even at this level and for this kind of work, however, some knowledge of computer arithmetic and of programming is often necessary. Knowledge of computer arithmetic may be necessary in order to understand anomalous numerical results. The ability to program—that is, the ability to implement an algorithm in a computer language—may be necessary if the analyses are nonstandard. The programming required for this may be development of macros or scripts in a high-level language.

The involvement of many BS- and MS-level statisticians in data analysis is primarily to do programming in support of others doing the actual analysis. In this type of work, the requirements include knowledge of computer arithmetic, basic algorithms, and data structures, and programming in specific languages such as Fortran or C, together with subroutine or function libraries such as GSL, IMSL, or Nag.

For research statisticians, and by that term, I mean those who are engaged in developing and extending the theory and methodology of statistics, deeper knowledge of computing is sometimes an absolute requirement, and always at the least, a useful skill. Many of the recent developments in statistics rely heavily on computing, and hence work in those areas requires some understanding of the computational issues. No matter the area, however, there are two ways in which computational skills are useful in the development of statistical theory and methods. A "quick-and-dirty" Monte Carlo study can make a research statistician more efficient in the pursuit of useful statistical methods. Methods can be roughly evaluated quickly, and promising methods explored more thoroughly. Another way in which computing can make the research statistician more efficient is in handling some of the tedious algebraic manipulations in a derivation. Packages for symbolic computations, such as Maple and Mathematica, may take some of the fun out of deriving a result, but they also reduce the errors in going from one line to another. Symbolic computational software can also be useful in quick-and-dirty explorations of ideas. Andrews and Stafford (2000) and Rose and Smith (2002) provided many illustrations of how software for symbolic computations can be useful in work in mathematical statistics. (Both of these books emphasize Mathematica.)

I have mentioned several areas in which computing expertise may be necessary or useful. Of course, my categorization of areas may differ slightly from how someone else might identify the various areas of computing. Because I will refer to these areas later, and to emphasize that I am not considering general "computer literacy" (E-mail, spreadsheets, surfing the Web, text processing, and so on), I list the main areas of computational expertise important for statisticians:

1. Software packages for data manipulation and analysis, and for rapid prototyping (i.e., trying out methods).
2. Software packages for symbolic computations and for rapid prototyping.
3. Computer arithmetic. This means how numerical data are represented in the computer and how operations on real numbers are simulated in the computer.
4. Programming. This means the implementation of a given algorithm in computer software (reliably, efficiently, etc.).
5. Algorithms and methods. This means specific methods, such as Newton's method, the EM algorithm, and so on. (Notice that algorithms can be structured modularly, as in the two examples cited; one might be part of another.) The list of methods can be categorized into some standard areas of numerical analysis, such as numerical linear algebra, quadrature, optimization, transforms, and random number generation.
6. Data structures.

James E. Gentle is Professor, School of Computational Sciences, George Mason University, Fairfax, VA 22030 (E-mail: jgentle@gmu.edu).

© *2004 American Statistical Association* *DOI: 10.1198/0003130042908*

(The numbering in this list is for convenient reference; it does not imply an ordering of importance, or a chronological ordering for learning the areas.)

It should be clear what is included in each of these areas, but perhaps an additional word about number 3, computer arithmetic, is in order. Some of the details of this topic may be rather arcane, but it is important for people who deal with numerical computations to understand that the computer works only with a subset of the real numbers. In the following, as in my teaching, I use the symbol $\mathbb{R}$ to represent the real numbers. This is a special kind of mathematical object, a field. I use the symbol $\mathbb{F}$ to represent the object that the computer uses to simulate $\mathbb{R}$. These are called "floating-point numbers," although, confusingly, in some computer languages the keyword `real` is used to designate these numbers. The object denoted by $\mathbb{F}$ is a finite subset of $\mathbb{R}$ (parameterized by four numbers, b, p, $e_{\min}$, and $e_{\max}$); it is not, however, a field (nor is it a ring or any other object that mathematicians commonly define and study). This object differs from $\mathbb{R}$ in some interesting (and important!) ways. First, aside from its being finite, its elements are not evenly distributed along the real number line. Second, while there are two basic operations in $\mathbb{F}$, just as there are in a field, these two operations (called "addition" and "multiplication") are not always the same as the corresponding two operations in $\mathbb{R}$, although they provide useful simulations of the familiar addition and multiplication. Neither of these operations is associative in $\mathbb{F}$.

While most statisticians do not need to know many of the details of $\mathbb{F}$, they need to be aware of some of the consequences of its properties. There are many convenient storylines that illustrate differences in $\mathbb{F}$ and $\mathbb{R}$. Some of the most striking of these involve evaluation of series. Consider, for example, the harmonic series $\sum_{i=1}^{\infty} i^{-1}$. Everyone knows this series is divergent in $\mathbb{R}$ (although the divergence is very slow; check it out!). In $\mathbb{F}$, because addition is not associative, an expression such as $\sum_{i=1}^{n} i^{-1}$ is not well-defined. (Of course, we know what expression in $\mathbb{R}$ we are attempting to simulate, so we may choose the associations of the operations so as to come as close to its value in $\mathbb{R}$ as possible—in fact, we could say this is one of the main objectives in numerical analysis: to determine the associations of the nonassociative operations of computer addition and multiplication so as to achieve a result that corresponds most closely with the desired result; that is, the result of an interpretation of the expression in $\mathbb{R}$.) Suppose we interpret $\sum_{i=1}^{n} i^{-1}$ as meaning $(\cdots((1 + 1/2) + 1/3)\cdots)$. (This would not be the best way, but let's ignore that now.) Choose a large value for n, or choose ∞ by just letting the computer run. After awhile the sum no longer changes; the sequence has converged. (I am not playing with the word "converge" here; under whatever way "converge" is defined, the harmonic series defined in terms of specified associations of the additions is convergent in $\mathbb{F}$. Under any particular specification of associations, it is convergent; but the associations must be defined because the limit is different for different ones.) Perhaps more amazingly, the series $\sum_{i=1}^{\infty} i$ is convergent in $\mathbb{F}$. (It should be no more nor no less amazing—all we need to remember is one storyline.) Not only do these facts tell us something important about computer arithmetic, they have significant implications for computations we are very likely to perform, such as the approximation of an expression by a finite series. These facts also tell us we have to be careful in interpreting results obtained on the computer. Consideration of these points in a course leads naturally to discussion of methods for extended precision and of software implementing computations in extended precision. (See Gentle 1998, chap. 1, for further discussions of computer arithmetic in both $\mathbb{F}$ and another system $\mathbb{I}$ and how this arithmetic differs from arithmetic in $\mathbb{R}$.)

2. WHERE AND HOW DO STATISTICIANS LEARN ABOUT COMPUTING?

For many statisticians, the aspects of their jobs that involve computing are largely self-taught. This is sometimes the most efficient way to learn, but it often leaves serious gaps in one's knowledge. For those of us engaged in the education of statisticians, there are important questions of what, how, and when to provide systematic training in computing. The answers to these questions have changed over the years, and will continue to change in coming years. The answers also depend to large extent on the level and the orientation of the educational program.

An undergraduate program in statistics, I believe, should include a course in numerical analysis (probably taught in the mathematics department) and at least one application-oriented course in computer science. Some of the statistics courses in the undergraduate program should of course require extensive usage of software for data analysis. As least some assignments in data analysis should carry the student beyond straightforward usage of packaged software.

I will address most of my comments below to "general purpose" master's and PhD programs in statistics, although later I will describe some of the components of a program oriented specifically toward computational statistics.

Training in any area is sometimes best provided in the "just-in-time" mode. This may be the case when that area is supplemental to the student's main focus of study. Thus, often statistical methods are taught in a course in engineering, or in a course in psychology, for example. Likewise, computational methods are sometimes appropriately taught in statistics courses. This is particularly true for teaching of computer software packages for statistical methods. It is convenient to provide an introduction to a statistical analysis package in a course in applied statistics, and then in that and other courses in applied statistics, go deeper into the usage of the software. Also, it is appropriate to introduce and provide examples in the use of symbolic mathematical packages in a course in mathematical statistics. I believe expertise in the first two areas I listed above are best learned in the just-in-time mode, at least initially. Occasionally, perhaps some more in-depth expertise in these packages that requires studying them specifically is required.

In both of these areas (data analysis and mathematical analysis), the availability of the software has changed the way statisticians do their work, and it is no longer necessary (or appropriate) to have students plow through hand calculations for data analysis or tedious derivations for mathematical statistics. This does not mean, of course, that the software does the data analysis or the mathematical analysis; it can only do arithmetic, algebra, and calculus. I believe that courses in applied statistics that do not use industrial-grade data analysis software and courses in mathematical statistics that do not use standard symbolic mathematical software do not provide proper education for graduate students in statistics. Courses that only cover statistical analysis packages or symbolic mathematical packages, however, are generally deficient in intellectual content. I have never believed that courses in "statistical computing" that just cover packages have a place in the statistics curriculum. There is a need, however, for university short courses that cover packages—and such short courses should not carry academic credit but neither should they carry the enormous fees of commercial enterprises. (When I use the term "statistical computing" below, it does not mean using data analysis packages.)

Before I move on to other areas, I will make just a brief statement to clarify my position (although the issue of the use of software for symbolic computations perhaps warrants more extensive discussion). I believe solid mathematical training is an absolute must for a statistician. I also believe, however, that software for symbolic computations can facilitate the acquisition of a grounding in the mathematics. I do not believe that the use of such software or the use of the computer in such applications poses any greater danger to understanding than the use of a computer or calculator to perform the computations for a t test poses.

What about the other areas of computing, such as numerical analysis, algorithm development, and simulation or other computationally intensive methods? Where should these aspects of a well-rounded training in statistics be taught? Some of the statistical procedures, such as Monte Carlo tests, Markov chain Monte Carlo (MCMC) methods, and bootstrapping techniques fit naturally into various applied and theoretical statistics courses, and should at least be introduced in those courses. There is, however, easily enough material concerning these procedures, which fall into an area called *computational statistics*, to justify a separate course. Such a course could provide a solid basis for such things as large-scale exploratory analysis and computational inference in many areas of statistical methodology. Much of the work in a course in computational statistics can be carried out in an interactive array language, such as Matlab or S-Plus. Although some ability to use a programming language like Fortran or C/C++ would be useful, it is not clear whether an ability to use such a language should be a prerequisite for the course in computational statistics.

The traditional topics of *statistical computing*, which include numerical analysis, algorithms, and computer graphics with applications in statistics, are difficult to incorporate into regular courses in applied or theoretical statistics. There is a place in the statistics curriculum for a course in "computational methods for statistics," although such a course would not be taken by all students. This course would cover material such as in Lange (1999) or Monahan (2001).

A course in statistical computing should require an ability to program in a language like Fortran 2000 or C/C++, and should involve careful implementation of algorithms in one of those languages as well as use of existing libraries through those languages. Principles of computer arithmetic, error analysis and control, and convergence of sequences in finite precision should be covered thoroughly. Methods in numerical linear algebra, optimization, both discrete and continuous, random number generation, function approximation, and numerical quadrature should receive careful consideration.

The important facts and known results in statistical computing have expanded greatly over the years; nevertheless, when looking back over an old book on statistical computing published in 1980, I am sometimes amazed at how little the important topics and the basic principles have changed. This indicates that while the details are ephemeral, the fundamentals are essential. Understanding the fundamentals helps in mastering details and in keeping them in their proper place.

3. STATISTICAL COMPUTING AND COMPUTATIONAL STATISTICS AT GEORGE MASON UNIVERSITY

This section briefly describes how computing is integrated into some graduate curricula in statistics with which I am familiar. My intent is not to promote these programs. They are convenient examples because they exhibit a range of combinations of statistics and computing.

At George Mason University, there are three degree programs in statistics, two PhD degrees and one MS degree. One of the PhD degrees and the MS degree do not require any particular level of computing skills, although some of the required courses make heavy use of SAS, so all students in those two programs learn at least some SAS. Other statistics courses in the program use R or S-Plus, as well as a few other statistics packages, and two courses in the PhD program use Maple or Mathematica to a certain extent. There are a number of short courses on various software packages. These are usually taught in all-day weekend sessions. They carry one hour of academic credit, but it is not applicable toward a statistics degree.

The other PhD degree at George Mason University, which is a PhD in computational sciences with an emphasis in statistics, has a strong computational component. This degree program, which is rather unusual for statistics, is the one with which I am more closely affiliated. We do not have a course in statistical computing per se, but rather, there is a required course in numerical methods that uses Matlab and a second required course in computational science that requires either C or Fortran. These two courses together cover topics in the last four areas I listed above, but the courses do not have any particular emphasis on statistics. (They are taken by all students in the computational sciences program, which includes other areas of concentration, such as computational physics, computational biology, and climate dynamics.) One interesting aspect of one of these courses is the requirement of a team project that involves development of a software system to address some specific scientific problem. The teams consist of from three to five students each. Team projects are also required in some other courses in the program.

Both of the statistics PhD degree programs at George Mason require a two-course sequence in mathematical statistics at the "baby measure theory" level. The current text is Shao (2003). Mathematica and Maple are used in these courses to illustrate their power in mechanical derivations. R or S-Plus is also used in these courses in mathematical statistics for such things as MCMC, for studying approaches of finite sample to asymptotic behaviors, and for comparing inferential procedures. (The textbook and other texts in this area do not use or discuss any software.) The 2003–2004 year is the first year that Shao has been used. The current plan is to cover the book in the two semesters.) There are two statistical mathematics courses and an advanced-calculus level mathematical statistics course that build up to this mathematical statistics course sequence, and the individual students' backgrounds determine which, if any, of these courses to take. There are no specific courses in applied statistics that are required, but most students take at least three or four in traditional areas such as regression, categorical analysis, nonparametrics, and so on. There is also a course in geometrical methods in statistics. Although the geometrical topics have broad applicability, an important area of application is in statistical graphics. All statistics students in the computational sciences program are encouraged, but not required, to take additional courses in numerical analysis and visualization.

Most statistics students in the computational sciences PhD program and some in the MS program or the other PhD program take a course in computational statistics. This course, which uses Gentle (2002), covers a number of computationally intensive methods in statistics. Some of these methods use simulations and some involve interactive iterations. An area of application that is emphasized is the discovery of structure in multidimensional data. Although the course is about *statistical methods*, the numerical algorithms underlying the methods are considered in more detail than is the usual case in other courses about statistical methods. The course uses R or S-Plus. Although the course does not require Fortran or C, the advantages of such languages are discussed and most students actually do use one or the other. The course requires a term project in which the student replicates and extends a published Monte Carlo study in statistics. The students are required to build Web sites where they put the results of their Monte Carlo studies and other reports on their projects. (The requirement to build a Web site first became a part of this course in 1995, which was before the browser wars. Building Web pages is not an area of computer usage that I have discussed in this article. Neither is use of LaTeX, but I think such aspects of "computer literacy" are useful for statisticians, and course requirements should include acquisition of these skills whenever and wherever practical.)

4. FURTHER COMMENTS AND SUMMARY

Looking back over my comments above, I realize I did not say much about graphics. I think I do not need to. The use of computer graphics is an integral part of almost any data analysis. In the work of the data analyst, graphical capabilities are available in the same software that I categorized in "area 1," or perhaps "area 2." At the programming level for graphics (i.e., not just issuing a "plot" command in a software package, but what is included in "area 4" above), there are, of course, many issues. I have not discussed any specific programming issues here; there are many, and they cover a wide range of applications. Algorithms for graphics, such as projections and rotations, would be included in "area 5" that I listed above. Again, I have not discussed any of these specific algorithms in this article.

I have advocated that use of computing be ubiquitous in the statistics curriculum; data analysis software in applied statistics courses, and flexible programming packages and symbolic computation software in mathematical statistics courses. The training in this kind of software is just-in-time; almost all training in software should be. There are other aspects of computing that are relevant for statisticians, however. These are what I have called "statistical computing" (and to me that term does not mean use of statistical packages—it includes the last four of the areas I listed above).

The educational program I am affiliated with requires two courses covering statistical computing and other topics in numerical analysis, and offers an additional course in computational statistics. This is rather extreme, but I believe that all graduate programs in statistics should offer at least one course that covers statistical computing, and that this course should require preliminary knowledge of Fortran or C/C++ and should develop competence in numerical computations using one or both of those standard languages. I believe that such a course should be taken by a substantial proportion of both MS and PhD students.

Most statisticians will continue to learn much of what they know about computing through trial and error and in self-study. This is an effective learning method if, as students, they are given a solid base in the fundamentals of scientific computing, that is, in the last four of the six areas listed above, and if they are required to use the computer throughout their course work.

I thank the editor and three reviewers for their comments, which certainly prompted me to think more about this topic—and to write more about it than I initially intended!

[Received July 2003. Revised August 2003.]

REFERENCES

Andrews, D. F., and Stafford, J. E. H. (2000), *Symbolic Computation for Statistical Inference*, Oxford, UK: Oxford University Press.

Gentle, J. E. (1998), *Numerical Linear Applications for Applications in Statistics*, New York: Springer.

——— (2002), *Elements of Computational Statistics*, New York: Springer.

Lange, K. (1999), *Numerical Analysis for Statisticians*, New York: Springer.

Monahan, J. F. (2001), *Numerical Methods of Statistics*, Cambridge, UK: Cambridge University Press.

Rose, C., and Smith, M. D. (2002), *Mathematical Statistics with Mathematica*, New York: Springer.

Shao, J. (2003), *Mathematical Statistics* (2nd ed.), New York: Springer.