THE FUTURE OF STATISTICAL SOFTWARE

Proceedings of a Forum

National Research Council

The Future of Statistical Software: Proceedings of a Forum

Panel on Guidelines for Statistical Software, National Research Council

ISBN: 0-309-58377-2, 100 pages, 8.5 x 11, (1991)

THE FUTURE OF STATISTICAL SOFTWARE

Proceedings of a Forum

Panel on Guidelines for Statistical Software
Committee on Applied and Theoretical Statistics
Board on Mathematical Sciences
Commission on Physical Sciences, Mathematics, and Applications
National Research Council

National Academy Press
Washington, D.C. 1991

NOTICE: The project that is the subject of this report was approved by the Governing Board of the National Research Council, whose members are drawn from the councils of the National Academy of Sciences, the National Academy of Engineering, and the Institute of Medicine.

The National Academy of Sciences is a private, nonprofit, self-perpetuating society of distinguished scholars engaged in scientific and engineering research, dedicated to the furtherance of science and technology and to their use for the general welfare. Upon the authority of the charter granted to it by the Congress in 1863, the Academy has a mandate that requires it to advise the federal government on scientific and technical matters. Dr. Frank Press is president of the National Academy of Sciences.

The National Academy of Engineering was established in 1964, under the charter of the National Academy of Sciences, as a parallel organization of outstanding engineers. It is autonomous in its administration and in the selection of its members, sharing with the National Academy of Sciences the responsibility for advising the federal government. The National Academy of Engineering also sponsors engineering programs aimed at meeting national needs, encourages education and research, and recognizes the superior achievement of engineers. Dr. Robert M. White is president of the National Academy of Engineering.

The Institute of Medicine was established in 1970 by the National Academy of Sciences to secure the services of eminent members of appropriate professions in the examination of policy matters pertaining to the health of the public. The Institute acts under the responsibility given to the National Academy of Sciences by its congressional charter to be an adviser to the federal government and, upon its own initiative, to identify issues of medical care, research, and education. Dr. Stuart Bondurant is acting president of the Institute of Medicine.

The National Research Council was organized by the National Academy of Sciences in 1916 to associate the broad community of science and technology with the Academy's purposes of furthering knowledge and advising the federal government. Functioning in accordance with general policies determined by the Academy, the Council has become the principal operating agency of both the National Academy of Sciences and the National Academy of Engineering in providing services to the government, the public, and the scientific and engineering communities. The Council is administered jointly by both Academies and the Institute of Medicine. Dr. Frank Press and Dr. Robert M. White are chairman and vice chairman, respectively, of the National Research Council.

The National Research Council established the Board on Mathematical Sciences in 1984. The objectives of the Board are to maintain awareness and active concern for the health of the mathematical sciences and to serve as the focal point in the National Research Council for issues connected with the mathematical sciences. In addition, the Board is designed to conduct studies for federal agencies and maintain liaison with the mathematical sciences communities and academia, professional societies, and industry.

Support for this project was provided by the National Science Foundation and the National Institute of Environmental Health Sciences.

Library of Congress Catalog Card No. 91-66874

International Standard Book Number 0-309-04599-1

Additional copies of this report are available from:
National Academy Press
2101 Constitution Avenue, N.W.
Washington, D.C. 20418

S467
Printed in the United States of America

PANEL ON GUIDELINES FOR STATISTICAL SOFTWARE

WILLIAM F. EDDY, Carnegie Mellon University, *Chair*
SALLY E. HOWE, National Institute of Standards and Technology
BARBARA F. RYAN, Minitab, Inc.
ROBERT F. TEITEL, Abt Associates, Inc.
FORREST W. youNG, University of North Carolina
JOHN R. TUCKER, Staff Officer

COMMITTEE ON APPLIED AND THEORETICAL STATISTICS

WILLIAM F. EDDY, Carnegie Mellon University, *Chair*
YVONNE BISHOP, U.S. Department of Energy
DONALD P. GAVER, Naval Postgraduate School
PREM K. GOEL, Ohio State University
DOUGLAS M. HAWKINS, University of Minnesota
DAVID G. HOEL, National Institute of Environmental Health Sciences
JON KETTENRING, Bellcore
CARL N. MORRIS, Harvard University
KARL E. PEACE, Biopharmaceutical Research Consultants
JAYARAM SETHURAMAN, Florida State University
JOHN R. TUCKER, Staff Officer

BOARD ON MATHEMATICAL SCIENCES

Shmuel Winograd, IBM T.J. Watson Research Center, *Chair*
Ronald Douglas, State University of New York-Stony Brook, *Vice-Chair*
Lawrence D. Brown, Cornell University
Sun-Yung A. Chang, University of California at Los Angeles
Joel E. Cohen, Rockefeller University
Avner Friedman, University of Minnesota
John F. Geweke, University of Minnesota
James Glimm, State University of New York-Stony Brook
Phillip A. Griffiths, Institute for Advanced Study
Diane Lambert, AT&T Bell Laboratories
Gerald J. Lieberman, Stanford University
Ronald F. Peierls, Brookhaven National Laboratory
Jerome Sacks, National Institute of Statistical Sciences

Ex Officio Member

William F. Eddy, Carnegie Mellon University Chair, Committee on Applied and Theoretical Statistics

Staff

John E. Lavery, Director
Jo Neville, Administrative Secretary
Ruth E. O'Brien, Staff Associate
Hans Oser, Staff Officer
John R. Tucker, Staff Officer
James A. Voytuk, Senior Staff Officer
Scott T. Weidman, Senior Staff Officer
Barbara Wright, Administrative Assistant

COMMISSION ON PHYSICAL SCIENCES, MATHEMATICS, AND APPLICATIONS

Preface

The Panel on Guidelines for Statistical Software was organized in 1990 by the National Research Council's Committee on Applied and Theoretical Statistics for the purpose of

- documenting, assessing, and prioritizing problem areas regarding the quality and reliability of statistical software;
- presenting to producers and users prototype guidelines in high-priority areas for the evaluation (based on established statistical principles) of statistical software packages; and
- making recommendations in the form of a plan for further discussion, research, testing, and implementation of guidelines involving the statistical computing, user, and producer communities.

The findings of the panel will be presented in a future report and at meetings of concerned groups, including professional societies, to stimulate such further work. The panel's guidelines will be accompanied by benchmark test data or descriptive material from which such data can be constructed. The panel will not endorse or censure specific statistical software products, but rather will offer general guidelines and broad objectives and evaluation criteria useful to statistical software users and developers, and designed to facilitate and focus further work on the subject.

On February 22, 1991, the panel held a public forum, “The Future of Statistical Software,” so as to gather material for its deliberations from a wide range of statistical scientists from academe, industry, and government. These proceedings have been compiled to document that input. However, the opinions expressed in this volume are those of the speakers or discussants and do not necessarily represent the views of the Panel on Guidelines for Statistical Software or of the National Research Council.

Contents

Morning Session Opening Remarks

Barbara Ryan
Minitab, Inc.

The question faced by the Panel on Guidelines for Statistical Software is how to provide effective guidance to people involved with statistical software. The audience we are trying to address consists of people who have to review software for one reason or another--for their own personal use, for use within a company or a university, or perhaps for presentation in a trade publication, professional journal, or popular magazine. We want to address people who are trying to understand statistical software, recognize good software, and know how to look at software to assess whether it will meet their needs.

Another group we expect the report to influence is the vendors who produce software. We want the vendors to produce the best possible software for the users. In the late 1970s, when the American Statistical Association got involved in evaluating statistical software, there was some concern from the vendor community about how that was done. But it had one major effect: it improved the quality of statistical software. Very few vendors wanted to appear to be uninformed, and so they made sure their software met some of the criteria that the statisticians considered very important. Our panel hopes that its future report, to which this forum will provide input, will also influence vendors to improve their products in ways that will help people to do statistical work well.

The panel has identified three qualities of statistical software as important: richness, exactness, and guidance. Although others are also important, these were selected as a means to structure the panel's efforts. Richness will be emphasized this morning and guidance will be featured this afternoon; exactness will arise throughout the day as appropriate. By richness, we mean: What does the software do, what type of analysis? Does it have depth? Does it have breadth? Those qualities are not necessarily always good things. For some people, too much depth and too much breadth are confusing. So richness is not automatically a good quality, but it is an important quality.

Richness for the One-Way ANOVA Layout

Keith E. Muller[1]
University of North Carolina, Chapel Hill

STATEMENT OF THE PROBLEM

Rationale of Approach

The Panel on Guidelines for Statistical Software has organized its examination of statistical software around the concepts of exactness, richness, and guidance. The one-way layout presents a ubiquitous task, and hence a stimulating prototype for devising guidelines. This presentation covers richness in the one-way layout. In the section below titled “Richness Dimensions,” a number of dimensions are proposed as the basis for evaluating the richness of statistical software. Each of these is examined through the same three steps: (1) a brief definition is given, (2) richness is detailed for the one-way layout in the dimension under discussion, and (3) the specific description is followed by a discussion of general principles.

Creating general and useful principles will require continuing interactive discussion among many interested observers; the comments reported here are intended to stimulate further discussion. Such discussion will help avoid embedding statistical fads in guidelines. Specific evaluations of software unavoidably depend on the philosophy of the evaluator. Hence evaluation guidelines must be ecumenical in coverage, yet able to be narrowed for a particular task. Undoubtedly the proposed structure and topics are not definitive. Despite that, the present author does believe that any alternate approach must cover all of the issues considered here.

Problem Boundaries.

A number of terms must first be defined, at least loosely. **Statistical software** will be taken to be any collection of computer programs and associated information intended to directly aid the production of any sort of statistical analysis. Note that this definition includes software that itself may not produce statistical analysis. For example, a program

[1] The author gratefully acknowledges the stimulation of summaries of meetings of the Panel on Guidelines for Statistical Software. The basic approach in which this discussion is embedded was determined by those deliberations. In addition, some specific examples are taken from the summaries.

used to create a file used as input to an analysis program may fall under this definition. **Exactness** consists of a choice of an acceptable algorithm and the accurate implementation of the algorithm. For example, asymptotic formulae for variance estimation or p-value calculations should not be used in small samples whenever more precise calculations are available. The algorithm should either tolerate stressful data configurations, or detect them and gracefully report an inability to proceed. **Guidance** consists of the help provided by the structure and features of the software in conducting a correct and effective analysis. This includes assistance in choosing appropriate instructions, as well as assistance in deciding whether a particular approach is valid. **Richness** consists of how fully the software can do the analysis. The term **coverage** may be preferred by some. Throughout, **the user** will be taken to refer to the person executing the software.

The definitions of exactness, guidance, and richness all overlap somewhat. Distinctions between guidance and richness are particularly important in defining the boundaries of the present discussion. At one extreme, providing maximal guidance leads to an expert-system approach. In that case, richness ceases to exist as a separate property, being determined by the range of tolerated inputs and guidance accuracy. At the other extreme, providing minimal guidance leads to documenting features of the software and nothing else. An intermediate amount of guidance would be the automatic inclusion of diagnostics concerning the assumptions of the method of analysis. In contrast, richness describes the availability and convenience of creating such diagnostics. In considering guidance, the validity of the analysis approach for the data at hand is always in question. In contrast, when discussing richness it will be assumed throughout that using the software for the data at hand is a valid endeavor.

The one-way layout in analysis of variance (ANOVA) is used as the basic example. For the sake of brevity, familiarity with traditional approaches will be assumed. Kirk [1982] provided a comprehensive treatment of a large range of ANOVA models. In one-way ANOVA, a continuous response variable is examined to assess whether it is related to a categorical predictor variable. The predictor values may be strictly nominal, ordered categories, or interval scale values. Typically two to ten distinct predictor values are present. A number of regularity conditions must be assumed for both non-parametric as well as parametric methods in order to ensure the validity of statistical analysis. These may be loosely grouped into assumptions concerning (1) existence, (2) independence, (3) model, and (4) distribution. Traditional parametric fixed-effect ANOVA requires independent and identically distributed (i.i.d.) Gaussian scores within each category and categories that differ only by expected value.

RICHNESS DIMENSIONS

1. Epistemological Goals

Definition The **epistemological goal(s)** for an analysis consists of the standards by which truth is judged, the standards by which decisions are made, and the purpose of the analysis.

One-Way Examples Example 1: the user wishes to evaluate whether red, green, or blue text can be read most rapidly on a computer screen. Example 2: the user wishes to estimate the location and scale of the amount of hypoxic cells in biopsies taken from a number of different types of cancerous tumors. Example 3: the user wishes to discover whether a new drug regimen maintains kidney function better than current practice. Example 4: the user wishes to decide whether a single time-release capsule has the same therapeutic effect as three smaller doses delivered once every eight hours. Example 5: the user wishes to examine the effect on hypertension of the dietary presence of one foodstuff from a large list.

Comments Epistemology is the study of the basis and limits of knowledge. Every user approaches an analysis task with a particular philosophy, whether explicit or not (even to the user). One's philosophy determines how to decide what is true, and what is worth deciding. Statisticians' philosophies vary substantially on a number of dimensions. For example, one may prefer a Frequentist, Bayesian, or Decision-Theoretic approach. More generally, statisticians describe estimation and inference as separate activities. Furthermore, one may be adamant about distinguishing between confirmatory and exploratory analysis, or actively opposed to the distinction. One may favor a parametric, robust, or non-parametric strategy.

General Principles Users have widely varying philosophies and purposes in using software. Software authors and reviewers should report the epistemology upon which they based their work. It should be emphasized that this does not demand statistical ecumenism. Instead such clarification will allow users of software and readers of software reviews to recognize whether the bases for evaluation are shared.

2. Methods

Definition **Methods** consist of all the techniques and algorithms that can be implemented within particular software.

One-Way Examples First consider the traditional parametric model assuming Gaussian errors, from a Frequentist perspective. Applications of estimation methods include estimation of primary parameters, such as cell means in a cell mean coding, and estimation of linear combinations of primary parameters (which are secondary parameters, and contrasts), such as mean differences or trend contrasts. Testing methods include the general linear hypothesis test and the many kinds of multiple comparison methods available [Miller, 1981]. Note that estimation of parameter variances and the specification of confidence intervals allow an (embedded) alternate approach for scalar parameters.

A Bayesian perspective requires implementing different methods for some of the same tasks, but also methods for tasks not listed here (such as posterior density estimation).

One may move away from the traditional model in many directions. Concerns about the robustness of the traditional model are implicit in all such moves. Examples include methods for evaluating the validity of assumptions [regression diagnostics; see Belsley et al., 1980], semi-parametric modifications such as down-weighting extreme values, and rank-transformation [Puri and Sen, 1985].

Comments The effectiveness of the implementation depends on how conveniently the interface links the statistical software with other software and the user. Many of the methods can be implemented with graphical displays. Diagnostics and multiple comparison methods are especially appropriate. Interfaces in general are discussed in the sections below titled “Inputs” and “Outputs.”

Richness of methods unavoidably intertwines with guidance and exactness. Consider the following applications, which were not given as examples: weighted least squares, both exact and approximate; estimating nonlinear functions of parameters and associated confidence intervals; random effects model estimation and testing, all subjects tested in all conditions (repeated measures); power calculation; and the analysis of dispersion. Should these applications be covered by one-way layout software, nominally centered on ANOVA? This issue will be addressed in the discussion of structure below.

General Principles Methods should be judged on (1) exactness, (2) breadth, and (3) interfaces. Exactness includes numerical accuracy, numerical efficiency, and optimality of technique (for example, never using asymptotic approximations in small samples when exact calculations are practical). Breadth can be evaluated only after having specified the target coverage desired. User interfaces should include communication from the user to the software (control) and feedback to the user about, for instance, any branches taken by the software.

3. Inputs

Definition **Inputs** are the statistical assumptions, data files, and control files. “Files” include signals from user interfaces such as keys, light-pens, joysticks, and mouse keys, as well as information organized on computer-readable media.

One-Way Examples The assumptions required for least squares optimality of the traditional fixed-effect model may be summarized as homoscedasticity of variance, existence of finite second moments, independence of observations, and model linearity (trivially met in this case). Assuming Gaussian errors leads to maximum likelihood optimality of the least squares calculations, and allows closed-form calculation of optimal likelihood ratio tests. One may wish to choose the parameter structure of the model, such as cell mean. One may also wish to choose particular contrasts to estimate, such as trends. Data are often stored in text format and also in formats created by various proprietary data management or data analysis software packages.

Comments Assumptions must be treated somewhere, and so they are included here, although they may merit separate treatment. Attention to satisfying assumptions contributes strongly to good data analysis.

The evaluation of software can be dominated by the quality of the input interface with the user. For example, software that depends on field-dependent references to variables may trip up even a sophisticated user. Computer designers and scientists have focused on the control files. Many convenient data entry packages are available, although unknown to most users. Software structure has also been recognized as an important determiner of input adequacy. Relatively little effort has been expended on data file importing. This is often difficult even across software, within a platform. Crossing platforms and software can be extremely difficult.

The great majority of data analysis methods are only defined for rectangular arrays of observations, allowing perhaps some missing data or other irregularities. Notable exceptions center on the work in classification and taxonomy. Database management software necessarily supports a great variety of data arrangements and relationship patterns. The conversion process may be inconvenient. Furthermore, care must be taken to produce an analysis file that allows the questions of interest to be addressed. These same comments are relevant also to the discussion of structure below, and to the discussion of guidance.

General Principles Considerations about the validity of assumptions should be embedded in input requirements of any statistical software. Accepting a broad range of inputs may substantially enhance the utility of the software. Users should be able to control, if they wish, the choice of algorithms, such as coding schemes. These desires must be balanced with the critical needs of efficiency and simplicity of use. Input correctness should be checked extensively. Error messages should be self-contained and indicate possible corrections. Clever structuring may prevent many errors from occurring.

4. Outputs

Definition **Outputs** are the collection of sensory displays provided immediately to the user, as well as any non-transitory record stored on paper, or on digital or other media.

One-Way Examples For the traditional Gaussian errors approach, example analysis outputs include an ANOVA source table, parameter estimates and associated variance estimates, and plots of means. Example diagnostic outputs include tests of homogeneity or normality, and box-plots for each cell. All can be produced on a character printer or a graphics device, or stored on digital media.

Comments Most current presentations of statistics could be enhanced by the addition of carefully chosen graphics displays. Currently, the depth of one's employer's pockets substantially determines the ability to render graphics with adequate resolution. Software tends to be very platform-specific, with only the beginnings of graphics interchange standards. For static displays, some optimism may be appropriate in that systems of modest cost are rapidly approaching the limits of the human visual system. Current dynamic displays are hardware-and cost-limited.

A user may wish to validate a particular invocation of software, conduct an analysis not provided as an option, document properly, “check-point” a procedure, or interface with other software. All require the ability to direct any output to digital media. Conscientious data analysis, coupled with the power of computers, invites such approaches. Digital file output enables extensibility, both within and between packages. Many different platforms and types of software are used for data analysis. Such diversity provides as many disadvantages as advantages.

General Principles The user may wish to direct any calculated result, including graphics and other displays, to a storage file for use at another time, on another computer, or in a different place. Such files should be easily portable and self-documenting. Users will strongly prefer software that supports standard interchange formats: science operates on the basis of sharing information. Software vendors, however, depend on secrecy of information to allow them to stay in business and make a profit. Vendors and scientists will need to cooperate and be creative to meet the user preferences for openness and ease of interchange while simultaneously protecting legitimate business interests.

5. Options

Definition **Options** are the alternative epistemological goals, methods, inputs, outputs, structures, internal paths, external paths, and documentation that may be invoked in addition to or in lieu of the default choices.

One-Way Examples Options that might be desired in the traditional one-way ANOVA include choice of coding scheme, deletion of the intercept from the model, creation of confidence intervals for mean differences, specification of the categorical variable as a random effect, and diagnostics for homogeneity of distribution across groups.

Comments The goals, structure, and audience of software mostly determine what options can and should be available. Given a particular goal and audience, the structure of options will reward or frustrate, depending upon the skills of the author. The availability of many options empowers the user who is able to control them efficiently. However, the same breadth may overwhelm and mislead less knowledgeable users.

General Principles The choice of options should be based on software goals. The choice of defaults should be based on the audience. Layering options may be used to resolve the conflict between the needs of the novice and the sophisticate. Certain desirable options are discussed in other sections of this paper, including “Inputs” and “Outputs” above.

6. Structure

Definition The **structure** of statistical software consists of the module definition and branching schemes employed in design and execution.

One-Way Examples Currently, most ANOVA software reports source tables by default and multiple comparisons at the user's specific request. For software with diagnostics available, few packages report them by default.

Comments The structure of a program embodies the designer's philosophy about the analysis goals and guidance appropriate for the expected audiences. Diversity of audiences appears to require layers of options. Limited structures constrain richness, while complex structures reduce efficiency and user difficulties. Designers must understand the conceptual, analytical, and numerical tasks in any statistical method in order to produce appropriate and efficient structure.

The description of the example being considered as “the one-way layout” may be contrasted with the description “one-way fixed-effect ANOVA.” The former describes the format of a collection of data values, while the latter fully specifies a model and associated analysis, including the required data format. The two descriptions correspond to radically different structures and labels for the elements of the structure. In turn, documentation and the audiences that can be served are strongly affected.

General Principles Ideal statistical software would provide seamless modularity. Such software would always present an efficient and simple-to-use interface with the user and other software. The methods and use of the product would derive from a consistent set of concepts, not a collection of tricks and gyrations. Good structure incorporates principles of perception and learning from the behavioral sciences and principles of numerical analysis and program design from the computational sciences.

7. Internal Paths.

Definition The **internal paths** of statistical software are the branches that may be followed due either to the dictates of the algorithm and the data or to choices made by the user.

One-Way Examples In a one-way ANOVA, one may wish to evaluate diagnostics on residuals before choosing to examine any output involved with inference. Much software always produces a source table. Software may allow the user to specify conditional execution of step-down tests, such as trend tests. Depending upon the coding scheme, a program may use an orthogonalization scheme or a simple elimination algorithm.

Comments Sophisticated users may desire control of branches internal to a single module. Such control may allow the naive user to bungle an analysis. Such control may allow the sophisticated user to tune the performance of the program to the application.

General Principles Sophisticated users desire control of all internal paths. Access to such control must be guarded with appropriate warning information. This recommended approach should be evaluated in light of the guidance standards and audiences.

8. External Paths

Definition The **external paths** of statistical software are the branches that may be followed by the user and data into, out of, and back into the software.

One-Way Examples The user may wish to conduct diagnostic analysis on alternate transformations of the data. The results may then be input to a summary analysis. In turn the user then needs to implement the preferred analysis.

Comments One rarely uses software in isolation. Convenient and efficient paths into and out of the software greatly facilitate quality data analysis.

Statisticians will continue to create better methods that are computationally intensive for whatever computing machinery becomes available. The ability to check-point such calculations would be advantageous. For example, some current iterative programs require manual intervention even to avoid most of the iterative calculations on a subsequent invocation.

General Principles Interfaces (both input and output) with other modules in the software should be provided. Convenient abilities to temporarily suspend execution, check-point, and conduct analysis recursively may substantially enhance the utility and convenience of the software.

9. Documentation

Definition **Documentation** consists of all information intended to aid the use of statistical software.

One-Way Examples Traditionally paperback books designed to be reference manuals have been the primary documentation available to the user. Such manuals focus on describing the vocabulary and grammar of the language needed to control such things as the choice of response variable, the choice of the categorical predictor, labeling, and analysis options.

Comments One plausible standard for perfection of software would be the need for no documentation. Extensive documentation may reflect either richness (and guidance) or awkwardness and unnecessary complexity.

Many types of documentation may be provided. Software-focused information usually resides in manuals for language reference, system management, and description of algorithms. Tutorials, collections of examples, and statistics texts focused on a particular piece of software assist the training of users in both the software and the statistical methods. User groups and toll-free telephone numbers may be supported, reflecting either the vendor's sensitivity or defensiveness.

Many formats may be used for documentation. The paperback book has been challenged by on-line documentation, at least for truly interactive software. The recent successful marketing of electronic books in Japan provides yet another step toward the handling of all information digitally. Arguments over ring-bound versus spine-bound versus on-line manuals will eventually also involve new formats.

Documentation can make good software look bad and bad software look good. Effective documentation requires the same attention to structure as do the algorithms. The top-most layer of documentation may be thought of as metadocumentation, documentation of documentation. Proper layering and branching can help the user.

Surprisingly, many existing manuals do not provide examples of actual code in all cases. In describing a particular programming statement, or a sequence of clicks or keys, a template description may not suffice. An appendix of formulas and a list of algorithmic steps may greatly aid understanding and using software. Professional standards may demand verifying the acceptability of the techniques relative to the data at hand.

General Principles Software may be documented in many formats. Metadocumentation can help the user take full advantage of documentation. Documentation should be structured, based on the same principles as the software. A reference manual, although always necessary for the sophisticated user, does not, by itself, provide adequate documentation. Algorithms and formulas should be detailed. Truly complete examples should be included. The extent of tutorials and statistical information embedded in documentation depends on the guidance goals and the audiences. Sophisticated users may dislike the presence of statistical information and advice in documentation, while novice users often crave it. System implementation documentation should be available.

10. Audiences

Definition The **audiences** for statistical software are the user groups that may be distinguished from each other because of their different approaches to and uses of the software.

One-Way Examples A piece of software may be used by a person with a doctorate in statistics and by a person with literally no statistical training. The same software may be used by a person with a master's degree in computer science and by a high school student frightened by computers.

Comments For statistical software, user sophistication varies in (1) knowledge of statistical theory, (2) knowledge of computing theory, (3) proficiency with data analysis, (4) facility with computing, and (5) experience with research data management. Natural language sophistication and physiological limitations, such as color blindness or response speed, may be relevant in some applications.

General Principles Designers, programmers, documenters, and reviewers of statistical software need to be explicitly and continuously sensitive to the audiences of interest. Software and documentation structure should reflect the often disparate needs of the novice and the sophisticate.

WHAT NEXT?

Step Back

The basic position presented above is that richness varies on a large number of continuous, correlated dimensions. It is also argued that the creation and evaluation of statistical software should occur with respect to explicit target goals and target audiences. This suggests that careful specification of the task, based on exactness, guidance, and richness, should always be the first step.

Jump In (Continuous Involvement)

The process of creating and evaluating software improves with effective interaction between producers and consumers. Such continuous involvement will lead to the creation of good products. However, the products will not be completely successful unless the decision makers and the “fashion” leaders of the user community can be educated about general and specific guidelines for good statistical software. Therefore, even after the Panel on Guidelines for Statistical Software releases its final recommendations in a future report, it will be necessary for those of us interested in guidelines to stay involved to make the user community aware of the panel's guidelines and encourage their acceptance.

REFERENCES

Belsley, D.A., E. Kuh, and R.E. Welsch, 1980, *Regression Diagnostics,* John Wiley & Sons, New York.
Kirk, R.E., 1982, *Experimental Design,* Brooks/Cole, Belmont, Calif.
Miller, R.G., 1981, *Simultaneous Statistical Inference,* Springer-Verlag, New York.
Puri, M.L., and P.K. Sen, 1985, *Nonparametric Methods in General Linear Models,* John Wiley & Sons, New York.

Serendipitous Data and Future Statistical Software.

Paul F. Velleman
Cornell University

Modern statistics is usually considered to have first appeared around the beginning of this century in a form that resembled its scientific father more than its mathematical mother.

As statistics matured during the middle of this century, statisticians developed mathematical foundations for much of modern statistics. Along the way they developed methods that were optimal in some sense. For example, maximum likelihood statistics--which, when applied under the most commonly used assumptions, include the most commonly used methods--have many properties that make them the best choice when certain assumptions about the model and the data are true. Data from suitably designed and suitably randomized studies were the focus of data analysis based on these insights.

However, much real-world data is serendipitous. By that I mean that it arises not from designed experiments with planned factors and randomization, nor from sample surveys, but rather as a by-product of other activities. Serendipitous data reside in databases, spreadsheets, and accounting records throughout business and industry. They are published by government and trade organizations. They arise as a by-product of designed studies when unexpected patterns appear but cannot be formally investigated because they are not part of the original protocol. Serendipitous data forms the basis for some sciences and social sciences because it is, for the moment, the best we can do.

Concern with optimal properties of statistics follows the traditions of mathematics in which elegant results are valued for their internal consistency and completeness but need not relate directly to the real world. By contrast, a scientist would reject even the most elegant theory for the small sin of failing to describe the observed world. Traditional statistics are often inappropriate for serendipitous data because we cannot reasonably make the assumptions they require. Much of the technology of classical statistics and of traditional statistics software is designed to analyze data from designed studies. This is a vital function, but it is not sufficient for the future.

In his landmark paper, “The Future of Data Analysis,” John Tukey identified the difference between mathematical and scientific statistics, and called for a rebirth of scientific statistics. To quote Lyle Jones, the editor of Volumes III and IV of Tukey's *Collected Works* [Jones, 1986, p. iv],

> The publication was a major event in the history of statistics. In retrospect, it marked a turning point for statistics that elevated the status of scientific statistics and cleared the path for the acceptance of exploratory data analysis as a legitimate branch of statistics.

Through Tukey's work, and that of others, data analysis that follows the paradigms of scientific statistics has been called Exploratory Data Analysis (EDA). Recently, EDA and the statistical graphics that often accompany it have emerged as important themes of computer-based statistical data analysis. An important aspect of these advances is that, contrary to traditional methods, they do not require data from designed experiments or random samples; they can work with serendipitous data. By examining the differences in these two philosophies of statistical data analysis, we can see important trends in the future of statistics software.

All statistical data analyses work with models or descriptions of the data and the data's relationship to the world. Functional models describe patterns and relationships among variables. Stochastic models try to account for randomness and error in the data in terms of probabilities, and provide a basis for inference. Traditional statistical analyses work with a specified functional model and an assumed stochastic model. Exploratory methods examine and refine the functional model based on the data, and are designed to work regardless of the stochastic model.

Statistics software has traditionally supported mathematical statistics. The data analyst is expected to specify the functional model before the analysis can proceed. (Indeed, that specification usually identifies the appropriate computing module.) The stochastic model is assumed by the choice of analysis and testing methods. Statistics packages that support these analyses offer a large battery of tests. They avoid overwhelming the typical user because a typical path through the dense thicket of choices is itself relatively simple.

The many branches in this design (Figure 1) encourage modularity, which in turn encourages a diversity of alternative tests and the growth of large, versatile packages. Most of the conclusions drawn from the analysis derive from the hypothesis tests. Printing (in Figure 1) includes plotting, but simply adding graphics modules to such a program cannot turn it into a suitable platform for EDA. For that we need a different philosophy of data analysis software design.

Software to support scientific statistics must support exploration of the functional model and be forgiving of weak knowledge of the stochastic model. It must thus provide many plots and displays, offer flexible data management, be highly interconnected, and depend on methods other than traditional hypothesis tests to reveal data structure. A schematic might look like Figure 2. Note that there is no exit from this diagram. Most of the paths are bi-directional, and many are omitted. The data analyst learns about the data from the process of analysis rather than from the ultimate hypothesis test. Indeed, there may never be a hypothesis test.

Software to support scientific statistics is typically not modular because each module must communicate with all the others. The complexity can grow factorially. It is thus harder to add new capabilities to programs designed for scientific statistics because they cannot simply plug in as new modules. However, additions that are carefully designed benefit from the synergy of all capabilities working together.

The user interface of such software is particularly important because the data analyst must “live” in the computing environment while exploring the data and refining the

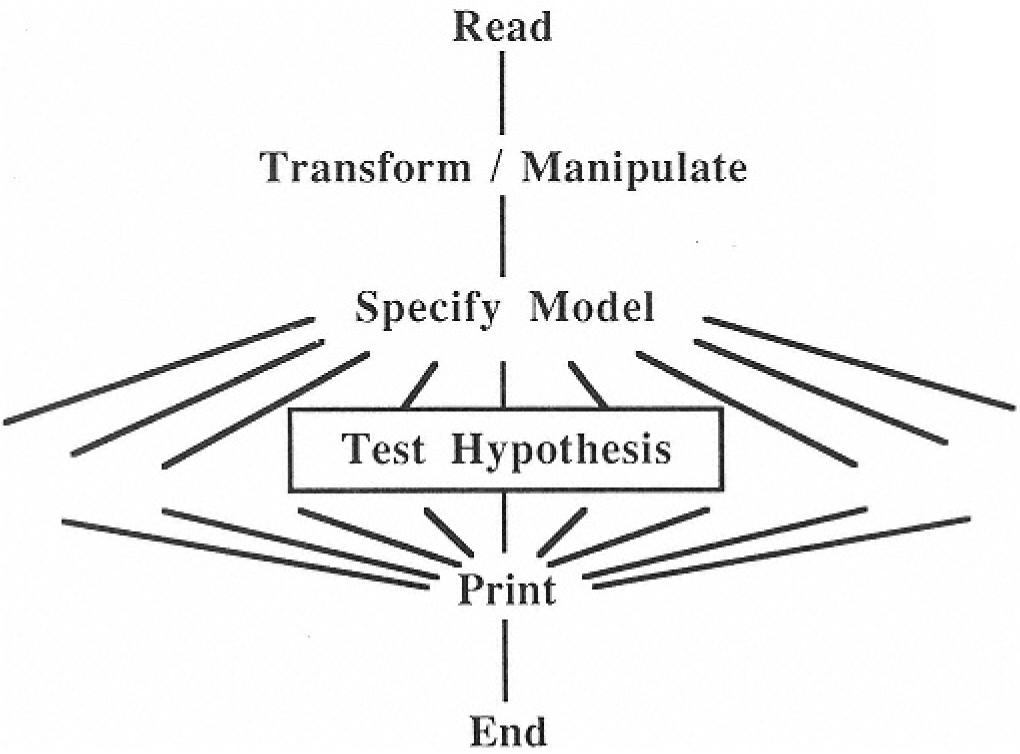

FIGURE 1: Schematic of software to support mathematical statistics.

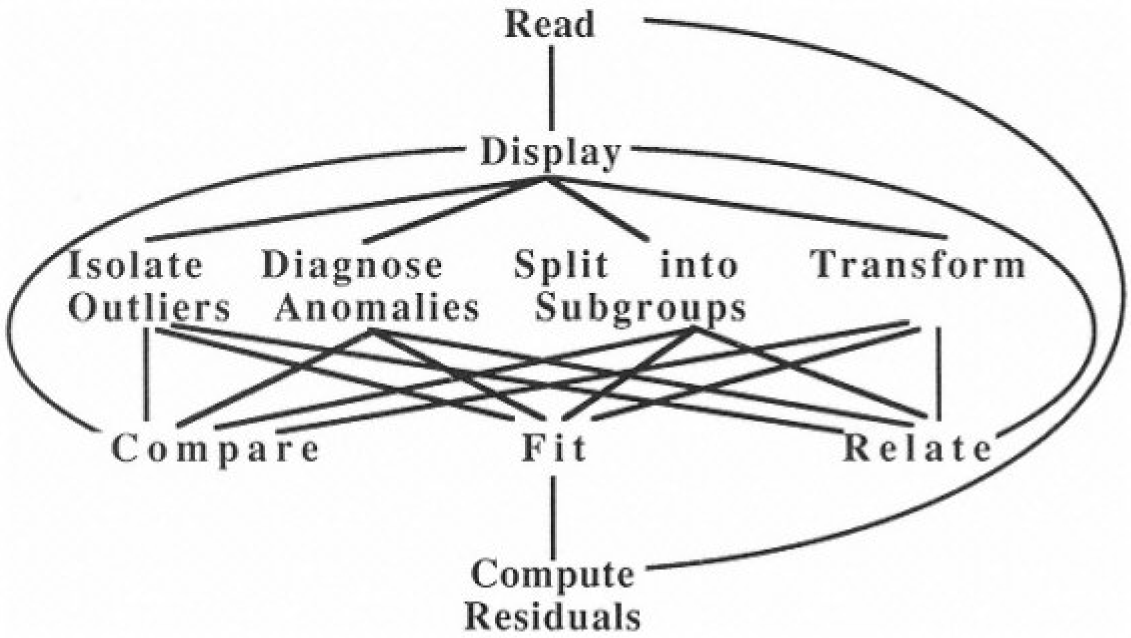

FIGURE 2: Schematic of software to support scientific statistics.

functional model (rather than simply passing through on a relatively straight path, as would be typical of traditional packages).

The ideals of mathematical and scientific statistics are two ends of a continuum. Good statistics software can fit almost anywhere along this continuum. However, it is probably impossible for any one program to serve both ends well. Any program rich enough in statistics methods and options to meet the needs of classical statistics will find it hard to offer the directness, speed, and integration required for data exploration.

WHERE IS STATISTICAL SOFTWARE GOING?

Many programs are moving to fill the middle ranges of the continuum. Programs such as SPSS, SAS, and Systat that were once geared to the extreme mathematical statistics end of the spectrum now appear in versions for desktop computers, and more flexible interfaces and better graphics have begun to be developed. Programs such as S, Data Desk, and X-Lisp Stat that pioneered in the data analysis end of the spectrum have had more capabilities added so that they no longer concentrate only on data exploration and display.

Nevertheless, I do not believe that we will all meet in the middle.

INNOVATIONS IN COMPUTING WILL OFFER NEW OPPORTUNITIES

Computing power will continue to grow, and new data analysis and graphics methods will develop to take advantage of it. Many of these will extend scientific statistics more than mathematical statistics, although both kinds of data analysis will improve.

As operating systems become more sophisticated, it will become increasingly common (and increasingly easy) to work with several programs, using each for what it does best and moving data and results among them freely. Thus, for example, one might open a favorite word processor, a presentation graphics program, a traditional mathematical statistics program, and a scientific data analysis program. One would then explore and graph data in the data analysis program, copying results to and writing commentary in the word processor. One might then move to the mathematical statistics program for specialized computations or specific tests available only there (again copying results to the word processor). Finally, the presentation graphics program could be used to generate a few displays to illustrate the major points of the analysis, and those displays again copied to the word processor to complete the report. Development in this direction will be good for statistical computing. There will be less pressure on statistics programs to be all things to all people, and more encouragement to define a role and fill it well. It will be easier to develop new innovative software because the practical barriers that make it difficult to develop and introduce new statistics software will be lower.

Networking will also improve dramatically, making it easier to obtain data from a variety of sources. Much of this data will be serendipitous, having been collected for other purposes. Nonetheless, additional data are likely to enhance analysis and understanding.

CHALLENGES FOR STATISTICAL SOFTWARE

There are a number of clear challenges facing someone with the goal of producing better statistics software. First, it takes a long time to design and implement a statistics package. Common wisdom is that 10 person-years is minimum, but the major packages have hundreds of person-years invested. Second, the user community is diverse. No matter what is in a package, someone will have a legitimate reason to want something slightly different. Many of these requests are reasonable, yet a package that meets all of them may satisfy nobody. Third, full support of software is difficult and expensive. While documentation can now be produced with desktop publishing, it is still very hard to design and write. Finally, designing and programming for multiple platforms often means designing for the lowest common capabilities. What may be worse, many of the difficulties arising in writing portable software affect capabilities of particular interest to modern computer-based data analysis, such as the speed and look of graphics, methods of efficient and effective data management, and the design and implementation of the user interface.

IS THE COMMERCIAL MARKETPLACE THE BEST SOURCE OF STATISTICAL SOFTWARE INNOVATION?

I answer this question in the affirmative. Software developed in academia or research facilities has rarely reached usefulness or ready availability without being commercialized. Commercial software continues to be innovative (although it also tends to copy the innovations of others). Commercial software is usually safer to use because the inevitable bugs and errors are more likely to get fixed. I also believe that the marketplace has shown a general ability to select statistics software. “Survival of the fittest” has been known to kill off some good programs, but weak packages tend to survive only in niche markets even when large sums are spent on advertising and marketing.

Nevertheless, commercial software development is a chancy business, demanding an unusual combination of skill, knowledge, and luck. The most innovative interface designers may lack the specialized knowledge to program a numerically stable least squares or to get the degrees of freedom right for an unbalanced factorial with missing cells. The best algorithms may be hidden behind an arcane command language or burdened with awkward data management.

We need to encourage innovation, but we also need to protect users from software that generates wrong results or encourages bad data analyses.

WHAT CAN WE DO TO ENCOURAGE INNOVATION?

To encourage innovations, we should encourage small players. While I have great respect for the giants of the statistics software industry, I think that monopoly rarely promotes innovation.

Developing standards that make it easier to work with several packages will also encourage innovation. For example, we need a better standard for passing tables of data among programs while preserving background information such as variable name, case label, units, and formats. Some might argue that we should work toward a unified interface for working with statistics software, but I believe that this would stifle innovation.

We can reduce the need to re-invent. For someone with a new idea about data analysis or graphics, it can take an inordinate amount of time and effort to implement standard algorithms that are already known. Books such as *Numerical Recipes* [Press et al., 1986] help for some scientific applications, but have little for statistics. For advanced statistical methods, Heiberger's book *Computing for the Analysis of Designed Experiments* [Heiberger, 1990] stands alone.

I support the trend toward using multiple programs. If a new concept can be proved in a relatively small, focused program and used readily with established programs, it will be easier to develop, distribute, and use. Users could then move data in and out of the program easily, continuing their analysis in their other preferred packages. Programs focused on a particular approach or audience (e.g., time series, quality control, econometrics) could integrate in this way with more established programs.

The pace of innovation could be speeded if we could convince academia that the creation of innovative software is a legitimate intellectual contribution to the discipline. We can attract fresh talent to statistical computing only if we provide publication paths and professional rewards. The *Journal of Computational and Graphical Statistics,* begun recently by the American Statistical Association, the Institute of Mathematical Statistics, and the Interface Foundation, and other new journals in this field may help here. Some university copyright regulations also penalize software development relative to authoring books or papers, but there is no consistency from school to school. We could work to establish standard policies that promote rather than stifle software innovation.

WHAT CAN WE DO TO ENSURE QUALITY?

Software reviewing is difficult and still haphazard. We need to recruit competent reviewers and encourage reviews that combine informed descriptions of a package's operation style, reasonable checks of its correctness, and well-designed studies of its performance. All complex software has bugs and errors. Responsible developers want to hear of them so they can be fixed. But people tend to believe that if the program doesn't work, it must be their fault ("After all, it's published, it's been out for so long someone must have noticed this before, computers don't make mistakes …"). We need to encourage users to recognize, document, and report bugs and errors.

A library of tested algorithms can help reduce errors. New programs always have errors. (So do old ones, but--we hope--fewer and less serious ones.) By providing tested programs and stable algorithms we can reduce the incidence of bugs and errors. Even if the algorithm is reprogrammed, it helps to have a benchmark for testing and debugging.

WHAT CAN WE DO TO PROMOTE PROGRESS?

- We must encourage those who have serendipitous data to examine it. (Typical thinking: "I don't have any data … well, I do have personnel records, sales records, accounting records, expenses, materials costs … but that's not data.")
- We must teach scientific statistics as well as mathematical statistics.
- We must encourage innovation.

REFERENCES.

Heiberger, R., 1990, *Computing for the Analysis of Designed Experiments,* John Wiley & Sons, New York.

Jones, L. (ed.), 1986, *The Collected Works of John W. Tukey, Vols. III and IV,* Wadsworth Advanced Books & Software, Monterey, Calif.

Press, W.H., B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling, 1986, *Numerical Recipes, The Art of Scientific Computing,* Cambridge University Press, New York.

Morning Discussion

TURKAN GARDENIER (Equal Employment Opportunity Commission): Professor Velleman's comments really spoke to the heart. After five or six years of dealing with military simulations for our designed experiments, where I had people run multivariance regression models in two weeks and analyze the nice multivariance regression model, I'm now in a setting where all of the data is serendipitous.

We held a seminar yesterday in which many attendees asked whether they could assume that a data set from some company was part of the universe of employee records. The statisticians looked at the data and said, “Not really; there are interventions in company records over a period of time.” By our not permitting that assumption, there was criticism expressed as to our not being real statisticians, because if it could not be assumed that this year's data was a random sample from a population, what were we doing there? Thanks to what Professor Velleman presented today, I am going back and reporting the innovations for serendipitous data that could be applied.

As a statistician working with serendipitous data, let me make another comment. We need lag time, both as data analysts as well as statisticians. I am dealing with a lot of litigation cases as a statistical expert, telling attorneys what data to collect. Some people come to our office with partially collected data; such data is very hard to analyze and have the analyses stand up in court.

Within our organization, we write memoranda of understanding. It's part of a statistician's professional responsibility to work with the right types of data, to make the right assumptions before using a computer, and to do statistical significance tests. Having lag time available ties in with ex post facto collection of the right types of data that could interface with interactive data analysis.

PAUL VELLEMAN: I think you are right. I mentioned that we need to teach about scientific statistics. For years we have taught the mathematical statistics approach as *the* approach in all introductory statistics courses. As a result, the world is full of people who think that the way one does statistics is to test a hypothesis. Many of our clients know only that much statistics, and this has in effect made our lives more difficult. It will make our future lives easier if we start teaching more about scientific statistics, rather than just hypothesis testing.

CLIFTON BAILEY (Health Care Financing Administration): The HCFA deals with all the Medicare data. I certainly concur regarding serendipitous data. We try to analyze the 6 million persons who have a hospitalization in each year, within a universe of 10 million hospitalizations from the 34 million beneficiaries. The data are provided on bills, and one needs different kinds of tools and techniques to deal with many of the issues in these types of analyses, e.g., the diagnostics.

I like Paul Velleman's example involving the pathway through an analysis. Many times we want to make comparisons, while taking one pathway, about what would have happened if another analysis, another pathway, had been used. When I look at standard statistical packages, I frequently see that they put in a log likelihood, but they do not do it in the same way. Many of them leave out the constants or formulate the likelihood differently and then do not tell how it is done. I want to be able to compare across pathways in that larger perspective, to do exploratory analyses in that context.

SALLY HOWE (National Institute of Standards and Technology): One of the things that you face that other people often don't face is that you have very large data sets. Is that an obstruction to doing your work, that there is not software available for very large data sets?

CLIFTON BAILEY: Yes, the available software is very limited when you get into large data sets, unless you deal with samples. But you want to be able to ask finer questions because you have all of that data, and so you do not want to merely use samples--or you want to use combinations of samples and then look at the subsets using the base line against the sample.

There are many complex issues involved in doing that. However, we can put a plot up and look at residuals for our data sets that are generated in the psychology laboratory or in many of the non-serendipitous database contexts. We can look at that on a graph; we can scan down a column in a table. But we need other techniques and ways of doing exploratory analyses with large data sets.

Another agency, the Agency for Health Care Policy Research, is funding interdisciplinary teams to make use of these kinds of data. There are at least a dozen research teams that are focusing on patient outcomes. Every one of those teams is facing this problem, as is our agency, and I am sure that many others are also.

SALLY HOWE: Do you see any additional obstructions that the previous speakers have not yet mentioned?

CLIFTON BAILEY: I recently needed to have a user-provided procedure (or proc) in SAS modified by one of the authors, because the outputs would not handle the large volumes of numbers. When the number of observations went over 150,000, it would not run. Many of the procs get into trouble when there are more than 50,000 observations or some similar constraint.

PAUL TUKEY (Bellcore): This problem, dealing with these very large data sets, is one that more and more people are having to face, and we should be very mindful of it. The very fact that everything is computerized, with network access to other computers and databases available, means that this problem of large data sets is going to arise more and more frequently. Some different statistical and computational tools are needed for it. Random sampling is one approach, but an easy and statistically valid way to do the random sampling is needed, and that is not always so simple to specify. People like Dan Carr [George Mason University], who is here today, and Wes Nicholson [Battelle Pacific Northwest Laboratory] have explicitly thought about the issue of how to deal with large data sets. There are ways to do it.

When you have a lot of data points, one issue is that you can no longer do convenient interactive computing. It can take five minutes to run through the data set once, doing something absolutely trivial.

There is also a statistical issue involved. When you have an enormous number of observations, you suddenly confront the horrifying fact that statistical significance is not what should be examined, because everything in sight becomes statistically significant. When that is the case, what should replace statistical significance? Other ways of determining what is of practical significance are needed. Perhaps formal calculations of statistical significance can still be used, but interpreted differently, e.g., used only as a benchmark for comparing situations. I completely agree that this issue of large data sets is very important.

FORREST YOUNG: I too agree with what Paul Velleman is saying and with what Clifton Bailey has brought up. In exploratory methods, the definition of “very large” is a lot smaller than it is in confirmatory methods, by their very nature. To handle 10,000 observations in exploratory methods is really difficult. To handle 10 million, perhaps, in confirmatory methods is really difficult. That is their very nature.

ROBERT HAMER (Virginia Commonwealth University): I agree with Forrest that some exploratory methods do not work well with large data sets. In fact, with sufficiently large data sets, almost any plotting technique will break down in a sense, because the result is a totally black page; every single point is darkened.

PAUL TUKEY: See Dan Carr about that.

MIKE GUILFOYLE (University of Pennsylvania): I help people analyze data, and this problem about scale intensiveness is very, very important. Software sophistication has to be considered in light of the specific task in which the users are involved.

Many of these statistical software packages were involved at their origins with teaching. Later they moved from teaching to research, and now from research the focus is turning to production. This is what Clifton Bailey from HCFA is saying. When large production runs are involved, the clever things performed at the touch of a mouse on small research or educational data sets cannot be done. My large production runs are done with batch processing and involve multi-volumes of tapes, a context in which the mouse-click cleverness is lost.

I am surprised that no speakers have yet mentioned Efron's work in jackknifing and bootstrapping [see, e.g., Efron and Tibshirani (1991), or Efron (1988)], which may be useful at least at the research level, to see if the model does or does not work. It is also a computer-intensive process, for which many people do not have the resources. But there are people thinking about such approaches, and the literature on those approaches might be of interest to statistical software producers.

All of the speakers alluded to software packages as black boxes. But beyond that, one needs to think about technology. People who use these clever little packages on workstations become very good at mastering the technology, but they then assume that they understand the underlying processes. However, it is more complicated than that, because those people may or may not understand statistics, and may or may not understand the computing involved. There is a nexus between computing and statistics, but you do not know whether it's a fifty-fifty split. One person may be reasonably competent in statistics and also reasonably competent at mastering the technology, while another is very good at handling the technology but knows nothing about the statistics. I worry that these technological interfaces make things too easy; people can sit there and fake it. It goes beyond the old regime of taking some data, running the regression, and asking if this is the best r^2 that can be obtained. It is much more complicated.

PAUL TUKEY: Yes, you can fake it with a lot of these packages, and you could fake it with the old packages, too. Part of the salvation might be what Daryl Pregibon will discuss this afternoon, an expert system that does not let you get away with doing something that is grossly inappropriate, or at least forces you to confront it, and will not quietly acquiesce.

In a way, a package cannot force a user to do something responsible. But at least it can responsibly bring things to the user's attention.

RAYMOND JASON (National Public Radio): I am not a degreed statistician; I am a user of statistical software. I have been assigned on occasion to design experiments, carry them out, and analyze them. The focus this morning and for the program this afternoon seems to be on analysis and not on the design of experiments. Of course, a fully designed experiment gives data that are analyzable by any package. In my work, I have found that I was not supported at all by software or its documentation in the design of the experiments. If a goal is to improve the real-world utility of statistical software, experimental design is a necessary responsibility that needs to be addressed.

Two ways of addressing it would be to specify standards for expert systems that do design of experiments, or at the very least to come up with some suitable warning to be applied to documentation. Statistical software is advertised and available to the general public through mail order outlets. You do not have to be a member of the American Statistical Association to be enticed by the software, and you do not have to be wealthy or associated with large companies to buy the software. It is definitely targeted at people such as myself. Yet, I came very close to a major statistical analysis disaster. Only because of work with other experts, certainly not because of the documentation of the software, was disaster prevented.[2]

BARBARA RYAN: Those comments of Raymond Jason are excellent. I also deal with many people who are not degreed statisticians. There is a real concern here because they are going to do statistical analyses. So unless we help those people, either through education or through software guidelines or other means, they are going to make mistakes.

Regarding his other comment about designing experiments, it is true that most of the software available is for either exploring or analyzing, not to help in designing. There are some attempts with diagnostics to warn when there are problems, but for up-front design, less software has been developed to help.

KEITH MULLER: Design is my favorite topic, but I find it the most fuzzily defined task that I face as a statistician. Therefore, it would be the most difficult to implement in a program package. Experimental design is a collection of heuristics at this stage, rather than an algorithm or a collection of algorithms.

The dimension I identified under the “Audiences” heading in my talk is that of user sophistication. Statistical software ought to try to identify itself with respect to its target audience. That at least would be an attempt to be responsible in distributing and marketing software.

PAUL TUKEY: I also agree that the packages being designed now should have better design modules. People developing the software tend not to be doing a lot of design experiments, and so experimental design tends to be a glaring omission.

Years ago, while I was a graduate student, I did some work with a Russian statistician who was visiting in London. He had just written a book containing an idea that greatly impressed me. His idea was that design and analysis are really one and the same, not two different things, and all should be integrated. He had some very good ideas on the “how to,” very practical ways of designing efficient experiments, not purely abstract mathematical things. We need to build on those kinds of ideas and get those things into our software packages. At least in situations where we have an opportunity to design experiments, we ought to take advantage of it. There are big gains to be had by doing so.

HERBERT EBER (Psychological Resources, Inc.): The answer to the questions of what to do with huge data sets, and with the fact that everything is then significant, has been around for a while. It is called confidence limits, power analysis, and effect size. There *are* packages available. I am aware of one in progress that will do away with significance

[2] An article in the *Journal of Quality Technology* by Chris Nachtsheim of the University of Minnesota [Nachtsheim, 1987] evaluates a number of software packages written to help people design experiments--ed.

concepts almost completely and talk about confidence intervals. So alternatives do exist.

WILLIAM PAGE (Institute of Medicine): Concerning a previous point, we have not really talked about sampling yet. The way you collect the data has something to do with the way you analyze the data. Either we are putting things in a simple random sample box, such as our ANOVAs, or we're putting them in a serendipity box. There ought to be a way to handle something in between. Do you have a complex sample survey? Then you should be using the ANOVA box.

This may pertain to the guidance issue of this afternoon. The first line produced by the package might ask, “Do you have a simple random sample, yes or no?” If the user punches the help button, then a light flashes on and the output reads, “Do you have money to pay a statistician?”

KEITH MULLER: On the issue of statistical significance, I try to teach my students the distinction between statistical significance and practical or scientific importance, and that the latter is what the user is actually interested in.

RICHARD JONES (Philip Morris Research Center): There is an area of statistical software that is not being addressed here at all. Many scientific instruments in laboratories have built-in software that does regressions, or hypothesis testing. Some researchers, for instance, use these instruments and blindly accept whatever is produced. I recently had an opportunity, fortunately, to catch a chemist taking results from a totally automated analysis. He was using a four-parameter equation that left me incredulous. When I asked why he used that, he said that it was because it gave a correlation coefficient of .998. I said, “Why don't you just do a log transform?” He replied, “Because that correlation coefficient is only .994.” This gentleman was perfectly serious about this. This software built into instruments is in general use. Though not part of the big statistical packages that have been discussed, it is just as important to the scientific community and to their understanding of things.

PAUL TUKEY: We have standards from IEEE for how to do arithmetic computations to ensure that different machines get the same answers. An effort is needed to develop some standard statistical computing algorithms. Some of these things already exist, of course. But specifying the actual code or pseudo-code would allow these algorithms to be rendered in different languages, so that they can certifiably produce the same answers and do the basic building-block statistical kinds of things from which one can build analyses. The major package vendors could adopt these and replace whatever they are doing with software that adheres to the standards, and the people building these instruments could, at least in the future machines, build in some coherence.

PAUL VELLEMAN: I endorse that very strongly. There was a mention of providing test data sets. As a comparison, I think that test data sets are inevitably a failure, because the result is programs that get the right answer on the test data sets, but not necessarily anywhere else. Standard algorithms and implementations of them would compel the same answer across a wide variety of data sets. If a particular implementation or use then fails to give the same answer as other methods, at least the situation is well defined, and so it can be fixed. This is something that should be looked into.

BARBARA RYAN: I think the issue raised by Richard Jones is an issue of education. There are many ways to do statistical analysis. Often the procedures are built into devices, as small packages that were developed in-house. There's a huge amount of such statistical software currently in use that the “professional statistics community” almost ignores. I have a sense that we dismiss it as being so simplistic and superficial that we are not even going to look at it.

The trouble is there are thousands of people who are using it. It is more an issue of education. Maybe I misunderstood his question, but it's really what happens with a log transform to your r^2. This is fundamentally an issue of education, not whether you are getting the right or wrong answers. So if a fairly naive user, as far as data analysis is concerned, gets involved in statistics but does not know what he or she is doing, how do you address that misuse of statistics, when such statistical software is so available to everyone?

WILLIAM EDDY: I want to make a comment about standard algorithms versus standard data sets. If you read the IEEE floating-point arithmetic standard 754, you will see that it does not specify how to do the arithmetic. What it specifies is what the results of arithmetic operations will be. Therefore, the standard is actually articulated in terms of standard data sets, rather than in terms of standard algorithms. If we are to emulate such an organization, we have a difficult task ahead of us. Algorithms are the easy way out.

KEITH MULLER: I have a problem with an algorithm standard. Let me give you an example in everyday life with which you are all familiar--headlights on your automobile. In the mid-1930s there were all kinds of bad headlights available. Therefore the United States created an equipment standard that said sealed beam headlights were required. In Europe, the standard used was a performance standard. The equipment standard held back development in the United States of quartz halogen bulbs, which are far superior in performance. It took a revision of the law to permit their use here. Consequently, I would urge us to specify performance standards rather than algorithmic standards.

PAUL GAMES (Department of Human Development and Family Studies, Pennsylvania State University): One of the things that I am most disturbed by in statistics is what some people promote as causal analysis. This is where they take what amounts to correlational data that has been collected in strange ways and, after merely getting regression weights and using what they call “the causal path,” produce fantastic interpretations of the experimental outcomes. Is there anything being done in statistical packages that might induce a little sophistication in those people?

FORREST YOUNG: The recent development along that line is that those kinds of analyses have now been added to the major statistical packages.

KEITH MULLER: This is an issue of philosophy and education. If you don't value such causal analyses, then you should teach your consultees accordingly. That is a statistical issue, and not a statistical computing issue, I would argue.

BARBARA RYAN: There is a question of how much expertise and training we can build into software. People sometimes learn about statistics first from a software package manual. There is a philosophical issue of how much teaching you can provide through software vehicles. Much is being provided by training courses. Many packages offer a lot of training with them, but provided through the software rather than through the university or independent workshops run by institutes for professionals.

It may also be a practical issue. If people keep learning from packages, perhaps the best way will then be to provide more statistical education and guidance through a software vehicle.

PAUL VELLEMAN: I see that more as an opportunity than a problem. The biggest problem in teaching statistics is convincing your students that they really want to know this. When a person already has a statistics package in front of him and wants to understand it, that is the time to teach him. The Minitab student handbook, which was really the first tutorial package-based manual, was therefore an important innovation.

BARBARA RYAN: There is a problem when the people writing the manuals are not statisticians. It takes the educational role away from people who are the real educators. One must find the right balance.

KEITH MULLER: Concerning the issue of large data sets that several members of the audience raised earlier, I presented a paper with some co-authors a few years ago [Muller, et al., 1982] in which we talked about the analysis of “not small” data sets. I suggested there that one needed to take the base 10 logarithm of the number of observations to classify the task that one faced. One could classify small data sets as those involving a hundred observations or fewer, not small as those in the 1,000 to 50,000 or 100,000 range, large as 100,000 to 1 million, and very large as 10 million or more. If we were to classify statistical software according to the size of data set for which it does work, it would help the user because it is obvious that people run into problems and that software does not transport across those soft boundaries.

Also, I neglected to mention a paper by Richard Roistacher on a proposal for an interchange format [Roistacher, 1978]. That appeared a few years ago, and there has been a thundering silence following its appearance.

WILLIAM EDDY: There's been about 15 years of silence following that.

ROBERT TEITEL: At the Interface conference that was held in North Carolina in 1978 or 1979, Roistacher got to the point of having most of the vendors accepting that standard in principle. But then, as I understand it, his agency contract ran out of money, so that didn't go anywhere.

The notion of small or medium or large cannot be done in absolute terms. “Small-medium-large” is relative to the equipment you are using. Many people would consider 10,000 observations on a hundred variables to be enormous, if you are trying to run it on a PC. I like to define “large” as any size data set that you have difficulty handling on the equipment you are using.

CLIFTON BAILEY: If we had analog data on all of the Medicare patients like that which is collected at bedside, our data sets would be very, very small.

DANIEL CARR (George Mason University): I worked on the analysis of a large data set project years ago. Leo Breiman [University of California at Berkeley] has raised the issue that complexity of the data is more important than the large sample size. For instance, if someone said to me, “I have 500 variables, what do I do?”, I would say, “That is not the ‘large’ I like to work with. I like to have hundreds of thousands of observations with only three variables.” Complexity is an issue in the different types of data.

I am very interested in interfaces to large databases. Most of the data that is collected is never seen by humans and I think that is a tragedy, because a lot of this data is very important. So I think statistical packages need more interfaces to standard government data sets. I went to the extreme of trying to interface GRAS and EST; GRAS is a geographical information system. The neat thing about GRAS is, it already had the tools to read lots of government tapes. Having that as a part of standard statistical packages would be great.

At some point, I believe we are going to have to think differently about how we analyze large data sets. Some things are just too big to keep. In fact, a lot of data is summarized at the census level. So I think statisticians are going to have to think about flow-through analysis at some point. It may take a new generation of people to actually do that.

Another topic brought up was what I call data analysis management. More and more, there is emphasis on quality assurance in analysis. Most of that means proving what you did. It does not mean having to do it right, but at least proving what you did. I think that needs to be built into the software, so that we can keep a record of exactly what we did and can clean up the record. When I do things interactively I keep a log of it, and then I go back and clean out the mistakes and re-run it. But I think we need to have this kind of record, and it needs to be annotated so that it is meaningful later on. A lot of times, I go back to my old analysis, look at it, and ask, “What was I doing?” I often think maybe I made a mistake. Then two days later I realize I really did know what I was doing, so it was right. But I had forgotten in the meantime. So there's a need for this quality assurance feature, data analysis management with annotation, which may include dictation, voice, whatever is easy.

I would also like to see some high-end graphics tools. I am envious of people right now who can make movies readily. It is a very powerful communications medium that ought to be part of statistical software. I would like to see more integration of mathematics tools like Mathematica. Somebody ought to be addressing memory management; eight megabytes on a Spark workstation may not be enough, depending on the software I am using.

I would like to have more involvement with standards. For example, I am not sure that the standards that are developed are always optimal for statistical analysis. For instance, there are some defaults on the Iris workstations for projection that may not produce exactly the projections I would like for stereo. But at some point they even get built into the hardware, and so I have to program around them. It would be nice if in some areas we could get involved with the standards, both for hardware and software, and that boundary is getting closer all the time.

REFERENCES

Efron, Bradley, 1988, Computer-intensive methods in statistical regression, *SIAM Review,* Vol. 30, No. 3, 421–449.

Efron, Bradley, and Robert Tibshirani, 1991, Statistical data analysis in the computer age, *Science,* Vol. 253, 390–395.

Muller, K.E., J.C. Smith, and J.S. Bass, 1982, Managing “not small” datasets in a research environment, *SUGI '82--Proceedings of the Seventh Annual SAS User's Group International Conference.*

Nachtsheim, Christopher J., 1987, Tools for computer-aided design of experiments, *Journal of Quality Technology,* Vol. 19, No. 3, 132–160.

Roistacher, R.C., 1978, Data interchange file: progress toward design and implementation, in *Proceedings of the 11th Symposium on the Interface: Statistics and Computing Science,* Interface Foundation, Reston, Va., pp. 274–284.

Afternoon Session Opening Remarks

Forrest Young
University of North Carolina, Chapel Hill

There was talk at various times this morning about standardization and occasionally about certification. The Panel on Guidelines for Statistical Software is about neither of those, and it is important to emphasize that. Our business is guidelines, not issuing seals of approval.

If you think particularly about the three topics of exactness, richness, and guidance, it is hard to know how one would decide for the last two, richness and guidance, that something deserves a seal of approval. Making such judgments for exactness is a possibility, although I am not saying I think that is a good thing to do. The panel aims only to state guidelines, not to set standards or issue seals of approval. It is certainly possible, though, to set standards for exactness. For richness or guidance, however, standards--let alone certification--may not be possible.

Another theme that came up several times in the morning was layering, that there should be different layers of the software system. I tend to see this as related to this afternoon's featured topic of guidance in that there can perhaps be an outer layer of a statistical system whose purpose is to guide the relatively unsophisticated user.

In my ideal data analysis environment, such a layer would be there for the more naive user, but would not have to be there; it need not be used by a more sophisticated user. There would be several layers. Perhaps the innermost layer would be just a language. A complete system would need to have more layers put on the outside to help people who are less sophisticated in terms of the data analysis, but who are very interested in the application.

Another theme from this morning was that of strategy, which is a central idea in guidance. Paul Tukey mentioned that one ought to have a strategy for doing regression modeling. Also, Paul Velleman presented two strategies. One is an original strategy for doing statistical analysis based on batch submission of analyses, where one first reads in the data and then specifies the strategy, afterward producing output. That is a very linear strategy without any choices in it. Later, he presented a much more involved strategy, more in tune with exploratory data analysis, where data is read at the beginning and displayed, whereupon the user is faced with a lot of options having to do with outliers, with diagnosing problems in the data, or with putting the data into sub-groups and transforming the data. Basically, that is another idea of a strategy in data analysis. Strategies are important for providing guidance.

In a paper I presented at the ASA conference in August of 1990 on that topic [Lubinsky and Young, 1990], there were a couple of slides on guidance showing my ideas along this line. Figure 3 is a mock-up of a proof-of-concept system that David Lubinsky of AT&T Bell Laboratories and I worked on. There is a window with a cyclic graph in it. As Paul Velleman pointed out this morning, it has an entry point, but no exit. This represents the process of data analysis. It is never finished. But you can exit at any point you want. There is no specified plan of things that must be done before you can quit. But when you do exit, the system would suggest a thing to do.

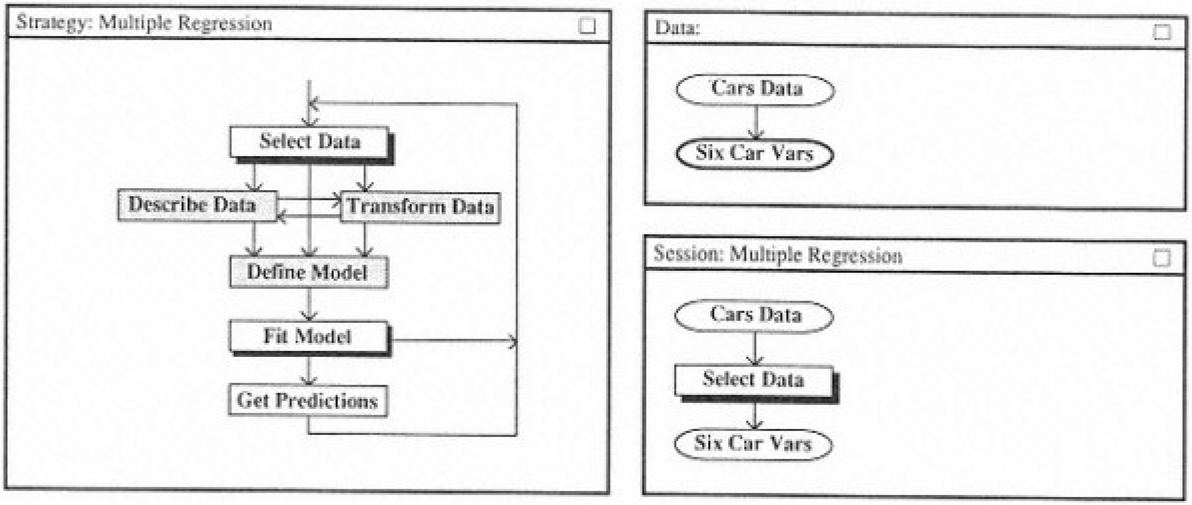

FIGURE 3: One possible way of guiding a data analysis. Reprinted, with permission, from Lubinsky and Young [1990]. Copyright © 1990 by American Statistical Association.

For example, the grayed-in box is suggesting that the first thing to do is to select the data. When that has been done, a sub-strategy might be given, a recursive definition of a strategy. A new strategy box opens up that focuses both on variables and observations or cases. When that is finished, that box closes.

Then the user goes to the next set of possible things that the strategy would suggest, either describing the data, transforming the data, or defining a model. As the flow indicates, if you describe the data, you still can again transform data or define the model, and conversely for transforming. But once you have a model defined, the only thing the strategies then suggest you do is to fit the model. Fitting the model itself is recursively defined. Within that one would see a more involved strategy depicting what to do.

This is one possible way of guiding a data analysis. Where does this strategy graph come from? It comes from an expert. Somewhere, an expert at multiple regression must have sat down and created this graph. In fact, this graph was created by Lubinsky and me after looking at the book by Daniel and Wood [1980], where such a strategy for doing multiple regression appears on the inside front cover. There are also analogous graphs presented for principal components in a factor analysis, for example. Such sources for guidance strategies are available, and experts can certainly be consulted for strategies to guide data analyses.

REFERENCES

Daniel, C., and F.S. Wood, 1980, *Fitting Equations to Data,* John Wiley & Sons, New York.

Lubinsky, D.J., and F.W. Young, 1990, Guiding data analysis, *Proceedings of Section on Computational Statistics,* American Statistical Association, Alexandria, Va.

An Industry View

Andrew Kirsch
3M

INTRODUCTION

For over 30 years, substantial efforts have been made at 3M to transfer statistical skills and thinking from professional statisticians to engineers, scientists, and business people. Our goal is to decentralize statistical knowledge and services, and our perspective on statistical software reflects this goal.

Statistical software plays a key role in the dissemination process. In training, software allows the students to concentrate on concepts rather than calculations. In application, software provides a link to what was learned in class and a fast means to apply it.

The author is a member of a central group, with consulting, training, and software responsibilities, that acts as a catalyst for this process. Knowledge of the (internal) clients' needs comes from consulting and training interactions and from formal surveys. Until recently this central group was also active in internal software development. Such experience imparts an awareness of the formidable challenges that software providers face in serving the statistical needs of industrial users. This presentation focuses on requirements and critical success factors for statistical software intended for the non-statistician, industrial user.

REQUIREMENTS FOR “INDUSTRIAL-GRADE” SOFTWARE

The requirements for “industrial-grade” software are fairly demanding. In addition to such givens as acceptable cost and core statistical capabilities, it is critical that the software be available for a wide variety of hardware environments. Near-identical versions must be available on commonly used microcomputers and mainframes to facilitate transparent data connectivity, communications on technical matters, and training. The alternative is another Tower of Babel. The nature of the users themselves imposes further requirements. A wide range of user skill levels must be accommodated, from novices to near-experts. The basic user can be easily overwhelmed by too many options, while the advanced user can be frustrated by inflexibility.

The non-statistician sees statistical design and analysis as part of a larger task of product development, process control, or process management. Such a user expects the software to perform the statistical tasks properly and not to inhibit performance of the larger task. For this reason, it is important that the software should interface with commonly used database, spreadsheet, and word processing programs. Graphical display of results in presentation-quality output (both on screen and as hard copy) is also important. Beyond these, the user will be delighted by other features, such as the automatic creation of data entry forms or uncoded response surface plots that go the extra mile to aid the solution of the larger task. Strong prejudices regarding statistical software are apparent even among non-statistician users, but this is more than just bullheadedness. In an atmosphere of stiff business competition where time to market is vital to success, there is limited time available to learn new support tools. The industrial user will be pleased by new capabilities but desires upwardly compatible releases.

Other software requirements include availability of support (either locally or by the vendor), complete and readable documentation, and, in an age of increasingly global businesses, suitability for persons with limited English-language skills.

But the most crucial requirement for “industrial-grade” statistical software, beyond core capabilities, is ease of use for infrequent users! This overriding need is suggested by an internal company survey that showed that, with the exception of basic statistical summaries, plots, and charts, most statistical methods were used by an engineer or scientist on a once-per-month or once-per-quarter basis. From a human factors perspective, this suggests that most industrial users cannot be expected to remember a complex set of commands, protocols, or movements in order to use the statistical software. Appropriate software must rely on *recognition* (e.g., of icons or application-oriented menus) rather than *recall* (e.g., of commands or algorithm-oriented menus) to satisfy these users.

Of course, ease of use means more than just recognition aids for the infrequent user. Other important facets are:

- well-planned display of information;
- understandable and consistent terminology;
- reasonable response time; and
- helpful error handling and correction.

But the author wants to stress sensitivity to the needs of the infrequent user, which are so often overlooked. While a visually attractive package will sell some copies, only those that genuinely meet the requirements of the infrequent user will survive in the long run.

IMPLICATIONS FOR RICHNESS.

Given the wide range of user skill levels that must be accommodated by statistical software, creative methods must be employed to ensure the appropriate richness for each user. It is unacceptable to provide software that is designed to meet the needs of the advanced user and assume that the basic user can simply ignore any options or outputs that are not needed. Experience indicates that such a strategy will scare off many basic users. On the other hand, providing different software for basic and advanced users induces an artificial barrier to learning and communication. An alternative is to provide software that can “grow” as the user's skills grow. On output, for instance, this means that different layers of output can be selected reflecting increasing degrees of richness.

While configurable output is already available in many statistical packages, another kind of configuration has yet to be widely exploited. This is the concept of configurable options. In a menu-driven program, this means that the user (or local expert) can configure the menus to exclude certain options that are not needed and only add “clutter” to the program. An example of a non-statistical package employing this feature is Microsoft *Word for Windows*®. With configurable options, the user or local expert could eliminate unused or undesired options of specific output (e.g., Durbin-Watson statistic) or of whole methods (e.g., nonlinear regression).

Another aspect of richness that is important for the advanced user is the ability to add frequently used capabilities via some sort of “macro” command sequence. As mentioned earlier, strong prejudices exist among statistical software users. A new package may well be rejected by a user or group of users because it lacks only one or two capabilities that the user(s) have come to depend on. The capability to build macros helps overcome such barriers. Even better is the capability to incorporate such macros into the normal program flow as menu items or icons--a capability also available on *Word for Windows.* This is the “flip side” of configurable options.

With configurable options, statistical software can avoid the “one-size-fits-all” fallacy, without inducing artificial barriers to learning or communications.

IMPLICATIONS FOR GUIDANCE

The novice user, limited by time constraints and capacity to recall, benefits greatly by appropriate guidance. Such a user is not primarily concerned with guidance as to what the software is doing at a computational level; he or she will generally rely on the trainer or local expert to vouch for its validity and appropriateness. The novice user is more concerned about guidance on how to progress through a reasonable design or analysis from start to finish and how to interpret the output.

Clearly it is easier to provide guidance for progressing through a specific analysis (e.g., regression model fitting) than for a general exploratory analysis. The same can be said for software providing design capabilities: guidance on the particulars of a class of designs (e.g., Plackett-Burman designs) will be easier to provide than guidance on the selection of an appropriate class of designs. For guidance through these more specific kinds of design or analysis problems, it is helpful to have the software options organized *by user task* rather than by *statistical algorithm.* For example, it is preferable that algorithms for residual analysis be available within both the linear regression and ANOVA options rather than requiring the user to move to a separate option for residual analysis from each starting point.

Another useful tool for guidance on a specific design or analysis is on-line help for interpreting the output. It is interesting that on-line help for command specification or option selection has been available for years, yet on-line help for output is much less widely used. It seems that many statisticians feel uncomfortable about the idea of a canned interpretation of the results of an ANOVA or regression analysis. Yet even a modest degree of memory-jogging would be enormously helpful. Why should a non-statistician be expected to remember that in a regression analysis, a small p-value on the lack-of-fit test is “bad” (i.e., evidence of lack of fit), while a small p-value on the test for the model is “good” (i.e., evidence of a meaningful model)? Will professional sensibilities be offended if an on-line help statement simply reminds the user (upon request) of the definition of a p-value and the implications of a p-value close to 0 for that particular test?

The author, too, is wary of excessive guidance that would encourage a “black box” attitude toward design or analysis, but feels that statisticians and other software developers must overcome a case of scruples in this regard. A middle ground really is possible.

Guidance on the more general kinds of design and analysis issues invites the development of knowledge-based (expert) systems. A major difficulty here is the nature of statistical knowledge--it is a generic methodology applied to diverse subject-matter areas. A knowledge-based system for, say, medical diagnosis could be relatively stand alone, but effective application of statistical methods requires the integration of subject matter considerations. Any stand-alone system for statistical methods risks segregating statistics from subject matter knowledge in a dangerous way.

For example, in a drying process, oven temperature and product mass can be controlled. Without some incorporation of the basic physics regarding the multiplicative effect of these variables on heat transferred, the stand-alone program might erroneously recommend a 2 × 2 design (where the AB and ab conditions are identical for heat transfer) or suggest fitting an additive model.

One option is to create statistical expert systems to which subject-matter knowledge can be appended. Short of that, statistical knowledge-based systems can exhibit an appropriate degree of modesty by *proscribing* rather than *prescribing:* pointing out options that are untenable and suggesting plausible options to explore further, rather than trying to identify one or a few solutions that are “the best.” The system can identify appropriate memory-jogging questions from a catalog of questions such as the following:

- Has an appropriate degree of replication been included in the design?
- Was this data collected in a completely randomized fashion (as the type of analysis might suggest), or was it collected in blocks?

Needless to say, all of these features would be available upon the user's request rather than imposed without recourse by the program.

CONCLUSION

Far too many software development dollars are being devoted to adding new capabilities and far too few to enhancing ease of use. It might shock some software providers to realize that the biggest competitor for the recommended statistical software at 3M is not another statistics package, but Lotus *1-2-3*®!

Menu-driven software and windowed software for non-statistical applications have raised the ease-of-use expectations of statistical software users. There is a substantial fraction of potential users in industry that will not “buy in” to a statistical software solution that does not combine state-of-the-art ease of use with core capabilities, acceptable cost, and multiple hardware availability.

The future of statistical software is not just a technical issue; it is also a business issue. The providers of most statistical software are private, profit-making companies. These firms often rely on new releases with added capabilities to produce current revenue. In the industrial market, far more revenue is available by providing enhanced ease of use than by adding non-core capabilities. It is hoped that future competition among software products will be more on the basis of guidance and configurability than on the basis of additional non-core capabilities.

Guidance for One-Way ANOVA

William DuMouchel
BBN Software Products

GOALS OF GUIDANCE

The first goal of guidance is to permit the occasional, infrequent user whose main business is not statistical data analysis (such as a scientist or engineer) to achieve the benefits of using basic statistical procedures. These benefits are the ability to make comparisons, predictions, and so forth, with measures of uncertainty attached. One of the key notions is to understand what is meant by “measures of uncertainty” and how to convey them in the computer output, while avoiding the most common pitfalls and inappropriate applications that one can fall into.

The second goal of guidance is to overcome the barriers that prevent technical professionals from using statistical models. Such barriers were covered quite well by Andrew Kirsch in the preceding talk. One barrier is in dealing with people who have not had courses in statistics or, worse yet, who have had a poorly taught statistics course. Another is that non-deterministic thinking is just not the natural evolutionary way our brain seems to have developed. So it is an unfamiliar and different concept to many.

Further, statistical jargon is quite alienating, in the same way that any jargon is alienating. Statisticians in particular, though, seem to have developed such a multitude of techniques that have different names. At first they seem very arbitrary and unrelated. Even if individuals can produce an analysis by working slowly through some of the software, they do not feel confident enough about the analysis to write a report or explain it to a supervisor. That could be a barrier to their attempting to do the analysis at all. Moreover, in trying to overcome those sorts of barriers, there are many software design barriers. One must identify the motivating philosophy, in attempting to deal with these issues, because there are many potential solutions that might conflict with other perceived truths in the statistics community.

PHILOSOPHY OF GUIDANCE

I was impressed by the degree to which all of the previous speakers seemed to embody the same kind of philosophy as mine. There seems to be a secular trend in the philosophy of statistics, and the textbooks have not caught up with it. The kind of textbooks and the type of statistical teaching that were so prevalent in the 1950s and 1960s and perhaps even into the 1970s are no longer accepted by expert users, as exemplified by today's speakers. Unfortunately, there is a Frankenstein monster out there of hypothesis testing and p values, and so on, that is impossible to stop. Most people think that statistics is hypothesis testing. There is a statistical education issue here for which I do not have a quick solution.

So here are the principles of my philosophy of guidance. Graphics should be somehow totally integrated, and one should not ever think of doing a data analysis without a graph. The focus should be on the task rather than the technique, emphasizing the commonality of different analysis problems. By keying on the commonality, what is learned in one scenario will help in another one. Different sample sizes, designs, and distributions must be smoothly supported. Merely having equal or unequal numbers in each group should not require that the user suddenly go to a different chapter of the user's manual. An occasional or infrequent user who doesn't understand why that should be necessary will be totally alienated. As mentioned before, hypothesis testing should be de-emphasized in favor of point and interval estimation. Simple, easy-to-visualize techniques should be chosen. Lastly, the statistical software should help as much as possible with the interpretation of the results and with the assembling of the report.

RECOGNIZING THE ONE-WAY ANOVA PROBLEM

With these guidance ideas in mind, one of the first things to note is that “one-way ANOVA” is, of course, jargon. What does that really mean? How are people to know that they should use a program called one-way ANOVA when they need to compare different groups?

It is easy to give guidance if the scientific questions can be rephrased in terms of simple variables. A statistical variable is not a natural thing. In statistics training, a random variable is drummed into the student earlier. It is better to phrase all of the statistical scientific questions in terms of questions about relations between variables. Instead of comparing the rats on this diet and the rats on that diet, one wants to know if there is a relationship in rats between weight gain and diet, with weight gain being one variable and diet another.

That is not a natural use of the language for most people. Yet software is much better used if the user has to think about a database of random variables and relationships between those variables. Forcing users to do that is, in a sense, a disadvantage of software, but also an ultimate advantage for users; if people understand and think about random variables, it will greatly help them to think about statistical issues in the right manner. Having users think in terms of variables may be doing them a favor.

Meta-data includes the description of variables in terms of their units, the types of the data, and so forth. Software should include specific spots for that kind of data; it is the means by which guidance on software becomes feasible. If one creates a data dictionary, each entry should include a variable name, a description, some indication as to whether it is categorical or a measurement scale, and what its range of values is.

A partition is a quite handy further refinement of this, if a variable can have several values and there is interest in a coarser strain of a given value. It may be easier for the software to address such subsets of values. With this situation, it is relatively easy to provide assistance. But a step must be taken to get the user to create those kinds of databases. Afterward, it is easy to talk about a response variable versus a predictor variable, or a factor versus a response. One might then ask, How is this categorical variable associated with that continuous variable? Short dialogues with the computer could address that. Once that dialogue is completed, the software should immediately display on the screen a graphical representation that has been integrated with the statistical analysis or inference procedure.

I do not much mind if a student confuses the definition of a distribution with the definition of a histogram, since one is a picture of the other. A histogram is something one can study, draw, and get a feel for, whereas a distribution is more an abstract concept. I would not mind if that student confused the issue that a one-way ANOVA is a method for looking at a set of box plots, namely, the representation of a continuous versus a categorical variable, and focused on the distributions in each category.

I believe the idea that a one-way ANOVA is an F-test is entirely wrong. A one-way ANOVA is merely a method for zeroing in on what a box plot might tell. Of course, there are many different ways one can do this. One way is to examine the plot and notice that there are a few outliers. There are quite a few directions one might want to go in that case: some statistical model or analysis tack may be preferable, depending on the data, or the system might suggest using means as a representation rather than medians, since there are not too many outliers.

ADAPTIVE FITTING PROCEDURE

Where to go after looking at the box plot is rather data dependent and also dependent on any other goals associated with the given data. There should be some automatic screening or adaptive fitting procedure as guidance, in which the software itself does the kinds of things that most statisticians would recommend. This includes such things as recognizing horribly skewed distributions, or recognizing when a response variable only takes two or three values, examining whether or not there are differences between the spreads in each group. The statistical software should then compose a model description and make some recommendations.

GUIDANCE FOR INTERPRETATION.

There are many different problems in interpreting such data, even though one-way ANOVA is thought of as one of the simplest of all statistical problems. But, as we heard this morning, even such a simple problem can be partitioned into many different tests. One can give descriptions of the model and/or data, explore the residuals, explain the ANOVA table, produce confidence intervals--to understand esoterica such as the simultaneous versus the non-simultaneous approach to estimating confidence intervals--and make confidence intervals for fitted values.

Again, software can help with that task. As an example, consider having software that produces a boilerplate description of the data being examined; for example,

> Data are drawn from the NURTUREDAT dataset. The variables GAIN vs DIET are modeled with N = 45. DIET is an unranked categorical variable. There are 3 levels of DIET all with sample size 15. The means of GAIN range from 81.1 (DIET = control) to 152.1 (DIET = liquid). There are three values of GAIN classed as extreme observations by the boxplot criterion.

Why would one want software to produce such a simple boilerplate description? From many years of teaching in various universities, I have learned that it is amazingly hard to produce a student who can reliably write such a paragraph. In actual fact, it is hard to get students to focus on describing these quantitative issues. The same is true with getting them to explain the single box plot and how to express in a couple of sentences a description of a confidence interval for the mean. Thus the software might also be capable of producing something such as

> If DIET = liquid, half of the 15 values of GAIN are within the boxed area of the plot, an interquartile range (IQR) of 28. There is 1 value classed as an extreme observation (more than 1.5 IQR from the nearest quartile). The group mean is 152.1, and the true mean is between 139 and 165.3 with 95% confidence.

These are the kinds of boilerplate descriptions that infrequent users especially, but even the people who are right in the middle of their course, have trouble producing.

Of course, that is even truer when it comes to explaining the results of an F test for ANOVA. So again, one could have a display such as

> There is strong evidence that DIET affects GAIN, since an F as large as 34.44 would only occur about 0% of the time if there were no consistent differences between DIET groups.

Such a display assures that the user does not reverse the situation and say that the F test is not significant when in fact it is.

Many statisticians would feel unhappy about software that produces such sentences, saying that there are not enough qualifications. On the other hand, if too many sentences are produced in qualification for that boilerplate, people are going to write front ends for the system in order to screen off the first four sentences of every paragraph, knowing those sentences will not say anything worthwhile.

Aside from the issue of data dependence interpretations, there are trickier issues, such as what one means by a simultaneous confidence interval. Suppose the following standard sentence is produced describing how to interpret a particular confidence interval:

> The true mean of GAIN where DIET = liquid, minus the true mean where DIET = solid, is approximately 10.9, and is between -14.7 and 36.7 using simultaneous 95% confidence intervals.

What does “simultaneous” mean? There needs to be at least some explanatory glossary, perhaps as an option, so that when a strange word is encountered a couple of sentences helping to define it can be given, such as

> Simultaneous 95% (or 99%, etc.) confidence intervals are wider, and therefore more reliable, than 95% nonsimultaneous intervals, because they contain the true difference for EVERY comparison in 95% of experiments while the later intervals are merely designed to be accurate in 95% of comparisons, even within one experiment. Use nonsimultaneous intervals only if the comparisons being displayed were of primary interest at the design stage of the experiment; otherwise you risk being misled by a chance result.

One of the primary areas where guidance is needed is in residual analysis, something we are all supposed to do. For those infrequent users, it is again a little intimidating. One can have many plots available, but it is not exactly clear to those infrequent users which they should examine. Part of the guidance can come from the structure of the menu system. One can make it quite easy to look at residuals, rather than having to save residuals as a separate variable and leave the original analysis to start up another analysis where some residual plots are done. By making residual analysis very easy, the user can be encouraged to try some of the menu items (e.g., which graphs to look at, what to look for on the graph, point out features of this graph) and see what happens.

GUIDANCE FOR REPORT WRITING

Another issue that software needs to address is guidance for report writing. Having the software keep some kind of log or diary is important. A log is a verbatim log, which has the beauty that when it is replayed, a repetition of all the actions takes place. A diary is oriented more toward an end user, as a means of keeping track of what the user has done. The diary needs to contain enough information to reproduce the analysis but not necessarily in that linear mode of the same steps that the user went through.

In order to reproduce a given analysis, all one really needs is an object state record that encodes such things as which cases were included and which model was being used. One does not necessarily need to record all the steps that led there. This kind of diary should collect interpretations produced by the system, as well as recording transformations and variable definitions, and it should also have a place to record the notes that the user might put in, along with references to tables and graphs that were saved. In the end, the user will have a compilation of information that gives solid help in assembling a report. Those boilerplate sentences have the right statistical jargon and are used appropriately with the right parts of speech, so that at least they can be captured and put into a report.

GUIDANCE REGARDING TACIT TECHNICAL ASSUMPTIONS

I now want to talk a bit more technically about some problems related to giving guidance. How can one overcome what are major pitfalls, violations of the major statistical assumptions made in the statistical model, in the one-way ANOVA layout? One assumption is that the variances are supposed to be equal in each group, and the other is that the mean is a good summary because the data is not too outlier prone.

Adjusting for Unequal Dispersions

The first problem concerns adjusting for unequal dispersions. When one looks for textbook advice on alternatives, the textbooks do not usually give very explicit advice, but rather provide implicit advice. Often, one part of a textbook will say that the variances are to be assumed equal, and it will not say what to do when they are not equal. There may be an inference of "first do this and then do that," but not something explicitly stated.

The difficulty with most comparisons of variances is that they are of very low power. The idea that one should always assume variances to be equal just because it cannot be proven otherwise is a tricky one. A large sample versus a small sample will yield radically different power. When there are many groups versus few groups, even the comparison of variances becomes quite muddied. If the distribution is not an exactly normal distribution, the typical outlier test for comparing variances is known to be biased. Regarding testing versus estimation, if one has huge sample sizes and one variance is proved significantly to be 20 percent bigger than another variance, that is not at all relevant to the question of what kind of procedure one should use for one-way ANOVA.

The infrequent and non-statistician user does not want to be concerned with all these technicalities. He or she wants software that will handle it all automatically.

One approach is tantamount to applying, in the background, an empirically based shrinkage estimator to the measures of dispersion, and using that estimator as the foundation of a guidance or automatic methodology. One does not want to be too sensitive to an outlier-prone distribution; one must distinguish between having an outlier and having a wider dispersion.

It is important that this estimator be based on the sample sizes. If there is a sample size 10 and a sample size 100 for the two groups to be compared, it is certain that one of the interquartile ranges is estimated much more accurately than the other. On the other hand, if there are 10 groups each of size 4, the fact that a few of those have interquartile ranges quite different from the average must not be overemphasized.

After obtaining such an estimator, one should, according to one perspective, do nothing unless the differences in the estimated interquartile ranges are great. After having shrunk them toward the average, however, if one is, say, double the other, then a method that assumes unequal variances might be recommended. In that case, it is presumed that the variances in each group are proportional to the squares of the shrinkage estimates of the interquartile ranges.

Adjusting for Outlier-prone Data

The second issue, adjusting for outlier-prone data, raises the issue of how the software can be more robust when comparing measures of location. As there is a huge literature on robust estimations, one again faces the question of how much complication to include. Most infrequent users, or scientists and industrial engineers, merely want the answer; they do not want to focus on the various techniques they could have chosen to get that answer. One must of course try to prevent misuses that can occur when the results of a computer package are religiously applied, and to make sure that a few extreme responses do not distort the actual statistical estimates that are being presented. On the other hand, if the data are not very outlier prone, most people would prefer to use the familiar least-squares estimates, and techniques based on means. This avoids having the software user continually defend the fact that the answers differ a little from those given by some other package.

When some non-classical approach is warranted, it should not necessitate learning an entirely new software interface. That is one of the biggest troubles regarding the so-called non-parametric techniques that were developed and popularized in the 1950s and 1960s. They are accompanied by a whole range of limitations whereby, although the same scientific problem is at issue, elegant solutions may not be available in some cases due to a technicality; very different-looking computer output might attend problems that appear superficially similar to a casual user. To facilitate focusing on the task instead of the technique, one might have to sacrifice some theoretical rigor that an alternate technique might have. Also, there is the question of how to choose between the more-and less-rigorous versions. An explicit threshold or criteria must be available.

In the context of one-way ANOVA, when the robust estimation is recommended, what might be done? The box plot has been seen as a generic graph describing the one-way ANOVA problem. In this robust version, the box plot of course represents the median as one of the more prominent points for each group. So it would seem that the median would be the natural thing to take as the alternative robust estimate, in order to have a tie-in with the graph that was being used to drive the analysis. The problem with using the median in a general linear model framework is that the median does not have an easily obtained sampling distribution. A more approximate approach is forced. One such approach is to have several samples rather than just one sample, and to use as a general measure of dispersion a multiple of the interquartile range of the residuals, after subtracting off each group median.

If the object is to provide an ease of use that allows the software user to focus on the task rather than the technique, some new techniques may have to be invented in conjunction with this. The users never see any of this; they just see a system suggestion that the confidence intervals for contrast be based on medians. They can override that suggestion if they want. Then they merely get confidence intervals for contrasts and predictions without needing to go into an entirely different software framework.

In summary, for the one-way ANOVA layout, in-software guidance is possible. There are, however, more complicated scenarios that could be called one-way ANOVA that are not covered by my remarks, e.g., issues about sampling methods and the validity of inference to target populations.

There is no doubt that whenever a piece of software provides some kind of guidance, it will offend a certain fraction of the statistics community. This is because whenever you give a problem to several statisticians, they each will come back with different answers. Statistics seems to be an art, and very hard to standardize.

More than anything else, the means to providing better guidance is to make the entire data analysis process more transparent. In the definition of one-way ANOVA, one should be looking at a box plot and determining if there is more that can be used there. If a person can interactively point and click, and so really get hold of a box plot, presumably more can be done with it. Perhaps a manipulation metaphor can make the goals of statistics more transparent and concrete, as well as the uncertainty measures produced by a statistical program. This is going to produce more for the guidance and overall understanding than boilerplate dialogues that attempt to mimic the discussion that an expert statistician might have with a client. Still, in order to produce that transparency, there will always have to be practical compromises with elegant theory before the ever-increasing numbers of data analysts can have ready access to the benefits of statistical methods.

Incorporating Statistical Expertise into Data Analysis Software

Daryl Pregibon
AT&T Bell Laboratories

OVERVIEW

Nearly 10 years have passed since initial efforts to put statistical expertise into data analysis software were reported. It is fair to say that the ambitious goals articulated then have not yet been realized. The short history of such efforts is reviewed here with a view toward identifying what went wrong. Since the need to encode statistical expertise in software will go on, this is a necessary step to making progress in the future.

Statistical software is guided by statistical practice. The first-generation packages were aimed at data reduction. Batch processing had no competition, and users had little choice. The late 1960s brought an era of challenging assumptions. Robust estimators were introduced “by the thousands” (a slight exaggeration), but these were slow to enter into statistical software, as emphasis was on numerical accuracy. By the late 1970s, diagnostics for and generalizations of classical models were developed, and graphical methods started to gain some legitimacy. Interactive computing was available, but statistical software was slow to capitalize on it. By the late 1980s, computer-intensive methods made huge inroads. Resampling techniques and semi-parametric models freed investigators from the shackles of normal theory and linear models. Dynamic graphical techniques were developed to handle increasingly complex data.

The capabilities and uses of statistical software have progressed smoothly and unerringly from data *reduction* to data *production.* Accomplished statisticians argue that alternative model formulations, diagnostics, plots, and so on are necessary for a proper analysis of data. Inexperienced users of statistical software are overwhelmed. Initial efforts to incorporate statistical expertise into software were aimed at helping inexperienced users navigate through the statistical software jungle that had been created. The typical design had the software perform much of the diagnostic checking in the background and report to the user only those checks that had failed, possibly providing direction on what might be done to alleviate the problem.

Not surprisingly, such ideas were not enthusiastically embraced by the statistics community. Few of the criticisms were legitimate, as most were concerned with the impossibility of automating the “art” of data analysis. Statisticians seemed to be making a distinction between providing statistical expertise in textbooks as opposed to via software. Today the commonly held view is that the latter is no more a threat to one's individual methods and prejudices than is the former.

Given the weak support by peers in the field, and the difficulties inherent with trying to encode expertise into software, some attempts were made to build tools to help those interested in specific statistical topics get started. These tool-building projects were even more ambitious than earlier efforts and hardly got off the ground, in part because existing hardware and software environments were too fragile and unfriendly. But the major factor limiting the number of people using these tools was the recognition that (subject matter) context was hard to ignore and even harder to incorporate into software than the statistical methodology itself. Just how much context is required in an analysis? When is it used? How is it used? The problems in thoughtfully integrating context into software seemed overwhelming.

There was an attempt to finesse the context problem by trying to *accommodate* rather than *integrate* context into software. Specifically, the idea was to mimic for the whole analysis what a variable selection procedure does for multiple regression, that is, to provide a multitude of context-free “answers” to choose from. Context guides the ultimate decision about which analysis is appropriate, just as it guides the decision about which variables to use in multiple regression. The separation of the purely algorithmic and the context-dependent aspects of an analysis seems attractive from the point of view of exploiting the relative strengths of computers (brute-force computation) and humans (thinking). Nevertheless, this idea also lacked support and recently died of island fever. (It existed on a workstation that no one used or cared to learn to use.)

So where does smart statistical software stand today? The need for it still exists, from the point of view of the naive user, just as it did 10 years ago. But it is doubtful that this need is sufficient to encourage statisticians to get involved; writing books is much easier. But there is another need, this one selfish, that may be enough to effect increased participation. Specifically, the greatest interest in data analysis has always been in the process itself. The data guides the analysis; it forces action and typically changes the usual course of an analysis. The effect of this on inferences, the bread and butter of statistics, is hard to characterize, but no longer possible to ignore. By encoding into software the statistician's expertise in data analysis, and by directing statisticians' infatuation with resampling methodology, there is now a unique opportunity to study the data analysis process itself. This will allow the operating characteristics of several tests applied in sequence--or even an entire analysis, as opposed to the properties of a single test or estimator--to be understood. This is an exciting prospect.

The time is also right for such endeavors to succeed, as long as initial goals are kept fairly limited in scope. The main advantage now favoring success is the availability of statistical computing environments with the capabilities to support the style of programming required. Previous attempts had all tried to access or otherwise recreate the statistical computing environment from the outside. Keeping within the boundaries of the statistical computing environment eliminates the need to learn a new language or operating system, thereby increasing the chance that developers *and* potential users will experiment with early prototypes. Both are necessary for the successful incorporation of statistical expertise into data analysis software.

WHAT IS MEANT BY STATISTICAL EXPERTISE?

What do I mean by statistical expertise? Let me recommend *How to Solve It* by George Polya [Polya, 1957]. It is a beautiful book on applying mathematics to real-world problems. Polya differentiates four steps in the mathematical problem-solving process: (1) understanding the problem, (2) devising a plan to solve the problem, (3) carrying out the plan, and (4) looking back on the method of solution and learning from it.

All four steps are essential in mathematical problem solving. For instance, devising a plan might consist of, say, using induction. Carrying out the plan would be the actual technical steps involved. Having proved a specific fact, one might look back and see it as a special case of something else and then be able to generalize the proof and perhaps solve a broader class of problems.

What kind of expertise do I want to put into software? Many of the steps that Polya outlines are very context dependent. Knowledge of the area in which the work is being done is needed. I am not talking about integrating context into software. That is ultimately going to be important, but it cannot be done yet. The expertise of concern here is that of carrying out the plan, the sequence of steps used once the decision has been made to do, say, a regression analysis or a one-way analysis of variance. Probably the most interesting things statisticians do take place before that.

Statistical expertise needs to be put into software for at least three reasons. The first of course is to provide better analysis for non-statisticians, to provide guidance in the use of those techniques that statisticians think are useful. The second is to stimulate the development of better software environments for statisticians. Sometimes statisticians actually have to stoop to analyzing data. It would be nice to have help in doing some of the things one would like to do but has neither the time nor the graduate students for. The third is to study the data analysis process itself, and that is my motivating interest. Throughout American or even global industry, there is much advocacy of statistical process control and of understanding processes. Statisticians have a process they espouse but do not know anything about. It is the process of putting together many tiny pieces, the process called data analysis, and is not really understood. Encoding these pieces provides a platform from which to study this process that was invented to tell people what to do, and about which little is known.

One of the most compelling reasons for resuming efforts to try to infuse guidance into statistical software, and to implement plans, is to better understand the process of analyzing data. But some of this is also motivated by folly. Part of my earlier career dealt with regression diagnostics, which is how to turn a small-sample problem into a large-sample problem. One can increase the size of the data set by orders of magnitude. Just 100 or 200 data points can easily be increased to thousands or tens of thousands. The larger set is highly correlated, of course, and may be reiterating the same information, but it can be produced in great quantity.

In the good old days, there was data reduction. This is what analysis of variance did. What began as a big body of data was reduced to mean and standard errors. Today with all the computing and statistics advances, the opposite end of the spectrum has been reached. Ever more data is produced, overwhelming the users. Some of that has to be suppressed.

WHO NEEDS SOFTWARE WITH STATISTICAL EXPERTISE?

The audiences for statistical software are many and varied. Infrequent users probably make up the majority of users of statistical software. They want active systems, systems that take control. In other words, they want a black box.

Most professional statisticians are probably frequent users. These users want to be in control, want passive systems that work on command. One might call such a passive system a *glass box,* indicating that its users can see what it is doing inside and can understand the reasoning that is being used. If such users do not like what they see in the box, they will throw away the answer.

But there is a range of things in between. Problems are inevitable because, in having to deal with users with very diverse needs and wants, it is hard to please all users. When building expertise into software it must be remembered that good data analysis relies on pattern recognition, and consequently graphics should be heavily integrated into the process. Most of what is seen cannot be simply quantified with a single number. Plots are done in order to see the unexpected.

LIMITATIONS TO THE INCORPORATION OF STATISTICAL EXPERTISE

A lot of the experience that goes into data analysis cannot be very easily captured. Moreover, good data analysis relies on the problem context; statistics is not applied in a void. It is applied in biology, as well as in business forecasting. The context is very important, although it is very hard to understand when and where it is important. Related to this is the difficulty of developing the sequence of required steps, the strategy. Finally, implementing that strategy is hard to do. Some hard trade-offs must be made; some engineering decisions that are not really statistically sound must be made. When a simulation is run, crude rules of thumb evolve and get implemented. Thus hard engineering decisions must be made in order to make any progress here.

One guiding principle to follow consistently when incorporating expertise into software, and not merely for statistical software, is this: whenever something is understood well enough, automate it. Matrix inversion has been automated, because it is believed to be well understood. No one wants to see how a matrix inversion routine works, and so it is automated.

Procedures that are not well understood require user interaction. In all of the systems described below, there are various levels of this automation-interaction trade-off.

EFFORTS TO BUILD DATA ANALYSIS SOFTWARE

REX

Going back into recent history, there was a system that Bill Gale and I were involved with called REX (circa 1982), an acronym for Regression EXpert [Gale, 1986a]. It was what one might call a front end to a statistics system, where the thing between the statistics system expertise and the user was an interface called REX. It was a rule-based interpreter in which the user never talked directly to the statistics expertise, but only talked to it through this intermediary. The user would say something such as, "regression Y on X." That was, in fact, the syntax, and was approximately all that the user could say. Then, if everything went right and this case was for the most part textbook data that satisfied a whole bunch of tests, REX would come out with a report, including plots, on the laser printer.

If one of the tests failed, REX would say, "I found a problem and here is how I would suggest you fix it." After REX offered the suggestion, the user could pick one of five alternatives: implement that suggestion, show the user a plot, explain why that suggestion was being made, describe alternatives (if any), or quit as a last resort.

REX was fairly dogmatic in that respect. If it found a severe problem and if the user refused all the suggestions, it would just say that it was not going to continue. Such intransigence was fine; in an environment where there were no users, it was easy to get away with that. If there had been any users, one can imagine what would have happened: the users would have simply used some other package that would have given them the answers.

What did REX do? It encoded a static plan for simple linear regression. It allowed a non-statistician to get a single good analysis of the regression, and it provided limited advice and explanation. It was an attempt to provide a playground for statisticians to build, study, and calibrate their own statistical plans.

On the subject of building, REX was actually built by a sort of bootstrapping. There was no history available on how to do something such as this. Instead, about a half-dozen examples were taken, all fairly small simple regression problems. Your speaker then did many analyses and kept a detailed diary of what was done, and why it was done. By knowing when something was done in the analysis sequence, one could study those steps and try to extract what commonality there was. Next, Bill Gale provided an architecture and a language to encode that strategy in the form of a *fixed* decision tree with if-then rules associated with each node, such as is shown in Figure 4. Each procedure encoded one strategy based on about a half-dozen examples.

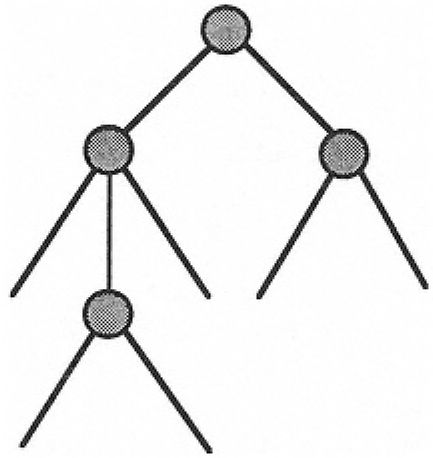

FIGURE 4: Example of a fixed decision tree.

It had been hoped that the language provided in REX would be a fertile playground for others to bootstrap their own interests, whether in time series, cluster analysis, or whatever. We knew of no other way to build strategies.

As to "features," please notice the use of quotation marks. Whenever a software developer mentions a feature, beware. Features are often blunders or inadequacies. In the case of REX, I will even tell you which ones are which.

REX had a variable automate-interact cycle. This is a positive feature whereby, if the data were clean, the user really did not have to interact with the system. In that situation, the user encountered no bothers. It was not like a menu system in which one has to work around the whole menu tree every time. But if the data set was problematic, REX would halt frequently and ask the user for more information so that it could continue.

With REX, the user was insulated from the statistical expertise. That is one of those dubious features. Many users are quite happy that they never have to see any underlying statistical expertise. Others are crying to do something on their own. This is related to a third so-called feature, that REX was in control. If the user had a different ordering of how he or she would like to do regression--e.g., looked at X first and then at Y, and later just wanted to look at Y first and then X--that could not be done. It was very pragmatic; one could not deviate from the path. Again, a certain class of users would be perfectly happy with that.

Another feature was that the system and language were designed to serve as an expert system shell that could be adapted for types of analyses other than regression.

Several things were learned from the work on REX. The first was that statisticians wanted more control. There were no users, rather merely statisticians looking over my shoulder to see how it was working. Automatically, people reacted negatively. They would not have done it that way. In contrast, non-statisticians to whom it was shown loved it. They wanted less control. In fact they did not want the system--they wanted answers.

To its credit, some of the people who did like it actually learned from REX. Someone who did not know much statistics or perhaps had had a course 5 or 10 years before could actually learn something. It was almost like an electronic textbook in that once you had an example, it could be an effective learning tool.

The most dismaying discovery of all was that not only did the statisticians around me dislike it, but they were also not even interested in building their own strategies. This was because the environment was a bit deficient, and the plan formulation--working through examples and manually extracting commonality in analyses--is a hard thing to do.

REX died shortly thereafter. The underlying operating system kept changing, and it just became too painful to keep it alive. There were bigger and better things to do. It was decided to next attack the second part of the problem, to get more statisticians involved in building expertise into software. As it was known to be a painful task, the desire was to build an expert system building tool, named Student [Gale, 1986b].

Student

I was fairly confident that the only way to end up with strategies and plans for understanding the data analysis process was by recording and analyzing the working of examples. Yet taking the trees of examples, assimilating what was done, and encoding it all into software was still a hard process. So Student was an ambitious attempt to look over the shoulder and be a big brother to the statistician. It was intended to watch the statistician analyzing examples, to capture the demands that were issued, to maintain a consistency between what the statistician did for the current example versus what had been done on previous examples, and to pinpoint why the statistician was doing something new this time. The statistician would have to say, e.g., “this case was concerned with time theories,” or “there was a time component,” and Student would thereby differentiate the cases and then proceed.

The idea was that the statistician would build a system encapsulating his or her expertise simply by working examples, during which this system would encode the sequence of steps used into software. In terms of the architecture, everything resided under the watchful eye of Student, with the statistician analyzing new data within the statistics package. Student watched that person analyze the data and tried to keep him or her honest by saying, “What you are doing now differs from what you did in this previous case; can you distinguish why you did it this way this time and that way that time?” However, for any analysis there is always an analysis path. Student was merely going to fill out the complete tree that may be involved in a more complicated problem.

The Student system also had features. Statisticians did not have to learn an unfamiliar language or representation, but simply used the statistics package that they ordinarily used, with no intermediate step (in contrast to REX, which had been written in LISP).

"Knowledge engineering" was a buzz word 10 years ago in building such systems. A knowledge engineer was someone who picked the brain of an expert and encoded that knowledge. With Student, the program was doing that, and so one only needed to engineer a program, and it would subsequently do all the work for any number of problems.

The system was again designed so that others could modify it, override it, and extend it to other analyses. Basically, that "feature" did not work, and there are two reasons why. One is that the Student system again was written in LISP. It actually ran on a different computer than that which was running the statistical software. With a network between the two, there were numerous logistical problems that got in the way and made things very tedious.

Perhaps the other reason it died was the gradual realization of the problem concerning context. It is just not sufficient to capture the sequences of commands that a statistician has issued in the analysis of data. Context is never captured in key strokes. There is almost always some "aha!" phase in an analysis, when someone is working at a knot and all of a sudden something pops out. Almost always that "aha!" is related to the context. Beyond the technical troubles in building and debugging a system such as Student was the realization that, in actuality, the wrong problem was being solved. That was what caused the project to be abandoned before progress was ever made.

TESS

That did not bring efforts to a complete halt, however. In 1987, a completely different approach was tried next with a system called TESS. The realization that context is important, and that how to incorporate it into strategy was unknown, led to the selection of an end-around approach. Context would be accommodated rather than incorporated. In much the same way that computer chess programs worked, the accommodation would be done by sheer brute-force computation.

Most statisticians have used subset regression or stepwise regression. These programs are simply number crunchers and do not know anything about context. They do not know, for instance, that a variable *A* is cheaper to collect than variable *B* or that the two variables are both measuring the same thing. Statisticians think that these regressions are useful, that they help to obtain a qualitative ordering on the variables and thereby perhaps help give clues to which classes of models are interesting. After the subset selection is done and the class of models is considered, context is typically brought in to pick one of the models, after which that chosen model is used to make inferences.

The idea of TESS (Tree-based Environment for developing Statistical Strategy) was to expand that game (of looking at the subset) to the overall selection of the model. For example, in performing a regression with a search-based approach, one defines a space of descriptions D for a class of regression data sets Y, where those data sets are actually ordered pairs, y and x. The goal is to get a hierarchy of the descriptions. Since some of the regressions are going to fit better than others and some will be more concise than others, one tries to order them. After all possible regression descriptions have been enumerated, the space D is searched for good ones. The user interface of this system is radically different from those of the previous two. The procedure is to tell the computer, “Give me 10 minutes worth of regressions of y on x.” (Two transformations were involved to re-express variables. Often, samples or data would be split into two, and outliers would be split off.) At the end of 10 minutes, a plot is made for each description in the space. There is a measure of accuracy, and also a measure of parsimony, and so a browsing list is started. This can be thought of as a CP plot. For each element of a CP plot, the accuracy is measured by CP, and the verbosity is measured by the number of parameters.

For TESS, that concept was generalized. When attempting to instill guidance into software, this overall approach will be important. For TESS, it was necessary to develop a vocabulary of what seemed to be important in describing regression data, i.e., the qualitative features of the data. These were not merely slopes and intercepts, but rather qualitative features closely linked with how data are represented and summarized. To organize all these things, a hierarchy was imposed and then a search procedure devised to traverse this hierarchy. The hope was to find good descriptions first, because this search could in principle continue forever.

TESS had three notable “features”: it was coded from scratch outside of any statistical system; it had a single automate-interact cycle, which permitted context to guide the user in selecting a good description; and once again, there were broad aspirations that the design would stress the environment tools so as to allow others to imitate it.

General Observations on TESS and Student

TESS and Student were very different in how they dealt with the issue of context, but there were several similarities. Both embodied great hopes that they would provide environments in which statisticians could ultimately explore, build, and analyze their own strategies. Ultimately, both used tree-like representations for plans. They both used examples in the plan-building process. And lastly, both are extinct.

Why did others not follow suit? There are a number of reasons. It was asking a lot to expect an individual to develop a strategy for a potentially complex task, and to learn a new language or system in which to implement it. These plans with which Student and TESS were concerned are very different from what is ordinary, usual thinking. A statistician, by training, comes up with a very isolated procedure; for example when one looks for normality, only a test for normality is applied, and it is assumed that everything else is out of the picture. Once all the complications are brought in, it is more than most people can sort out.

The other barrier was always learning a new language or system to implement these things. Even if we got Student running, there was a problem in how we designed it from the knowledge-engineering point of view. It was always outside the statistical system; there was always some additional learning needed.

The implication of all of this is the need to aim at smaller and simpler tasks. Also, an overriding issue is to stay within the bounds of the statistical system. There is a great deal of potential here that did not exist 5 years ago. With some modern statistical-computing languages, one can do interesting things within a system and not have to go outside it.

Mini-expert Functions

A final type of data-analysis software that possesses statistical expertise is a mini-expert function. These could pervade a statistical package, so that for every statistical function in the language (provided it is a command-type language), many associated mini-expert functions can be defined.

The first example of a mini-expert function is interpret(funcall, options), which provides interpretation of the *returned* output from the function call; i.e., it post-processes the function call's output. For instance, the regression command regress(x, y) might produce some regression data strategy. Applying the interpret function to that output, via the command “interpret(regress(x, y)),” might result in the response “regression is linear but there is evidence of an outlier.” The idea here is that “interpret” is merely a function in the language like any other function, with suitable options.

The next(funcall, options) mini-expert function has much the same syntax. When it is given a function call and options, it will provide a suggestion for the next function call. It could serve as a post-processor to the interpret function, and entering “next(regress(x, y))” might result in the response “try a smooth of y on x since there is curvature in the regression.” Options might include ranking, or memory to store information as to the source of the data that it was given.

Let us give an example of how the user interface would work with mini-expert functions. In languages where the value of the last expression is automatically stored, by successively entering the commands “smooth(x, y)” and “interpret(tutorial = T),” the interpreter would interpret the value of that smooth by default. It would not have to be explicitly given the function call name itself. As a second example, if the value of whatever had been done were assigned to a variable, say “z ← tree(x, y),” a plot of the variable could be done, e.g., “treeplot(z).” Then the variable could be given to the function next via “next(z)” in order to have that mini-expert supply a message suggesting what to do next.

For those who really want a black box, the system can be run in black-box mode. If one enters “regress(x, y)” followed by “next(do it = T), next(do it = T), …,” the system will always do whatever was suggested. In this way, the system can accommodate a range of users, from those who want to make choices at every step of the data analysis to those who want a complete black-box expert system.

One point to note: these mini-expert functions have not even been designed yet. This is more or less a plan for the future. As to "features," these functions possess a short automate-interact cycle, the user remains in complete control (but can relinquish it if desired), they are suitable for novices and experts alike, they exist within the statistical system, they are designed for others to imitate, and they force the developer to think hard about what a function returns.

CONCLUDING OBSERVATIONS

As to the future of efforts to incorporate statistical expertise into software, progress is being made, but it has been very indirect and quite modest. Work in this area may subtly influence designers of statistical packages and languages to change. Effort in this direction parallels the contributions of artificial intelligence research in the areas of interactive computing, windowing, integrated programming environments, new languages, new data structures, and new architectures.

There are some packages available commercially that attempt to incorporate statistical expertise, if one is interested in trying such software. Since the particular systems that were described in this presentation do not exist now, anyone with such an interest would have to look for what is available currently.

Finally, let me mention some interesting work that may stimulate readers who feel that incorporating software into data analysis plans is important. John Adams, a recent graduate from the University of Minnesota, wrote a thesis directed at trying to understand the effects of all possible combinations of the things that are done in regression for a huge designed experiment [Adams, 1990]. The design not only involved crossing all of the various factors, but also involved different types of data configurations, number of co-variances, correlation structure, and so on. The goal was to learn how all those multiple procedures fit together.

Prior to any incorporation of statistical expertise into data analysis software, that sort of study must first be done. In a way, it amounts to defining a meta-procedure: statistical expertise will not be broadly incorporated into software until that kind of understanding becomes a standard part of the literature.

REFERENCES.

Adams, J.L., 1990, Evaluating regression strategies, Ph.D. dissertation, Department of Statistics, University of Minnesota, Minneapolis.

Gale, W.A., 1986a, REX review, in *Artificial Intelligence and Statistics,* W.A. Gale, ed., Addison Wesley, Menlo Park, Calif.

Gale, W.A., 1986b, Student--Phase 1, in *Artificial Intelligence and Statistics,* W.A. Gale, ed., Addison Wesley, Menlo Park, Calif.

Polya, G., 1957, *How to Solve It,* 2nd edition, Princeton University Press, Princeton, N.J.

Afternoon Discussion

TURKAN GARDENIER (Equal Employment Opportunity Commission): I totally agree with Andrew Kirsch in his opinions on the need for configurable output. In litigation cases, an offense-defense strategy is played in which one has to selectively provide information to the opposing party without giving too much information. If too much is provided, not only might they fail to comprehend it, but--attorneys have told me--it could also be used against the person providing it.

ANDREW KIRSCH: The same risks are faced by engineers who must present their data analyses to others.

TURKAN GARDENIER: It has sometimes been necessary to cut and paste the standard statistical output, which is not kosher. If asked, “Did you present all the information the computer gave?” do you answer yes or no? If you reply that the omitted information is not relevant to the issue, they say, “Present everything,” and then they ask you to explain everything in depositions. The deposition consumes three hours on some trivial case in the computer output that is totally irrelevant to the issue, and frequently confuses the opposing party rather than shedding light on the issue. There needs to be a road map for selecting information to be displayed, depending on the capability of the user to either understand it or use it.

ANDREW KIRSCH: That is as much an issue in learning as it is in litigation. I tell my students that there are professionals whose careers are based on adding new kinds of tests and checks on various kinds of statistical analyses. In order to know everything about the output, individuals have to make a career of it themselves.

PAUL VELLEMAN: There seems to be a mild contradiction in this. I love Andrew's conclusion that ease of use is more important. Somehow, software vendors never hear “please give us more ease of use” from users. They hear “please give us another feature.” Andrew began with a list of capabilities. If a vendor's package does not have one of those capabilities, and the vendor wants 3M to use that package, that missing capability must be added, rather than the program being made easier to use. If greater ease of use is of prime importance, and I believe it is, then that is what the package developers need to hear from the user community.

ANDREW KIRSCH: I agree. That very comment, on being willing to set aside adding new capabilities for the sake of ease of use, has been made by 3M to some people present today.

By the way, 3M does not try to meet all four of those categories (acceptable cost;

core statistical capabilities; availability for a variety of hardware environments, with interfaces to common database, spreadsheet, word processing programs and ready availability of support along with complete, readable documentation; and lastly, ease of use for infrequent users) with one software. 3M has separate software for acceptance sampling, design of experiments, and so on. When going through an evaluation process, it is really hard to say what are core capabilities.

ERIC STELLWAGEN (Business Forecast Systems, Inc.): As a developer of software, we never hear from our users concerning the vast majority of products that go out the door. The reason is that we strive very hard to make them easy to use.

Now, we have fallen into the trap of having different products to aim at different markets. Our most sophisticated product, statistically speaking, is the one on which we get the greatest number of comments, such as, “It does not have this test,” or “Do you not know about this other technique?” But those comments are coming not from infrequent users, but from people who are using the product every day in their work.

The vast majority of our clients are the ones who once a month take a program off the shelf, do their sales forecast, and put it back on the shelf until next month. So the points expressed at this forum mirror exactly my experience.

PAUL TUKEY: I think we have to make a clear distinction between ease of use and oversimplification of the technique. I, too, believe in ease of use. For instance, one of the reasons I like S is that it has high-level objects that are easy to use, because they are self-describing. This means I do not have to be constantly reminding the system of all the attributes.

But there is the danger of mixing up ease of use with pretending that the world is simple when it is not. If naive people do naive analyses, e.g., fit a straight line when a straight line is not appropriate to fit, they may miss the things that are happening, because they are using Lotus. Frankly, Lotus is not all that easy to use for statistics. Try to program a regression in the Lotus programming language; it is horrendous.

This is not a plea for oversimplified statistics. Some may worry that if software is made too easy to use, it sweeps the hard problems under the rug and everybody is left to believe that the world is always simple, when that is not necessarily so.

ANDREW KIRSCH: I tried to mention the idea of ease of flow through the program as well as that of limiting the amount of output generated. But certainly there is a danger in limiting generated output if you reduce the amount of data generated to the point that you are vastly oversimplifying. So users must keep focused on those things that are most important, because this large laundry list of output, of potential things to consider and act on, does not provide much focus.

FORREST YOUNG: Returning to the topic of guidance, one way of making systems easier to use is to not “cripple” them by taking options out or perhaps hiding options, but to make suggestions to the user as to which options, what kinds of analyses, are the ones that are appropriate at this point. The software should not give the appearance that there is only one thing the user can do, namely the thing that is being suggested by the system. Rather, the program should identify the things some expert thinks would be reasonable things to do next. That would certainly make systems easier to use for relatively new users.

TURKAN GARDENIER: Recently I received a demonstration copy of a menu-driven tutorial called Statistical Navigator that explains when to use statistical tests. It encompasses most of the modules but does not analyze data. It provides menus, with questions and answers and with information about the data it is given, and gives advice about what test to use.

DARYL PREGIBON: There is a publication that has probably been out about 10 years from the University of Michigan, I believe from the School of Social Sciences. It is a small monograph or paperback that basically does what that software implements. This Navigator tutorial is probably one of the first to implement the ideas into software.

PAUL TUKEY: Daryl, your enthusiasm is clear about solving the expert system problem in the near future. What are your thoughts on extracting some of the lessons from that experience and incorporating pieces into some more classical kinds of statistical software packages, so that one has a better understanding of what are the good diagnostics and so on? Such programs would not necessarily always say what should be done next, but could at least perform some operations in the background.

DARYL PREGIBON: Bill DuMouchel mentioned something related to that, the concept of meta-data. These are qualitative features, such as the units of measurements and the range of the variable, that can and should be integrated into the software. One can go to great lengths to provide a certain amount of guidance, but it also requires some discipline. If you have a procedure to do a Box-Cox analysis, you want it to involve more than just the numerical values of X and Y; you also want this meta-data. Many statisticians have probably written software procedures for Box-Cox, but how many of you have written any that truly ask or require this meta-data to be present? Yet in Box-Cox's paper [Box and Cox, 1964], approximately the first five pages were on the qualitative aspects of data transformations. There were various guidelines there, and after following them one can then do optimization.

Anyone who has created software that implements only the procedure is at fault. Statisticians have factored out all that qualitative meta-data from their routines. There is much room for improvement in applying the lessons that have already been learned, to simply bring that existing guidance into analyses. That would drastically improve the level of guidance available in statistical software and prohibit indiscriminate transforming of variables when there is no rationale for doing so on physical or other grounds.

KEITH MULLER: This goes back to the ultimate expert question where, when individuals come to a statistical consultant, often they are asking the wrong question of the data. Daryl's idea of meta-methods, in which a shell is built around the little pieces to try to build a bit bigger piece, is very attractive. But the insolubility of that problem of asking the wrong question may be the reason that many of us were originally so cynical about the feasibility of expert systems.

DARYL PREGIBON: I emphatically agree. The most interesting and difficult parts of an analysis problem are recognizing at the outset what the problem really is, and understanding what to disregard in order to reduce the problem to something for which one can draw up a sequence of steps.

CARL RUSSELL (U.S. Army Operational Test and Evaluation Command): When you mentioned mini-steps, I expected you to discuss the kind of implementation that is similar to your “next” function that is in both Jump and Data Desk. For those packages, whenever the user has an individual window, he or she can go up and essentially get it next. Those little steps are already implemented, at least in a couple of packages.

DARYL PREGIBON: Everyone should explore this approach in more detail. Those vendors probably saw the light before I did on this, which is why they were successful where I was not.

WILLIAM DUMOUCHEL: Every menu system helps to some extent in that manner, because there is a particular structure in which you want to do one set of analyses, while only in certain contexts have you completed other analyses.

FORREST YOUNG: What is your opinion about the future of expert systems?

WILLIAM DUMOUCHEL: That is a rather general question. I agree with Daryl about how hard the problem is in the absence of making use of context. The best thing a software company can do is to provide tools for users to design their own on-site expert environments. If a general tool is offered that has menu-building tools in it as well as other kinds of extensibility options, then a group at 3M who know what kind of data they usually use can fine-tune it and put in expertise.

The future effort toward incorporating expertise will be to try to make everything more concrete. Desktop metaphors seem to work wonders in many aspects of how to use computers quickly and easily. If we can think of other metaphors that help in the data analysis software line, then somehow we can make this whole process more transparent. Expertise can be more an invisible rather than an explicit thing.

CLIFTON BAILEY: With expert systems, one is often asking to deal with models that are inherently not in the class of models that are normally considered in everyday statistical work. Also, in putting expertise into a product, one needs to be aware that there are certain symmetries and structures that would never be followed in a particular context.

ANDREW KIRSCH: To reinforce a well-made point of Daryl's, the necessary antecedent to statisticians making progress in the world of expert systems is to rethink and specify what it is that statisticians do. It is not just automating some system, but is more akin to asking in advance whether it is laid out in the best possible way. Simply automating may be an inferior practice. The problem here is not inferior practice, but that the practice has not been adequately specified.

AL BEST (Virginia Commonwealth University): These ideas are all very exciting, but clearly a lot more needs to be known before much can be done that is concrete in promulgating guidelines. Should there be guidelines on whether there needs to be a “next” function, or guidelines on whether a user should be allowed to get only a system table without a graph? There are many questions here.

REFERENCE

Box, G.E.P., and D.R. Cox, 1964, An analysis of transformations, *J. Roy. Stat. Soc. B,* Vol. 26, 211–252 (with discussion).

Closing Remarks

William Eddy
Carnegie Mellon University

This is a good moment for me to make a long series of rambling remarks and comments that may or may not address that last question.

First, I want to thank all of you for coming and providing a number of very insightful dimensions that the panel had not considered in their deliberations before today. I want to thank all the speakers and my colleagues on the panel. I also want to thank the National Science Foundation (NSF) for its financial support, and I would like to thank the staff at the National Research Council who organized this forum.

As was mentioned this morning, the panel welcomes additional input from you, preferably in writing. To comment on the presentations starting from this morning, Paul Velleman made a statement to the effect that good statistical software drives bad statistical software out of the marketplace. I happen to not believe that. In fact, for quite a long time I have been saying there is an explicit Gresham's law for statistical software, namely, the bad drives out the good. A1 Thaler of the NSF recently pointed out to me that my view on this is not completely correct, that rather there is a modified law: if the software is good enough, it does not have to get any better. That is certainly the case. As a supporting example, I am currently teaching a course to undergraduates. The particular statistical package being used is identical to the one I used in 1972. This package has not changed an iota in 19 years. It was good enough, and so it did not need to change.

Underneath all of these discussions is software. Software is one of the most unusual commodities that humans have ever ever encountered. It has a very unique feature in that if the tiniest change is made in the input, in the program, in the controls, or in anything, there can be huge changes in what results; software is in this way discontinuous. This fact drives a great deal of work in software engineering and reliability, but it should also drive our thinking about statistical software.

In relation to this, Bill DuMouchel's mention of switching from means to medians intrigued me because I know that there exist sets of numbers that are epsilon-indifferent, but which would produce substantially different answers in his system. That switch is not a smooth transition, but is instead a switch that discontinuously kicks in. The fact that it is such a switch troubles me, because I like things to vary in a smooth way. If there is one single goal that I would set for software, it is that ultimately the response would be a smooth function of the input. I do not see any way to achieve this, and cannot offer a solution.

Another thing that troubles me is that there are software systems available in the marketplace that do not meet what I would consider minimally acceptable quality standards on any kind of measure you select. These systems are not the 6 or 10 or 20 big ones that everyone knows. There are 300 of them available out there. A lot of bad software is being foisted on unsuspecting customers, and I am deeply concerned about that.

To put it in concrete terms, what does it take for the developer of a statistical package to be able to claim that the package does linear least-squares regression? What minimal functionality has to be delivered for that? There exists software that does not meet any acceptability criteria--no matter how little you ask.

Another matter troubling me, even more so now that I am a confirmed UNIX user, is the inconsistent interfaces in our statistical software systems. Vendors undoubtably view this as a feature, it being what distinguishes package A from package B. But as a user, I cannot think of anything I hate more than having to read some detestable manual to find out what confounded things I have to communicate to this package in order to do what I want, since the package is not the one I usually use. This interface inconsistency is not only in the human side of things, where it is readily obvious, but also on the computer side. There is now an activity within the American Statistical Association to develop a way to move data systematically from package to package.

Users of statistical software cannot force vendors to do anything, but gentle pleas can be made to make life a little easier. That is what this activity is really about, simplifying my life as well as yours and those of many others out there.

There was discussion this morning on standardizing algorithms or test data sets as ways to confirm the veracity of programs. That is important, but what is even more important is the tremendous amount of wasted human energy when yet another program is written to compute $X^TX^{-1}X^TY$. There must be 400 programs out on the market to do that, and they were each written independently. That is simply stupid. Although every vendor will say its product has a new little twist that is better than the previous vendor's package, the fact is that vendors are wasting their resources doing the wrong things.

We heard several speakers this afternoon discuss the importance of simplifying the output and the analysis. If vendors would expend effort there instead of adding bells and whistles or in writing another wretched regression program, everyone would be much better off. The future report of the Panel on Guidelines for Statistical Software will include, I hope, some indications to software vendors as to where they should be focusing their efforts.

I do not know the solution to all these problems. Standards are an obvious partial solution. When a minimum standard is set everybody agrees on what it is and, when all abide by the standard, life becomes simpler. However, life does not become optimal. If optimality is desired, then simplicity must usually be relinquished. Nevertheless, standards are important, not so much in their being enforced, but in their providing targets or levels of expectation of what is to be met.

I hope this panel's future report will influence statistical research in positive ways, in addition to having positive effects on software development, and thereby affect the software infrastructure that is more and more enveloping our lives at every turn.

The critical thing that is most needed in software is adaptability. I loved Daryl Pregibon's “do.it = True” command. That is an optimal command. From now on, that is all my computer is going to do, so that I do not have to deal with it anymore. An afterthought: an essentially similar function is needed in regard to documentation, namely, “GO AWAY!” Why is it that documentation has not evolved very far beyond the old program textbooks? There is much room for improvement here, and it is sorely needed.

One interesting thing expressed today was that maybe it is *statistics,* and not just the software, that needs to be changed. As said repeatedly by several speakers, statisticians tend to focus on the procedure rather than the task. That is an educational problem that we as academics need to address in the way statistics is taught, and so this relates to methodological weaknesses in statistical training. The numbers that come out of a statistical package are not the solution; numbers are the problem. There is far too much focus on having the software produce seven-digit numbers with decimal points in the right place. Statistical software must be made more sophisticated, to the point that it “talks” to the user and facilitates correct and appropriate analyses. A paraphrase of an old aphorism says that you can lead a fool to data, but you can't force him to analyze it correctly. Future statistical software should also guide the user to correct analyses.

Appendixes

Appendix A
Speakers

Keith E. Muller

Keith E. Muller is an associate professor of biostatistics at the University of North Carolina in Chapel Hill. He earned B.S. and M.A. degrees in psychology from Bradley University. He subsequently earned a Ph.D. in quantitative psychology and an M.S. in statistics at the University of North Carolina at Chapel Hill. His enthusiasm for and training in quantitative methods began during his undergraduate years in the late 1960s. Having grown tired of earning money doing complex ANOVAs on calculators, he took his first computing course, FORTRAN and beginning numerical analysis. This was followed by a two-semester graduate sequence in numerical methods for computers (focusing on linear and nonlinear systems). Other courses followed. As with many middle-aged statisticians, his statistical computing experiences encompass a range of equipment, software, and applications. He is a member of the American Psychological Association, the American Statistical Association, the Psi Chi Psychology Honorary, and the Psychometric Society, and he is an associate review editor for the *Journal of the American Statistical Association.*

Paul F. Velleman

Paul Velleman teaches statistics at Cornell University, where he chairs the Department of Economic and Social Statistics. He is a fellow of the American Statistical Association and past chair of the Association's Section on Statistical Computing. Professor Velleman is the developer of *Data Desk*®, one of the major statistics and graphics programs on the Macintosh. Professor Velleman has published research in modern data analysis methods, graphical data analysis methods, and statistical computing. His book (co-authored with David Hoaglin), *ABC's of EDA,* made computer implementations of exploratory data analysis methods widely available.

Andrew Kirsch

Andrew Kirsch is a statistical specialist with 3M in St. Paul, Minnesota. He holds an M.S. in statistics from the University of Wisconsin and an M.A. in applied mathematics from the University of Massachusetts. He has worked with 3M plants and laboratories for the past 9 years on a wide variety of products and processes, from electronics to contact lenses. He also has responsibilities in the areas of software, education, supervision, and methods research.

William DuMouchel

William DuMouchel has been chief statistical scientist at the Software Products Division of Bolt Beranek Newman Inc. since 1987. He earned a Ph.D. in statistics from Yale University in 1971. Before coming to BBN Software Products he held faculty appointments at the University of California, Berkeley, University of Michigan, Massachusetts Institute of Technology, and Harvard University. He has been elected a fellow of the American Statistical Association and of the Institute of Mathematical Statistics, and he is a member of the International Statistical Institute. His research interests include statistical computing, Bayesian statistics and meta-analysis, and environmental and insurance risk assessment.

Daryl Pregibon

Daryl Pregibon is head of the Statistics and Data Analysis Research Department at AT&T Bell Laboratories in Murray Hill, New Jersey. He received his Ph.D. in statistics from the University of Toronto in 1979. Dr. Pregibon spent one postdoctoral year at Princeton University and another at the University of Washington. He has been at AT&T Bell Laboratories for the past 9 years. Dr. Pregibon has been involved in the development of knowledge-based software for statistical data analysis since his coming to Bell Laboratories. He is a co-developer, with W.A. Gale, of the first prototype expert system for regression analysis, REX. He has contributed substantially to the theory, application, and computational aspects of generalized linear models. His current interests concern interactive statistical graphics and tree-based modeling. Dr. Pregibon is a fellow of the American Statistical Association and a member of the International Statistical Institute.

Appendix B
Position Statements: Additional Material Submitted by Symposium Participants

R. Clifton Bailey

Health Standards and Quality Bureau
Health Care Financing Administration

At the symposium, Andrew Kirsch noted that looking at the physical context of some statistical problems sometimes exposes the fact that certain additive functional forms can be contrary to known physical relationships. In his example, a product was instead the natural construct. Wilson (1952) discusses some examples such as symmetry (§11.5, pp. 308–312), limiting cases (p. 308), and dimensional analysis (§11.12, pp. 322–328). I would note that such considerations quickly lead one to consider functional relationships that are not linear in the parameters. The supporting statistical software has been of limited utility in dealing with these functional relationships. The picture becomes more complex when one introduces stochastic components to the models and complex sample designs for collecting data.

Nonlinear regression deals nicely with the problems when the stochastic element is limited to a measurement error. However, many formulations do not readily permit the use of implicitly defined functions. However, for those of us who must perform operational analyses without the luxury of developing special tools to deal with diverse problems, adequate tools have not been easily accessible.

Expert systems are designed to prevent the incorrect application of a procedure. In addition to asking whether we have correctly used a procedure according to accepted standards, we must also ask whether the methods are useful for our solving problems. One of the very strengths of statistical methods derived for designed experiments is the very weakness of these methods. The strength is to have methods that work without regard to the underlying functional relationships. We really can design methods that work in this way by controlling the design. These methods answer the preliminary questions such as whether there is a difference or not. Any subject-matter specialist quickly wants to go beyond these questions to understand the structure of a problem. Furthermore, there are many important data sources other than designed studies or experiments. Graphical techniques are so appealing partly because our analytical tools to reveal the more complex structures are so limited and underdeveloped. I hope that graphical techniques will drive analysts to consider more complex structural relationships in the analytical context.

As I said following Paul Velleman's talk, it would be nice if statisticians would think more globally. For example, in using the results of a survival analysis package for a specific analysis, it would be useful to know how the likelihood was formulated. Better yet, some agreement on the formulation would permit comparison across various models and packages. For example, a log-likelihood for a Weibel, constant hazard, or other model form could be compared if there were some agreement in how these are formulated. Some packages do not provide this comparability, even within a given procedure. Part of the reason is that some drop constants that are not relevant for the function to be optimized, the log-likelihood. Others use the density function in the formulation, while still others consider events to occur in an interval (which must be stated) and formulate the probability of the event occurring in that interval. I prefer this latter approach because of its generality. Events need to have the same interval and censored events are part of the same formulation.

Some problems fit nicely on personal computers, while others have large volumes of data that are centrally managed. Successful interaction with these data depends on the links between the data sources and the tools for analysis. Many specialized and intriguing enhancements appear on a variety of the changing platforms. I think a diversity of procedures and implementations is useful. One implementation may provide a capability not available in another, and results can be compared from various sources. However, it is difficult to keep track of these and to make them work on evolving platforms and in evolving environments. The speakers indicated nicely that too much effort is devoted to reproducing the similar algorithms in every setting and not enough to the enrichment of our tools.

Let me point out some of the problems faced by statisticians in operating agencies. We are not in the position of spending long periods on research issues. Furthermore, we are faced with managers of computing centers who want to support only standard software. This applies to mainframe and personal computer software. The danger, in the spirit expressed by Leo Breiman, is that we find nails to apply our hammers to. We do this by reformulating our problem to fit one for which we know the solution. The result is a correct solution to the wrong problem because our tools are oriented to solving the textbook problems.

We also encounter difficulties applying many routines to larger data sets--50,000 to 10 million cases. Furthermore, the diagnostics for dealing with large data sets need to be very different from those envisioned for smaller data sets for which we can easily scan residuals, plots and other simple displays, or tables for all of our data. For example, techniques for focusing on a few outlier observations need to be generalized to focus on outlier sets. I have found that having a convenient means for constraining or fixing parameters can be used in many interesting ways to solve complex problems such as those faced when trying to study the effect of many (several thousand) indicator variables in analyzing large data sets.

The needs are different for exploratory work where we are trying to understand relationships, and for production activities such as making 6000 tables or graphs for a publication.

The extensive use of contracting-out for solution of problems fragments the system and tends to work against having the technical expertise and resources appropriately concentrated to recognize and develop appropriate solutions. (See the various presentations by W. Edwards Deming on the system and profound knowledge, e.g., his February 16, 1991, talk at the annual meeting of the AAAS in Washington, D.C.)

A few years ago I found the available options were inadequate to the task of probability analysis. Can you image a procedure that produces the same graph no matter what the problem? Actually the graphs were correct but not useful since the scale on the graph changed with the problem. This does not facilitate graphical comparisons. The problem at hand required an elaborate work-around with the available package (see Bailey and Eynon, 1988).

Such creative use of imperfect tools is always required. I remember learning that a major government agency computer center (EPA) was going to abandon support for SPSS on the mainframe. This was a management decision without contact with the statistical community in the agency. Part of the problem arose because users could not readily obtain manuals. At another agency, SAS is the principal mainframe package available. While this is a powerful package suitable for many important statistical activities, other packages provide strengths not found in SAS.

References

Bailey, R. Clifton, and Barrett P. Eynon, 1988, Toxicity testing of drilling fluids: Assessing laboratory performance and variability, *Chemical and Biological Characterization of Sludges, Sediments, Dredge Spoils, and Drilling Muds,* ASTM STP 976, J.J. Lictenberg, J. A. Winter, C. I. Weber, and L. Fradkin, eds., American Society for Testing and Materials, Philadelphia, pp. 334–374.

Wilson, E. Bright, Jr., 1952, *An Introduction to Scientific Research,* Dover, New York.

Michael P. Cohen.

National Center for Education Statistics

I have a few comments on suggested standards and guidelines. These comments reflect my opinion only. My experience has been in sampling, design, and estimation for large, complex statistical surveys, including the U.S. Consumer Price Index and the Integrated Postsecondary Education Data System.

1. While the forum emphasized the proliferation of software, there remains a dearth of statistical procedures for analyses of data from complex surveys within general-purpose packages. I am aware of *special* -purpose packages such as SESUDAN, WESVAR, and PC-CARP.
2. Many statistical packages still do not treat *weighted* data very well. Even procedures that allow weights often treat them as a way of indicating multiple occurrences of the same observed value.

James R. Knaub, Jr.

Energy Information Administration Department of Energy

In a letter to *The American Statistician* [Knaub, 1987], I made some comments on the practical interpretation of hypothesis tests which are pertinent to statistical software. I feel that such software, by ignoring type II error analyses for simple alternatives, has helped make hypothesis testing disreputable, when it should be one of our viable tools. I hope that future generations of statistical software will take into account the need for type II error analysis.

Reference

Knaub, James R., Jr., 1987, Practical interpretation of hypothesis tests, *The American Statistician,* Vol. 41, No. 3, p. 246.

Jay Magidson

Statistical Innovations, Inc.

Overall, I believe that the panel members and invited speakers at the symposium represent a much too narrow range of interests and opinions to adequately represent the majority of users of statistical software. In general, I believe that the interests of business users, less sophisticated users, and less frequent users are underrepresented.

Among more sophisticated users, the emphasis on Exploratory Data Analysis (EDA) of quantitative data, and applications from the health sciences and quality control were *overrepresented.* In particular, I believe that the interests of survey researchers from the social sciences who analyze primarily categorical data are being largely neglected by the panel.

Most of the presentations point out the importance of EDA methods in appropriately dealing with violations of the assumptions of traditional techniques for quantitative analysis. However, there is a major revolution taking place in the analysis of categorical data that was totally neglected at the symposium. The revolution in categorical analysis from simple cross-tabulation software to log-linear models, latent class analysis, association models, and other chi-squared-based techniques is even more remarkable than EDA techniques since it is based on a unified theory. There is no need to adjust for the unrealistic assumptions of normality and linearity, since such assumptions are not made.

As interest in categorical methods continues to grow, in a few years these methods may well represent the majority of statistical applications. The choice of regression and one-way ANOVA as examples at the seminar illustrates the overemphasis on EDA and quantitative analysis. Regression and ANOVA are the same type of technique. One-way ANOVA was chosen simply to illustrate box plots. Cross-tabulation was totally ignored.

While my own academic background is econometrics, and while I have taught statistics at Tufts and at Boston University, I am currently president and chief statistician of a consulting firm where businesses are my primary clients. I am also the developer of a statistical package that is selling at the rate of more than 50 copies a month and has over 500 users--primarily business users. Thus, I am very familiar with business use of statistics.

I have also been the software editor of the *Journal of Marketing Research,* and head of the software committee at ABT Associates prior to forming Statistical Innovations Inc. in 1981. I have published on both quantitative and categorical multivariate techniques. I have also organized and conducted statistical workshops with Professor Leo A. Goodman, and through these courses have trained several hundred practicing researchers. Thus, I speak from my experience when I strongly recommend that these under-representations present on your panel be corrected.

Appendix C
Forum Participants

Susan Ahmed
National Center for Education Statistics

Yahia Ahmed
Internal Revenue Service

Nicholas Alberti
Bureau of the Census

Irwin Anolik
Bureau of the Census

Edmund L. Auchter
Arlington, Virginia

R. Clifton Bailey
Health Care Financing Administration

Bruce M. Barnhill
Biodecision Laboratories, Inc.

Austin Barron
American University

Marilyn Barron
Agency for Health Care Policy Research

Curtis Barton
Food and Drug Administration

Al Best
Virginia Commonwealth University

Paul Black
Decision Science Consortium

Larry Bobbitt
Bureau of the Census

Mary Ellen Bock
Purdue University

Tom Broene
Bell Atlantic, Inc.

Gregory Campbell
National Institutes of Health

Daniel B. Carr
George Mason University

Judy Chen
Food and Drug Administration

James Clair
Merck, Sharp and Dohme, Inc.

Michael P. Cohen
National Center for Education Statistics

James M. Davenport
Virginia Commonwealth University

Daniel Denman
University of Maryland

R. Michael Dummer
Glenside, Pennsylvania

William DuMouchel
BBN Software Products, Inc.

Herbert W. Eber
Psychological Resources, Inc.

William F. Eddy
Carnegie Mellon University

Dean Fennell
Department of Energy

Robert J. Frank
National Security Agency

Russell Freed
Michigan State University

Paul A. Games
Pennsylvania State University

Turkan K. Gardenier
Equal Employment Opportunity Commission

Edward J. Gilroy
U.S. Geological Survey

J. Jeffrey Goebel
Department of Agriculture

Jeff Goldner
New Unit

David Grier
George Washington University

M.J. Guilfoyle
University of Pennsylvania

Robert M. Hamer
Virginia Commonwealth University

Tom Herzog
Federal Housing Administration

Katherine S. Hogye
Food and Drug Administration

Sally E. Howe
National Institute of Standards and Technology

M. Raymond Jason
National Public Radio

Robert W. Jernigan
American University

Robert E. Johnson
Virginia Commonwealth University

Richard M. Jones
Philip Morris Research Center

Ian H. Keith
Potomac Systems Engineering, Inc.

David S. Keller
SmithKline Beecham Animal Health

Andrew Kirsch
3M

Charlie Kish
Medical College of Virginia

James R. Knaub, Jr.
Department of Energy

Robert Koyak
Department of Justice

Charles Lin
SAS Institute, Inc.

Murray H. Loew
George Washington University

Jay Magidson
Statistical Innovations, Inc.

Marvin S. Margolis
Millersville University

William E. McGarvey
National Institutes of Health

Ted Mihalisin
Mihalisin Associates, Inc

Reza Modarres
American University

Keith E. Muller
University of North Carolina

Sally M. Muller
University of North Carolina

Ruth E. O'Brien
Board on Mathematical Sciences

Michael W. O'Donnell
Food and Drug Administration

William Page
Institute of Medicine

Jon K. Peck
SPSS, Inc.

Daryl Pregibon
AT&T Bell Laboratories

Terry J. Reedy
Newark, Delaware

Alfredo Rojas
SmithKline Beecham

Carl T. Russell
Operational Test and Evaluation
Command, U.S. Army

Barbara F. Ryan
Minitab, Inc.

Rama Sastry
Department of Energy

John H. Schuenemeyer
University of Delaware

Babu V. Shah
Research Triangle Institute

Alison Shaw
Cornell University

Jay Silva
U.S. Army Research Institute

P. Simpson
Medical College of Virginia

Willard A. Stanback
Norfolk State University

Eric Stellwagen
Business Forecast Systems, Inc.

Mike Sullivan
STSC, Inc.

Robert Suriano
American Petroleum Institute

Bettie M. Teevan
Army Research Institute for Behavioral
Science

Robert F. Teitel
Abt Associates, Inc.

Al Thaler
National Science Foundation

John R. Tucker
Board on Mathematical Sciences

Paul A. Tukey
Bell Communications Research

Paul F. Velleman
Cornell University

T.S. Weng
Food and Drug Administration

William B. Whiston
Small Business Administration

Karen Wilcock
ENVIRON Corporation

Lisa Wilkinson
BBN Software Products, Inc.

Steve Wilson
Food and Drug Administration

Philip Wirtz
George Washington University

Franklin Womack
U.S. Army Concepts Analysis Agency

Forrest W. Young
University of North Carolina

William Yu
Health Care Financing Administration

Constantine Zervos
Food and Drug Administration