The usefulness of the L_S book would be greatly increased by including more complete descriptions of each function, organized either alphabetically as in the S book or by topic as is done in the description of the Common Lisp language (Steele, 1984).

I end with some minor criticisms of the L_S book.

First, there is disconcerting ambiguity throughout the book about which version of Lisp underlies L_S. It would have been better to commit to presenting L_S as it is implemented in XLISP, and then if/when it is released for Common Lisp, this could be accompanied by a document describing the differences. It is confusing that some ideas are presented generally when only one case applies to the current implementation of L_S. Examples of this are the discussion of lexical and dynamic scoping, and tail recursion, neither of which are relevant to L_S.

Second, the coverage of Lisp is varied. For example, though macros are mentioned obliquely in the text, their use is never discussed. This might be have been a concious choice since Abelson and Sussman (1985) deprecate the use of macros in their description of Lisp. But macros play an important part in Lisp programming and should not have been ignored.

Third, the index is not complete (for example, no entry on eval), but this applies to both books.

Fourth, there is too much Lisp code. I am someone who has a high tolerance for reading code and loves nothing better than wallowing around in piles of parentheses, but the density of Lisp code was too much even for me in some of the sections. Much better to my mind would have been to supply all of the Lisp code with the L_S system, and refer to relevant new ideas as they are introduced. This is worst in the last chapter, discussing dynamic graphics examples. But also in some of the earlier chapters, there is too much reliance on presenting code. Sometimes an entire function is presented many times as its evolution is discussed. Again, it would have been better just to present the relevant changes to the function in each new version.

7. CONCLUSION

By providing a broad range of statistical and mathematical primitives and an interpretive language, combined with good graphical facilities, together, these two systems define the state of the art in computing environments for statisticians. Each system is better suited to certain applications, and for data analysis and research, statisticians can only benefit by acquainting themselves with both.

ACKNOWLEDGMENTS

Thanks are due to John Chambers, Rick Becker, Allan Wilks and Luke Tierney who all provided me with valuable comments on this review.

Rejoinder

Luke Tierney

I would like to thank the reviewers for their comments and for their efforts in working through the book and the software, and I would like thank the editor for this opportunity to comment briefly on a few of the issues raised in the reviews.

FUNCTIONALITY AND EXTENSIBILITY

Several of the reviews point out that the basic Lisp-Stat system does not include a wide range of specialized analyses. This is quite deliberate. A major advantage of an extensible system is that it allows experts in using and developing a particular methodology to provide tools for implementing the methodology. If the Lisp-Stat system is found to be useful then, over time, this should lead to a wider set of better tools than can be provided by a single implementor or small group of implementors.

A comparison with the evolution of the S system may be helpful. The basic S system as described in Becker, Chambers and Wilks (1988) also does not directly support a side range of different analyses. But few sites now provide only the basic S system. Most augment it with the facilities of S-Plus, a variety of locally written code, selections of code

Luke Tierney is Professor, School of Statistics, University of Minnesota, Minneapolis, Minnesota 55455.

contributed to `statlib`, or a combination of all three. Statisticians who find the Lisp-Stat system useful will develop, and in many cases already have developed, their own functions and objects. Many of these functions and objects would be useful to others and will, hopefully, be made available through the `statlib` archive.

GRAPHICAL USER INTERFACE ELEMENTS

In designing the graphical interface to Lisp-Stat, a primary objective was to provide a set of tools that could be used in the same way on a variety of native user interfaces. As a result, it was necessary to reduce the tools provided to a common denominator that could be implemented portably. Several useful but not essential items seemed difficult to implement in a portable way and were therefore excluded. Lubinsky points out two such items: support for multiple fonts and pixmap handling tools. Others include the ability to use dialog items and graphs in the same window, support for scrollable multi-line text fields in dialogs and support for multiple plots in a single window.

In the last few years, higher level toolkits for programming graphical user interfaces have evolved, or perhaps just my understanding of these toolkits has improved, to a point where it seems possible to add many of these features to the basic Lisp-Stat system. I hope to experiment with some of these additions in the near future.

DOCUMENTATION

The documentation provided for the functions available in Lisp-Stat is indeed rather limited. For functions that are also available in Common Lisp, the book *Common Lisp: The Reference* (Franz, Inc., 1988) provides perhaps the most complete documentation. This book gives a full description and a set of examples for each Common Lisp function and macro. It would be useful to develop similar descriptions for the functions and objects that are specific to Lisp-Stat.

Developing more effective forms of on-line documentation is more challenging. Both MS Windows and Macintosh System 7 attempt to provide a standard form of on-line documentation. Both systems allow programs to provide context-sensitive help, and both systems provide a uniform documentation format across applications. The hypertextual approach of MS Windows appears to provide more flexibility for supporting various browsing strategies. Neither system is designed to support an extensible system. Nevertheless, it should be possible to take advantage of these systems for documenting at least the standard portions of Lisp-Stat. A help file for the new MS Windows version of XLISP-STAT may be available in the near future.

Support for nonlinear on-line documentation structures on UNIX systems is at present less uniform. Perhaps the most widely available system is the GNU emacs info system.

GRAPHICAL INTERFACES AND EXTENSIBILITY

Weihs raises the issue of a possible conflict between providing a convenient graphical user interface for a program and allowing the program to be used in batch mode, or as a component of a larger program. This is a challenge provided by the combination of graphical interface tools with an extensible environment. A guideline I have found useful is to avoid making a program dependent on is interface by using interface tools merely as convenient devices for sending messages to objects representing the program. In Weihs' example, instead of having the dialog itself take certain actions when the **OK** button is clicked, I would simply have the button action consist of a message like

```
(send omega :set-technique 'pca-cov)
```

for the choice shown in the dialog. If the only purpose of the dialog is to send a message like this, then it will be possible in include such an expression in a batch file or the code of another program without the need to use the dialog itself.

An advantage of this approach is that one can have an option in which the messages sent by a dialog are shown to the user as they are sent. This could help the user make the transition form interactive use of the program to using it as a building block for more complicated analyses.

SOME NOTES ON LISP-STAT IMPLEMENTATIONS

I would like to conclude these comments with some notes on current Lisp-Stat implementations and the future development of the systems.

Versions of the XLISP-STAT implementation are now available for UNIX systems using X11, the Macintosh, DOS systems using MS Windows 3.0, and the Commodore Amiga. The port to the Amiga was done by Jim Lindsey. At the time of writing, the MS Windows version is still experimental, but it should stabilize in the next few months. I do not currently plan any further ports, for example to the Display Postscript system used by NeXT workstations, but I would be happy to provide assistance to anyone interested in porting XLISP-STAT to other systems.

The XLISP-based implementations should continue to evolve. There is a XLISP community that has been working for some time to reduce differences between XLISP and Common Lisp. As these modifications become available, they will be incorporated in XLISP-STAT. As an example, many of the sequence functions that currently only apply to lists will be extended to vectors as well. Another feature that may be included is the ability to save an XLISP-STAT workspace.

A Common Lisp version of Lisp-Stat based on the Austin Kyoto Common Lisp (AKCL) system, which is available free of charge for many UNIX systems, is currently in preparation. The nongraphics portion is essentially complete, and a first implementation of the graphics portion may be completed in the next six months. Since this implementation will reuse much of the C code from the XLISP implementation, it will depend heavily on the AKCL foreign function interface. It should not be too difficult to adapt to foreign function interfaces provided by other Common Lisp systems.

Beyond these basic implementations, the future evolution of the Lisp-Stat system is in the hands of its users. Many interesting ideas have already been implemented using Lisp-Stat. But, at the time of writing, few have been made available to others through the `statlib` archive. I would therefore like to close by encouraging readers who have developed code in Lisp-Stat, or any other language, that might be of use to others to consider making their work available through the `statlib` archive. The first step is to send an electronic mail message to the internet address *statlib@lib.stat.cmu.edu* containing the single line

```
send index
```

The response will provide information on software available as well as instructions on submitting software to the archive.

REFERENCES

ABELSON, H. and SUSSMAN, G. (1985). *Structure and Interpretation of Computer Programs*. MIT Press.

BAXTER, R. I., CAMERON, M. A., FISHER, N. I. and HOFFMANN, B. (1991). Using multiple views for data analysis. In *Computing Science and Statistics: Proceedings of the 23rd Symposium on the Interface*. To appear.

BECKER, R. A. and CLEVELAND, W. S. (1987). Brushing scatterplots. *Technometrics* **29** 127-142.

BECKER, R. A., CHAMBERS, J. A. and WILKS, A. R. (1988). *The New S Language*. Wadsworth, Pacific Grove, Calif.

BUJA, A., ASIMOV, D., HURLEY, C. and MCDONALD, J. A. (1988). Elements of a viewing pipeline for data analysis. In *Dynamic Graphics for Statistics* (W. S. Cleveland and M. E. McGill, eds.) 227-308. Wadsworth, Pacific Grove, Calif.

CLEVELAND, W. S. and MCGILL, M. E., eds. (1988). *Dynamic Graphics for Statistics*. Wadsworth, Pacific Grove, Calif.

DONOHO, A. W., DONOHO, D. L. and GASKO, M. (1985). MACSPIN Graphical Data Analysis Software. D^2 Software, Austin, Tex.

Franz, Inc. (1988). *Common Lisp: The Reference*. Addison-Wesley, Reading, Mass.

HASLETT, J., WILLS, G. and UNWIN, A. R. (1990). SPIDER-An interactive statistical tool for the analysis of spatial data. *International Journal of Geographical Information Systems* **4** 285-296.

ISP (1988). *Apollo ISP-SGS/DGS User's Guide*. Artemis Systems, Carlisle, Pa.

KIRCHEN, A. and WEIHS, C. (1984). The IAS-System Bonn: An interactive software system for econometric modelling. Technical Report 152, Institut für Gesellschafts und Wirtschaftswissenschaften.

MCDONALD, J. A. and PEDERSEN, J. (1988). Computing environments for data analysis. III. Programming environments. *SIAM J. Sci. Statist. Comput.* **9** 380-400.

OLDFORD, R. W. and PETERS, S. C. (1984). DINDE: Towards more sophisticated software environments for statistics. *SIAM J. Sci. Statist. Comput.* **9** 191-211.

S-Plus (1990). *S-Plus User's Manual*. Statistical Sciences, Seattle, Wash.

SAS-INSIGHT (1991). *SAS-INSIGHT User's Guide, Version 6*. SAS Institute, Cary, N.C.

STEELE, G. L. (1984). *Common Lisp: The Language*. Digital Press, Burlington, Mass.

STUETZLE, W. (1987). Plot windows. *J. Amer. Statist. Assoc.* **82** 466-475.

STAETZLE, W. (1988). Plot windows. In *Dynamic Graphics for Statistics* (W. S. Cleveland and M. E. McGill, eds.) 225-243. Wadsworth, Pacific Grove, Calif.

VELLEMAN, P. F. and VELLEMAN, A. Y. (1988). *DataDesk Handbook*. Odesta Corporation, Northbrook, Ill.

WEIHS, C. and SCHMIDLI, H. (1990a). OMEGA—online multivariate exploratory graphical analysis: Routine search for structure (with discussion). *Statist. Sci.* **5** 175-226.

WEIHS, C. and SCHMIDLI, H. (1990b). Multivariate exploratory data analysis in chemical industry. Technical report 9011, Mathematical Applications, CIBA-GEIGY, Basel, Switzerland. (Also appears in *Proceedings of the European Conference on Analytical Chemistry 1990* in *Microchimica Acta*.)

WINSTON, P. H. and HORN, B. K. P. (1989). *LISP: 3rd Edition*. Addison-Wesley, Reading, Mass.

WIRTH, N. (1976). *Algorithms + Data Structures = Programs*. Prentice-Hall, Englewood Cliffs, N.J.

YOUNG, F. W. and RHEINGANS, P. (1991). Visualizing structure in high-dimensional data. *IBM J. Res. Develop.* **35**. To appear.

YOUNG, F. W., FALDOWSKI, R. A. and HARRIS, D. F. (1990). The spreadplot: A graphical spreadsheet of algebraically linked dynamic plots. In *Proceedings of the Section on Statistical Graphics*. Amer. Statist. Assoc., Alexandria, Va. To appear.