

# WHY WRITE STATISTICAL SOFTWARE? THE CASE OF ROBUST STATISTICAL METHODS

ARNOLD J. STROMBERG

ABSTRACT. Robust statistical methods are designed to work well when classical assumptions, typically normality and/or the lack of outliers, are violated. Almost everyone agrees on the value of robust statistical procedures. Nonetheless, after more than 40 years and thousands of papers, few robust methods were available in standard statistical software packages until very recently.

This paper argues that one of the primary reasons for the lack of robust statistical methods in standard statistical software packages is the fact that few developers of statistical methods are willing to write user-friendly and readable software for the methods they develop, regardless of the usefulness of the method. Recent changes in academic statistics make it highly desirable for all developers of statistical methods to provide usable code for their statistical methods.

## 1. INTRODUCTION

Academic research in statistics often involves the introduction of new statistical methods. The method is often justified theoretically, in simulations, and by examples. Obviously, the simulations and examples require the method be programmed. Most academics write code only for their own

---

*Date:* April 27, 2004.

The author thanks Jan de Leeuw and Nicholas Cox for advice that significantly improved the paper.

use. They have numerous excuses for not writing more user-friendly code.

An incomplete list includes:

- Code isn't publishable.
- Software isn't fundable and doesn't help your career.
- Writing code is no fun.
- You don't want to embarrass yourself with your poor programming skills.
- No one will use it.
- It takes extra time.

The remainder of this paper addresses these excuses, discusses the case of robust regression and concludes with further comments.

## 2. CODE ISN'T PUBLISHABLE

That is no longer true. In particular, as this article shows, statistical software can be published in the electronic Journal of Statistical Software (JSS) which can be found at [www.jstatsoft.org](http://www.jstatsoft.org). Abstracts are published in the Journal of Computational and Graphical Statistics (JCGS). Quoting from the JSS website, JSS will publish:

- (1) Manuals, user's guides, and other forms of description of statistical software, together with the actual software in human-readable form.
- (2) Code snippets – small code projects, any language.
- (3) Special issues on topics in statistical computing.
- (4) A yearly special issue documenting progress of major statistical software projects.
- (5) Reviews and comparisons of statistical software.
- (6) Reviews of books using statistical software. (Recently added.)

The typical JSS paper will have a section explaining the statistical technique, a section explaining the code, a section with the actual code, and a section with examples. All sections will be made browsable as well as downloadable. The papers and code should be accessible to a broad community of practitioners, teachers, and researchers in the field of statistics and beyond.

### 3. SOFTWARE ISN'T FUNDABLE AND DOESN'T HELP MY CAREER

In reality, it is theoretical research in statistics that rarely gets funded. For example, in the United States, the National Science Foundation (NSF) wants "useful and innovative" to quote an NSF Statistics and Probability Program Director. "Useful" can be taken in several ways, but clearly user-friendly code helps argue for usefulness. If no one can use the method, then it could be argued that the method isn't useful. A great way to show that statistical methods are useful is to have collaborators who can apply your methods. The advantage of collaborators includes the fact they have "real" problems, thus at least doubling your funding options. They also at least double your publication options. These collaborators will also support you and your department. "Innovative" means that NSF, and presumably other funding agencies, are not overly interested in funding incremental steps forward in any particular area. They would much rather see innovative approaches to new problems. Of course, innovative does not mean crazy. One must still convince the funding agency that the project is doable. User-friendly code is one excellent way to do that.

Historically, many junior academics thought that the best way to get funding was to collaborate with a funded senior colleague. Although it occasionally

works, there are serious disadvantages of this strategy. Many funding agencies give priority to junior, not senior, faculty. Having a senior colleague as a collaborator may actually decrease the likelihood of funding. More importantly, when junior academics are evaluated, there is a tendency to think that the junior faculty member got the grant because he or she collaborated with a senior colleague.

Other funding agencies want medical applications or military applications. Often junior faculty are discouraged from being Co-Principal Investigators (Co-PIs) on such grants because "it's just consulting" and the academic should be "writing papers instead." If this consulting takes away from research, this point is well taken. On the other hand, this funding can be used to reduce teaching loads, thus providing more time for research. Consulting papers also provide breadth to one's resume. Being a Co-PI also often leads to interesting real research problems, for which the statistician can be the PI.

#### 4. WRITING CODE IS NO FUN

Writing code may not be fun, but seeing researchers use your methods is lots of fun! If your method is useful, researchers will use it if they know about it and have the tools. Statistical software applications must be published in a variety of journals in order to have an impact.

#### 5. I DON'T WANT TO EMBARRASS MYSELF WITH MY POOR PROGRAMMING SKILLS

What counts is code that works, not how pretty it is or even how fast it is. Obviously, making code readable prior to publication is important, but it is far more important to get useful code out so it can be used. In fact, it seems

far more valuable to spend time on robustifying the code so it crashes less frequently, then making the code pretty.

#### 6. NO ONE WILL USE MY SOFTWARE

No one has a chance to use your method if you don't provide code, which is worse than no one using the code at all. If you write code in a readable and user-friendly way, and the statistical method is itself useful, my experience is that several groups will be interested. Researchers interested in comparing their new method with yours will use it (and reference it). Anyone reading about your method in a methods journal will be thrilled to have a chance to try the method on their data. They will reference your papers and code for others to use. User-friendly code that stands the test of time may then be incorporated into standard statistical packages.

#### 7. WRITING CODE TAKES EXTRA TIME

It is true that writing user-friendly readable code take more time than code that can only be used by its author. Since code must be functional in order for authors to run simulations and examples, it is frequently only a bit more effort to make the code user-friendly. The availability of R to everyone makes it a good platform, other factors being equal.

#### 8. EXAMPLE: THE CASE OF ROBUST REGRESSION

Multiple linear regression is one of the most used of all statistical methods. It's easy to convince everyone that outliers can have devastating effects on a linear regression fit. The need for robust linear regression estimators that can automatically downweight outliers is clear. In fact Huber proposed the

M-estimator nearly 40 years ago. M-estimators will be available for the first time in base SAS in Version 9.0.

Why did it take so long for robust regression to be incorporated into base SAS? There are many reasons, including conservatism, the desire to provide what the market wants and the need to supply technical support. Another important part of the reason is that at the time M estimators were proposed, statisticians' main emphasis was on theory and not applications, so there was little incentive to program methods. Yet another reason was that computing anything in 1964 was hard. Statisticians worked on improvements to M estimators and came up with many. This started extensive discussion on which robust regression estimate was the best. Applied statisticians understood the need for robust estimators, but it was now unclear which robust regression estimator should be incorporated into software packages. The 1972 Princeton Robustness Study was an effort to compare robust location estimators via computer simulations. A general, if possibly unfounded, conclusion from the study was taken to be that a high breakdown point estimator was desirable for a robust estimator. Unfortunately, this conclusion further delayed the implementation of robust estimators because high breakdown estimators were computationally intensive and no high breakdown regression estimate had been found. Such estimators are still hard to compute today so that conclusion, although correct, likely greatly increased the time needed to get robust regression estimators into base SAS. In 1984, Peter Rousseeuw introduced the Least Median of Squares (LMS) estimator. He and others provided code and thus made robust regression estimation possible in practice. Improvements to LMS were quickly introduced, each one with slightly improved theoretical properties, but few practical comparisons were done because major journals were unwilling to

publish such comparisons. Again, this caused confusion again about what robust regression estimator should be used in practice. It took until Version 9.0 to get robust regression into base SAS. Had applied statisticians been able to compute M estimates from the beginning, it seems very likely that they would have been incorporated into standard statistical packages. That not being the case, it took almost 40 years before most researchers could compute this useful estimator easily.

#### 9. EXAMPLE: THE EVOLUTION OF A GOOD STATISTICAL IDEA

Suppose you have an idea for a new statistical method. Ideally the method addresses a real problem proposed by a collaborator who is not a statistician. How should you proceed? First, you have to find out if the idea is right. There are two choices, start with the theory or the simulations. If you start with the theory and the idea is wrong, you are going to waste a lot of time trying to prove something that isn't true. It's better to start with simulations in either SAS, Matlab, S-PLUS, R, or other vectorizable software. If the idea works, work on the grant application and the proof. The simulations convince the funding agency that the idea is good. The theory will follow with appropriate assumptions. Once the theory and simulations are done, use your collaborators' data and submit it to a statistical methods journal. While that paper is under review, work on user-friendly software for the method. Submit the code and documentation to the Journal of Statistical Software. In the meantime, your collaborator is writing a methods paper for their field showing off this new method. One good idea, three publications and a grant. How could life be any better?

## 10. CONCLUSION

Reviewing the excuses for not writing usable code found in the Introduction to this paper, it is clear that most of them are baseless:

- Code is publishable.
- Software is fundable and helps your career.
- Writing code is fun, especially when others use it.
- Your programming skills may be better than you think.
- Everyone will use useful code.
- It takes little extra time.

DEPARTMENT OF STATISTICS, UNIVERSITY OF KENTUCKY

*E-mail address:* `astro@ms.uky.edu`