Comput Econ (2010) 35:371–394
DOI 10.1007/s10614-010-9204-4

# Should Economists Use Open Source Software for Doing Research?

**A. Talha Yalta · A. Yasemin Yalta**

Accepted: 17 February 2010 / Published online: 5 March 2010


**Abstract** We survey the literature on the accuracy of econometric software. We also assess the advantages of open source software from the point of view of reliability and discuss its potential in applied economics, which has now become fully dependent on computers. As a case study, we apply various accuracy tests on GNU Regression, Econometrics and Time-series Library (gretl) and demonstrate that the open source nature of the program made it possible to see the cause, facilitated a rapid fix, and enabled verifying the correction of a number of flaws that we uncovered. We also run the same tests on four widely-used proprietary econometric packages and observe the known accuracy errors that remained uncorrected for more than 5 years.

**Keywords** Open source · Econometric software · Gretl · Accuracy · Software reliability

## 1 Introduction

Yes, and the reasons are quite a few,[1] but the ones that we focus in this study are reliability and accountability. These are important issues because conducting research

[1] See Yalta and Lucchetti (2008) for a discussion of some of the practical advantages of using open source software in general and the Linux operating system in particular in the field of economics.

A. T. Yalta (✉)
Department of Economics, TOBB University of Economics and Technology, Sogutozu Caddesi No. 43, Sogutozu, 06560 Ankara, Turkey
e-mail: yalta@etu.edu.tr; talhayalta@gmail.com

A. Y. Yalta
Department of Economics, Hacettepe University, 06800 Ankara, Turkey
e-mail: yyalta@hacettepe.edu.tr

in economics today is fully dependent on computers and econometric software.[2] This is unlike the situation that existed when economists first used the programmable electronic computer about only 60 years ago. Renfro (2004b) discusses how it essentially took an entire day in the late 1940s to compute the regression parameters for a single equation. With the help of faster and more and more powerful computers, this process became a matter of less than an hour in the early 1960s and less than a minute in the 1980s. Today it takes less than a second (and often just a mouse click) on a standard PC to make calculations considered almost impossible a few decades ago. However, we now also have a considerable number of studies showing that the increasingly complex tools that we use for doing research can and do have important flaws, imperfections, or inconsistencies; an issue not addressed in the mainstream journals except McCullough and Vinod (1999, 2003b).

Open source as a software development model is receiving substantial attention thanks to the tremendously successful open source projects in the recent years. 439 of the world's fastest 500 supercomputers are now running the Linux operating system, which is estimated by the Linux Foundation to worth 1.4 billion US dollars for the kernel alone. The lesser known OpenBSD operating system, which is focused on code correctness and rigorous auditing, has seen only two security flaws in the last 10 years. The Apache HTTP server is powering 49% of all the web servers comprising the Internet. The Firefox browser has set the world record for the most downloaded software in a single day and reached a market share of 22% despite its biggest rival Explorer comes preinstalled on almost all new computers. As of March 2009, SourceForge.net, the biggest hosting services provider for open source developers, lists over 230,000 open source projects and more than 2 million registered users. Among the successful open source projects is the statistical package R, which has become widely popular among data analysts as well as economists inside corporations and academia.[3]

The open source movement is growing rapidly[4] and resulting in research on a range of topics such as the motivations to contribute to open source projects (Hertel et al. 2003; Roberts et al. 2006) as well as the outcomes and the competitive dynamics of contributing (West 2003; Bonaccorsi et al. 2006). There also exist studies such as

[2] By 'econometric software,' we refer to a range of programs commonly used by economists for doing computations. For example, spreadsheets such as MS Excel are often utilized for econometric analysis, if preliminary. Also commonly used are packages such as Mathematica, MatLab, R, and S-Plus that promote themselves first and foremost as programming languages. There are comprehensive GUI driven programs as well as various add-ins, modules, and libraries; both commercial or noncommercial. See Renfro (2004b) for a discussion of the difficulty involved in defining and classifying econometric software.

[3] On the estimated value of Linux and its deployment on supercomputers, see http://www.linuxfoundation.org/publications/estimatnglinux.php and http://www.top500.org respectively. On the security of FreeBSD, see http://www.openbsd.org/. On the popularity of Apache, see http://news.netcraft.com/archives/2009/02/18/february_2009_web_server_survey.html. On the market share of Firefox and its world download record, see http://marketshare.hitslink.com and http://www.spreadfirefox.com/en-US/worldrecord/ respectively. On the size of SourceForge, see http://apps.sourceforge.net/trac/sourceforge/wiki/What%20is%20SourceForge.net. On the success of R, see http://www.nytimes.com/2009/01/07/technology/business-computing/07program.html (all accessed on March 19, 2009).

[4] Using a panel dataset from SourceForge, Lerner et al. (2006) find that the median growth in the number of contributors and contributions to a sample of 98 open source projects were respectively 36 and 50% between 2001 and 2004.

Maurer and Scotchmer (2006) and Lerner and Tirole (2001, 2005a), which review and explain this phenomenon from the perspective of the standard economic theory. Our objective, on the other hand, is to assess the benefits of using this resource in applied economic research. To accomplish this, in the next section we survey the literature on the flaws and accuracy errors of econometric software. After discussing the difficulties in addressing with proprietary programs the issue of reliability in a scientific setting, we discuss in Sect. 3 how open source software can have an inherent advantage in this department. As a case study, we apply in Sect. 4 the so called Wilkinson tests to the open source gretl econometric package and show how it was possible to see the causes of the various problems and verify their subsequent corrections. In Sect. 5, we discuss the dissimilar case of closed source proprietary programs and draw some conclusions in Sect. 6.

## 2 The Issue of Accuracy

Textbooks on econometrics usually give the impression that all one has to do is to use a computer to apply different tests and estimation techniques. On the other hand, computer calculations have a finite precision and the results obtained using computers are subject to various accuracy flaws. For example, it comes as a surprise to some economists that in a simple regression model finding the parameter vector $\mathbf{b}$ by solving the linear equation

$$\mathbf{X}'\mathbf{X}\mathbf{b} = \mathbf{X}'\mathbf{y}, \tag{1}$$

where $\mathbf{X}$ is a data matrix and $\mathbf{y}$ is a column vector of observations on the dependent variable, results in the following approximation by the computer:

$$(\mathbf{X}'\mathbf{X} + \mathbf{E})\mathbf{b}^* = \mathbf{X}'\mathbf{y} + \mathbf{e}. \tag{2}$$

Here, $\mathbf{E}$ and $\mathbf{e}$ respectively represent an initial error matrix and an initial error vector, which cause the inexact solution $\mathbf{b}^*$. These errors are inevitable and will be encountered in all computers with finite precision. The existence of computer based numerical errors, together with human errors such as choosing an insufficient algorithm, improperly implementing a sufficient algorithm, or both means that the accuracy of econometric software should never be taken for granted. Scientific software is a work in progress and therefore should be subject to rigorous testing and continuous monitoring.[5]

Today it is unimaginable to do research without assistance from econometric software. On the other hand, the issue of the reliability of such software, by and large, has been overlooked by the economics profession despite the numerous studies pointing out the existing problems. Table 1 below lists the research in the last 15 years that has invariably found errors, imperfections, or inconsistencies in the commonly used

[5] For further information regarding the various accuracy issues in computer calculations, see Altman et al. (2004).

**Table 1** Studies exposing software errors and inconsistencies in the last 15 years

| Study | Packages | Tested/applied |
|---|---|---|
| Lovell and Selover (1994) | 4 | AR(1) correction, OLS, NLS |
| Newbold et al. (1994) | 17 | ARMA |
| Sawitzki (1994a) | 9 | Wilkinson's tests |
| Knüsel (1995) | 1 | Statistical distributions |
| Silk (1996) | 3 | Systems estimation |
| Bankhofer and Hilbert (1997) | 9 | Wilkinson's tests[a] |
| Knüsel (1998) | 1 | Statistical distributions |
| McCullough and Renfro (1998) | 7 | GARCH |
| McCullough (1999a) | 3 | McCullough's method |
| McCullough (1999b) | 4 | McCullough's method |
| McCullough and Vinod (1999) | 4 | FIML[a] |
| McCullough and Wilson (1999) | 1 | McCullough's method |
| Vinod (2000) | 1 | McCullough's method |
| Altman and McDonald (2001) | 4 | StRD |
| Brooks et al. (2001) | 9 | GARCH, EGARCH |
| Knüsel (2002) | 1 | Statistical distributions |
| McCullough and Wilson (2002) | 1 | McCullough's method |
| McKenzie and Takaoka (2002) | 1 | StRD[a] |
| Altman and McDonald (2003) | 3 | Probit and Logit |
| Baiocchi and Distaso (2003) | 1 | Statistical distributions, OLS |
| Brooks et al. (2003) | 4 | MGARCH |
| Kitchen et al. (2003) | 2 | StRD |
| McCullough and Vinod (2003a) | 4 | GARCH |
| Bruno and Bonis (2004) | 3 | Panel data estimation |
| McCullough (2004b) | 5 | Wilkinson's tests |
| McCullough (2004a) | 1 | McCullough's method |
| Stokes (2004) | 6 | Probit |
| Choi and Kiefer (2005) | 1 | Wilkinson's, OLS, NLS, GARCH |
| Knüsel (2005) | 1 | Statistical distributions |
| McCullough and Wilson (2005) | 1 | McCullough's method |
| Zeileis and Kleiber (2005) | 1 | Multiple structural change |
| Heiser (2006) | 1 | McCullough's method[a] |
| Keeling and Pavur (2007) | 9 | StRD |
| McKenzie and Takaoka (2007) | 1 | StRD, GARCH[a] |
| Yalta (2007) | 1 | McCullough's method |
| Yalta and Yalta (2007) | 1 | McCullough's method |
| McCullough (2008) | 1 | Random number generation |
| McCullough and Heiser (2008) | 1 | McCullough's method[a] |
| Yalta (2008) | 3 | Statistical distributions |

**Table 1** continued

| Study | Packages | Tested/applied |
|---|---|---|
| Yalta and Jenal (2009) | 1 | ARMA |
| McCullough (2009) | 3 | ARMA |

[a] Reported on some additional econometric or statistical functionality also

econometric programs.[6] A great majority of the errors and flaws reported by these studies are simply unacceptable in a scientific program. Sometimes the issue involves computer output that is downright wrong, while it is sometimes related to small or large inconsistencies between packages. Two or more programs returning different answers, if slightly, should be enough reason to make the user pause and think which, if any, of the results is the correct one but we almost never see researchers using a second package to verify their computations.[7]

Because it is impossible to test all of the procedures offered by different programs, the studies on software reliability generally employ an introductory or an intermediate level testing approach such as Wilkinson (1985) 'Statistics Quiz,' the Statistical Reference Datasets (StRD) by the U.S. National Institute of Standards and Technology, or McCullough (1998) set of tests. Among these methods; Wilkinson's tests involve a single dataset and four groups of tests focusing on basic calculations related to graphics, data management, correlation, tabulation, and linear regression. The StRD offers a number of datasets along with certified computational results for testing the accuracy of such functionality as univariate summary statistics, analysis of variance, linear regression, and nonlinear regression. McCullough's method formalizes the StRD and expands on it by adding further tests for assessing the reliability of random number generation as well as statistical distributions.

Currently, the accuracy of many econometric methods such as Kalman filtering, multinomial logit, VAR, VEC, or GMM cannot be assessed simply because of the lack of benchmarks. This is why it is noticeable in Table 1 that what has been tested constitutes a relatively small percentage of the functionality typically available in a modern econometric package. It is obvious that the tools that we are using are imperfect and do contain errors known or yet to be discovered. The important question then is whether these flaws can be identified and fixed objectively so that inaccurate research results based on software flaws can be corrected in a fashion consistent with the scientific method. The answer to this question is generally "No." Today, the programs used by many researchers are proprietary commercial packages, exact contents of which are unknown to all except the software vendors. This in turn presents

[6] We decide not to identify by name the individual programs because no software is perfect and flaws will be found as well in other programs not covered in these studies.

[7] Suppose a researcher concerned with scientific integrity takes the time to double check his results with an alternative program only to obtain a different answer. Resolving this discrepancy brings further costs and the researcher knows it is unlikely that a journal will publish two sets of results. Consequently, the researcher has no incentive to verify his results using a second package (McCullough 2000). That is, unless the researcher is interested in implementing the model on different software until he gets the answer he wants (Altman and McDonald 2003).

important challenges in sustaining the reliability of computational results in the field of economics.

The studies indicate that different software vendors show different levels of concern to computational accuracy and not all developers genuinely care about numerical reliability. Sawitzki (1994a) applies the Wilkinson tests on nine different data analysis systems, finds flaws in each of them, and reports that the vendor reaction to these errors varied ranging between "cooperative concern and rude comments." Yalta (2007) finds that the various numerical issues in the GAUSS software package, reported first by Knüsel (1995, 1996) and then by Vinod (2000), were not properly fixed after more than 7 years and several major revisions. Microsoft Excel is widely used for statistical analysis of data and McCullough and Heiser (2008) show that the errors found in Excel97 were still either not fixed or wrongly fixed in Excel2007 despite several earlier studies pointing out those problems.

There exist studies such as McKenzie and Takaoka (2007) and Keeling and Pavur (2007) which report improved performance in comparison to earlier testing but this is not something that the user can take for granted. Accuracy is not a static rating but an ongoing process that is costly to maintain and difficult to measure. This can lower its priority in a commercial environment. McCullough (2001) argues that the reason Microsoft does not fix errors in Excel can be due to a profit maximizing strategy of allocating the resources on features such as a glitzy GUI that have more potential to create new revenue. Indeed, Yalta (2008) observes that the 33 page Excel 2007 product guide emphasized an overhauled user interface, better data integration, faster calculation performance, and "dramatic visual effects" without giving a word's consideration to numerical accuracy. Renfro (2004b) discusses the development costs and various expansion problems that provided in the past some incentive for econometric software vendors to "paper over the cracks and hope that no one steps there."

Proprietary software has the disadvantage of being a black box so that it is extremely difficult to see how it exactly works and whether the various algorithms are implemented as claimed by the vendor. There can be undocumented changes so that two different versions of the same software system produce different answers. Altman and McDonald (2003) find in the subsequent versions of a well-known econometric package discrepancies that change the overall conclusions of two studies published in a prestigious journal. Sometimes it is even impossible to run an accuracy test because software companies can choose not to disclose details regarding the algorithms used in their products. McCullough (2009) reports his astonishment finding out that many packages offering the unconditional least squares method do not document the stopping rule. What is more, among the four developers that he contacts, all except one refuse to reveal this information. Documented in the literature are cases where the software vendor knowingly misleads the users. Wilkinson (1994) mentions of a software company claiming to use in their product an algorithm which in fact only appeared in another package that they imitated. McCullough (2008) discusses how the Microsoft Corporation twice falsely claimed fixing the bad random number generator in Excel by implementing the Wichmann and Hill (1982) RNG. He also shows that Excel 2007 had "an unknown and undocumented RNG of unknown period that is not known to pass any standard tests of randomness." Yalta and Jenal (2009) find in the context of an ARIMA model that the least-squares estimation option in the XLSTAT program was

grossly erroneous. When they report this, the vendor replies that they already knew "the least-squares method (was) not reliable in some cases." They were simply letting the users use a bad function thinking it was correct.

It is a fact that by using proprietary programs, we are giving up the ownership of the tools that we use for conducting scientific research. We simply become licensees who, by agreeing to the end-user license agreement (EULA), forego in the beginning any rights to knowing or studying the internals of our tools and sharing them with our peers for the verification of our research. Some of the commercial programs even require a continuous payment of license fees, which means that a researcher will not be able to revisit his own work if a license extension is not obtained in the future.[8] All these point to a situation where the research process in economics, particularly the dissemination of econometric results, may not be as scientific as one might think. It is also one of the reasons why today research replication can be almost impossible in the field of economics (but see the next section on this).

## 3 The Reliability of Open Source Software

There are a growing number of researchers who think that scientific software is too serious a matter to be left to the software vendors. These individuals choose to use free/libre[9] and open source software (FLOSS) in conducting scientific research.

FLOSS is based on an approach that fundamentally differs from the traditional commercial software development model, where paid workers create intellectual property for a private company. It is based on a continuously evolving group of users and developers interacting with each other to create software in the classical spirit of scientific collaboration. The resulting program is distributed under a software license that seeks to keep the individual contributions free and available.[10] The community around such programs usually involves a relatively small number of paid or unpaid 'core developers' who have the technical expertise in addition to various programming skills. There is usually a larger group of 'contributors' helping with the documentation, testing functionality, fixing small errors, or maintaining ports of the program on different platforms. The third and the largest group is composed of the 'users' who can also participate in the development process by submitting bug reports, requesting new features, and helping over the Internet other users learn or make better use of the program (Lakhani and von Hippel 2003). Over time, it is possible that a user becomes a contributor and a contributor becomes a core developer. The reasons underlying

[8] In a different study comparing the accuracy of the output from several programs, one of the authors used a one month trial version of an econometric package. Currently, it is impossible for the author to replicate his computations unless he pays an agreed upon sum of money to the software vendor.

[9] Because the English word 'free' cannot fully capture the principle of freedom valued highly by the users of such programs, the French/Spanish word 'libre' is adopted in order to distinguish freeware (gratis software) from free (libre) software liberating computer users from proprietary software under restrictive licensing terms.

[10] There is a multitude of such licenses including the Free Software Foundation's relatively well-known General Public License (GPL), the latest version 3 of which has been officially released in 2007. For a comparison of the various open source software licenses, see Lerner and Tirole (2005b).

**Table 2** Some of the open source programs useful for doing research in economics

| Project | Category | Year | Developers | SLOC | Effort |
|---|---|---|---|---|---|
| Scilab | Numerical analysis | 1994 | 35 | 1,234,895 | 341 |
| GNU Octave | Numerical analysis | 1988 | 74 | 853,439 | 238 |
| Maxima | Algebra | 1998[a] | 17 | 616,576 | 167 |
| R | Statistics | 1997 | 13 | 549,780[b] | 151 |
| SciPy | Mathematical library | 2001 | 31 | 455,903 | 124 |
| Gnumeric | Spreadsheet | 2001 | 9 | 384,341 | 100 |
| Gretl | Econometrics | 2000[a] | 10 | 361,393 | 94 |
| Sage | Mathematics | 2005 | 142 | 195,602 | 51 |
| PSPP | Statistics | 1998 | 3 | 152,593 | 39 |
| Gnuplot | Scientific plotting | 1986 | 6 | 95,380 | 24 |

*Source*: Ohloh.net. The year information is from the project web pages (accessed March 2, 2009)
[a] Shows the year the program became open source
[b] Base system only. The more than 1700 contributed R extension packages are not included

participation in the process include delayed benefits such as learning, promotion, and peer recognition. These motivations find theoretical support in social psychology (Clary et al. 1998) as well as labor economics and industrial organization theory (Lerner and Tirole 2002).[11]

Table 2 shows some of the better known FLOSS tools useful for doing research in economics along with the year of initiation, total source lines of code (SLOC) and an estimation of the effort involved in person years based on the Constructive Cost Model (COCOMO 81) for software costing.[12] As can be seen from the table, most of these programs are relatively young and they show significant progress thanks to a relatively large number of contributors involved in their development.[13] In fact, some of these projects have already reached a level of maturity ranging from useful to indispensable for research purposes. For example, the R programming language and environment has become a de facto standard for the development of new methods in statistics as indicated by a total of 1726 contributed packages allowing specialized statistical techniques.[14] R is becoming widely used in economics as well and there is now a number of textbooks such as Cryer and Chan (2008), Kleiber and Zeileis (2008) and Vinod (2008) devoted to applied econometrics using this program.

The biggest advantage of FLOSS from the perspective of reliability and accountability is the public accessibility of the source code. With a closed source program, the

[11] See Schwarz and Takhteyev (to appear) for a review of the historical evolution of open source software.

[12] The COCOMO 81 model (Boehm 1981) is used for a rough estimation of software costs. Due to lack of data, it is currently not possible to employ more advanced methods such as the COCOMO II (Boehm et al. 2000), which take into account additional factors such as hardware constraints and developer experience.

[13] It is important to point out that econometric software is really a cottage industry where almost all commercial programs are written and maintained by at most a handful of developers. See Renfro (2004b) for a detailed account of the market for econometric software.

[14] The contributed R packages are available at the Comprehensive R Archive Network (CRAN), accessed on March 19, 2009.

programming code is only available to a limited number of individuals namely company employees who are more likely to focus on bugs and enhancements that make commercial sense. Moreover, as Johnson (2006) argues, career concerns can lead paid programmers to collide and avoid revealing flaws in each others' code. Such principal agent problems do not affect FLOSS, which is based on a large group of people from within the economics profession coming together to study, understand, and improve the software without a direct monetary concern. As an example, Listing 1 below shows the gretl source code for the command `make_Omega` in `plugin/jrestrict.c`, which updates the estimate of the covariance matrix in the context of testing restrictions on a vector error correction model (VECM).[15] This transparency and accessibility is not something a user can expect to see in a proprietary program.[16] It also leads to FLOSS being subject to public debate and makes its development a collaborative, merit based process similar in several ways to academics (Lerner and Tirole 2005a; von Krogh and Spaeth 2007). In fact, a piece of open source software is akin to a publication in an academic journal in the sense that it has been vetted by competent experts in the field before being accepted, especially in bigger projects. While everyone can study the code and make suggestions, a revision can only be made by those with a 'commit privilege' obtained through demonstrated competence and knowledge of the subject.[17] Public discussion and review by several top developers and contributors help FLOSS to become more reliable than a closed source alternative for which the user has to take the vendor's word alone.

The collaborative nature of the open source process results in the adoption of several mechanisms and practices that can be of importance in the department of software reliability. One such exercise is the large-scale code reuse within FLOSS projects (Haefliger et al. 2008).[18] Open source licenses such as the GPL allow the sharing of programming code between similarly licensed projects. Yalta (2008) shows that the statistical distributions of the spreadsheet program Gnumeric were more accurate than the alternative OpenOffice Calc and Microsoft Excel applications. This is to be expected because Gnumeric is rising on the shoulder of the R statistical environment by using its math/stats library for these calculations. Similarly, gretl uses the libraries LAPAC for linear algebra, FFTW for fast Fourier transformation, Cephes for statistical distributions, GMP for multiple precision arithmetic, and the programs GnuPlot

[15] As of gretl 1.8.0. The latest version of this file can be seen at http://gretl.cvs.sourceforge.net/viewvc/gretl/gretl/plugin/jrestrict.c.

[16] Some vendors include the pseudo-code, a step by step spelling out of a particular algorithm, for some of the procedures used in their software. Also, some proprietary programs include various 'open' scripts performing a variety of procedures. These, however, can be unreliable because they depend on calls to the proprietary program. Trying to port a GAUSS script to R, Zeileis and Kleiber (2005) report having to go through a numerical detective work only to find in GAUSS itself a programming error that rendered invalid the results of an earlier study.

[17] See von Krogh et al. (2003) for an analysis of the processes through which new individuals join and start contributing code to existing FLOSS projects.

[18] There exists limited source code reuse in closed-source programs through commercial licensing as well. Also, proprietary programs are known to use software free of any copyright or license restrictions. Nerlove (2004) discusses how the public domain EISPAC, LINPACK, and MINPACK libraries originally developed by the Argonne National Laboratories became the building blocks of such packages as GAUSS, MATLAB, SAS, TSP, and LIMDEP.

**Listing 1** Gretl Source Code Updating the Covariance Matrix Estimate in a VECM

```
/*
    Update Omega using:

    S_{00} - S_{01} \beta \alpha' - \alpha \beta' S_{10} +
        \alpha \beta' S_{11} \beta \alpha'

    then invert into iOmega if wanted.
*/

static int make_Omega (Jwrap *J, int code)
{
    int err = 0;

    gretl_matrix_copy_values (J->Omega, J->S00);

    gretl_matrix_multiply_mod (J->alpha, GRETL_MOD_NONE,
                               J->beta, GRETL_MOD_TRANSPOSE,
                               J->Pi, GRETL_MOD_NONE);

    gretl_matrix_multiply_mod (J->S01, GRETL_MOD_NONE,
                               J->Pi, GRETL_MOD_TRANSPOSE,
                               J->Tmppp, GRETL_MOD_NONE);

    gretl_matrix_add_self_transpose (J->Tmppp);
    gretl_matrix_subtract_from (J->Omega, J->Tmppp);

    gretl_matrix_qform (J->Pi, GRETL_MOD_NONE, J->S11,
                        J->Omega, GRETL_MOD_CUMULATE);

    if (code == OMEGA_PLUS) {
        gretl_matrix_copy_values (J->iOmega, J->Omega);
        err = gretl_invert_symmetric_matrix (J->iOmega);
    }

    return err;
}
```

for graphics as well as TRAMO/SEATS and X-12-ARIMA for seasonal adjustment. This not only saves development effort but also improves reliability because the mentioned are all open, well-known, and thoroughly tested tools and programs. A second instrument that can improve the reliability of FLOSS is the use of public bug tracking systems. Unlike the ones employed internally in private software companies, a public issue tracking tool such as Bugzilla lets users enter bug reports directly and allows anyone to see all the reported facts about software defects and known errors. In addition, the system usually supports submission of feature requests and voting by the users on the priority of different issues. The third mechanism is the transparent and participative Internet-based revision control approach employed in FLOSS development. Whenever a developer 'commits' a change, the updated software becomes instantaneously visible and available for all users to 'check out' from the online repository, resulting in better testing.[19] This is unlike commercial software, where a new version of the program can take weeks or months as well as upgrade costs to reach the users and the information regarding the changes is limited to what is supplied by

[19] While FLOSS can certainly be built from source code, in practice many users are likely to use precompiled packages supplied by the developers or various software distributors. These packages can be updated with a lag.

the vendor. Although it is evident that these three mechanisms can improve software reliability, their significance and relative importance warrant further investigation.

In an academic setting, FLOSS and its specific tools and practices have the important consequence of facilitating research replication. The hallmark of science is reproducibility, which helps extend and improve existing research and makes it possible to monitor published results for quality (Vinod 2001). Yet, there is a growing number of studies showing that replicable economic research is the exception and not the rule.[20] Among the impediments to replication is the use of blackbox software, whose details on implemented algorithms are accepted as trade secrets owned by a private company. This, and the fact that errors are inevitable in living software, make it arduous to replicate computational results by even those who have the access to the appertaining commercial programs.[21] Moreover, a proprietary package which is widely-used today can become unavailable in the future. Renfro (2004a,b) discusses end-of-lifed econometric programs such as EAL, ESP, Workbench, DAMSEL, EPS, and XSIM. Also, Koch and Haag (1995) review 82 statistical software packages many of which are not available today. This is a problem because it can be anywhere between unnecessarily difficult and near impossible to revisit research conducted with an extinct program, especially if the code or data is only available in a proprietary machine-readable format. FLOSS can avoid these problems because it is used and redistributed free of charge. Equally easy is obtaining the earlier versions. There is no single person or an entity that can decide to discontinue it. Future users can adapt it or forward-port it to new platforms if necessary. The data and work files are stored in open formats ideal for archiving. See Baiocchi (2007) for an illustration of how scripts using only open source applications can reproduce a project in its entirety from data collection and preparation to analysis and dissemination.

While having important advantages in the department of accuracy and reliability, open source is not without drawbacks. The non-monetizable incentives can result in both over-supply and under-supply of FLOSS or its various components. Indeed, Maurer and Scotchmer (2006) discuss how signalling incentives and the desire to work on intellectually stimulating projects lead to an abundance of highly technical open source projects such as operating systems, programming languages, and virtual machine monitors; while under-serving less interesting programs and tasks such as writing documentation, ensuring backward compatibility, and preparing easy-to-use interfaces and utilities. In particular, unavailable or limited documentation is a commonly encountered problem. For example, while can be considered generally adequate, gretl's some 250 page user guide is dwarfed by the three volume Stata Base Reference Manual comprised of over 2000 pages.[22] Second, the success of FLOSS depends on a critical mass of contributors. With new ideas or products, however, a project can

[20] See Anderson et al. (2008) for a detailed review.

[21] It is important to note that the costs of commercial software can undermine replication significantly. To illustrate, a package costing 500 US dollars corresponds to roughly 13% of monthly per capita GDP in the US, however, this value becomes about 45% in Turkey, and 100% in China.

[22] On the other hand, unlike most commercial econometric packages, gretl's user interface currently supports thirteen languages while its documentation is available in four different languages with two more under translation.

become less exciting and the resultant lack of developers can ultimately lead to 'abandoning.'[23] Projects can also be diverted to an unintended direction ('hijacking') or split into separate projects ('forking'). These can result in wasted developer time and confuse users. Third, driven primarily by the interests of the developers, open source can be less responsive to users, especially those who are not programmers. Further, the profusion of open source licenses is needlessly puzzling. Open Source Initiative (OSI) lists 65 of such licenses, and this is a subset.[24] Finally, it is worth mentioning that proprietary software is reacting to FLOSS by adopting policies that emulate some of its strengths and virtues. Lerner and Tirole (2005a) discuss Microsoft's 'Shared Source' initiative, which lets select third-parties look at the source code of some products under a confidentiality agreement. In addition, Microsoft started in 2006 the Codeplex hosting service, which seeks shared development of open source software between firms and volunteers. Software companies are also emphasizing more on a work environment that respects motives like reciprocity, alturism as well as "being part of a team" (Maurer and Scotchmer 2006). With these new tendencies, the gap between open source and proprietary software is becoming narrower, at least to some extent.

## 4 Fixing of Flaws in Open Source Software

As a case study assessing the reliability of FLOSS in a scientific setting, we applied the Wilkinson tests on gretl, the flagship FLOSS program in the domain of econometric software.[25] Gretl supports a wide variety of estimators and time-series methods accessible via its GUI as well as through its support for scripting. It is written in the C programming language and distributed under the terms of the GPL version 3. The popularity of the program has been increasing and according to the project's web host SourceForge.net, it was downloaded more than 100,000 times in 2008.[26] See Baiocchi and Distaso (2003), Mixon and Smith (2006), Yalta and Yalta (2007), and Rosenblad (2008) for reviews of gretl versions 0.997, 1.51, 1.6.0, and 1.7.3 respectively.

The Wilkinson (1985) tests, discussed in detail by Sawitzki (1994b), is a well-known entry level collection of tests that have been applied in the past to various different statistical and econometric programs. The process is composed of four group of tests focusing on reading data (IA, IB), descriptive statistics (IIA–IIF), missing values

[23] Over time, the abandoned program can also become un-runnable if it depends upon old compliers and libraries. On the other hand, it is possible as well with renewed interest that an abandoned project becomes active again in the future.

[24] Available online at http://www.opensource.org/licenses (accessed on December 10, 2009).

[25] We have been observer and participant in several FLOSS projects including gretl for a long time. Although we never got involved in the coding process of gretl; we made contributions in the form of submitting bug reports, testing the accuracy of various functions, and helping the translation efforts. As a subscriber to the gretl-users mailing list, our personal experience over the years has been that various bug reports are usually corrected in a very short period, often within 48 h.

[26] The number of downloads may not match the number of users for several reasons but mostly because there has been more than one new releases of the program throughout 2008. In an econometric study, Lucchetti (2009) observes in the gretl download data a strong upward trend and estimates that the users of the program have increased steadily at a rate of 43% per year between 2006 and 2009.

**Table 3** The data set NASTY

| X | ZERO | MISS | BIG | LITTLE | HUGE | TINY | ROUND |
|---|---|---|---|---|---|---|---|
| 1 | 0 | NA | 99999991 | 0.99999991 | 1e+012 | 1e−012 | 0.5 |
| 2 | 0 | NA | 99999992 | 0.99999992 | 2e+012 | 2e−012 | 1.5 |
| 3 | 0 | NA | 99999993 | 0.99999993 | 3e+012 | 3e−012 | 2.5 |
| 4 | 0 | NA | 99999994 | 0.99999994 | 4e+012 | 4e−012 | 3.5 |
| 5 | 0 | NA | 99999995 | 0.99999995 | 5e+012 | 5e−012 | 4.5 |
| 6 | 0 | NA | 99999996 | 0.99999996 | 6e+012 | 6e−012 | 5.5 |
| 7 | 0 | NA | 99999997 | 0.99999997 | 7e+012 | 7e−012 | 6.5 |
| 8 | 0 | NA | 99999998 | 0.99999998 | 8e+012 | 8e−012 | 7.5 |
| 9 | 0 | NA | 99999999 | 0.99999999 | 9e+012 | 9e−012 | 8.5 |

(IIIA–IIIC), and linear regression (IVA–IVD). The tests are easy to apply and elegantly designed to reveal defects in statistical software by using a small and effective data set NASTY shown in Table 3. The reason we choose to employ this approach is because it is basic, it is long-familiar in the software industry, and the flaws it is designed to expose have well-known solutions so that a reliable program can and should pass all of the tests. It is important to note that our focus in this study is not the data or the proposed tests per se, but the mechanism ultimately leading to the proper correction of the various flaws encountered after applying these tests.

Reporting on the testing process and its outcomes is important for two reasons. First, it shows how a hands-on approach can reveal in econometric software surprising flaws that can cause invalid and misleading results. A degree of circumspection on behalf of the users, in turn, can help achieve fewer errors and more reliable output in applied economic research. Second, it demonstrates how it is possible with FLOSS to verify the causes of software errors as well as the validity of the subsequent corrections. This in turn can help the uninitiated economist understand the merits of the open source process and better compare it with closed source software.

### 4.1 Tests IA and IB

The first two tests involve reading ASCII data files. Gretl was able to read into memory the NASTY data set in CSV and ASCII formats without problems. Upon loading the data, however, we noticed that gretl's default options for displaying the values was deficient. Table 4 shows the output of gretl's 'display data' window. Here, the series BIG changing between observations 4–5 and LITTLE changing between 5 and 6 is not an error since BIG is evidently governed by switching from regular to scientific notation, while LITTLE is governed by rounding. The real problem is that the display output could not be controlled because gretl's 'reformat' option was only available for a window displaying a single series. This could potentially mislead the user who, for example, might conclude that LITTLE is constant for observations 1 to 5, while BIG is constant through 5 to 9.

**Table 4** Gretl's 'display data' output

| Obs | X | ZERO | MISS | BIG | LITTLE | HUGE | TINY | ROUND |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 0 | | 99999991 | 0.9999999 | 1e+012 | 1e−012 | 0.5 |
| 2 | 2 | 0 | | 99999992 | 0.9999999 | 2e+012 | 2e−012 | 1.5 |
| 3 | 3 | 0 | | 99999993 | 0.9999999 | 3e+012 | 3e−012 | 2.5 |
| 4 | 4 | 0 | | 99999994 | 0.9999999 | 4e+012 | 4e−012 | 3.5 |
| 5 | 5 | 0 | | 1E+8 | 0.9999999 | 5e+012 | 5e−012 | 4.5 |
| 6 | 6 | 0 | | 1E+8 | 1.0000000 | 6e+012 | 6e−012 | 5.5 |
| 7 | 7 | 0 | | 1E+8 | 1.0000000 | 7e+012 | 7e−012 | 6.5 |
| 8 | 8 | 0 | | 1E+8 | 1.0000000 | 8e+012 | 8e−012 | 7.5 |
| 9 | 9 | 0 | | 1E+8 | 1.0000000 | 9e+012 | 9e−012 | 8.5 |

We reported the problem to gretl developers and within three days the program received an update so that the 'reformat' function is now available for a display window showing an arbitrary number of series. The changes were in the form of CVS commits of the modified versions of the `lib/src/printout.c` and `gui2/series_view.c` files related with the display data window.[27] The contents of these source files along with all the revision details can be accessed and examined at http://gretl.cvs.sourceforge.net/.

## 4.2 Test IIA

The second group of Wilkinson tests evaluates the ability to do basic arithmetic and compute various summary statistics. In the case of Test IIA, the program is asked to simply print ROUND with only one digit. This requires rounding done automatically by the program, although the user is allowed to specify formatting statements if necessary. The answer should be the numbers from 1 to 9.[28]

The gretl command `printf "%.0f", ROUND` returned {0, 2, 2, 4, 4, 6, 6, 8, 8}, failing this test. This behavior was due to the gretl commands `print`, `printf`, and `sprintf`; which are based on the corresponding print commands in the C programming language. Depending on the system C library, often these commands round numeric values using 'unbiased rounding' or the 'round-to-even' method so that '.5' does not round up when the preceding digit is even. This is useful during computation because it avoids results becoming upward biased, a potentially serious problem for economic data where trends are important. On the other hand, rounding for the purpose of printing is not about the calculation but the presentation of the results. Here, the expectation of the user is that the output is not presented such that $R(1.5) = R(3.5) = 2$.

Gretl's text printing routines for various objects are controlled by the two source files `lib/src/printout.c` and `lib/src/printscan.c`. Upon our reporting

[27] Revision 1.375 and revisions 1.44–1.49 respectively.

[28] 0.5 rounded to 1 significant digits is still 0.5 since the leading zeros are not significant digits. By stating "Print ROUND with only one digit," Wilkinson (1985) obviously implies rounding to 1 decimal places.

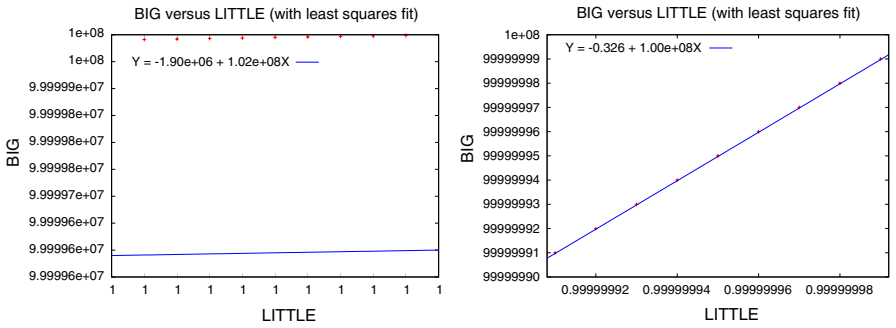


**Fig. 1** Gretl plots of BIG against LITTLE, Before and After

of the potential problems associated with the rounding approach used for displaying values, gretl developers updated these files so that an improved version presenting the data as expected by the users was available on the Internet after 4 days.[29]

### 4.3 Test IIB

Test IIB assesses the graphing accuracy through plotting HUGE against TINY, BIG against LITTLE, and X against ZERO in separate scatterplots. For the first two cases, the answer is a 45° line. For the last one, the answer should be a vertical line. It is not uncommon to get surprising results as can be seen in the left part of Fig. 1 showing gretl's results for BIG versus LITTLE. Here, the first problem is the automatically generated OLS line, which is fitted badly. Second, the horizontal tick-marks are showing all 1's. This is unacceptable because, even if the points are laying on a straight line, it is not possible to know whether they are placed correctly. In addition to these problems, gretl also failed to plot X against ZERO and gave the message "Can't plot with an empty x range!" This too is a failure because, although the graph was not produced, the error message was misleading.

The bad placement of the regression line was in fact a result of gretl's not passing enough digits to Gnuplot for the specification of the fitted line. The problem with the axes was due to not actively specifying the tic-marks for Gnuplot, which by default prints these values with six digits of precision due to typographical concerns. In the case of LITTLE, however, the tick-marks would become all 1's after rounding, resulting in the error. Finally, the failure to plot X against ZERO was because a non-zero range was not forced for Gnuplot when a degenerate x variable was given.

Gretl's graphing behavior causing the problems discussed above is specified in the source files `lib/src/graphing.c`, `lib/src/plotspec.c` and `lib/src/gretl_matrix.c`. After 6 days of our reporting the errors, these files

[29] `lib/src/printout.c` revisions 1.372, 1.373, 1.375–1.378 and `lib/src/printscan.c` revisions 1.21–1.25.

were revised in the online repositories so that the users could obtain the correct graphics including the BIG against LITTLE plot shown on the right part of Fig. 1.[30]

### 4.4 Test IIC

This test involves the computation of various basic descriptive statistics for all the series. The expectation is that the computed mean is equal to the fifth value for each variable. The standard deviation should be 0 for ZERO, missing/undefined for MISSING, and a power of 2.738612788 with some degree of precision for the remaining series.

Gretl automatically removed MISS from the data set and displayed all the values correctly except the standard deviation of TINY, which was reported as 0.0000. This was due to a printing error since the value was actually calculated correctly but was then screened out as effectively zero in view of the limited precision of computer arithmetic. Consequently, the fix involved the tightening of the tolerance in this type of screening in the source file `lib/src/gretl_matrix.c`. Within 24 h of our reporting of the error, gretl's Internet repositories received the revision 1.388 of this file correcting the error by adding four new lines after lines 8044 and 8145 each and modifying the four lines between lines 8115 and 8120. These changes, like the others, can be examined using SourceForge's 'viewvc' interface accessible at http://gretl.cvs.sourceforge.net/viewvc/gretl/.

### 4.5 Test IID

In this test, the program is asked to compute the correlation coefficients and the Spearman's rank correlations between all the variables. All correlations must be unity while those involve ZERO and MISS should be reported as missing or undefined. Gretl computed all these statistics correctly except the Spearman's $\rho$ values involving the series ZERO, which were uniformly reported as:

```
17976931348623157081452742373170435679807056752584499659
89174768031572607800285387605895586327668781715404589535
14382464234321326889464182768467546703537516986049910576
55128207624549009038932894407586850845513394230458323690
32229481658085593321233482747978262041447231687381771809
19299881250404026184124858368.00000000
```

The value above, which is on the order of $10^{309}$, is in fact the largest double precision floating point number that can be represented by most computers. This value has the specially assigned meaning Not Available DouBLe (NADBL) and it is not to be printed as a numerical value. Here, the erroneous output is immediately noticeable, however, it is possible in other cases that the answer is only slightly wrong, misleading the user by

[30] `lib/src/graphing.c` revisions 1.409, 1.410, 1.412–1.416, `lib/src/plotspec.c` revisions 1.42, 1.43, and `lib/src/gretl_matrix.c` revision 1.393 respectively.

going unnoticed. It is therefore simply unacceptable that a program meeting a boundary condition fails to realize it. This flaw was fixed with the 1.392 and 1.411 revisions of the `lib/src/gretl_matrix.c` and `lib/src/graphing.c` files, which were made available by the developers within the next 24 h of our discovery of the error.

### 4.6 Tests IIE Through IVF

In Test IIE, X is tabulated against X using BIG as a case weight. We skipped this test since gretl does not offer this strictly statistical procedure. Test IIF involves BIG regressed on X to see whether the constant term and the regression coefficients are correctly estimated as 99999990 and 1 respectively. Gretl passed this test.

The third group of tests is regarding how missing values are handled by the program. Missing values are common in some areas of economics and therefore require special consideration. For the case of Test IIIA, a new variable TEST is created using the definition `IF MISS=3 THEN TEST=1 ELSE TEST=0`. The following Test IIIB involves transforming MISS using `IF MISS=<missing> THEN MISS=MISS+1`. The answer should be 2 or `<missing>` for the first test and `<missing>` for the second test for all values of TEST. We used the gretl command `genr TEST = 2 - (MISS=3)` to apply Test IIIA and the command `genr MISS = MISS + 1` to apply Test IIIB. Gretl returned the correct answers in both cases.

The fourth group of tests involves regression analysis and the first test in this category intends to push the computer arithmetic to its limits with an estimation of the model

$$X = \beta_1 + \beta_2 X^2 + \beta_3 X^3 + \beta_4 X^4 + \beta_5 X^5 + \beta_6 X^6 + \beta_7 X^7 + \beta_8 X^8 + \beta_9 X^9. \quad (3)$$

As Sawitzki (1994b) points out, the primary objective of this test is not the coefficient estimates but the overall regression, where all the standard errors are 0 and the $R^2$ is unity. Test IVB involves the regression of X on X, which has the obvious solution X=0+1X. Test IVC is a singular model where X is regressed on BIG and LITTLE. The expectation is that the program detects the linear dependence between the two regressors and informs about the problem. Finally, Test IVD regresses ZERO on X, thereby testing for the exceptional case where the regressand is constant. Gretl passed these four tests by computing all of the required values accurately and by stopping or giving an appropriate warning message when necessary.

## 5 The Case with Proprietary Packages

After seeing how quickly a FLOSS tool can fix reported software defects, we decided to focus our attention to some of the well-known closed source programs and their performance in this department. Over 5 years ago, McCullough (2004b) applied the Wilkinson's tests on five widely-used commercial econometric software and in each

of them exposed various accuracy flaws. We decided to reapply the tests on the new versions of these packages.[31] Here is what we found.

It appears that Package1 has fixed the problem of reporting correlation coefficients in excess of unity. Also, the program can now correctly compute the polynomial regression of test IVA. On the other hand, the correlation matrix becomes all missing values when MISS is included among the variables, with or without choosing the 'balanced sample' option. The program is able to graph the BIG against LITTLE plot with accurate axes and the dots laying on a 45° path. Adding an OLS fitted line, however, results in a vertical line, which is obviously incorrect.

We could not apply the tests on Package2 because, unlike the other packages, the demo version offered by the vendor only allows using several built-in data sets. As a result, it was not possible without payment to know whether or not they have fixed the flaws in their product.

We found no change in Package3. The correlations involving ZERO are still given as 0. The program still returns coefficient estimates without warning about singularity for the regression of X on a constant, BIG, and LITTLE (Test IVC). We also observed other errors not mentioned by McCullough such as computing the means and the standard deviations of LITTLE and TINY incorrectly as 0 or 1 (and these are printed with six significant digits). Moreover, the HUGE against TINY plot has wrong axis tick-marks for HUGE.

There are some performance improvements in Package4, which now correctly gives missing values when 1 added to MISS. Also, the inaccurate standard deviations and/or correlation computations involving X, BIG, LITTLE, and MISS are fixed. On the other hand, the correlations and the Spearman's rank correlations of ZERO with the other variables continue to be calculated erroneously as 0 and 1 respectively. Moreover, the correlation of MISS with itself is still 0 while the correlation of ZERO with itself is still 1. In addition, the program now cannot correctly plot BIG against LITTLE, and HUGE against TINY.

The developers of Package5 must have paid attention to the earlier testing so that the correlations of ZERO are now correctly reported as missing. Their web site also includes a batch file for easily running the Wilkinson tests. We noticed, however, that the text based plots of BIG against LITTLE and HUGE against TINY have bad axis labels, resulting in the failure in Test IIB.

When it comes to proprietary software, our experience was similar to that of Sawitzki (1994a) in showing that different software vendors indeed have different levels of concern to computational reliability. After 5 years since McCullough first applied these tests on five commercial econometric packages it is our understanding that two of the software vendors have fixed all of the reported errors, while one vendor fixed some of them, and one vendor fixed none of them. There still exist problems in all of the four packages that we were able to test. Why the various accuracy flaws were not fixed is puzzling, especially considering the fact that applying these relatively simple tests on all of the four packages took us about just a day's effort. We contacted the individual software vendors regarding the problems that we encountered.

[31] Again, we prefer not to identify these packages by name, however, the interested reader can do this by referring to McCullough's study.

Package1 pointed out that they are releasing monthly updates and the correlation matrix problem has been found and fixed several months ago. They also acknowledged the problem with the fitted OLS lines and assured us that this would be corrected within several weeks. Package5 also showed a cooperative concern and fixed the one remaining plotting problem after 3 weeks. Our exchange with Package4, however, warrants further discussion.

Upon reporting the errors, Package4 responded that they were "satisfied that <Package4> is both accurate and reliable." Regarding the issue of returning 1 for the correlation of zero with itself, they first argued that the formula for the coefficient of correlation was not designed for the 0/0 case. Their reference was the 'Ask a Scientist' Internet service intended primarily for K-12 students and teachers. When we wanted to know their mathematical definition of correlation, they responded that it was "common sense" for two series such as $y = \{1, 1, 1, 1\}$ and $x = \{1, 1, 1, 1\}$ to have a correlation of 1. We insisted that correlation is a concept regarding variation and they needed a theoretical justification such as convergence proof for their correlation calculation but their final 'solution' was putting in a new option to produce Not a Number (NaN) if so desired by the user. As for the inaccurate plots, they first claimed that it was possible to override the default values manually to create the plots required. When we asked for the parameters producing the correct graphs; they acknowledged the error, told us this was due to the external program that they used for graphing, and advised contacting the other software. It was when we showed them the accurate plot produced by the other software that they finally accepted to fix the problem in their program but their proposed solution would not correct the error completely.

After our discussion of the persisting errors, we decided to ask vendors whether they are willing to supply a trial copy for someone who wants to replicate results in an article. We also inquired, since computational results can be software dependent, whether it is possible to obtain earlier versions of the various commercial packages. The response that we received from Package1 can help show the conflict of interest posed by using proprietary programs in a scientific setting:

> There is little business sense (after all we are a business, and we cater primarily for business customers rather than for academic research) justification in us providing free copies of <Package1> whenever someone just wants to "replicate results in an article." You are right that not everyone has <Package1>, but it is not in our interest to provide free copies to everyone who doesn't.

Regarding the availability of an older version:

> We no longer distribute <version 4> so I don't think there is any way you could get hold of a copy.

## 6 Conclusion

Powerful computers and advanced software tools made the increasingly complex computations in applied economic research faster and easier but not necessarily more accurate. In the first part of this paper, we presented an extensive account of the subject of software reliability, which does not necessarily mean error free. One has to accept the

fact that complex living software contains errors and imperfections. On the other hand, studies in the last 15 years show that commercial software vendors can also introduce various difficulties to the research process by not correcting the known errors, avoiding to give details on the algorithms, or providing false information regarding their programs. Closed source software can hurt the reliability of computational results by making it impossible to study and verify the programming code performing the myriad functions expected from today's typical econometric package. It also complicates the process of research replication, which is already an exception and not a rule in the field of economics.

The open source movement, which has started to pick momentum after 1998, is now resulting in scientific software reaching and in some cases surpassing in terms of features and usability some of the proprietary alternatives. This new paradigm also brings its own set of inefficiencies such as an over-supply or under-supply of certain types of software, a surplus of licenses as well as the potential for wasted effort due to 'hijacking,' 'forking,' and 'abandoning.' When it comes to reliability and accountability, however, FLOSS helps avoid some of the difficulties associated with proprietary programs. Open source development is a transparent and merit based process similar in some ways to academics. The availability of the source code enables its verification by a large number of people within the economics profession. Because it is free, everyone has access to it. It is flexible and future proof. These not only result in software of a high standard, but also facilitate peer review and help advance research replication.

In an attempt to assess reliability and accountability, we applied an entry level test suite of accuracy on the gretl econometric package and discovered a number of software defects. However, because gretl is open source, our experience was considerably different in comparison to earlier studies assessing various proprietary packages. First, here it was possible to access the source code and see the exact cause of the problems. Second, unlike the other studies, all of the errors were corrected within a week of our reporting. Moreover, each time there was a revision to one of the source files, the updated version of the program was immediately available for download and inspection. This in turn allowed the instant verification of the correction of the errors instead of having to rely only on the developer's claim on accuracy.[32] It is important to note that these improvements were not motivated by an expectation of monetary compensation. Rather, they were due to inducements not unfamiliar to academicians doing scientific research with the objective of creating a public good.

When we applied the same tests on four widely-used proprietary econometric programs, we found that the various flaws uncovered and reported in an earlier study were not necessarily corrected. Despite the 5 years passing, only two of the software vendors have fixed all of the reported errors and still there were problems in all of the packages that we were able to test. It is perhaps misplaced to judge whether a particular package is reliable based on the performance in a single, if well-rounded, suite of accuracy tests. On the other hand, our method by all means qualifies only as an entry level test and one can argue that a stringent test is worth the effort once a package

[32] It is worth mentioning that not only the contents, but also the revision details of all gretl source files can be accessed, examined, and compared starting from version 0.97 released in 2001. One relatively easy way of doing this is by using the SourceForge 'viewvc' GUI accessible at http://gretl.cvs.sourceforge.net/.

passes such a basic procedure (Sawitzki 1994b). Economists need and deserve to use in their research the best available algorithms. The fact that, despite earlier reporting, it is still possible to see flaws and errors in basic functions such as graphing or computing sample standard deviations hardly inspires confidence in the accuracy of the more advanced methods offered by these programs. Hence, our experience shows that, due to being transparent, an open source program can be considered inherently more reliable than a closed source alternative.

Software is the catalyzer that makes the economic theory operational and it is important that the mechanism producing the scientific software is itself open and collaborative. As a result, it is our understanding that the economics profession can benefit from researchers and journal editors who better realize the computational realities as well as the importance of research replication and take a proactive stand by supporting the use of open source software for doing research in economics.

**Acknowledgments** We would like to thank Allin Cottrell from gretl and the support departments of the four proprietary econometric packages for their kind and prompt responses to our reports and inquiries. We also would like to express our appreciation to the anonymous referee, whose comments led to significant improvements to this manuscript. The usual disclaimer applies.

## References

Altman, M., & McDonald, M. P. (2001). Choosing reliable scientific software. *PS: Political Science and Politics, 34*, 681–687.

Altman, M., & McDonald, M. P. (2003). Replication with attention to numerical accuracy. *Political Analysis, 11*, 302–307.

Altman, M., Gill, J., & McDonald, M. P. (2004). *Numerical issues in statistical computing for the social scientist* (1st ed.). New Jersey: Wiley.

Anderson, R. G., Greene, W. H., McCullough, B. D., & Vinod, H. D. (2008). The role of data-code archives in the future of economic research. *Journal of Economic Methodology, 15*, 99–119.

Baiocchi, G. (2007). Reproducible research in computational economics: Guidelines, integrated approaches, and open source software. *Computational Economics, 30*, 19–40.

Baiocchi, G., & Distaso, W. (2003). GRETL: Econometric software for the GNU generation. *Journal of Applied Econometrics, 18*, 105–110.

Bankhofer, U., & Hilbert, A. (1997). Statistical software packages for windows—a market survey. *Statistical Papers, 38*, 377–471.

Boehm, B. W. (1981). *Software engineering economics* (1st ed.). New Jersey: Prentice Hall.

Boehm, B. W., Abts, C., Brown, A. W., Chulani, S., Clark, B. K., Horowitz, E., Madachy, R., Reifer, D. J., & Steece, B. (2000). *Software cost estimation with Cocomo II* (1st ed.). New Jersey: Prentice Hall.

Bonaccorsi, A., Giannangeli, S., & Rossi, C. (2006). Entry strategies under competing standards: Hybrid business models in the open source software industry. *Management Science, 52*, 1085–1098.

Brooks, C., Burke, S. P., & Persand, G. (2001). Benchmarks and the accuracy of GARCH model estimation. *International Journal of Forecasting, 17*, 45–56.

Brooks, C., Burke, S. P., & Persand, G. (2003). Multivariate GARCH models: Software choice and estimation issues. *Journal of Applied Econometrics, 18*, 725–734.

Bruno, G., & Bonis, R. D. (2004). A comparative study of alternative econometric packages with an application to Italian deposit interest rates. *Journal of Economic and Social Measurement, 29*, 271–295.

Choi, H. S., & Kiefer, N. M. (2005). Software evaluation: EasyReg international. *International Journal of Forecasting, 21*, 609–616.

Clary, E. G., Ridge, R. D., Stukas, A. A., Snyder, M., Copeland, J., Haugen, J., & Miene, P. (1998). Understanding and assessing the motivations of volunteers: A functional approach. *Journal of Personality and Social Psychology, 74*, 1516–1530.

Cryer, J. D., & Chan, K. S. (2008). *Time series analysis: With applications in R* (2nd ed.). New York, NY: Springer.

Haefliger, S., von Krogh, G., & Spaeth, S. (2008). Code reuse in open source software. *Management Science, 54*, 180–195.

Heiser, D. A. (2006). Microsoft Excel 2000 and 2003 faults, problems, workarounds and fixes. *Computational Statistics and Data Analysis, 51*, 1442–1443. http://www.daheiser.info/excel/frontpage.html.

Hertel, G., Niedner, S., & Herrmann, S. (2003). Motivation of software developers in open source projects: An internet-based survey of contributors to the Linux kernel. *Research Policy, 32*, 1159–1177.

Johnson, J. P. (2006). Collaboration, peer review and open source software. *Information Economics and Policy, 18*, 477–497.

Keeling, K. B., & Pavur, R. J. (2007). A comparative study of the reliability of nine statistical software packages. *Computational Statistics and Data Analysis, 51*, 3811–3831.

Kitchen, A. M., Drachenberg, R., & Symanzik, J. (2003). Assessing the reliability of web-based statistical software. *Computational Statistics, 18*, 107–122.

Kleiber, C., & Zeileis, A. (2008). *Applied Econometrics with R* (1st ed.). New York, NY: Springer.

Knüsel, L. (1995). On the accuracy of the statistical distributions in GAUSS. *Computational Statistics and Data Analysis, 20*, 699–702.

Knüsel, L. (1996). Telegrams. *Computational Statistics and Data Analysis, 21*, 116.

Knüsel, L (1998). On the accuracy of statistical distributions in Microsoft Excel 97. *Computational Statistics and Data Analysis, 26*, 375–377.

Knüsel, L. (2002). On the reliability of Microsoft Excel XP for statistical purposes. *Computational Statistics and Data Analysis, 39*, 109–110.

Knüsel, L. (2005). On the accuracy of statistical distributions in Microsoft Excel 2003. *Computational Statistics and Data Analysis, 48*, 445–449.

Koch, A., & Haag, U. (1995). The statistical software guide '94/95. A. Koch & U. Haag (Eds.). *Computational Statistics and Data Analysis, 19*, 237–261.

Lakhani, K. R., & von Hippel, E. (2003). How open source software works: "Free" user-to-user assistance. *Research Policy, 32*, 923–943.

Lerner, J., & Tirole, J. (2001). The open source movement: Key research questions. *European Economic Review, 45*, 819–826.

Lerner, J., & Tirole, J. (2002). Some simple economics of open source. *Journal of Industrial Economics, 50*, 197–234.

Lerner, J., & Tirole, J. (2005a). The economics of technology sharing: Open source and beyond. *Journal of Economic Perspectives, 19*, 99–120.

Lerner, J., & Tirole, J. (2005b). The scope of open source licensing. *Journal of Law Economics and Organization, 21*, 20–56.

Lerner, J., Pathak, P. A., & Tirole, J. (2006). The dynamics of open-source contributors. *American Economic Review, 96*, 114–118.

Lovell, M. C., & Selover, D. D. (1994). Econometric software accidents. *The Economic Journal, 104*, 713–725.

Lucchetti, R. (2009). Who uses gretl? An analysis of the SourceForge download data. In *Econometrics with gretl. Proceedings of the gretl conference 2009, Bilbao, Spain* (pp. 45–55).

Maurer, S. M., & Scotchmer, S. (2006). Open source software: The new IP paradigm. In T. Hendershott (Ed.), *Handbook of economics and information systems* (pp. 285–319). Amsterdam: Elsevier.

McCullough, B. D. (1998). Assessing the reliability of statistical software: Part I. *American Statistician, 52*, 358–366.

McCullough, B. D. (1999a). Assessing the reliability of statistical software: Part II. *American Statistician, 53*, 149–159.

McCullough, B. D. (1999b). Econometric software reliability: Eviews, LIMDEP, CHASM and TSP. *Journal of Applied Econometrics, 14*, 191–202.

McCullough, B. D. (2000). Is it safe to assume that software is accurate. *International Journal of Forecasting, 16*, 349–357.

McCullough, B. D. (2001). Does Microsoft fix errors in Excel? In *Proceedings of the 2001 joint statistical meetings*. Alexandria, VA: American Statistical Association.
McCullough, B. D. (2004a). Fixing statistical errors in spreadsheet software: The cases of Gnumeric and Excel. CSDA Statistical Software Newsletter. Retrieved December 10, 2008, from http://www.csdassn.org/software_reports/gnumeric.pdf.
McCullough, B. D. (2004b). Wilkinson's tests and econometric software. *Journal of Economic and Social Measurement, 29*, 261–270.
McCullough, B. D. (2008). Microsoft Excel's 'not the Wichmann–Hill' random number generators. *Computational Statistics and Data Analysis, 52*, 4587–4593.
McCullough, B. D. (2009). Testing econometric software. In T. C. Mills & K. Patterson (Eds.), *Handbook of econometrics* (pp. 1293–1320). New York: Pargrave.
McCullough, B. D., & Heiser, D. A. (2008). On the accuracy of statistical procedures in Microsoft Excel 2007. *Computational Statistics and Data Analysis, 52*, 4570–4578.
McCullough, B. D., & Renfro, C. G. (1998). Benchmarks and software standards: A case study of GARCH procedures. *Journal of Economic and Social Measurement, 25*, 59–71.
McCullough, B. D., & Vinod, H. D. (1999). The numerical reliability of econometric software. *Journal of Economic Literature, 37*, 633–655.
McCullough, B. D., & Vinod, H. D. (2003a). Comment: Econometrics and software. *Journal of Economic Perspectives, 17*, 223–224.
McCullough, B. D., & Vinod, H. D. (2003b). Verifying the solution from a nonlinear solver: A case study. *The American Economic Review, 93*, 873–892.
McCullough, B. D., & Wilson, B. (1999). On the accuracy of statistical procedures in Microsoft EXCEL 97. *Computational Statistics and Data Analysis, 31*, 27–37.
McCullough, B. D., & Wilson, B. (2002). On the accuracy of statistical procedures in Microsoft Excel 2000 and Excel XP. *Computational Statistics and Data Analysis, 40*, 713–721.
McCullough, B. D., & Wilson, B. (2005). On the accuracy of statistical procedures in Microsoft Excel 2003. *Computational Statistics and Data Analysis, 49*, 1244–1252.
McKenzie, C. R., & Takaoka, S. (2002). 2002: A LIMDEP odyssey. *Journal of Applied Econometrics, 18*, 241–247.
McKenzie, C. R., & Takaoka, S. (2007). Eviews 5.1. *Journal of Applied Econometrics, 22*, 1145–1152.
Mixon, J. W., & Smith, R. J. (2006). Teaching undergraduate econometrics with GRETL. *Journal of Applied Econometrics, 21*, 1103–1107.
Nerlove, M. (2004). Programming languages: A short history for economists. *Journal of Economic and Social Measurement, 29*, 189–203.
Newbold, P., Agiakloglou, C., & Miller, J. (1994). Adventures with ARIMA software. *International Journal of Forecasting, 10*, 573–581.
Renfro, C. G. (2004a). A compendium of existing econometric software packages. *Journal of Economic and Social Measurement, 29*, 359–409.
Renfro, C. G. (2004b). Econometric software: The first fifty years in perspective. *Journal of Economic and Social Measurement, 29*, 9–107.
Roberts, J. A., Hann, I. H., & Slaughter, S. A. (2006). Understanding the motivations, participation and performance of open source software developers: A longitudinal study of the Apache projects. *Management Science, 52*, 984–999.
Rosenblad, A. (2008). Gretl 1.7.3. *Journal of Statistical Software, 25*, 19. http://www.jstatsoft.org/v25/s01.
Sawitzki, G. (1994a). Report on the numerical reliability of data analysis systems. *Computational Statistics and Data Analysis, 18*, 289–301.
Sawitzki, G. (1994b). Testing numerical reliability of data analysis systems. *Computational Statistics and Data Analysis, 18*, 269–286.
Schwarz, M., & Takhteyev, Y. (to appear). Half a century of public software institutions: Open source as a solution to the hold-up problem. *Journal of Public Economic Theory*.
Silk, J. (1996). Systems estimation: A comparison of SAS, Shazam and TSP. *Journal of Applied Econometrics, 11*, 437–450.
Stokes, H. H. (2004). On the advantage of using two or more econometric software systems to solve the same problem. *Journal of Economic and Social Measurement, 29*, 307–320.
Vinod, H. D. (2000). Review of GAUSS for Windows, including its numerical accuracy. *Journal of Applied Econometrics, 15*, 211–220.

Vinod, H. D. (2001). Care and feeding of reproducible econometrics. *Journal of Econometrics, 100*, 87–88.
Vinod, H. D. (2008). *Hands-on intermediate econometrics using R* (1st ed.). New Jersey: World Scientific Publishing Company.
von Krogh, G., & Spaeth, S. (2007). The open source software phenomenon: Characteristics that promote research. *Journal of Strategic Information Systems, 16*, 236–253.
von Krogh, G., Spaeth, S., & Lakhani, K. R. (2003). Community, joining, and specialization in open source software innovation: A case study. *Research Policy, 32*, 1217–1241.
West, J. (2003). How open is open enough? melding proprietary and open source platform strategies. *Reseach Policy, 32*, 1258–1286.
Wichmann, B. A., & Hill, I. D. (1982). Algorithm AS 183: An efficient and portable pseudo-random number generator. *Applied Statistics, 31*, 188–190.
Wilkinson, L. (1985). *Statistics quiz: Problems which reveal deficiencies in statistical programs* (1st ed.). Evanston, IL: SYSTAT.
Wilkinson, L. (1994). Practical guidelines for testing statistical software. In P. Dirschedl & R. Ostermann (Eds.), *Computational statistics, 25th conference on statistical computing at Schloss Reisenburg*. Physica, Verlag.
Yalta, A. T. (2007). The numerical reliability of GAUSS 8.0. *The American Statistician, 61*, 262–268.
Yalta, A. T. (2008). The accuracy of statistical distributions in Microsoft ® Excel 2007. *Computational Statistics and Data Analysis, 52*, 4579–4586.
Yalta, A. T., & Jenal, O. (2009). On the importance of verifying forecasting results. *International Journal of Forecasting, 25*, 62–73.
Yalta, A. T., & Lucchetti, R. (2008). The GNU/Linux platform and freedom respecting software for economists. *Journal of Applied Econometrics, 23*, 279–286.
Yalta, A. T., & Yalta, A. Y. (2007). GRETL 1.6.0 and its numerical accuracy. *Journal of Applied Econometrics, 22*, 849–854.
Zeileis, A., & Kleiber, C. (2005). Validating multiple structural change models—a case study. *Journal of Applied Econometrics, 20*, 685–690.