Preface

Charles G. Renfro
601 West 113th Street, #12G, New York, NY 10025, USA
E-mail: cgrenfro@modler.com

This is the second preface that has been written for this special volume on econometric software and its evaluation. The first also started out in the same way to be a short introductory piece, intended to describe the various articles that appear in order to provide you, the reader, with some indication of where each fits into the overall scheme of things. As an opening gambit, it seemed reasonable to begin that introduction by making a few remarks on the early history of the development of this type of software, which has commonly been thought to have been rooted in the efforts of a small band of people sometime in the mid-1960s, although it has long been known that Houthakker, Klein, and Leontief began to use the computer in some way in the late 1940s and early 1950s – but apart from such vague notions, there were mists swirling about. Therefore, having pieced together two or three paragraphs essentially based on personal recollection and some gleanings from articles by Klein, Leontief and others, I sent these out to a dozen or so econometricians, using of course email. Almost immediately, I was startled to find in my inbox something on the order of 25 responses, for of course nearly everyone who has ever created an econometric software package is still alive and the bush telegraph quickly spread the word.

That original introduction in expanded form is now the first article in this volume [10]. It contains a substantial amount of detail about the broad history of the development of econometric software based on the personal recollections of many people. Although I take responsibility for all errors made and all opinions stated that are not enclosed in quotation marks, the article as it stands is the direct result of the patience and cooperation of these people, most of whom have been kind enough to read multiple versions, as the article has steadily grown in size, and to provide a steady stream of information, comments, and suggestions. Moreover, it now represents a diligent effort on the part of all these people to track down information on the history of econometric software development worldwide. Although no doubt still incomplete, for not everyone who has played a part could be tracked down and interrogated under a bare light bulb, the article nevertheless provides, both in its text and its references, enough information that a later generation might be able to make use of it, together with other materials, to actually write a history that puts all this into the proper perspective.

Of course, you may be saying to yourself, why a history of econometric software? And why now? The short answer to these questions is that for economists the computer has increasingly become the primary applied research tool, and it is software that makes the computer work. Moreover, it matters that this software should be the best that it can be, for not only does it permit necessary calculations to be performed but it also determines, for better or worse over time, how easy or how difficult the applied research process will be for each succeeding generation of economists. This assertion assumes of course the availability of the necessary data, and that observations can be obtained relatively easily – but in the day of the Internet, data distribution is also a matter of software. And, in addition, there is the consideration that both the quality and the amount of possible research, as a matter of time spent, may be crucially dependent on just how good that software is, both in its computational properties and as a time saver. All these considerations argue for greater attention being paid to the process of the development of such software, particularly inasmuch as – unlike disciplines such as physics, which have been able over the years to attract substantial funding for the development of research tools – economics is a shoestring activity. We are not blest with major research centers supporting research software development, economists are relatively few in number, and, at the same time, we all wish to obtain software cheaply. These are not circumstances that on their own will naturally lead to the development of more and better software.

In any case, once the word got out about this special volume, the information streaming in from around the world began to fill a sizeable portion of a backup CD-Rom. Furthermore, it soon became clear that there was no point in simply citing quantities of unpublished material. There were too many good stories to tell, not only amusing ones about card decks being run over by delivery trucks or eggs being cooked on the top of the central processing unit of the first operative stored-program computer, but also revealing descriptions of computer-based research that illustrate the role of the computer in the progress of econometric theory and economic research. Consequently, it seemed a very good idea to encourage a number of people to expand their emails into short pieces that describe some of this work. As the second "article," you will find what is in fact a group of short articles, under the rubric *Some Milestones in Econometric Computing*, that together describe aspects of the development of econometric software, starting from the hand calculation era and continuing to relatively modern times. Complementing these, at the end of this issue is a piece entitled *A Compendium of existing econometric software packages*, which brings together descriptions of currently available econometric software packages, in order to provide a statement of where we are now. In almost all cases, these descriptions have been provided by the developers of the individual packages. The exception to this rule is the description of certain software packages that have an historical importance but are no longer supported.

The articles following the Milestones section essentially consist of two types, the first being those that consider characteristics of the computing environment and the second a group focused on the evaluation of econometric software. In the first category, the article by Houston Stokes [12] considers in some detail a number of aspects of the design of econometric software. As Stokes points out, at the beginning of its use by economists the computer was seen simply as an enhancement of hand methods of computing – of the electromechanical desktop calculator in particular – with the then-prevailing sense of the computer being simply that it was a faster way to do what had been done before. However, in recent years especially, with the development of the Internet, the computer has taken on a persona quite different than that of just being fast.

In fact, the Internet is only part of the story. For example, consider word processing: the capability to edit a printed document once it has been created and then simply reprint it is another order of ease of use – as compared to the need, as recently as the late 1970s, to retype entire documents repeatedly as successive changes were made, including (memorably for many people) doctoral dissertations. Other such examples come easily to mind: whereas once an economists might "program the computer," implying essentially making the computer perform a specific set of calculations, today increasingly programs are *designed*, with consideration equally given to the user interface and the process of managing the data used internally, making a variety of the calculations, and then providing an intelligible display of the final results. A number of the facilities that now exist to support software design, as well as the improvement in operating systems and architectural issues associated with various design configurations, are considered in the Stokes paper.

Marc Nerlove [8], in contrast, focuses on the characteristics of programming languages, as these have developed since the early 1950s. The native language of a computer – namely, machine language – is a set of primitive operations that may, for example, simply copy a number from one memory location to another or perform a specific, simple arithmetic operation that today is normally built into the hardware of the central processing unit. The order in which these primitive operations are performed is what determines the work done, inasmuch as complex procedures can be created by combining simple operations in a specific order. Originally, computers needed to be programmed at this primitive level. However, since 1956, with the development of Fortran and subsequently other higher level programming languages, it has increasingly become possible for people to program computers in a way that, although still formalized in style, is nonetheless closer to a "natural" language, each of the statements of which are more content-rich from a human perspective. This evolution of programming languages has in turn set the stage for the development of a generally free-format human-computer interface that has permitted the development of econometric modeling languages, in the 1970s, and econometric programming languages during the 1980s and 1990s, in addition to providing the modern Graphical User Interface (GUI).

In contrast to the papers just mentioned, the next four are much more directly focused on the evaluation of the characteristics of specific econometric software packages. Since 1997, when papers by Bruce McCullough [3] and Charles Renfro [9] separately appeared that each noted and deplored the general absence of the numeric evaluation of econometric software – as opposed to scores of reviews that simply described the offerings and other surface characteristics of such software – there has been a series of papers, beginning with one in this journal in 1998 [4] and another by McCullough and Vinod [5] in the *Journal of Economic Literature* in 1999, that have begun to redress the balance. The latest is to be found in the June 2003 issue of the *American Economic Review* [7]. McCullough has been particularly prominent in both advocating the need for numeric evaluation and leading the charge. In this issue, McCullough [6] in this leading role considers a set of tests originally developed by Leland Wilkinson in the context of more general statistical packages and applies these to number of econometric software packages. The McCullough paper presents important results that need to be carefully considered by econometric software developers, but this paper also raises some general issues concerning both program design and the scope of evaluative tests. In particular, given the inherent imprecision of most economic data, there is, for example, a question whether memory space should be devoted to high precision storage (as opposed to calculation in high precision) simply to insure that a package pass a particular set of tests. It is also open for debate whether tests should be created specifically for econometric software packages, or if it is useful to import them from other, albeit closely related, disciplines.

The next paper by Giuseppe Bruno and Riccardo De Bonis [1] considers existing packages that permit the estimation of panel data. Beyond its specific findings, this paper raises a number of general evaluative issues that need further consideration. From the outset, Bruno and De Bonis encountered a common evaluative problem, which is the absence of a readily available benchmark. One of the effects of this problem is that it prevents a tester from determining more than that the same results can, or cannot, be obtained for two or more packages. Obviously, if the same results can be obtained, the finding would appear to create confidence in the packages' accuracy. If they cannot be, such findings at least flag a need to discover why. But, of course, in the first case, the inference crucially depends upon a presumption that the packages are independently developed, a presumption that itself needs to be examined, inasmuch as a number of econometric software packages contain essentially similar, if not identical code. It may or may not be widely known, but a number of the programs that date from the late 1970s or before involve some sharing of code, usually Fortran code. More recently, as packages have been written or re-written in C, C++ and other languages, the degree of development independence may have increased, but even this is not certain, for there are a number of published listings that are available, dating from even the 1990s, that have no doubt been used to create ostensibly different packages – not to mention the widespread existence of various machine-language code libraries.

There is in fact a general need for the much wider development of benchmark results, and not only to support independent evaluation. The benefit of the development of benchmarks also extends to the support provided to developers during software development. In the absence of the availability of such benchmarks, a developer not only needs to create the code, but also various numeric tests of that code, thus extending the required development time. In addition, it is also true that any test by a developer may be too closely related to the code developed, and thus not necessarily provide the equivalent of a well-founded independent test. Ideally, upon first public release of a software package, it already should have been tested intensively. Independent testing should simply confirm numeric accuracy, not expose inaccuracies. Such an ideal is ideal, but it is too much to expect its achievement in practice. However, if there were to be an increase in the number of available benchmarks, it is likely that both developers and evaluators would take advantage of this, leading to at least closer conformity between reality and the ideal.

The next paper, another by Houston Stokes [11], sheds additional light upon the several matters just discussed. This paper also considers the evaluation of two or more programs together, but from the point of view of a developer. A particularly interesting aspect of this paper is that it demonstrates, among other things, that as a developer it pays to look carefully at results, as well as to approach any particular estimation problem from more than one direction, as the title of his paper implies. The most significant finding is that the properties of estimators can critically depend upon the context in which they are applied, and that routine applications may be as likely to involve inherent computational problems as those of greater ostensible complexity. This paper furthermore demonstrates the importance, after the fact, of a full reporting of computational details as a part of any description of applied research – at least to the degree of identifying at the absolute minimum the software used, including also the platform and version used. Moreover, the paper additionally demonstrates that the careful documentation of a developer's own package can be self-educational.

The final evaluative paper of this volume is that by Ric Herbert, which considers the use of the widely used MATLAB package in a somewhat broader setting than econometrics as normally defined. The Herbert paper does not evaluate the numeric characteristics of this program, particularly issues of numeric accuracy, so much as consider aspects of the applicability of packages like this to economic modeling, to include the use of control theoretic techniques. MATLAB is an example of a specialized programming language that, as Herbert points out, fits "between the conventional programming languages (such as C++ and Java) and the higher level applications packages." As described in the Renfro paper [10], during the past 15 years or so, several programs have appeared that have the characteristics of high level programming languages. These in principle permit their users to exercise greater control over the results obtained, at the cost of taking on more of a programmers role. MATLAB is a member of a class of these packages, which generally do not support parameter estimation – particularly the variety of econometric methods, yet permit the construction and use of models. Inasmuch as parameters can be estimated in one package and the estimated equations then transferred to a package such as MATLAB, there is clearly a use overlap between this program and econometric software.

Apart of the software compendium mentioned earlier, the last paper in this volume is one by Tim Harrison and Charles Renfro [2], which considers data transfer protocols, both those that have been used widely for many years and two that are now proposed as extensions in order to meet future needs. In the early 1980s, Harrison, Renfro and others developed the TSD (Time Series Data) transfer format, originally in order to transfer time series data and associated documentation between the AREMOS and MODLER packages, but also between the mainframe and microcomputer versions of AREMOS. The underlying concept was that economic data series consist not only of observations, but also certain properties that include some sort of identifying name, an observational frequency and a particular date range of availability, at minimum. In addition, ideally, in order for a potential or actual user to understand exactly what the numbers refer to in particular case, additional information is needed, including a short written description, an indication of the original source, the units of measurement, and other such associated documentation. The TSD protocol permits the rapid and efficient transfer of documented time series data and slowly its use has become progressively more widespread. The format is now supported by such data vendors as Economy.Com, Global Insight (the merger of Chase Econometric Associates, Data Resources, and Wharton Econometric Forecasting Associates, all of which previously used it) and Haver Analytics, among others. It is also supported by a number of econometric software packages, as can be seen from the software Compendium. However, the Harrison-Renfro paper now proposes certain extensions to this protocol, including an XML version that permits the extended protocol to be better utilized in the context of the Internet.

This special volume has steadily grown in size during the past six to eight months, to the point that it now forms an entire volume. Notwithstanding, it is not intended to be, in any sense, the final word on the subject of econometric software and its evaluation, but rather only a first contribution to what could be a steadily wider and deeper investigation of this topic over time. The *Journal of Economic and Social Measurement* stands ready to consider for publication additional scholarly articles of the types just described. It stands ready to consider unreservedly articles that take issue with opinions expressed in the articles in this special volume, or that seek to provide a more accurate or more comprehensive historical statement concerning the design or development of econometric software, or the impact of that development on econometric theory or practice. Moreover, such contributions need not be restricted to econometric software as defined in this special volume. This journal today, as it has been the past 30 years, is dedicated to the investigation of all aspects of social and economic measurement and generally seeks to complement and broaden the subject focus of the disciplinary journals in order to encourage and support empirical research.

References

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

[2] T. Harrison and C.G. Renfro, TSX and TSE: A proposal for an extended time series data interchange and transfer format, *Journal of Economic and Social Measurement* **29** (2004), 339–358, this volume.

[3] B.D. McCullough, A review of RATS v4.2: Benchmarking numeric accuracy, *Journal of Applied Econometrics* **12** (1997), 181–190.

[4] B.D. McCullough and C.G. Renfro, Benchmarks and software standards: a case study of GARCH procedures, *Journal of Economic and Social Measurement* **25** (1998), 59–71.
[5] B.D. McCullough and H.D. Vinod, The numerical reliability of econometric software, *Journal of Economic Literature* **37**(2) (1999), 633–665.
[6] B.D. McCullough, Wilkinson's tests and econometric software, *Journal of Economic and Social Measurement* **29** (2004), 261–270, this volume.
[7] B.D. McCullough and H.D. Vinod, Verifying the solution from a nonlinear solver: a case study, *American Economic Review* **93**(3) (2003), 873–892.
[8] M. Nerlove, Programming languages: A short history for economists, *Journal of Economic and Social Measurement* **29** (2004), 189–203, this volume.
[9] C.G. Renfro, Normative Considerations in the development of a software package for econometric estimation, *Journal of Economic and Social Measurement* **23**(4) (1997), 277–330.
[10] C.G. Renfro, Econometric software: The first fifty years in perspective, *Journal of Economic and Social Measurement* **29** (2004), 9–107, this volume.
[11] H.H. Stokes, On the advantage of using two or more econometric software systems to solve the same problem, *Journal of Economic and Social Measurement* **29** (2004), 307–320, this volume.
[12] H.H. Stokes, The evolution of econometric software design: A developer's view, *Journal of Economic and Social Measurement* **29** (2004), 205–259, this volume.