The Numerical Reliability of GAUSS 8.0

A. Talha YALTA

A number of studies have investigated the numerical reliability of the GAUSS Mathematical and Statistical System by Aptech Inc. Those studies identified several critical accuracy errors for different computational methods. We conducted comprehensive tests of this widely used package on estimation, statistical distributions, and random number generation and found that GAUSS 8.0 still has serious problems, especially with statistical distributions and random number generation.

KEY WORDS: Software testing; Numerical accuracy; Statistical reference datasets (StRD).

1. INTRODUCTION

The numerical accuracy of statistical and econometric software cannot be taken for granted, and serious discrepancies can be found in many widely used programs. A number of studies have addressed these problems, for example, McCullough (1999a,b), Vinod (2000), Knüsel (2002), McCullough and Wilson (2002, 2005), and Keeling and Pavur (2007). Many of these studies also report that vendors, in general, can be slow to correct well-documented accuracy errors in their software. Accuracy testing is important for reflecting the profession's ongoing concern about the numerical reliability of its software, resulting in the production of better and more accurate programs.

Here, we review the numerical accuracy of GAUSS Mathematical and Statistical System (GAUSS) version 8.0 to investigate the steps taken by Aptech Systems Inc. to fix the inaccuracies identified in earlier studies by Knüsel (1995) and Vinod (2000). Our results indicate that GAUSS still has unacceptable problems, especially with statistical distributions and random number generation. We also discover that the numerical accuracy of the results produced by the GNU/Linux version of the software is higher—slightly but statistically significant—compared to the MS Windows version.

The following section discusses previous studies regarding the numerical reliability of GAUSS, provides details on the testing method used, and presents our findings after a comprehensive assessment of this popular econometric software. Section 3 offers the conclusions.

A. Talha Yalta is Assistant Professor, TOBB University of Economics and Technology, Söğütözü Caddesi No: 43; Söğütözü, 06560, Ankara, Turkey (E-mail: *yalta@etu.edu.tr*). I wish to thank H. D. Vinod and the referees for their helpful comments and suggestions. I also thank Gail Horecny and the Aptech support team for their efforts in providing rapid and useful answers for my inquiries regarding GAUSS and its inner workings.

2. THE NUMERICAL ACCURACY OF GAUSS

The numerical reliability of GAUSS has been the subject of a number of studies. First, using the SAS program (version 6.10) and his own ELV utility, Knüsel (1995) reported "unacceptable" inaccuracies in the statistical distribution functions in GAUSS. Vinod (2000) performed a more comprehensive analysis of accuracy on multiple fronts and found critical accuracy errors especially in linear and nonlinear regressions as well as serious nonrandomness problems. Blaming both the language itself and the programs offered as a part of the standard package, Vinod concluded by warning: "This should be a wake-up call, since the inaccuracies appear in bread-and-butter econometrics computations." In a recent replication study, Zeileis and Kleiber (2005) discussed a numerical problem "with no easy workaround," which affects GAUSS versions including but not limited to 3.2.32 through 6.08.

We adopt the same method employed by Vinod (2000). Initially proposed by McCullough (1998), this procedure is commonly used to assess the reliability of statistical software on three fronts: estimation, statistical distributions, and random number generation. In particular, we first evaluate the accuracy of estimation using the "Statistical Reference Datasets" (StRD) by the National Institute of Standards and Technology (NIST). Afterwards, we make an assessment of the random number generator and employ the ELV program by Knüsel (2003) to verify accuracy of the values computed by GAUSS for various statistical distributions.

The StRD is composed of five main categories, namely: analysis of variance, linear regression, Markov chain Monte Carlo, nonlinear regression, and univariate summary statistics. The reference datasets are further organized by level of difficulty (lower, average, and higher) according to the complexity of the model used and the stiffness of the data. For all the models, NIST provides certified values computed with multiple precision in 500

Table 1. Numerical tests of univariate summary statistics

		GAUSS SYSTEM 8.0		
Dataset	Diff.	Mean	Std. dev.	Autocr. coeff.
PiDigits	l	15	15	13
Lottery	l	15	15	15
Lew	l	15	15	15
Mavro	l	15	13.1	13.7
Michelso	a	15	13.9	13.4
NumAcc1	a	15	15	15
NumAcc2	a	15	15	13.7
NumAcc3	a	15	9.5	11.2
NumAcc4	h	15	8.3	9

 DOI: 10.1198/000313007X223720

Table 2. Numerical tests of linear least squares.

Dataset	Diff.	GAUSS for WINDOWS 3.2			GAUSS SYSTEM 8.0		
		Coeff.	SE	R-square	Coeff.	SE	R-square
Norris	l	12.2–12.9	10.5	15–15	12.1–12.9	11.1	15–15
Pontius	l	11.6–12.6	7.9	14.6–15	11.3–12.5	7.9	14.5–15
NoInt1	a	14.7–14.7	13.4	0–15	15–15	13.4	0–15
NoInt2	a	15–15	14.3	0–15	15–15	14.3	0.4–15
Filip	h	0–0	0	1.2–2.9	n.s.–0	n.s.	n.s.–3
Longley	h	8.5–11.2	10	13.4–14.8	8.5–11.2	10.3	12.6–15
Wampler1	h	6.1–9.3	0	11.4–15	4.6–9.3	0	10.5–15
Wampler2	h	9.4–12.7	4.4	11.4–15	9.4–13.4	0	10.8–15
Wampler3	h	6.1–9.2	6.3	11.4–15	4.6–9.6	5.5	10.5–15
Wampler4	h	6.1–8.4	10.1	11.4–15	4.6–8.3	9.5	10.5–15
Wampler5	h	6.1–6.4	10.5	11.4–13.5	4.6–6.3	11.6	10.5–13.2

Note: Numbers on the right show the NADs obtained with the "`olsqr2()`" function.

digits of accuracy, which were later rounded to 15 significant digits (11 for nonlinear least squares). We evaluated GAUSS in all departments except for Markov chain Monte Carlo and analysis of variance, for which there is no function available as a part of the standard package. We compared the output with the certified values from the StRD by calculating the number of accurate digits (NADs) based on the log relative errors defined as:

$$\text{LRE} = \max\left(0, -\log_{10}\left[\,|\,g - c\,|\,/\,|\,c\,|\,\right]\right), \qquad (1)$$

where g is the value computed by GAUSS and c is the NIST-certified correct value. In the case that c equals zero, NAD is given by the log-10 logarithm of the absolute error: LAE $= -\log_{10}[\,|\,g\,|\,]$. For problems with multiple parameters, we chose the least accurate of each vector based on the principle that the strength of the weakest link measures the ultimate strength of a chain. All tests were conducted using a 2.00 GHz Intel Centrino computer configured for dual booting both the GNU/Linux (Debian) and the MS Windows XP operating systems.

2.1 Numerical Tests of Univariate Summary Statistics

In the StRD univariate summary statistics test suite, NIST provides certified values for the mean, standard deviation, and the first-order autocorrelation coefficient for five "real-world" and four artificial datasets with the number of observations ranging from 3 to 5,000. NIST acknowledges that the first-order autocorrelation coefficient "may have several definitions." We used the definition employed in the StRD ($\text{cov}(y_t, y_{t-1})/\text{var}(y_t)$) for the sake of comparison. The column means and standard deviations were calculated using the GAUSS commands "`meanc()`" and "`stdc()`," respectively. The results, which concur with those of Vinod (2000), are provided in Table 1. The performance of GAUSS 8.0 on this suite of tests is *acceptable.*

2.2 Numerical Tests of Linear Least Squares

GAUSS offers several alternatives such as "`ols()`", "`olsqr()`", "`olsqr2()`", and the slash operator (bhat $=$ y/x) for solving least squares regressions. The slash operator and the "`olsqr()`" functions compute only the regression coefficients. McCullough (1998) argued that when there are multiple options, "the simplest or most obvious method should be used for calculating each statistic." We used the "`ols()`" and the "`olsqr2()`" commands, which were also chosen by Vinod (2000) for testing.

The "`olsqr2()`" command employs the QR decomposition, which is known to be more accurate in comparison to other techniques such as the LU factorization or Cholesky factorization. The "`ols()`" command uses neither of those methods because it is designed to compute regressions where the data is processed in small pieces. This feature is useful when the data are too large to fit into memory; however, it also results in further losses in numerical accuracy.

Table 2 compares the NADs for the coefficients, standard errors, and R^2 obtained with the "`ols()`" and "`olsqr2()`" commands for both GAUSS 3.2 for Windows and GAUSS System 8.0. There is a methodological disagreement among statisticians in computing the R^2 for models without an intercept as discussed by Ramanathan (2002, p. 151) . Apparently, the "`ols()`" function does not use the same definition employed by NIST, hence the zero NADs for R^2 for the "`noint1`" and "`noint2`" datasets. Also, "Filip" is a tenth-order polynomial and many software packages are known to fail this test by providing zero NADs without giving any warnings. Unlike the earlier version, the "`ols()`" function in GAUSS 8.0 correctly detected the near singularity in the covariance matrix of independent variables and gave an appropriate error message. This response is acceptable because the user is not misled. On the other hand, the "`olsqr2()`" command still returns zero or very low NADs for this test without any notification and needs to be fixed.

It is noticeable that there were modifications in both the "`ols()`" and the "`olsqr2()`" commands in GAUSS. However, it is difficult to say that the numerical accuracy of these functions has been significantly improved after more than seven years since errors have been reported. In fact, comparing the output of the two versions, one sees that the results have deteriorated in 20 cases out of 55 calculated statistics. It should be Aptech's priority to increase the accuracy of the basic "`ols()`" command by using the Cholesky update, which was not available

Table 3. Numerical tests of nonlinear least squares

		GAUSS for WINDOWS 3.2		GAUSS SYSTEM 8.0					
		Default Settings		Default Settings		Pref. Numeric		Pref. Analytic	
Dataset	Diff.	Coeff. –	SE	Coeff. –	SE	Coeff. –	SE	Coeff. –	SE
Misra1a	l	7.4 –	7.1	ns –	ns	9.8 –	9.5	8.8 –	6.6
Chwirut2	l	5.0 –	5.4	7.0 –	6.9	7.0 –	6.9	8.5 –	8.9
Chwirut1	l	5.6 –	0	7.6 –	7.1	8.4 –	7.3	8.6 –	8.9
Lanczos3	l	3.2 –	3.2	0 –	0	5.5 –	4.7	6.5 –	6.4
Gauss1	l	8.8 –	0	9.0 –	7.8	9.0 –	7.8	10.7 –	10.9
Gauss2	l	9.0 –	8.4	8.4 –	7.3	8.4 –	7.3	10.0 –	10.0
DanWood	l	7.9 –	0	6.5 –	6.7	9.1 –	7.8	11.3 –	11.2
Misra1b	l	8.5 –	8.5	ns –	ns	9.1 –	6.4	10.3 –	10.3
Kirby2	a	0 –	0	ns –	ns	ns –	ns	10.4 –	10.7
Hahn1	a	0 –	0	ns –	ns	ns –	ns	10.7 –	10.3
Nelson	a	0 –	0	ns –	ns	ns –	ns	ns –	ns
MGH17	a	0 –	0	ns –	ns	7.0* –	6.0*	9.7* –	9.2*
Lanczos1	a	0 –	0	0 –	0	9.3 –	1.7	9.3 –	1.8
Lanczos2	a	3.2 –	1.0	0 –	0	6.6 –	5.1	7.1 –	7.1
Gauss3	a	8.2 –	7.9	7.7 –	6.9	9.1 –	6.9	10.5 –	10.4
Misra1c	a	7.2 –	7.0	ns –	ns	9.2 –	6.2	9.9 –	9.7
Misra1d	a	3.0 –	0	9.5 –	6.4	9.5 –	6.4	11.4 –	10.9
Roszman1	a	7.1 –	1.4	5.4 –	6.0	7.6 –	7.2	7.1 –	7.6
EnsO	a	5.8 –	6.8	5.0 –	6.1	5.8 –	6.6	7.4 –	8.6
MGH09	h	5.3 –	0	2.4 –	2.6	6.8 –	6.8	7.6 –	7.7
Thurber	h	6.0 –	1.1	7.4 –	6.7	7.4 –	6.7	9.2 –	8.6
BoxBOD	h	8.2 –	8.0	6.4 –	6.4	8.6 –	8.6	9.3 –	9.3
Rat42	h	0 –	1.3	8.8 –	7.6	8.8 –	7.6	11.2 –	10.6
MGH10	h	ns –	ns	ns –	ns	7.5* –	5.3*	8.8* –	8.9*
Eckerle4	h	9.1 –	9.3	7.6 –	7.6	10.2 –	7.9	10.5 –	10.5
Rat43	h	6.7 –	0	6.7 –	6.3	7.9 –	6.4	9.6 –	9.9
Bennett5	h	ns –	ns	4.6 –	4.7	4.6 –	4.7	9.9 –	7.0

* For the MGH17 and MGH10 datasets, convergence was achieved using the easier start2 values.

in GAUSS when this command was first written. Meanwhile, the program's performance in this fundamental area can be considered *acceptable but unimpressive.*

2.3 Numerical Tests of Nonlinear Least Squares

GAUSS does not have a native implementation of nonlinear least squares. However, there are various add-on modules written *in* GAUSS for this estimation method. Vinod (2000) reported important accuracy errors for this test suite using Aptech's official "Constrained Optimization" (CO) module. However, CO is designed primarily for solving constrained nonlinear programming problems and the way convergence is determined by this module differs from unconstrained optimization problems similar to those provided by the StRD nonlinear regression test suite. Consequently, for our own testing, we decided to use the "Curve Fit" package, which is another premier application module by Aptech available at an extra cost and designed specifically for solving this type of problem with greater accuracy. By default, the Curve Fit module employs the Levenberg-Marquardt variation of the Gauss-Newton method, however, it is also possible to use the conjugate gradient method (the Polak-Ribiere variation) as the iteration technique. The primary method for convergence testing is the relative gradient and the default convergence tolerance is 1.0E-5.

For nonlinear estimation, the results obtained with any software using the default options is rarely as good as what is possible using some other setting and it is the responsibility of the user to improve the accuracy of output by learning how to use the software effectively. Therefore, in Table 3, we provide the regression coefficients and standard errors calculated in three different ways: In the first method, the estimations were carried out simply by using Curve Fit's default options. The second and third methods involve using our preferred combination of settings without and with the user-supplied analytical derivatives. The analytical derivatives not only deliver more accuracy, but also enable the solution of problems that could not be solved by their numerical approximations. For each dataset, the preferred combination of settings was determined after running 96 separate estimations which involve

1. using either the Levenberg-Marquardt method or the conjugate gradient method;
2. using either the analytical derivatives or their numerical counterparts;
3. using the two sets of start values provided by the StRD where the first set is further from the solution, hence considered more difficult compared to the second set;
4. repeating the above while increasing the convergence tolerance from 1.0E-3 to 1.0E-14 gradually.

With a failure to achieve convergence in eight datasets and reporting zero or very low NADs without any warning in five other datasets, the performance of the Curve Fit module with the default settings is rather poor. Once again, the StRD shows that users should not trust results produced with default settings. On the other hand, the results obtained with the preferred settings can be judged acceptable. Although some of the nonconvergence problems linger when numerical derivatives are employed, the user is not misled and the performance further improves with the use of analytical derivatives. Consequently, GAUSS' performance in the StRD nonlinear regressions test suite using the Curve Fit module, which is recommended by Aptech for this estimation method, is judged *acceptable*.

2.4 Randomness Tests of Random Number Generation

The quality of pseudo random numbers generated by computer software is critical especially for Monte Carlo simulations and bootstrap methods, which often require billions of calls to the random number generator (RNG). A good RNG must create reproducible output, have a very long period, and produce numbers which are not only uniformly distributed but also independent in a moderate number of dimensions, as discussed by Ripley (1990). Using the DIEHARD program by Marsaglia (1996), Vinod (2000) uncovers critical nonrandomness problems in GAUSS and advises researchers against using this software for RNG intensive tasks.

After Vinod's study and since version 3.6, GAUSS now offers two RNGs, namely "`rndi()`" and "`rndkmi()`." The "`rndi()`" algorithm, which was tested previously, is of linear congruential type with a short maximum period ($2^{32}-1$) and various other deficiencies. For example, one important problem with this class of RNGs is that the pseudo random numbers created will have nonrandom lower order bits. Also, when the random numbers are used as a source of points in an n-dimensional cube, as the number of dimensions increases, the points tend to cluster on plains.

The newer "`rndkmi()`" function introduced in GAUSS 3.6 is an implementation of the "recur-with-carry" type KISS+Monster algorithm by George Marsaglia of Florida State University. According to a technical report by Ford and Ford (2001) available for download from Aptech's Web site, this new generator claims to have an impressive period greater than 10^{8888}, provides more than 920 dimensions, and claims to pass all of the 18 randomness tests in the DIEHARD suite.

Aptech does not allow these claims to be independently verified, so users should be wary. Aptech claims the code as proprietary and does not make the algorithm or code available for inspection. It is crucial to have open access to the RNG and the importance of this issue is best seen in the case of Microsoft Excel discussed by McCullough and Wilson (2005). As a response to criticisms of the unacceptably bad RNGs in Excel, Microsoft modified the "RAND" function claiming to implement the Wichmann-Hill (1982) RNG in Excel 2003. However, Microsoft did not provide the source code, nor could the users enter a seed to verify Microsoft's claims. It was later understood that the new RNG was not the Wichmann-Hill (or its correct implementation) because the "RAND" function would occasionally produce negative numbers for the uniform [0,1]. L'Ecuyer and Simard (2006) showed that poor (or plain bad) generators can still be found in popular commercial statistical and simulation software and there is no reason to trust a "black box" RNG especially when there are excellent free/libre and open-source alternatives available.

Another issue with the new "`rndkmi()`" function is regarding Aptech's claim that it passes all of the 18 randomness tests in the DIEHARD suite. Although this marks an improvement over the old "`rndi()`" function, the DIEHARD program itself is passé. McCullough (2006) discussed a new program named TESTU01 by L'Ecuyer and Simard (2006), which provides a more comprehensive testing of uniform random number generators. Many RNGs that are known to pass the DIEHARD tests fail the TESTU01 tests. However, using TESTU01 for testing the "`rndkmi()`" command requires the source code of the program, which is not available. As a result, until Aptech implements an RNG that can be publicly verified to be capable of passing the tests in TESTU01, the performance of GAUSS in random number generation is considered *unacceptable*.

2.5 Numerical Tests of Statistical Distributions

Knüsel (1995) documented serious flaws in the computation of several elementary statistical distributions in the MS-DOS version of GAUSS 3.2.6 and Knüsel (1996) later found that the same errors were not fixed in version 3.2.13. Vinod (2000) also reported that these accuracy problems have not been corrected in GAUSS for Windows 3.2.37. More recently, while porting the code for the multiple structural change model by Bai and Perron (2003) to GNU-R, Zeileis and Kleiber (2005) discovered that, in GAUSS version 3.2.32, algorithm errors in the "`lncdfn`" function, which computes the natural log of Normal cdf, results in an underflow in the computation of confidence intervals for a nonstandard distribution beyond the 90.7% quantile, rendering the original results invalid. They also confirmed that this numerical problem persisted in GAUSS versions including but not limited to 5.0.22, 5.0.25, and 6.08 and that it was finally fixed by Aptech while the study was in review.

In order to see whether GAUSS 8.0 can finally be considered reliable in the statistical distributions department, we employed the second edition of Knüsel's (2003) ELV program, which is capable of calculating exact values of nine elementary statistical distributions for probabilities as small as 10^{-100}. Tables 4 through 8 compare critical values computed by GAUSS 3.2.6, GAUSS 8.0, and GNU-R 2.5.0 with their "exact" counterparts computed by ELV for the F, beta, and noncentral chi-square, noncentral F, and noncentral t-distributions, respectively. It is quite noticeable that Aptech has attempted to fix some of the problems in these functions and the program no longer hangs up or gives over/underflow errors. In addition, the documentation is updated to include more details regarding GAUSS's statistical algorithms. The noncentral t-distribution also *seems* to be accurate in the new version, however, there is plenty of room for improvement in this area as well. Several entries showing 0.00e+000 under GAUSS 8.0 prove that the F and Beta distributions are still very unstable and therefore unsafe to use. Also, the noncentrality parameters for the chi-square and the F-distributions need to be revised in order to comply with the ELV program and most other statistical packages including GNU-R.

Table 4. F-distribution with (n_1, n_2) degrees of freedom.

x, n_1, n_2	ELV	GAUSS 3.2.6	GAUSS 8.0	GNU-R 2.5.0
35, 2, 100	3.00e − 012	3.01e − 012	Exact	Exact
47, 2, 100	4.07e − 015	4.82e − 015	0.00e + 000	Exact
48, 2, 100	2.44e − 015	0.00e + 000	4.44e − 015	Exact
50, 2, 100	8.88e − 016	0.00e + 000	0.00e + 000	Exact
51, 2, 100	5.40e − 016	Exact	Exact	Exact
1000, 2, 100	7.75e − 067	Exact	Exact	Exact

Table 5. Beta distribution with parameters (a, b).

x, a, b	ELV	GAUSS 3.2.6	GAUSS 8.0	GNU-R 2.5.0
0.8, 100, 1	2.04e − 010	Exact	Exact	Exact
0.73, 100, 1	2.15e − 014	2.50e − 014	9.77e − 015	Exact
0.72, 100, 1	5.41e − 015	0.00e + 000	2.55e − 014	Exact
0.51, 100, 1	5.72e − 030	0.00e + 000	0.00e + 000	Exact
0.5, 100, 1	7.89e − 031	Exact	Exact	Exact
0.2, 100, 1	1.27e − 070	Exact	Exact	Exact

Table 6. Noncentral chi-square distribution with parameters (n, λ).

x, n, λ	ELV	GAUSS 3.2.6	GAUSS 8.0	GNU-R 2.5.0
1500, 20, 37	1.00e + 000	9.30e − 001	9.30e − 001	Exact
1500, 20, 38	1.00e + 000	Over/underflow(!)	6.85e − 001	Exact
1500, 20, 394	1.00e + 000	Underflow(!)	3.03e − 001	Exact

Table 7. Noncentral F-distribution with parameters (n_1, n_2, λ).

x, n_1, n_2, λ	ELV	GAUSS 3.2.6	GAUSS 8.0	GNU-R 2.5.0
100, 10, 1, 37	8.30e − 001	2.41e − 001	2.41e − 001	Exact
100, 10, 1, 38	8.28e − 001	Hangs up(!)	2.28e − 001	Exact
100, 10, 1, 39	8.26e − 001	Underflow(!)	2.16e − 001	Exact

Table 8. Noncentral t-distribution with parameters (n, δ).

x, n, δ	ELV	GAUSS 3.2.6	GAUSS 8.0	GNU-R 2.5.0
5, 3, 11.7	1.91e − 003	Exact	Exact	Exact
5, 3, 11.8	1.70e − 003	Overflow(!)	Exact	Exact
10, 3, 11.3	2.86e − 001	Exact	Exact	Exact
10, 3, 11.5	2.71e − 001	Overflow(!)	Exact	Exact
20, 3, 11.2	8.14e − 001	Exact	Exact	Exact
20, 3, 11.4	8.06e − 001	Overflow(!)	Exact	Exact

All in all, GAUSS' performance on statistical distributions is judged *unacceptable*.

2.6 Windows Version Compared to the GNU/Linux Version

GAUSS was first introduced in 1984 as a terminal application for MS-DOS and was initially written in the Assembler language. MS-DOS is a 16-bit operating system, and when GAUSS was later ported to the 32-bit MS Windows platform it was rewritten entirely in the C programming language. Consequently, the current source code for the number-crunching parts of GAUSS are identical for both the MS Windows and the GNU/Linux platforms.

During our testing of GAUSS, we discovered that, for the StRD nonlinear least squares test suite, the GNU/Linux version of the program fails to converge with analytical derivatives in the "MGH10" and the "MGH17" models and it also requires the easier set of start values to converge with the default settings in the "Gauss1" model. It is likely that Aptech's Curve Fit module was primarily developed and tested on the MS Windows platform and we suspect this is the reason why the GNU/Linux version has more difficulty in converging (11 datasets instead of 8 out of 27) in comparison with the Windows version.

For StRD's univariate summary statistics and the linear regression test suites—which involve more straightforward computations—it has not escaped our notice that the GNU/Linux version of GAUSS produces considerably higher NADs in 24 cases, while the NADs are lower in 8 cases. The differences can be quite large especially for the linear least squares procedure. For example, using the "`olsqr2()`" function, the lowest NAD for the coefficients of the Longley model under Windows was 11.2 with a GNU/Linux counterpart of 12.5. The mean difference in NADs was only 0.23, however, running a paired sample t test, we failed to reject the hypothesis that differences are significant at the 5% level when there are platform-dependent disparities in results. These small discrepancies in NADs are likely to be caused by the different C libraries and compilers used on the two platforms. Yalta and Yalta (2007) also mention similar differences between the GNU/Linux and MS Windows versions of GRETL, the free/libre and open-source econometrics package.

Research replication is the cornerstone of science and it is quickly becoming an important concern with today's trend toward openness in econometric analysis. McCullough and Vinod (2003) argued that a replicator needs not only access to the data and to the program code, but also knowledge of the software version and the operating system. As a result, our findings can be interpreted as additional proof that knowledge of the operating system is imperative.

3. CONCLUSION

There have been significant enhancements in the accuracy of GAUSS since version 3.2 and this shows that software developers do respond to benchmarking as argued by Keeling and Pavur (2007). Still, it is our understanding that GAUSS 8.0 has plenty of room for improvement. First of all, Aptech should correct the various accuracy flaws which can still be found even in the most basic functions. For example, Vinod's improved "`meanc2()`" and "`stdc2()`" procedures, posted online at the GAUSS Source Code Archive (*http://www.american.edu/academic.depts/cas/econ/gaussres/GAUSSIDX.HTM*), perform the computation of column means and standard deviations more accurately than GAUSS's own "`meanc()`" and "`stdc()`" commands. Also, the "`ols()`" and the "`olsqr2()`" commands, several statistical functions as well as the CurveFit add-on need further improvements. Moreover, having open access to the random number generator is crucial and GAUSS should follow the example of other software such as GNU-R, LIMDEP, and TSP in using a publicly vetted RNG. Finally, the number of commands and functions available in GAUSS looks impressive until one realizes that, added at different intervals, some of these programs perform similar operations with different levels of numerical accuracy. We believe that Aptech's attempts to keep syntax simple this way is a poor design choice, which can lead to the creation of less than optimal code by many users.

In conclusion, it seems that the primary feature that sets GAUSS apart from its alternatives is its speed and this is Aptech's main concern before making any modifications to the language. It is true that choosing a fast algorithm over a slower, albeit slightly more accurate, algorithm can facilitate estimation of a large and complex model by reducing a run that takes days into one that completes in only several hours. On the other hand, small accuracy sacrifices accumulate over time so that it becomes difficult for the researcher to be confident in the output after several million iterations. In a world that increasingly demands reproducible research results, and with computing costs plummeting, speed in execution of programs is slowly becoming a trivial issue. Moreover, many scientific departments today are also discovering parallel computing possibilities offered by various cluster management systems such as Beowulf and openMosix. These free/libre programs (available gratis and also libre as they liberate computer users from proprietary software under restrictive licensing terms) make it easier than ever to create super computers running on commodity hardware under the GNU/Linux platform. As a result, it is our opinion that GAUSS is currently suitable for rapid deployment of complex models and checking preliminary findings, however, researchers need to consider using a more accurate alternative for the final analysis. Also, journal editors should be judicious about results based on GAUSS versions 8.0 and earlier.

REFERENCES

Bai, J., and Perron, P. (2003), "Computation and Analysis of Multiple Structural Change Models," *Journal of Applied Econometrics*, 18, 1–22.

Ford, M. P., and Ford, D. J. (2001), "Investigation of GAUSS' Random Number Generators," {*http://www.aptech.com/papers/rndu36.pdf*}, [Online: accessed October 3, 2006].

Keeling, K. B., and Pavur, R. J. (2007), "A Comparative Study of the Reliability of Nine Statistical Software Packages," *Computational Statistics and Data Analysis*, 51, 3811–3831.

Knüsel, L. (1995), "On the Accuracy of the Statistical Distributions in GAUSS," *Computational Statistics and Data Analysis*, 20, 699–702.

——— (1996), "Telegrams," *Computational Statistics and Data Analysis*, 21, 116.

——— (2002), "On the Reliability of Microsoft Excel XP for Statistical Purposes," *Computational Statistics and Data Analysis*, 39, 109–110.

——— (2003), "Computation of Statistical Distributions—Documentation of

Program ELV." {*http://www.stat.uni-muenchen.de/~knuesel*}, [Online; retrieved October 3, 2006].

L'Ecuyer, P. L., and Simard, R. (2006), "TESTU01: A Software Library in ANSI C for Empirical Testing of Random Number Generators," *ACM Transactions on Mathematical Software*, under revision.

Marsaglia, G. (1996), "DIEHARD: a Battery Test of Randomness," *http://stat.fsu.edu/pub/diehard*. [Online: accessed October 3, 2006].

McCullough, B. D. (1998), "Assessing the Reliability of Statistical Software: Part I," *The American Statistician*, 52, 358–366.

——— (1999a), "Assessing the Reliability of Statistical Software: Part II," *The American Statistician*, 53, 149–159.

——— (1999b), "Econometric Software Reliability: Eviews, LIMDEP, CHASM and TSP," *Journal of Applied Econometrics*, 14, 191–202.

——— (2006), "A Review of TESTU01," *Journal of Applied Econometrics*, 21, 677–682.

McCullough, B. D., and Vinod, H. D. (2003), "Verifying the Solution from a Nonlinear Solver: a Case Study," *The American Economic Review*, 93, 873–892.

McCullough, B. D., and Wilson, B. (2002), "On the Accuracy of Statistical Procedures in Microsoft Excel 2000 and Excel XP," *Computational Statistics and Data Analysis*, 40, 713–721.

——— (2005), "On the Accuracy of Statistical Procedures in Microsoft Excel 2003," *Computational Statistics and Data Analysis*, 49, 1244–1252.

Ramanathan, R. (2002), *Introductory Econometrics with Applications* (5 ed.), Ohio: South-Western College Pub.

Ripley, B. D. (1990), "Thoughts on Pseudorandom Number Generators," *Journal of Computational and Applied Mathematics*, 31, 153–163.

Vinod, H. D. (2000), "Review of GAUSS for Windows, Including Its Numerical Accuracy," *Journal of Applied Econometrics*, 15, 211–220.

Wichmann, B. A., and Hill, I. D. (1982), "Algorithm AS 183: an Efficient and Portable Pseudo-Random Number Generator," *Applied Statistics*, 31, 188–190.

Yalta, A. T., and Yalta, A. Y. (2007), "GRETL 1.6.0 and Its Numerical Accuracy," *Journal of Applied Econometrics*, 22, forthcoming.

Zeileis, A., and Kleiber, C. (2005), "Validating Multiple Structural Change Models – A Case Study," *Journal of Applied Econometrics*, 20, 685–690.