Bridging the Gap Between Ox and Gauss using OxGauss
Forthcoming in *Journal of Applied Econometrics*

Sébastien Laurent

CeReFim (Université de Namur)

and CORE (Université catholique de Louvain).

E-mail: Sebastien.Laurent@fundp.ac.be.

Correspondence to 8 rempart de la vierge, B5000 Namur, Belgium.

Phone: +32 (0) 81 724869.

Fax: +32 (0) 81 724840.

and

Jean-Pierre Urbain

Department of Quantitative Economics, Universiteit Maastricht,

The Netherlands. E-mail: j.urbain@ke.unimaas.nl.

First Draft: October 2003
This version: April 2004

1 Introduction

The goal of this paper is to review and discuss OxGauss, a program available with recent versions of Ox.[1] OxGauss provides a way to run Gauss[2] programs in the Ox environment or to call an existing Gauss procedure under Ox in the same way that C or Fortran programs can be called from Gauss and Ox. Unlike the old g2ox program provided with Ox, OxGauss is *not* a Gauss-to-Ox translator.

Depending on the goal of the analysis and the user's experience, both features are noteworthy and useful. From an Ox user point of view, the main objective of OxGauss is to allow existing Gauss

[1]There are two versions of Ox. Oxconsole can be downloaded from *http://www.nuff.ox.ac.uk/Users/Doornik/*, which is the main Ox web page. The console version is free for educational purposes and academic research. The Professional Windows version, or commercial version comes with a nice interface for graphics known as GiveWin (available for purchase from Timberlake Consultants, *http://www.timberlake.co.uk*). Both versions are provided with a comprehensive documentation of OxGauss available in the file *OxAppendix.pdf*.

[2]Gauss is sold by Aptech Systems, 23804 S.E. Kent-Kangley Rd., Maple Valley, WA, 98038, USA; see *http://www.aptech.com/*.

programs to be called from Ox with only a minimum number of changes to these programs. This is beneficial to both Ox and Gauss users. It provides more visibility to both and hence increases the potential use of the underlying statistical technique. Furthermore, it can help with the migration from Gauss to Ox.

Running a pure Gauss code with OxGauss is attractive for non-Gauss and potentially even for non-Ox users because it allows the replication of published work using the console version of Ox. This is an interesting feature since the replicability of simulation and empirical results in econometrics is recognized as being an important aspect of research. An increasing number of researchers in econometrics are making their programs and routines freely available to the econometrics community. As such, OxGauss also provides much added value in that it provides the researcher with a free and rather simple solution to run Gauss programs.

This review is structured as follows. In Section 2, we review OxGauss and give some simple examples as well as a speed comparison between Ox, OxGauss and Gauss 3.5. Section 3 discusses graphics. Section 4 tests the usefulness of OxGauss in replicating the results of a broad number of research papers. Finally, Section 5 concludes.

2 OxGauss

As explained above, the purpose of OxGauss is twofold: calling Gauss programs from Ox, and running Gauss programs without having to install Gauss. The next two subsections illustrate these two features of OxGauss.

2.1 Calling Gauss programs from Ox

The first use of OxGauss is to allow Gauss procedures to be called from Ox. This helps in the transition to Ox, and it increases the amount of code available to Ox users.

To illustrate how Gauss programs can be called from Ox, we consider a small project that mixes both Gauss and Ox programs. The first file, *Gaussprocs.src*, consists of a code file that features the procedure *gengarch(omega,alpha,beta,nu,T_0,T,n)*, which simulates a GARCH model. This procedure has been written by Dick van Dijk (see Franses and van Dijk, 2000) and is downloadable from his web site `http://www.few.eur.nl/few/people/djvandijk/nltsmef/nltsmef.htm`.

To call this procedure from Ox codes, one first has to create a header file. This header file allows the declaration of the functions, constants and external variables so that these are known when required. This is also mandatory to avoid compilation errors in Ox, since functions and global variables have to be explicitly declared before their use. In our example, the header file (*Gaussprocs.h*) consists of the following instructions:

Gaussprocs.h

```
#include <oxstd.h>
namespace gauss
{
    gengarch(const omega,const alpha,const beta,const nu,const T_0,
             const T, const n);
    // Add new procedures here
}
```

Additional procedures can be added in *Gaussprocs.src*, but the header file has to be modified accordingly.[3] It is recommended to use the *.src* extension for the Gauss programs and *.h* for the header files.

In the example *GarchEstim.ox*, we use a Gauss procedure to generate 20,000 observations from a GARCH(1,1) process with Student-t errors. Then, we rely on the Ox package G@RCH 3.0 (see Laurent and Peters, 2002) to estimate a GARCH(1,1) model by Gaussian Quasi-Maximum likelihood. To do this, the Gauss code must be imported into the Ox program, along with the G@RCH package. The **#import** command has been extended so that OxGauss imports are defined by prefixing the file name with `gauss::`.

GarchEstim.ox

```
#include <oxstd.h>
#import <packages/Garch30/garch>
#import "gauss::Gaussprocs"                                        //<---
main()
{
    decl omega=0.2; decl alpha=0.1; decl beta=0.8; decl nu=10;
    decl T_0=1000; decl T=20000; decl n=1;
    decl y=gauss::gengarch(omega,alpha,beta,nu,T_0,T,n);           //<---
    decl garchobj;
    garchobj = new Garch();
    garchobj.Create(1, 1, 1, T, 1);
    garchobj.Append(y, "Y");
    garchobj.Select(Y_VAR, "Y",0,0 );
    garchobj.SetSelSample(-1, 1, -1, 1);
    garchobj.DISTRI(0);             //0 for Normal
    garchobj.GARCH_ORDERS(1,1);     //p order, q order
    garchobj.MODEL(1);              //1: GARCH
    garchobj.Initialization(<>);
    garchobj.DoEstimation();
    garchobj.Output();
    delete garchobj;
}
```

Note that when OxGauss functions or variables are accessed, they must also be prefixed with the identifier `gauss::`.

To run this program from the command line, the user simply has to enter `oxl GarchEstim.ox`. Alternatively, it can be launched from OxEdit. OxEdit 1.62 (or later) is a free but powerful text editor provided with both versions of Ox 3.3 and included in the Windows in-

[3] Arguments declared **const** can be referenced, but cannot be changed inside the function.

stallation program. Like GiveWin, OxEdit features syntax colouring of Ox programs, and context-sensitive help. When OxEdit is used for the first time, the user should execute the `Preferences/Add Predefined Modules` menu and select Ox. Ox and Gauss programs can then be run from the Modules menu without leaving OxEdit. See also the OxEdit web page `http://www.oxedit.com` for more details. Finally, users of Ox Professional can run Ox programs within GiveWin by using the menu `Modules/Start OxRun`.

2.2 Running Gauss programs

The second use of OxGauss is to allow the user to run directly a wide range of Gauss programs under Ox. As an example, we consider the Gauss package *Mixed Logit Estimation Routine for Panel Data* of Kenneth Train, David Revelt and Paul Ruud. The archive file *train0299.zip* (available at `http://elsa.berkeley.edu/Software/abstracts/train0296.html`) contains seven files including the code file `mxlp.g` and the data. This program has been written by Kenneth Train and used by him in a collection of papers (see the web site above for more details) dealing with mixed logit models.[4]

To save space, we do not report the 1396 lines of code of the main file `mxlp.g`. This program can be run on the command line by entering `oxl -g mxlp.g`. Alternatively, it can be launched from OxEdit (`Modules/OxGauss` menu) or within GiveWin by using the menu `Modules/Start OxGauss`. Note that while previous versions of OxGauss required a few modifications of the code,[5] the program is now almost fully compatible with the new version. The only problem is that the program estimates the model by maximum likelihood, allowing the user to choose either the maximization routine domax of Paul Ruud or the commercial package maxlik. Launching the program could lead to the following error message:

```
path...\mxlp.g (1372): maxlik file not found
path...\mxlp.g (1373): maxlik.ext include file not found
```

To solve this problem, one can either comment out lines 421 to 451 relative to the add-on maxlik (and use the domax procedure, i.e. the default option $OPTIM = 1$ in the program) or install the M@ximize package discussed in Section 4. As expected, the results are very similar, if not identical; the only differences are detected after the sixth decimal of the standard errors.

[4]Mixed logit (also called random-parameters logit) generalizes standard logit by allowing the parameter associated with each observed variable (e.g., its coefficient) to vary randomly across units (e.g. individuals or customers).

[5]On his web site (*http://facweb.arch.ohio-state.edu/pviton/support/oxgauss*), Philip Viton mentioned about six changes to the original Gauss code needed to run it without compilation errors with the version of OxGauss provided with Ox 3.2.

2.3 Understanding OxGauss

When an OxGauss program is run, it automatically includes the `\include\oxgauss.ox` file. This by itself imports the required files:[6]

\include\oxgauss.ox

```
#define OX_GAUSS
#import <g2ox>
#import <gauss::oxgauss>
```

These import statements ensure that `g2ox.h` and `oxgauss.h` are being included. Most of the OxGauss run-time system is in `\include\g2ox.ox` while the keywords are largely in `oxgauss.src`.

Most of the programs that link Gauss functions to Ox are gathered in the file `\include\g2ox.ox`. For instance, the output of the Gauss function `cumprodc(x)` is an $N \times K$ matrix with the cumulative products of the columns of the $N \times K$ matrix `x`. The Ox code given below (copied from the file `g2ox.ox`) shows how OxGauss interprets this function.

part of g2ox.ox

```
cumprodc(const mx)
{
    return ::cumprod(mx);
}
```

As indicated in this example, OxGauss does not translate the Gauss code into Ox. Instead, it makes a link between the Gauss function (here `cumprodc`) and its Ox counterpart (`cumprod`). When the corresponding Ox function does not exist, Ox code is written between the brackets which computes what the original Gauss function meant. It is important to note that not all Gauss functions are supported by OxGauss. For instance, there is no equivalent of the Gauss function `intgrat2` (for the computation of double integrals) in Ox 3.3. For this reason, the corresponding procedure in `g2ox.ox` just reports the error message `intgrat2() unsupported` (see below). However, if such a function becomes available in a future version of Ox, mapping `ingrat2` to the corresponding function in Ox will be very easy.

Tables A1 and A2 of the appendix (available from the JAE Data Archive website) give a list of all the Gauss functions supported by OxGauss. To simplify reading this list, we report pre-compiled functions (or directly mapped functions) like `sin` in Table A1 and open source functions (like `cumprodc`; see above) in Table A2. Adding all functions leads to a total of 420 functions recognized by OxGauss. Table A3 in the appendix gives a list of 64 Gauss functions not supported by the current version of Ox. From these tables, it is evident that most of the basic functions are dealt with by OxGauss and that the 15% of Gauss procedures left out are quite specialized.

[6] For ease of presentation, the filename is printed in the upper right corner of the window.

2.4 Speed Comparison

As pointed out by Cribari-Neto (1997), a principal strength of Ox is its speed, although Gauss also runs fast, and its speed performance is not far behind Ox. A recent and detailed comparison of several mathematical programs by Stefan Steinhaus (see `http://www.scientificweb.de/ncrunch/`) shows that Ox is the winner in terms of speed. Since OxGauss just implements a layer on Ox, OxGauss is expected to be comparable to Ox. But one may want to assess the speed loss and how it really compares to Gauss in terms of speed. To answer these two questions, we consider the benchmark tests proposed by Stefan Steinhaus (edition 3). Note that since the functions *intquad2* and *intquad3* (double and triple integration of functions) are not available in Ox 3.3, the corresponding tests have been discarded, which leads to a total of 14 tests. To perform the speed comparison, we first executed the Ox benchmark program `Benchox2.ox` with 5 replications of each test on a 2.6 GHz. Pentium 4 running under Windows XP. We then repeated the tests with the Gauss benchmark program `Benchga2.prg` using both OxGauss and Gauss 3.5. The results are reported in Table 1.

Table 1 Speed Comparison (times in seconds).

Operation	Ox 3.3	OxGauss	Gauss 3.5
Creation, trans. & reshaping of a 1000x1000 matrix:	0.203	0.206	0.197
1000x1000 random matrix to the power 1000:	0.216	0.219	0.216
Sorting of 2,000,000 random values:	1.243	1.475	1.625
FFT over 1,048,576 random values:	1.887	1.950	4.241
Determinant of a 1000x1000 random matrix:	2.225	2.053	2.975
Creation of an 1400x1400 Toeplitz matrix:	0.028	0.028	0.097
Inverse of a 1000x1000 random matrix:	5.365	5.344	7.191
Eigenvalues of a 600x600 random matrix:	7.816	8.522	6.715
Choleski decomposition of a 1000x1000 random matrix:	0.466	0.472	1.063
Creation of 1000x1000 cross-product matrix:	0.784	0.772	4.816
Calculation of 500000 fibonacci numbers:	0.250	0.256	0.209
Gamma function on a 1000x1000 random matrix:	0.156	0.156	0.366
Gaussian error function over a 1000x1000 random matrix:	0.225	0.210	0.444
Linear regression over a 1000x1000 random matrix:	3.387	3.412	4.484
Total of the 14 tests:	24.251	25.075	34.639

Benchmark programs were run (5 replications of each test) on a 2.6 GHz. Pentium 4 with 1 GB RAM running under Windows XP.

Broadly speaking, Table 1 shows that OxGauss compares very favorably to Ox 3.3 in terms of speed. Overall, Ox and OxGauss are a bit faster than Gauss 3.5, which is in line with the previous results of Stefan Steinhaus, who does not consider OxGauss.[7] Note that this benchmark program only

[7] When performing the same speed comparison using Gauss 6.0, Gauss was found to be marginally faster than earlier versions for these operations.

tests one specific aspect of speed, namely, some operations on very large matrices.

3 Graphics support in OxGauss

An important aspect of OxGauss is that it supports most of the graphical features of the Gauss library pgraph. As for the standard functions (see Section 2.3), the file `\oxgauss\src\pgraph.ox` now makes the link between pgraph and the Ox graphical package (oxdraw). For instance, the Gauss function `xy()` is linked to its Ox counterpart `DrawXMatrix()`.

Here is an example of a Gauss program (`grGauss.prg`) that draws a simple graph.

grGauss.prg

```
library pgraph;
    x=seqa(1,1,1000);
    y=rndn(1000,1);
    xlabel("X-axis");
    ylabel("Y-axis: Normal(0,1) draws");
    call xy(x, y);
end;
```

However, with version 3.3 of Ox, only Ox Professional for Windows supports on-screen graphics (through GiveWin). By default, non-Windows versions of Ox and Oxconsole have no support for graphs. Nevertheless, the user can rely on the Ox package GnuDraw developed by Charles Bos that allows the creation of GnuPlot (see `http://www.gnuplot.info`) graphics from Ox. The package is platform independent, free of charge, and downloadable from the author's homepage `http://www.tinbergen.nl/~cbos/`, along with the GnuPlot software.[8] The use of GnuDraw is meant to be simple; see Cribari-Neto and Zarkos (2003) for a comprehensive overview of the GnuDraw package. Interestingly, GnuDraw allows the use of the console version for a quick check of the graphical output of Gauss code. Therefore, academic institutions do not have to license the full professional version of Ox if Gauss programs only need to be replicated.

4 Replicating empirical results using OxGauss

As mentioned in the introduction, one key feature of OxGauss is that it allows one to replicate empirical results obtained using programs written in Gauss. To test OxGauss in a real-life situation, we downloaded from the Internet a huge number of Gauss programs. Here is a list of five web sites that we visited and from which both the data and the Gauss programs can be retrieved:

James Hamilton: `http://weber.ucsd.edu/~jhamilto/`

Bruce Hansen: `http: //www.ssc.wisc.edu/~bhansen/`

Chang-Jin Kim: `http://www.econ.washington.edu/user/cnelson/SSMARKOV.htm`

[8] A detailed help file *gnudraw.html* and a few examples are provided with the package, which makes its use very friendly.

Luc Bauwens: `http://www.core.ucl.ac.be/econometrics/bauwens.htm`
Rolf Tschernig: `http://www.personeel.unimaas.nl/r.tschernig/`

We also used the codes provided by Kim and Nelson (1999) in their book on Markov switching models (Chapters 3 to 11). Table A4 in the appendix gives the list of papers that we replicated (i.e. 27 references). Most of these papers rely quite heavily on nonlinear optimization techniques and thus require one of the optimizers from the Cml, Maxlik, or Optmum modules (see Section 2.2) of Gauss.

For almost all standard functions and procedures of Gauss, the results are easily replicated using OxGauss. However, running Gauss programs using the current version of OxGauss is problematic when the Gauss program uses a Gauss application module such as the three optimizers, or the Time Series, Linear Programming, CurveFit or Nonlinear Equations modules. To alleviate this problem, two solutions are at hand. The first solution is to import the corresponding Gauss libraries. This however requires the user to have a valid registered version of the application module, and hence the use of OxGauss is then of limited interest. The second solution is to create a new Ox package that links the names of the Gauss functions, global variables and procedures to the corresponding Ox functions. This is possible because OxGauss is easily extensible (as shown in the previous section).

To test the second solution, we wrote a set of procedures put in a package called M@ximize 1.0 that supports most of the important options of Cml, Maxlik, and Optmum. Note that the package *does not translate* the Gauss optimizers into Ox, nor does it clone the optimizers. Like pgraph, it makes the link between Gauss and Ox commands. The package is open source and freely available on the web at the following address: `http://www.core.ucl.ac.be/~laurent`.

We stress that we do not want to review M@ximize in this paper. The main goal of this section is to test the reliability of OxGauss.

Once the M@ximize package is installed by unzipping the file *m@ximize.zip* in the main directory of Ox, most of the codes can be run in their present form. However some marginal changes in the Gauss programs are sometimes needed. The most frequently encountered problems are:

- *Converting data files*. For instance, running the Gauss code related to reference 22 in Table A4 gives the "Invalid .FMT or .DAT file" error message. This problem occurs because old style Gauss data sets (v89 *.dht/.dat*) must be converted to the new Gauss format (v96 *.dat*). The program implement this conversion is `ox\lib\dht2dat`. The conversion can be run from the command line as:

  ```
  oxl lib/dht2dat old_datafile.dht new_datafile.dat
  ```

  Alternatively, the data files can be converted to the new format through GiveWin by first loading the *.dht* file and then saving the file into the new format.
- *Absence of extension*. To launch a Gauss code using OxEdit, the file needs an extension. It is common to use the extension *.src*.

- *Interactive mode*. Examples 1, 6, 9 and 13 use the Gauss function `cons` that requests an input from the keyboard (console) and puts it into a string. The typical use of this function is to generate a message like "Do you wish to continue (y or n)?" and accordingly branch in one of two directions. In other words, the program enters into interactive mode. In such a case, the program has to be launched using "Ox interactive", i.e. Oxli.exe under Windows instead of Oxl.exe (this mode is not supported by the console version).[9]

To illustrate, we consider the Gauss package written by Rolf Tschernig for Yang and Tschernig (1999). We focus on the example file *multband.tes* provided by the author, which estimates the asymptotic optimal vector bandwidth for simulated bivariate nonlinear regression models. This file is made up of about 190 lines of Gauss code and includes three libraries, namely, Optmum, pgraph and multband (a library provided by the author), as well as a set of three dll files. To use the package under Gauss, we first install the library multband by copying the file *multband.lcg* into the subdirectory `.\lib` of Gauss and the files *multband.src* (about 2900 lines of code) and *multband.dec* (declarations of global variables) into the subdirectory `.\src`. Finally, we copy the three dll files *locling.dll, density.dll* and *loccubg.dll* into the subdirectory `.\dlib`. Importantly, to use the package under OxGauss, one has to follow the same instructions and copy the files into the existing subdirectories `.\OxGauss\lib`, `.\OxGauss\src` and `.\OxGauss\dlib`.

Once this installation procedure has been completed, the example file *multband.tes* can be run.[10] Again, the results are identical up to the sixth decimal. The outputs obtained with OxGauss and Gauss 3.2 are reported in Table A5 of the appendix available from the JAE Data Archive.

5 Conclusion

OxGauss is an application that allows a user to run a wide range of Gauss programs under Ox without the need to have Gauss installed on his/her computer. OxGauss is potentially useful both for Gauss and Ox users. On the one hand, Gauss programs can efficiently be called under Ox. Thus Ox programmers willing to use existing Gauss procedures do not have to translate the procedures into Ox but can call the Gauss code directly under Ox. On the other hand, OxGauss can be used to run Gauss programs under Ox and hence to replicate the results of a paper for which Gauss code is made available by the author(s).

The effectiveness of OxGauss was illustrated by running a large number of Gauss programs that are freely available on the Internet and that use Gauss application modules requiring numerical op-

[9] When using OxEdit to run the Gauss code, an additional shortcut has to be created. The simple solution is to click on the menu VIEW/PREFERENCES/ADD/REMOVE MODULES. Then clone the OxGauss shortcut and in the Command line change Oxl.exe to Oxli.exe.

[10] Note that this example file simulates a sequence of 250 observations. To allow the comparison between Gauss and OxGauss, we changed the original code so that it now always uses the same random numbers.

timization (26 papers published in international journals and one book; see Table A4). In all cases the programs were found to be fully compatible with OxGauss in the senses that either no change or very minor changes were required to the original code and that the results were almost identical. Furthermore, as shown in the preceding section, one of the strengths of OxGauss is its transparency and extensibility. Therefore, even though some functions and Gauss application modules are currently not available in OxGauss, the packages GnuDraw and M@ximize are examples of possible solutions to further bridge the gap between Ox and Gauss.

To conclude, we believe that OxGauss goes a long way towards bridging the gap between Gauss and Ox user communities. We hope that Gauss users who already share their Gauss programs would start testing their compatibility with OxGauss. If it is required, they could make minor changes to their programs to ensure compatibility and indicate that their code is "OxGauss compliant".

6 Acknowledgment

While remaining responsible for any errors in this paper, the authors would like to thank James G. MacKinnon, the Software Review Editor, Pierre Giot, and Jean-Philippe Peters for useful comments, and Charles Bos and Jurgen Doornik for their accessibility and the numerous remarks they made on a previous version of the paper.

References

CRIBARI-NETO, F. (1997): "Econometric Programming Environments: Gauss, Ox and S-Plus," *Journal of Applied Econometrics*, 12, 77–89.

CRIBARI-NETO, F., and S. ZARKOS (2003): "Econometric and Statistical Computing Using Ox," *Computational Economics*, 21, 277–295.

FRANSES, P., and D. VAN DIJK (2000): *Non-Linear Series Models in Empirical Finance*. Cambridge University Press.

KIM, C.-J., and C. NELSON (1999): *State-Space Models with Regime Switching: Classical and Gibbs-Sampling Approaches with Applications*. The MIT Press.

LAURENT, S., and J.-P. PETERS (2002): "G@RCH 2.2 : An Ox Package for Estimating and Forecasting Various ARCH Models," *Journal of Economic Surveys*, 16, 447–485.