6/16/06
05:31

Teaching Computational Economics to Graduate Students

by
David A. Kendrick

Abstract:

The teaching of computational economics to graduate students has mostly been in a single course with a focus on algorithms and computer code. The shortcoming of this approach is that it neglects one of the most important aspects of computational economics - namely model development skills. These skills are the ability to conceptualize the science, engineering and economics of a problem and to convert that understanding first to a mathematical model and then to a computational representation in a software system. Thus I recommend that a two course sequence in computational economics be created for graduate students with the first course focusing on model development skills and the second course on algorithms and the speed and accuracy of computer codes. I believe that a model development course is most helpful to graduate students when it introduces the students to a wide variety of computational models created by past generations and ask them to first make small modification in order to better understand the models, the mathematics and the software. This, in turn, is followed by encouraging them to make more substantial modifications of the student's own choosing so as to move the models in directions that permit the students to address current economic problems. I think that the key element of this process is it enhancement of the *creative* abilities of our students.

1. Preface

The use of computational models in economics is a two-step procedure. The first step is the development of the model and the second is its solution. The first step involves the creation of a verbal understanding of the key elements of an economic problem, the translation of that understanding into mathematics and the specification of the mathematics in a computer software system. The second step involves the solution of the model with an appropriate algorithm in a fashion that is both fast and accurate.

Most of the emphasis on teaching computational economics in the past has been on the second (algorithmic) step. I argue here that this focus now needs to be broadened to include the first step of model development along with the second step of model solution.

In the last few decades remarkable progress has been made in the hardware and software that is used to solve economic models. During this era many economics graduate students have been taught how to create the mathematical algorithms to solve models and the computer coding methods to do this both efficiently and accurately. However, we have been slower to help our students develop skills as model developers. How can we improve on this?

I argue here that we need to begin thinking of a two-course sequence for teaching computational economics to graduate students. The first course would be designed to attract the broad set of students who are interested in economics but not necessarily in computational methods. The second course would be for those students who enjoy working with computers and with the algorithms that are used to solve models on those computers.

My focus in this paper is on methods of teaching model development skills to graduate students – thus it is on the first of these two courses. Some materials that are appropriate to this first step are in my book with Ruben Mercado and Hans Amman, i.e. Kendrick, Mercado and Amman (2006). For examples of books that emphasize the second step the reader is referred Judd (1998) and Miranda and Fackler (2002).

I begin in the second section of this paper with a discussion of methods of helping students to enhance their model development skills. One element of this is exposing the

students to models developed by previous generations in many areas of economics. A discussion of these areas is in the third section of the paper. The timing and rhythm of the course is described in the fourth section and the choice of software systems is elaborated in the fifth section.

2. Teaching Model Development Skills

In my experience it is, at first, not easy for many students to size-up economic problems in verbal terms, translate that understanding into mathematics, and then specify the mathematical model in a software system. However, these skills can be developed by "standing on the shoulders" of previous generations. Thus my approach is to expose the students to a wide variety of economic problems in order to see how their predecessors created the mathematical models and then the computational models to analyze these problems. In each area the students are encouraged to develop their own models on the foundations of the existing models.

At the start this is nothing more than solving the existing model with some changes in the parameters. For example, in a simple one-sector optimal growth model the students might solve the model several times with different discount rates to see how the optimal consumption path changes. Then the student might be encouraged to begin altering the specification of the model. For example, if the original model included only capital in the production function the student might add labor. This involves altering both the mathematical model and the specification of the model in the software. For most students this kind of step is easy enough to be accomplished fairly quickly and yet difficult enough to inspire substantial confidence that they can begin to shape the model to their own purposes.

A third step might involve the creation of a second sector in the growth model so that the model is converted from a single sector to a multi-sector model. This is considerably more difficult and might become the subject for a mid-term paper rather than for a weekly exercise in the course. However, the student who progresses to this level will be adding both to his mathematical skills and to his programming skills.

The key element here is that the students should began to realize that they can be *creative*

in developing economic models, i.e. that they can take a model developed by the previous generation of economists and convert it in a series of steps to a model that serves their own purpose and allows them to focus on the problems of interest to their own cohort.

Alex Meeraus, the developer of the GAMS software system, use to say that economists need to be more like architects. In the old days when an architect began a new project he did not start with a blank sheet of paper. Rather he went to the file cabinet and pulled out the blue prints from some previous buildings of the same type. Then elements from a number of buildings were used to create drawings of an entirely new structure. Modern architects would use computer files to accomplish the same purpose.

For economists the old blue prints are the models developed by their predecessors. One picks and chooses elements from the mathematical models and their representations in software systems to develop new models that address current problems. In my experience this kind of system of model development works well for most economics graduate students. It is very difficult for most students to sit down with a blank piece of paper and develop their own model from scratch. However, students are adept at tracking down previous models in their areas of interest, first solving these models with different parameter values, then solving them again and again with changes in the functional specifications until a solid understanding is gained of the strengths and weaknesses of the model. Then elements from one model are added to elements from a second model and mixed with some entirely new elements to create a model that focuses on the problem of current interest for the student.

Against this background a first course in computational economics should expose the students to a variety of models from many different areas of economics. The next section discusses a number of areas that I have found useful.

3. Shoulders to Stand On

The development of computational economics at this point in time in somewhat akin to the development of econometrics fifty years ago. At that time econometric methods were being applied to only a few areas in economics such as the estimation of consumption and investment functions in small macroeconomic models and of demand functions in microeconomic models. Over time the use of econometric tools spread

across broad fronts in economics.

Computational economics is now at an intermediate stage and is being applied to tens of areas; however, this coverage in spotty with established strength in some areas and just at the beginning of work in others. This first course in computational economics reflects this pattern and so I discuss below a number of areas that may be useful to graduate students. I begin with an outline followed by a discussion of parts of the outline.

- microeconomics
  - industrial models
  - game theory
  - computable general equilibrium models
  - agent-based models
- macroeconomics
  - growth
  - fluctuations
  - microeconomic foundations of macroeconomics
- finance
  - personal financial planning
  - portfolio models
- estimation
  - neural nets
  - database systems
- environmental economics
  - sectoral models
  - global warming
- international trade
  - sectoral models
  - computable general equilibrium models

In the outline above I have not tried to partition the field with each topic in one and only one box. Rather I have listed a single topic under two or more headings if this facilitates understanding.

3.1 Microeconomics

One of the earliest areas to develop in computational economics was the use of linear programming models to solve transportation problems. Problems of this type arose, for example, during the Second World War with shipments of supplies to the European and Pacific fronts from a number of ports on the east and west coasts of the U.S. The problem was to minimize the resources used in shipping the supplies in this network. Early contributors to this line of development in economics were Koopmans (1947) and Dorfman, Samuelson and Solow (1958). A recent manifestation of this line of development was a project at the World Bank which ran over many years and developed models focused on a number of different industries including steel, aluminum, copper, fertilizer, cement and pulp and paper. These models were national, regional or world wide in scope and analyzed when and where to construct new plants in a system characterized by strong economies of scale. Many of these models were solved with mixed integer programming methods. For examples of this type of work see Choksi, Meeraus and Stoutjesdijk (1980) and Kendrick, Meeraus and Alatorre (1984).

Most work in game theory in economics is analytical rather than numerical – still there are important areas where computational economics is already beginning to be used. The basic game theory models such as Cournot and Stackelberg models can be solved both analytically and numerically using the Mathematica software. Also, evolutionary games can be effectively solved using genetic algorithm methods coded in MATLAB. For examples of both of these developments see Kendrick, Mercado and Amman (2006).

Computable general equilibrium (CGE) models have a long history of development in computational economics. This work began with static models and in recent years has moved to dynamic models, viz. Kim (2002). These models have been applied to national and regional economies. For an example of a national model see the work on Australia by Peter Dixon and his colleagues, viz Dixon, Parmenter, Sutton and Vincent (1982). One example of multi-country regional models is the analysis of the North American

Free Trade Area (NAFTA) that was done with CGE models that included sub-models for Canada, the U.S. and Mexico. Also, much work on environmental modeling has been done with CGE models, viz. McKibbin and Wilcoxen (2002).

One of the newest types of computational microeconomic models is agent based models. This work started with the famous "sugarscape" models of Epstein and Axtell (1996) and has now grown into a virtual industry, viz Judd and Tesfatsion (2005). In these models computer simulations are used to analyze interactions between economic agents in space and time. The object-oriented capabilities of the C++ language lend themselves well to this class of models; however in recent years much work has been done in this field using MATLAB.

3.2 Macroeconomics

Though early research on growth theory was done with analytical models focused on steady state solutions, increasingly in recent years this work has shifted to numerical models in which one can analyze the entire path of the development of the economy over time. Also, the early models tended to have only one or two sectors while numerical methods have opened the door for analysis of models containing many sectors. The simplest Ramsey (1928) model lends itself nicely to solution even with so simple a tool as the optimization capability called the "Solver" that lies hidden in the Excel spreadsheet software. This model is easy to construct and solve and yet provides a good laboratory for graduate students to understand the basics of growth theory in a single sector model before moving on to the more advanced multisectoral models.

Growth models provide a good example of the two-step process for learning computational economics that was discussed above in the preface. A graduate student can develop a growth model in Excel (or even in GAMS) with considerable ease. The model can include one or more sectors, labor and capital inputs and even exports and imports. Human capital accumulation may be added and endogenous technical change included while the student experiments with the effects of various specifications on the results that are obtained from the model. All of this can be done without knowing the details of the solution procedures that are use by the Excel Solver or by the nonlinear programming codes that can be used with GAMS to solve such models. However, students who take a deeper interest in computational economics will want to continue to a

second course in the subject where they learn about nonlinear programming algorithms and the coding methods that can be used to make these algorithms both fast and accurate.

Models of economic fluctuations were among the first applications of computational economics dating from the work of Tinbergen (1950) and Klein (1947). Modern manifestations of this work are in textbooks, viz Hall and Taylor (1998) and these models can be solved with ease using software such as GAMS (see Mercado, Kendrick and Amman (1998)). More advanced treatment of these topics has moved in recent years into the use of dynamic stochastic models that are solved with dynamic programming or control theory methods as is discussed in Part III of Kendrick, Mercado and Amman (2006).

In recent years there has been much work on the microeconomic foundations of macroeconomics. Some of these models are dynamic programming models viz. Corbae (1993) and Adda and Cooper (2003). Also, there is much work on heterogeneous agent models starting from the seminal paper of Krusell and Smith (1998).

3.3 Finance

The simplest problem in finance – and one that can be highly motivating for graduate students – is how to finance a graduate education without emerging from graduate school overburdened with debt. Such a dynamic model can be constructed with a few state equations for stocks, bonds, checking account, student loans and credit card debt and readily solved.

A more advanced set of models in finance has to do with portfolio allocation in the tradition of Markowitz (1952). These models are specified as quadratic programming problems. The models offer another example of the kind of creativity we would like to encourage in graduate students. A student working with the portfolio allocation model might be solving it with a gradient method by calling the *fmincon* function in MATLAB. However, he or she might then realize that the original model does not include brokerage fees and adds these to the model. Then the criterion function is to maximize the return less a weighted variance term less brokerage fees. After some experimentation with this model the student decides to add an additional degree of realism to the model by making some of the brokerage fees fixed amounts per transaction rather than having all the fees

be proportional to the number of shares traded. For a time the student continues solving the model with the gradient function; however, he or she soon realizes that the revised model may have a non-convex criterion function and thus the problem may have local optima. Then the student switches from gradient methods to genetic algorithms in MATLAB in order to be able to do global optimization in problems that have local optima.

In a subsequent semester of study of computational economics the student might dig deeper into the algorithms and code for both gradient methods and for global optimization methods. Having begun with a finance problem that was of substantial personal interest and having made modifications to the model to improve its realism, the student then enters the second semester algorithms and code portion of his or her training highly motivated to master these methods. Moreover, the student may then move deeply enough into genetic algorithms to begin to make contributions to methodology of algorithms and codes used for this purpose.

3.4 Estimation

While most work in estimation properly belongs to the field of econometrics and not to the field of computational economics, one approach to estimation is commonly discussed in computational economics courses. This is neural nets – a field that got its start in computer science and migrated to economics and finance. For a clear discussion of this topic see Sargent (1993). I have found that neural net methods can be readily constructed and solved in Excel using the Solver add-on to do the optimization. The application of these methods to predicting stock market prices provides strong motivation for many graduate students and permits them to use readily available data.

Though it is not really an estimation method, I have grouped database systems under this heading because these systems are frequently used in support of econometric applications. However, database systems are also often used to answer queries about data that do not necessarily involve estimation. Relational database methods (viz Date (1977)) are widely used in business and industry – so much so that I have discovered that many students have encountered them in summer jobs and are motivated to develop creative applications.

3.5 Environmental Economics

Many environmental economic problems are local in geography and grow out of a single sector. Examples are the models of shrimp aquaculture in Thailand or of livestock management in the Kalahari as discussed in Duraiappah (2003). Others are global in scope such as the problem of global warming, viz Nordhaus (1992). However, both local and global models share the property that they interest graduate students who are concerned with environmental degradation. They also have the characteristic that they are effectively represented in software with a set-driven modeling language like GAMS. These models at first seem complex to students; however they discover that they can quickly come abreast of previous developments and can begin to improve on the existing models. Then they slowly develop the capability to create entirely new models.

3.6 International Trade

Computational models of international trade fall into three large classes. One group includes models of a single sector that may be applied to a multi-country region such as the model of the fertilizer industry in Andean Pac area in South America, viz. Mennes and Stoutjesdijk (1985). Alternatively such models may be developed for the entire world, viz the model of the aluminum industry created at the World Bank, viz. Brown, Dammert, Meeraus and Stoutjesdijk (1983).

A second group looks at multiple sectors but a single country. A model in this tradition is the one created by Dixon and his colleagues, that was mentioned above, and that was used to study the effects of reductions in trade barriers in Australia. This model was disaggregated to include large numbers of sectors, viz. Dixon, Parmenter, Sutton and Vincent (1982). Models like this have been developed not only in Australia but also in other places with the GEMPACK software that was created by the Australian group.

The third group is multisectoral models that have been used for the analysis of free trade areas such as the North American Free Trade Area (NAFTA). These NAFTA models included three computable general equilibrium sub models for Canada, the U.S. and Mexico.

As one can see from the lists above, there are many more types of models than can be covered in the fourteen weeks of the normal academic semester. I have found it useful to change the coverage from one semester to the next as the interest of the students changes and as some subfields within computational economics grow more rapidly than others. In a normal semester I will use about thirteen of the topics above - roughly one per week - as is discussed in the following section on the rhythm of the course.

4. Rhythm

The course has two rhythms – one being the rhythm within each week and the second being the rhythm over the entire semester. Consider first the rhythm within each week.

I have found it useful to divide each week's instruction into four segments. The first segment is an introduction to the science, technology and economics of the subject. The second segment is on the mathematical models and the third is about the specification of that mathematics in a software system. The last segment is a session in a computer lab where the students began work on the problem but have help available to get through the first difficulties incumbent on using new software or a new type of model.

For example, consider a week's instruction using the global warming models of Nordhaus (1992). The first class would be about the science of global warming from carbon emissions, to CO_2 concentrations in the atmosphere, to changes in world temperature levels. It would also cover carbon taxes and their effects on this process. The second class would discuss the ten or so dynamic nonlinear simultaneous equation system that embodies the science and economics of the subject. The third segment would show how those equations can be represented in the GAMS software. The four segments would turn the students loose in the computer lab to first reproduce the base line solutions and then to began modifying the model.

The students would then be given a week to make small or more substantial changes of their own choosing and to write a few pages about their experiments with the model. In my experience it is impressive how quickly graduate student can effectively move into a new subject area using mathematical methods that they might not already be familiar with and employing a software package that is new to them … and to do all this in one

week while modifying an existing model to nudge it in a direction that interest them.

By mid-semester the students are encourage to develop one of their short weekly papers into a larger research project in which they make more substantial changes in one of the weekly models. Thus after only a few weeks into the course they are being challenged to begin moving toward the creation of their own models. Many students enter this phase of the course with some trepidation but find that they can do much more than they first imagine.

A term paper ends the course and here the students are expected to make substantial modifications to an existing model or even to begin developing entirely new models of their own. My experience is that developing an entirely new model from scratch is a challenge – even for the most able graduate students – so I encourage my students to build on the existing models and even to take ideas from a variety of different models rather than to start from scratch. However, there are exceptions …

5. Which Software Systems to Use?

One's first thought when developing a graduate course in computational economics is what language (or software system) to use. Should it be Mathematica or MATLAB or GAUSS or GAMS or even Excel? For example Varian (1996) uses Mathematica, Lin (2001) uses GAUSS, Miranda and Fackler (2002) and Adda and Cooper use MATLAB, and Thompson and Thore (1992) use GAMS.

However, my advice is not to use a single software system, but rather to expose the students to a variety of systems with each type of model being specified in the language that facilitates efficient development and solution of that particular type of model. For example, game theory models are naturally specified in Mathematica where one can mix analytical and numerical methods. Dynamic macro models lend themselves well to solution with MATLAB where one can draw on the natural power of that software system with matrix operations in a dynamic context. Computable general equilibrium models are easily specified in the GAMS language with its set-driven capability, etc.

This quickly raises the concern that students cannot possibly be expected to master a number of different software systems within a single semester. However, my experience

is that I can, in a single semester, give the students an introduction to a variety of software systems so that they become acquainted with the capabilities of different systems. Moreover, by the time they finish a term paper using one of the systems on a model in their own particular area of interest, they will began to obtain a substantial degree of competence in that language - a competence that can be built on in a second course in computational economics that focuses more on algorithms and code and less on model development.

6. Conclusions

Computational economics has mostly been taught to graduate students in a single course with a focus on algorithms and on the speed and accuracy of computer codes. I believe that this leaves out one of the most important aspects of the subject, namely teaching the students model development skills. Thus I recommend that a two course sequence in computational economics be established for graduate students with the first course being on model development and the second course on algorithms and code.

For us, one of the most important elements of teaching computational economics to graduate students is to unleash their *creative* capabilities. Time and again I have seen the motivation and indeed, pleasure, that comes to graduate students when they realize that they can take charge of a problem area in economics and can create models that address problems that are important to their own times.

References

Adda, J. and R. Cooper (2003), *Dynamics Economics: Quantitative Methods and Applications*, The MIT Press, Cambridge, MA.

Brown, Martin, Alfredo Dammert, Alexander Meeraus and Ardy Stoutjesdijk (1983), "World Investment Analysis: The Case of Aluminum", World Bank Staff Working Papers, Number 603, The World Bank, Washington, D.C.

Choksi, Armeane M., Alexander Meeraus and Ardy J. Stoutjesdijk (1980), *The Planning of Investment Programs in the Fertilizer Industry*, The Johns Hopkins University Press, Baltimore.

Corbae, Dean (1993), "Relaxing the Cash-in-Advance Constraint at a Fixed Cost: Are Simple Trigger-Target Portfolio Rules Optimal?", *Journal of Economic Dynamics and Control*, **17**, 51-64.

Date, C. J. (1977), *An Introduction to Database Systems, 2nd ed.*, Addison-Wesley, Reading, MA.

Dixon, Peter B., B. R. Parmenter, John Sutton and D. P. Vincent (1982), *ORANI, A Multisectoral Model of the Australian Economy*, North Holland, Amsterdam.

Dorfman, Robert, Paul Samuelson and Robert Solow (1958), *Linear Programming and Economic Analysis*, McGraw-Hill Book Company, New York.

Duraiappah, Anantha Kumar (2003), *Computational Models in the Economics of Environment and Development*, Kluwer Academic Publishers, Dordrecht, The Netherlands.

Epstein, Joshua M. and Robert Axtell (1996), *Growing Artificial Societies: Social Science from the Bottom Up*, The MIT Press, Cambridge, Massachusetts.

Hall, Robert E. and John B. Taylor (1997), *Macroeconomics*, 5th edition, W. W. Norton & Company, New York.

Judd, Kenneth L. (1998), *Numerical Methods in Economics*, The MIT Press, Cambridge, MA.

Judd, Kenneth L. and Leigh Tesfatsion (2005), *Handbook of Computational Economics II: Agent-Based Computational Economics*, Handbooks in Economics Series, North Holland, Amsterdam (forthcoming).

Kendrick, David A., Alexander Meeraus and Jaime Alatorre (1984), *The Planning of Investment Programs in the Steel Industry*, The Johns Hopkins University Press, Baltimore.

Kendrick, David A., P. Ruben Mercado and Hans M. Amman (2006), *Computational Economics*, Princeton University Press, Princeton, New Jersey.

Kim, Seung Rae (2004), "Uncertainty, Political Preferences, and Stabilization: Stochastic Control Using Dynamic CGE Models", *Computational Economics*, **24**, 97-116.

Klein, Lawrence R. (1947), *The Keynesian Revolution*, Macmillan Co., London.

Koopmans, Tjalling (1947), "Optimum Utilization of the Transportation System" in Proceedings of the International Statistical Conferences, 1947, Washington, D.C. Vol. 5 (Vol. 5 reprinted as a supplement in 1949 to *Econometrica*, 17).

Krusell, Per and Anthony A. Smith, Jr. (1998), "Income and Wealth Heterogeneity in the Macroeconomy," *Journal of Political Economy*, **106**, 867-896.

Lin, Kuan Pin (2001), *Computational Econometrics: GAUSS Programming for Econometricians and Financial Analysts*, ETEXT Publishing, available at http://www.etext.net

Markowitz, Harry (1952), "Portfolio Selection", *The Journal of Finance*, **7**, 77-91.

McKibbin, Warwick J and Peter J. Wilcoxen, (2002), *Climate Change Policy After Kyoto: Blueprint for a Realistic Approach*, Brookings Institution, Washington, D.C.

Mennes, L. B. M. and Ardy J. Stoutjesdijk (1985), *Multicountry Investment Analysis*, The Johns Hopkins University Press, Baltimore, MD.

Mercado, P. Ruben, David A. Kendrick and Hans Amman (1998), "Teaching Macroeconomics with GAMS," *Computational Economics*, **12**, 125-149.

Miranda, Mario J. and Paul L. Fackler (2002), *Applied Computational Economics and Finance*, The MIT Press, Cambridge, MA.

Nordhaus, William D. (1992), "An Optimal Transition Path for Controlling Greenhouse Gases", *Science*, **258**, 1315-1319.

Ramsey, F. (1928), "A Mathematical Theory of Savings", *Economic Journal*, reprinted in K. J. Arrow and T. Scitovsky, eds. (1969), *Readings on Welfare Economics*, Richard D. Irwin, Homewood, IL.

Sargent, Thomas J. (1993), *Bounded Rationality in Macroeconomics*, Oxford University Press, Oxford, United Kingdom.

Thompson, Gerald L and Sten Thore (1992), *Computational Economics*, The Scientific Press, South San Francisco, CA.

Tinbergen, Jan (1950), *The Dynamics of Business Cycles: A Study in Economic Fluctuations*, University of Chicago Press, Chicago.

Varian, Hal R. (1996), *Computational Economics and Finance: Modeling and Analysis with Mathematica*, Springer-Verlag.