SOLVING NONLINEAR DYNAMIC STOCHASTIC MODELS: AN ALGORITHM COMPUTING VALUE FUNCTION BY SIMULATIONS*

Lilia Maliar and Serguei Maliar**

WP-AD 2004-37

Corresponding author: Lilia Maliar: Departamento de Fundamentos del Análisis Económico, Universidad de Alicante, Campus San Vicente del Raspeig, Ap. Correos 99, 03080 Alicante, Spain.
E-mail: maliarl@merlin.fae.ua.es

Editor: Instituto Valenciano de Investigaciones Económicas, S.A.

Primera Edición Octubre 2004.

Depósito Legal: V-4395-2004

* This research was partially supported by the Instituto Valenciano de Investigaciones Económicas and the Ministerio de Ciencia y Tecnología, BEC 2001-0535.
** L&S Maliar: Departamento de Fundamentos del Análisis Económico, Universidad de Alicante, Campus San Vicente del Raspeig, Ap. Correos 99, 03080 Alicante, Spain. E-mails: maliarl@merlin.fae.ua.es, maliars@merlin.fae.ua.es

SOLVING NONLINEAR DYNAMIC STOCHASTIC MODELS: AN ALGORITHM COMPUTING VALUE FUNCTION BY SIMULATIONS

Lilia Maliar and Serguei Maliar

ABSTRACT

This paper presents an algorithm for solving nonlinear dynamic stochastic models that computes value function by simulations. We argue that the proposed algorithm can be a useful alternative to the existing methods in some applications.

JEL Classification: C6; C63; C68

Key Words: Nonlinear stochastic models; Value function; Parameterized expectations; Monte Carlo simulations; Numerical solutions

1 Introduction

The study of dynamic economies often requires finding solutions to stochastic infinite-horizon optimization problems with continuous state and action spaces. The existing computational methods, typically, either solve for a value function satisfying the Bellman equation or compute decision rules satisfying first-order conditions (Euler equations).[1] In the paper, we develop a simple algorithm, which combines both approaches. It parametrizes value function, simulates time series satisfying first-order conditions and uses the resulting series to minimize the difference between the two sides of the Bellman equation. The algorithm is similar to Marcet's (1988) version of the Parameterized Expectations Algorithm (PEA) in that it uses Monte Carlo simulations for evaluating the conditional expectations. We argue that the algorithm proposed can be a useful alternative to the existing methods in some applications.

2 The problem

We focus on the class of stochastic infinite-horizon optimization problems in which both state and control variables can take a continuum of possible

[1] Rust (1996) and Marimon and Scott (1999) provide reviews of numerical methods used in economic dynamics.

values. We assume that the problem has a recursive formulation, so that its solution satisfies the Bellman equation:

$$V(z_t, u_t) = \max_{x_t} \{r(z_t, u_t, x_t) + \delta E_t [V(z_{t+1}, u_{t+1})]\} \tag{1}$$

$$\text{s.t.} \quad z_{t+1} = g(z_t, u_t, x_t), \tag{2}$$

$$\Pi(u_{t+1} = u' \mid u_t = u) \text{ for all } u', u \in U \subseteq R^{n_u}, \tag{3}$$

where (z_0, u_0) is given; $E_t[\cdot] \equiv E[\cdot \mid u_t]$ denotes the conditional expectation; $\delta \in (0,1)$ is the discount factor; z_t, u_t and x_t are vectors of n_z endogenous state variables, n_u exogenous state variables and n_x control variables, respectively; V is the value function; r is the return function; g is the transition equation for a vector of endogenous state variables; and finally, Π is the transitional probability function, associated with a first-order Markov process for the vector of exogenous state variables. We assume that r is concave and that g is such that the set $\{(z_{t+1}, z_t) : z_{t+1} = g(z_t, u_t, x_t), x_t \in R^{n_x}, u_t \in R^{n_u}\}$ is convex and compact.

We assume that a solution to the problem $(1)-(3)$ exists, and also that it is interior and unique. As such, an optimal allocation satisfies the first-order condition

$$\frac{\partial r(z_t, u_t, x_t)}{\partial x_t} + \frac{\partial g(z_t, u_t, x_t)}{\partial x_t} \cdot \delta E_t \left[\frac{\partial V(z_{t+1}, u_{t+1})}{\partial z_{t+1}}\right] = 0. \tag{4}$$

By the envelope theorem, we have

$$\frac{\partial V\left(z_t,u_t\right)}{\partial z_t}=\frac{\partial r\left(z_t,u_t,x_t\right)}{\partial z_t}+\frac{\partial g\left(z_t,u_t,x_t\right)}{\partial z_t}\cdot\delta E_t\left[\frac{\partial V\left(z_{t+1},u_{t+1}\right)}{\partial z_{t+1}}\right]. \tag{5}$$

Substituting $E_t\left[\frac{\partial V(z_{t+1},u_{t+1})}{\partial z_{t+1}}\right]$ from (4) to (5), updating the resulting condition and combining it with (4) yields

$$\frac{\partial r\left(z_t,u_t,x_t\right)/\partial x_t}{\partial g\left(z_t,u_t,x_t\right)/\partial x_t}=$$
$$\delta E_t\left[\frac{\partial r\left(z_{t+1},u_{t+1},x_{t+1}\right)}{\partial z_{t+1}}+\frac{\partial g\left(z_{t+1},u_{t+1},x_{t+1}\right)}{\partial z_{t+1}}\cdot\frac{\partial r\left(z_{t+1},u_{t+1},x_{t+1}\right)/\partial x_{t+1}}{\partial g\left(z_{t+1},u_{t+1},x_{t+1}\right)/\partial x_{t+1}}\right]. \tag{6}$$

Condition (6) is the so-called Euler equation.

There are two general approaches to solving the problem $(1)-(3)$. One is the value-iterative approach in which the optimal value function is computed with the Bellman equation (1). The other is the Euler equation approach, in which the optimal decision rules are calculated from the Euler equation (6) without computing the value function. The method we propose here combines both approaches. Specifically, it searches for the optimal decision rules satisfying first-order condition (4) and uses the Bellman equation (1) as a criterion for the accuracy of the solution. The formal description of the method is provided in the following section.

3 The algorithm

We approximate the true value function $V(z,u)$ by a parametric function $W(z,u;\beta)$, $\beta \in R^v$. Our objective is to find a vector of coefficients β^* such that $W(z,u;\beta^*)$ is the best approximation to $V(z,u)$ given the functional form chosen, i.e.

$$\beta^* = \arg\min_{\beta \in R^v} \| W(z,u;\beta) - V(z,u) \| .$$

We solve for β^* by using Monte Carlo simulations.

- *Step* 1. For an initial iteration $i = 0$, fix $\beta = \beta(0) \in R^v$. Fix initial conditions z_0 and u_0; draw and fix for all simulations a random series $\{u_t\}_{t=1}^T$ by using (3). Replace $\frac{\partial V(z_{t+1},u_{t+1})}{\partial z_{t+1}}$ in (4) by the approximation $\frac{\partial W(z_{t+1},u_{t+1};\beta)}{\partial z_{t+1}}$ and solve (2) and (4) with respect to z_{t+1} and x_t. We assume that a solution to (2), (4) exists and that it is unique.

- *Step* 2. Given $\beta \in R^v$, recursively calculate $\{z_{t+1}(\beta), u_{t+1}, x_t(\beta)\}_{t=1}^T$.

- *Step* 3. Construct the variable $\{w_t(\beta)\}_{t=1}^T$ such that

$$w_t \equiv r(z_t(\beta), u_t, x_t(\beta)) + \delta E [W(z_{t+1}(\beta), u_{t+1}; \beta) \mid u_t]$$

  and run a nonlinear least-square regression of this variable on explanatory function $W(z_t(\beta), u_t; \xi)$ to estimate the vector of parameters ξ.

Call the result $G(\beta)$

$$G(\beta) = \arg\min_{\xi \in R^v} \left\| w_t - W(z_t(\beta), u_t; \xi) \right\|.$$

- *Step* 4. Compute the vector $\beta(i+1)$ for the next iteration

$$\beta(i+1) = (1-\mu)\beta(i) + \mu G(\beta(i)),$$

where $\mu \in (0,1)$ is the updating parameter.

Iterate on *Steps* $2-4$ until $\beta^* = G(\beta^*)$ for all t.

The simulation procedure underlying our algorithm is similar to the one used in a version of the PEA, developed by Marcet (1988).[2] The difference is that under Marcet's PEA, simulations are employed for computing the equilibrium law of motion of the conditional expectation in the Euler equation (6), whereas, in our method, simulations are used to solve for value function.

Unfortunately, our method does not necessarily guarantee finding a solution. This drawback, however, is common to all numerical algorithms iterating on first-order conditions. The failure might occur if the approximation happens to be far away from the true solution. The simulated series then become highly non-stationary, so that the regression delivers meaningless results. To rule out explosive (implosive) strategies, we restrict the endogenous

[2]Marcet and Lorenzoni (1999) and Christiano and Fisher (2000) describe the PEA applications and provide further references.

state variables within a certain range $z_{t+1} \in [\underline{z}, \overline{z}]$ for all t.[3] This range is chosen so that the restriction can bind the simulated series on initial iterations when the solution is imprecise, however, it becomes completely irrelevant when the solution is refined.

4 Example

Consider a version of the two-sector neoclassical growth model with four types of exogenous shocks, two to technology in two different sectors, one to preferences and one to the depreciation rate:[4]

$$\max_{\{c_t, k_{t+1}, h_{t+1}\}_{t=0}^{\infty}} E_0 \sum_{t=0}^{\infty} \delta^t \theta_{3,t} \ln (c_t) \tag{7}$$

$$\text{s.t.} \quad c_t + k_{t+1} + h_{t+1} = (1 - \theta_{4,t} d)(k_t + h_t) + \theta_{1,t} k_t^{\alpha} + \theta_{2,t} h_t^{\alpha}, \tag{8}$$

where c_t, k_t and h_t are consumption and the capital stocks in the two sectors, respectively; $\alpha \in (0, 1)$; the process for a shock $i \in \{1, ..., 4\}$ is an AR(1), $\ln \theta_{i,t+1} = \rho_i \ln \theta_{i,t} + \epsilon_{i,t+1}$ with $\epsilon_{i,t+1} \sim N\left(0, \sigma_{\epsilon,i}^2\right)$; $\theta_{4,t}$ is the shock to the

[3]For the simulation-based PEA, Maliar and Maliar (2001) show that imposing bounds on the simulated series significantly enhances the convergence properties of the algorithm. One can also ensure convergence by starting iterations from a good initial guess, such as a known solution to some related model (see den Haan and Marcet, 1991), or a log-linear solution to the model in question (see Christiano and Fisher, 2000).

[4]**Note for EL's editor:** For your evaluation, we include a diskette with the MATLAB programs solving the example.

depreciation rate, d, such that $\theta_{4,t} d \in (0,1)$; and $\left(k_0, h_0, \{\theta_{i,0}\}_{i=1}^4\right)$ is given. Thus, there are two endogenous state variables, k_t and h_t, and four exogenous state variables $\{\theta_{i,t}\}_{i=1}^4$.

We approximate the value function by

$$
\begin{gathered}
V\left(k_t, h_t, \{\theta_{i,t}\}_{i=1}^4\right) \simeq W\left(k_t, h_t, \{\theta_{i,t}\}_{i=1}^4 ; \beta\right) = \\
\beta_1 + \beta_2 \ln k_t + \beta_3 \ln h_t + \beta_4 \ln \theta_{1,t} + \beta_5 \ln \theta_{2,t} + \beta_6 \ln \theta_{3,t} + \beta_7 \ln \theta_{4,t} \\
+\beta_8 \ln \theta_{1,t} \ln k_t + \beta_9 \ln \theta_{2,t} \ln k_t + \beta_{10} \ln \theta_{3,t} \ln k_t + \beta_{11} \ln \theta_{4,t} \ln k_t \\
+\beta_{12} \ln \theta_{1,t} \ln h_t + \beta_{13} \ln \theta_{2,t} \ln h_t + \beta_{14} \ln \theta_{3,t} \ln h_t + \beta_{15} \ln \theta_{4,t} \ln h_t
\end{gathered} \tag{9}
$$

with $\beta = \left(\{\beta_i\}_{i=1}^{15}\right)$. Since there are two endogenous state variables, (4) leads to two intertemporal conditions

$$
\begin{aligned}
\frac{\theta_{3,t}}{c_t} &= \delta E_t \left[\frac{\partial W\left(k_{t+1}, h_{t+1}, \{\theta_{i,t+1}\}_{i=1}^4 ; \beta\right)}{\partial k_{t+1}} \right] = \\
& \frac{\delta}{k_{t+1}} \left(\beta_2 + \beta_8 \rho_1 \ln \theta_{1,t} + \beta_9 \rho_2 \ln \theta_{2,t} + \beta_{10} \rho_3 \ln \theta_{3,t} + \beta_{11} \rho_4 \ln \theta_{4,t}\right),
\end{aligned} \tag{10}
$$

$$
\begin{aligned}
\frac{\theta_{3,t}}{c_t} &= \delta E_t \left[\frac{\partial W\left(k_{t+1}, h_{t+1}, \{\theta_{i,t+1}\}_{i=1}^4 ; \beta\right)}{\partial h_{t+1}} \right] = \\
& \frac{\delta}{h_{t+1}} \left(\beta_3 + \beta_{12} \rho_1 \ln \theta_{1,t} + \beta_{13} \rho_2 \ln \theta_{2,t} + \beta_{14} \rho_3 \ln \theta_{3,t} + \beta_{15} \rho_4 \ln \theta_{4,t}\right).
\end{aligned} \tag{11}
$$

By combining (8), (10), (11), we get

$$c_t = \frac{(1-\theta_{4,t}d)(k_t+h_t)+\theta_{1,t}k_t^{\alpha}+\theta_{2,t}h_t^{\alpha}}{1+\frac{\delta(\beta_2+\beta_3+(\beta_8+\beta_{12})\rho_1\ln\theta_{1,t}+(\beta_9+\beta_{13})\rho_2\ln\theta_{2,t}+(\beta_{10}+\beta_{14})\rho_3\ln\theta_{3,t}+(\beta_{11}+\beta_{15})\rho_4\ln\theta_{4,t})}{\theta_{3,t}}}.$$

Given c_t, (10) and (11) identify k_{t+1} and h_{t+1}, respectively.

As an initial guess, we choose β that matches the non-stochastic steady state

$$\beta_1 = \frac{\ln(c_{ss})}{1-\delta} - 2\beta_2\ln(k_{ss}), \quad \beta_2=\beta_3=\frac{k_{ss}}{\delta c_{ss}}, \quad \beta_4,...,\beta_{15}=\varepsilon,$$

where "ss" denotes steady state values, and ε is a small number (we take $\varepsilon = 10^{-5}$).[5] Here, we use the fact that $h_{ss}=k_{ss}$.

To simulate the model, we set: $\alpha=0.33$, $\delta=0.95$, $d=0.02$, $\rho_i=\rho=0.95$, $\sigma_{\epsilon,i}=\sigma_\epsilon\in\{0.005,0.05\}$, $i=1,...,4$, $\left(k_0,h_0,\{\theta_{i,0}\}_{i=1}^4\right)=\left(k_{ss},h_{ss},\{1\}_{i=1}^4\right)$. The updating parameter is set at $\mu=0.5$, and k_t and h_t are restricted to lie in the interval $[k_{ss}/5, 5k_{ss}]$. The convergence criterion is that the precision in the coefficient vector is less than 10^{-5}.

As a comparison, we also apply the algorithm for solving the one-sector neoclassical model where there are shocks only to technology. *Table 1* provides the simulation results under three values of simulation length, $T\in$

[5]The MATLAB subroutine "nlinfit", which we use to run the nonlinear regression, may fail to work appropriately if some coefficients are equal to zero.

$\{1000, 5000, 10000\}$. Observe that the expense for the two-sector model is about three times as high as that for the one-sector model. The increase in the computational time is the result of having eleven additional parameters in the regression. Given that there are two state variables in the one-sector model and that there are six state variables in the two-sector model, the proportional three-time increase in the computational expense seems to be modest.

5 Comparison with other methods

With a large number of state variables, our algorithm can be a cheap alternative to the traditional grid-based dynamic programming methods in situations where value function can be accurately approximated by low-degree polynomials.

In applications with several endogenous state variables, our algorithm also has an important advantage over Marcet's (1988) version of the PEA. Specifically, the PEA needs to parametrize and approximate as many conditional expectations as there are endogenous state variables in the model.[6] For instance, applying the PEA to the two-sector growth model would require parametrizing two conditional expectations by two different functions and

[6]For a discussion, see Marcet and Lorenzoni (1999), Example 7.6.

computing twice the amount of polynomial coefficients than we did. Three endogenous state variables would imply three regressions to run and would triple the number of the coefficients, etc. The need to simultaneously iterate on more than one decision function can not only increase the computational expense, but can also complicate the practical implementation of the PEA and lead to a problem of non-convergence. This does not happen with our method, in which, independently of the number of endogenous state variables, there is always just one value function to be approximated.

Our method may also be preferable to the conventional PEA in applications where value function enters the Euler equation. This can occur in models with endogenous business cycles, e.g., Andolfatto and MacDonald (1998), and Freeman, Hong and Peled (1999). The PEA operates on the Euler equation without calculating value function. Hence, if the PEA is applied to such a model, it would be necessary to approximate value function on each PEA's iteration somehow. In contrast, with our method, the approximate value function is always known.

6 Final remark

Just as different versions of the PEA exist in the literature (see Christiano and Fisher, 2000), one can consider different variants of our method. For example, instead of Monte Carlo simulations, it is possible to seek a solution on a grid and use a quadrature integration; the updating can be replaced by a gradient descendent method; under appropriate collocation of grid points, a nonlinear least-square regression can be substituted by a linear one. Such modifications may increase the method's speed and/or accuracy in some applications.

References

[1] Andolfatto, D. and G. MacDonald, 1998, Technology diffusion and aggregate dynamics, Review of Economic Dynamics 1, 338-370.

[2] Christiano, L. and J. Fisher, 2000, Algorithms for solving dynamic models with occasionally binding constraints, Journal of Economic Dynamics and Control 24, 1179-1232.

[3] den Haan, W., and A. Marcet, 1990, Solving the stochastic growth model by parameterizing expectations, Journal of Business and Economic Statistics 8, 31-34.

[4] Freeman, S., D. Hong and D. Peled, 1999, Endogenous cycles and growth with indivisible technological developments, Review of Economic Dynamics 2, 403-432.

[5] Maliar, L. and S. Maliar, 2002, Parameterized expectations algorithm and the moving bounds, Journal of Business and Economic Statistics, *(forthcoming).*

[6] Marcet, A., 1988, Solving nonlinear stochastic models by parametrizing expectations, Working paper, Carnegie Mellon University.

[7] Marcet, A. and G. Lorenzoni, 1999, The parameterized expectation approach: some practical issues, in: R. Marimon and A. Scott, eds., Computational Methods for Study of Dynamic Economies *(*New York: Oxford University Press) 143-171.

[8] Marimon, R. and A. Scott, 1999, Computational Methods for Study of Dynamic Economies (New York: Oxford University Press).

[9] Rust, J., 1996, Numerical dynamic programming in economics, in: H. Amman, D. Kendrick and J. Rust, eds., Handbook of Computational Economics *(*Amsterdam: Elsevier Science) 619-722.

Table 1. Computational time for the one- and two-sector neoclassical models.

	T=1000		T=5000		T=10000	
	σ_ε =0.005	σ_ε =0.05	σ_ε =0.005	σ_ε =0.05	σ_ε =0.005	σ_ε =0.05
One-sector model, time, sec	124	61	383	304	760	653
Two-sector model, time, sec	351	186	1581	977	2025	1704