A TECHNIQUE FOR SOLVING RATIONAL EXPECTATION MODELS

Jean-Louis BRILLET

*Institut National de la Statistique et des Etudes Economiques (INSEE), BP 100, 15 bd Gabriel Péri, 92244
Malakoff Cedex, France
e-mail : jean-louis.brillet@insee.fr*

Abstract : Solving large rational expectation models is now feasible on microcomputers, using the traditional Fair Taylor or the newer stacked-time algorithms**.** We propose another technique, which could present additional improvements both in convergence speed and in solution reliability. It is based on the notion of loop variable, and solves the system with a sequence of Newton-Raphson iterations based on a Jacobian matrix of reduced size. Practical tests are conducted at present, and results should be available soon. *Copyright ©1999 IFAC*

Keywords : algorithms, simulation, Jacobian, modelling, economics, estimation.

1 INTRODUCTION

In past years, the increase in computer speed and available memory has allowed solving rational expectations by another method than Fair-Taylor, even for large models on a large number of periods. And these methods, globally called « stacked-time » seem to be generally more efficient.

Our purpose is to consider a slightly different method, based on the notion of "loop variables", which limits the size of the technical problem, and hopefully speeds up the process.

2 THE GENERAL PROBLEM

Before trying to solve a rational expectation model, we can generally, through the addition of identities, eliminate its lags or leads larger than one, leading to a "Blanchard – Kahn" specification:

$$(1)\ f_t\,(\,y_t\,,\,y_{t-1},\,y_{t+1},\,x_t) = 0$$

with

- t representing a time index varying from 1 to T

- y a vector of endogenous variables (dimension n)
- x a vector of exogenous variables

y_0 and y_{T+1} are known.

Two methods are generally considered for solving such a system.

Fair - Taylor: starting from initial leads, each f_t is solved in turn from 1 to T, using a Newton-Raphson algorithm, until the leads converge.

Stacked-Time: The whole system is solved directly, by linearization of the system, block-triangularization, and global application of Newton-Raphson. See for instance Laffargue JP (1990), Hollinger P (1996).

Of these methods, enforcing the Blanchard Kahn conditions is more essential to the second one.

For a comparison of these algorithms, one can refer to Juillard M; Laxton D, (1996) or Juillard M, et al. (1996).

3 THE LOOP VARIABLES

This technique was first introduced by Gabay, *et al.* (1978).

Let us consider the model (without rational expectations):

$$(2)\ y_t = f_t\ (\ y_t\ ,\ y_{t-1},\ x_t)$$

Let us consider a particular ordering of the equations, and observe their formulations from the top down. The loop variables are defined as the ones that appear on the right hand side through their present value, before they are computed. Solving the model for one particular period we can drop the time indexes, leading to:

$$(3)\ y = f\ (y)$$

Calling y^b the subset of loop variables (cardinal n_b), it would be technically feasible to transform the set of formulations by replacing in turn, in the chosen order, the right-hand side variables of lower rank by their expressions (themselves possibly already modified). This process would leave only loop variables on the right side.

$$(4)\ y = g\ (y^b)$$

But of course the expressions would be much too complex for this system to be managed analytically. However, we can compute the value of y associated through g to a starting value of y^b through a very simple process: one has just to use the last computed values instead of the initial ones, which means performing a Gauss-Seidel (and not Ritz-Jordan) iteration.

This has important implications for the solution of the Newton-Raphson algorithm. By shocking each loop variable in turn, performing a single Gauss-Seidel iteration, and computing the difference to a non shocked iteration (in other words, through finite differences), we can compute each column of the Jacobian of

$$(5)\ y^b = g\ (y^b)$$

This allows applying the Newton-Raphson algorithm to a problem with the size of the number of loop variables. Dropping the b index in favor of the number of the iteration k, this gives:

$$(6)\quad y^K = y^{K-1} - (I - fg\ /\ fy)^{-1}_{y=y^{k-1}}\ (y^{K-1} - g(y^{K-1}))$$

or (7) :

$$y^K = (I - fg\ /\ fy)^{-1}_{y=y^{k-1}}\ (g(y^{K-1}) - (fg\ /\ fy)^{-1}_{y=y^{k-1}}\, y^{K-1})$$

In other words for a Newton - Raphson iteration we shall need:

- $n_b + 1$ Gauss Seidel iterations
- inverting a matrix of dimension n_b (a polynomial of degree three in n_b).

This method is obviously faster than the normal application of Newton - Raphson, which involves the inversion of a matrix of degree n[1]

[1] . But the actual diagnosis is not clear, as the larger matrix is also much sparser, and specific

4 APPLICATION TO RATIONAL EXPECTATIONS MODELS

The above technique can easily be applied to rational expectations models. We shall suppose for now that they appear under the identified form:

$$(8)\ y_t = f_t\ (\ y_t\ ,\ y_{t-1},\ ...\ y_{t-k},\ y_{t+1},\ x_t)$$

which is very often the case[2].

We shall suppose for now that the expectation variables appear with only one forward lag, but the lagged variables can appear with any number of backward ones.

It is well known, however, that a multi-period lag can be contracted to one by steps, by introducing additional identities. Applying this technique would have provided a Blanchard-Kahn form, useful for the application of some method, like the stacked-time algorithm.

$$(9)\ y_t = f_t\ (\ y_t\ ,\ y_{t-1},\ y_{t+1},\ x_t)$$

But we shall see that this technique does not bring any improvement to our method.

Let us now consider the global model, built by stacking one-period models. An equation is now associated with a variable and a period.

Its incidence matrix takes the following form (five periods is enough to present the general case).

$$\begin{bmatrix} A_{11} & A_{12} & 0 & 0 & 0 \\ A_{21} & A_{22} & A_{23} & 0 & 0 \\ A_{31} & A_{32} & A_{33} & A_{34} & 0 \\ A_{41} & A_{42} & A_{43} & A_{44} & A_{45} \\ A_{51} & A_{52} & A_{53} & A_{54} & A_{55} \end{bmatrix}$$

In this model, the notion of loop variable can still be applied to the variables "which are used for computations in an equation before they are computed themselves". To the set of original loop variables, we must now add the ones with a forward lag. There is a good chance, however, that variables have both characteristics, which apply generally to important elements of the model. Value added for instance can be an anticipated element, and appear in the Keynesian loop. This will reduce the size of the transformed problem, and the cost of subsequent computations.

We could consider applying the same method as above:

- Computing the Jacobian of the full system, by finite differences. We just have to apply a Gauss Seidel iteration to the full system, then shock in turn each of the new "loop" variables and compute the difference to the base values.

- Applying the standard Newton formula,

$$(6)\quad y^{K} = y^{K-1} - (I - \mathit{f}g\ /\ \mathit{f}y)^{-1}_{y=y^{k-1}}\ (y^{K-1} - g(\ y^{K-1}))$$

in which the dimension of the Jacobian is the product of the number of loop variables by the number of periods.

- Iterating until the solution is found.

One can see immediately the problem of this method. Solving rational expectations models often require a large number of periods, to eliminate terminal condition problems. Even with a moderate number of loop variables, the size of the problem can be quite big. For instance, a 300-equation model might have ten loop variables and five forward ones. Solving it for 50 periods would mean a Jacobian with dimension 750. This can still be manageable, but can prove expensive in computing time.

To improve the method, we shall follow the steps of the other algorithms in the field, and take into account the particular shape of the matrix.

If we compute the Jacobian according to our usual method, we shall get a matrix with the same shape as before, but with a considerably reduced size.

$$\begin{bmatrix} B_{11} & B_{12} & 0 & 0 & 0 \\ B_{21} & B_{22} & B_{23} & 0 & 0 \\ B_{31} & B_{32} & B_{33} & B_{34} & 0 \\ B_{41} & B_{42} & B_{43} & B_{44} & B_{45} \\ B_{51} & B_{52} & B_{53} & B_{54} & B_{55} \end{bmatrix}$$

A Newton iteration consists in applying to y a variation such that:

(I-B) Δy = (y-g(y))

Instead of inverting (I-B), we shall transform it to simplify the computation of Δy.

We have :

$$\begin{bmatrix} I-B_{11} & -B_{12} & 0 & 0 & 0 \\ -B_{21} & I-B_{22} & -B_{23} & 0 & 0 \\ -B_{31} & -B_{32} & I-B_{33} & -B_{34} & 0 \\ -B_{41} & -B_{42} & -B_{43} & I-B_{44} & -B_{45} \\ -B_{51} & -B_{52} & -B_{53} & -B_{54} & I-B_{55} \end{bmatrix} \Delta y = \begin{bmatrix} y_1-g_1 \\ y_2-g_2 \\ y_3-g_3 \\ y_4-g_4 \\ y_5-g_5 \end{bmatrix}$$

We shall start with the last line: by multiplying it on the left side by the inverse of (I-B55) we get :

$$\begin{bmatrix} I-B_{11} & -B_{12} & 0 & 0 & 0 \\ -B_{21} & I-B_{22} & -B_{23} & 0 & 0 \\ -B_{31} & -B_{32} & I-B_{33} & -B_{34} & 0 \\ -B_{41} & -B_{42} & -B_{43} & I-B_{44} & -B_{45} \\ C_{51} & C_{52} & C_{53} & C_{54} & I \end{bmatrix} \Delta y = \begin{bmatrix} y_1-g_1 \\ y_2-g_2 \\ y_3-g_3 \\ y_4-g_4 \\ d_5 \end{bmatrix}$$

With

$$C_{5i} = -(I - B_{55})^{-1} B_{5i}$$

$$d_{5=} (I - B_{55})^{-1} (y_5 - g_5)$$

Let us now proceed to line 4.

We want both to eliminate the term above the diagonal, and transform the diagonal term to I. For this we can apply to the fourth period lines of our Jacobian on the left by a transformation based on the last two periods:

$$C_{4i} = E \langle (-B_{4i}) + F \langle C_{5i}$$

Such that

$$E \langle (I - B\ \ ) + F \langle C\ \ = I$$

As we have

$$C_{55} = I$$

The second expression gives

$$F = E \langle B_{45}$$

And by substitution in the first:

$$E = (I - B_{44} + C_{45}B_{54})^{-1}$$

$$F = (I - B_{44} + C_{45}B_{54})^{-1}B_{45}$$

And finally

$$C_{4i} = (I - B_{44} + C_{45}B_{54})^{-1}(\delta_4^i I - B_{4i} + B_{45}C_{5i})$$

Of which we can check that

$$C_{44} = (I - B_{44} + C_{45}B_{54})^{-1}(I - B_{44} + B_{45}C_{54}) = I$$

$$C_{45} = (I - B_{44} + C_{45}B_{54})^{-1}(-B_{45} + B_{45}I) = 0$$

$$d_4 = (I - B_{44} + C_{45}B_{54})^{-1}(g_4 - y_4 + B_{45}d_5)$$

Of course we have to apply the same transformation to the right hand term:

Applying the same recursive method up to the first line::

$$C_{j,i} = (I - B_{j,j} + C_{j,j+1}B_{j+1,j})^{-1}(\delta_j^i I - B_{j,i} + B_{j,j+1}C_{j+1,i})$$

$$d_j = (I - B_{j,j} + C_{j,j+1}B_{j+1,j})^{-1}(g_j - y_j + B_{j,j+1}d_{j+1})$$

And finally :

$$\begin{pmatrix} I & 0 & 0 & 0 & 0 \\ C_{21} & I & 0 & 0 & 0 \\ C_{31} & C_{32} & I & 0 & 0 \\ C_{41} & C_{42} & C_{43} & I & 0 \\ C_{51} & C_{52} & C_{53} & C_{54} & I \end{pmatrix} \Delta y = \begin{pmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \end{pmatrix}$$

The (forward) solution of the system for Δy is immediate.

Comparison with other methods

Actually, this method looks very similar to the stacked time algorithm (Laffargue(1990), Juillard(1996)). The framework is the same, and the main difference lies in the use of loop variables. This option presents both plusses and minuses.

Minus :	Plus:
One has to produce Gauss Seidel iterations on the full model.	The Gauss-Seidel iterations are limited to the "loop" variables
The elementary matrixes are much fuller	The elementary matrixes are much smaller
The transformed Jacobian matrix must be full lower triangular with a higher band, not band triangular. But this means eliminating lags higher than one is unnecessary	One does not have to eliminate the lags higher than one.
One can use a priori knowledge on the actual forward and backward linkages to use sparse matrix inversion techniques	The identification of the loop variables is sufficient (and can be automatized), not the knowledge and storage of the lagged and forward individual links.
The lower triangular part of the Jacobian is full, which increases the number of computations, the more as the number of period increases	Formal changes in model equations are taken into account automatically.
This increases also the number of elements to be stored	After one period, the Gauss-Seidel iterations take into account downward influences which increase the accuracy of the Jacobian.
	The method generalizes to forward influences from several periods ahead, without increasing too much computation costs

5 THE MODEL

To evaluate the efficiency of this method, in terms of computing speed and convergence probability, tests must be undertaken. And if it is clear that the memory size needed is much lower than for the full problem, using sparse matrix techniques can also reduce the size of the problem drastically.

For this we shall use a small macro economic model of the French economy, itself a part of a larger multi-country model developed for the MACSIM project.

The goal of this project is to produce a tool for teaching international macroeconomics, by linking individual country models. One will find a short description of this package in Brillet (1998).

The international dimension means that the model framework can apply to most industrialized countries. Another interesting feature is that the behavorial equations in the model generally follow an error correction (ECM) format.

The initial version does not use rational expectations, and presents three loop variables, associated with the Keynesian cross, the price-wage interdependence, and (optionally) the exchange rate

In the first phase, we have introduced rational expectations in the following behaviors.

- In the investment equation: the firms are now supposed to know in advance the actual production level needed in the next period, which determines the optimal capacity level (with a target capacity utilization rate).

- Households know in advance their future revenue, which allows them to optimize their consumption behaviour.

This means we have five non pre-determined variables per period (actually only two for period 1, and three for period T).

In a second phase, we shall introduce rational expectations on inflation and interest rates. This will affect the behaviour of households (through the savings equation) and again the level of productive investment.

6 THE TESTS

Our tests use the following steps.

We produce simulations of the backward looking model over a very long period (several centuries) using exogenous assumptions compatible with a steady state growth path. This leads us to modify slightly some coefficients, for instance to eliminate some trends or set them to a theoretical value.

We also formally specify a long-term model, in which all lagged variables have been replaced by present ones, by applying the steady state growth rate.

Once we have found for the dynamic model a trajectory which presents growth rates close enough to steady state values, we compare it to the solution of the formal long-term model, to check that the long term solution has indeed been reached.

We replace in the appropriate formulations of the dynamic model the backward looking terms by forward-looking ones, and solve the model again. As we are on the steady state path, the solution will not change.

Finally, we apply a constant change to the long-term assumptions, for a certain number of final periods. We solve the forward-looking model again, using individual programs associated with one algorithm in the test, but identical in every other way.

Simulations are being be produced over 20 to 100 periods, comparing, in terms of speed and convergence reliability, our method with the following:

- Pure Gauss – Seidel
- Fair Taylor
- Stacked-time

The conditions of the tests will vary with the number of shocked periods, the horizon.

To get a complete picture, we shall also consider various starting points (at different distances from the solution) and various accuracy requirements (our method, which is pure Newton, should be more efficient for high accuracy requirements).

We shall also introduce a weighting parameter allowing to move by steps from a pure backward to a pure forward looking model, and observe how it affects the diagnosis.

At present, we have produced a program computing a long-term dynamic solution (using Newton and loop variables), that does indeed converge to the long-term model. And of course introducing rational expectations leaves the solution unchanged.

We have implemented our algorithm, as well as the Fair Taylor algorithm. The first tests seem to show that for relatively large numbers of periods (50 for example) our method compares quite well, especially (as could be expected) if the numerical influence of the forward variables is important[3]. But we still have to compare with stacked-time, which uses a block diagonal Jacobian instead of a lower triangular one.

For instance, supposing that wages depend on future inflation (introducing a multi-period wage-price loop) gives a much bigger advantage to our method than making investment depend on future production (investment is quite inert, and represents a small share of demand).

The full results will be available for the Boston CEFES meeting, as well as for the July IFAC Conference in Beijing. They will be updated on the site.

REFERENCES

Brillet JL (1998), Win MCD and MACSIM: Computer tools for teaching macroeconomics, *Proceedings of the 3rd Congress on Computing in Economics and Finance*, Kluwer Academic Publishers

Gabay D., Nepomiaschy P., Rachdi M., Ravellli A. (1978), Etude, résolution et optimisation de modèles macroéconomiques, *INRIA report,* 1978.

Hollinger P(1996), The Stacked-Time Simulator in TROLL: A Robust Algorithm for Solving Forward-Looking Models, *paper presented at the Second International Conference on Computing in Economics and Finance (Geneva, Switzerland*).

Juillard, M; and Laxton D, (1996), "A Robust and Efficient Method for Solving Nonlinear Rational Expectations Models," *International Monetary Fund Working Paper* WP/96/106

Juillard, Michel; Douglas Laxton, Peter McAdam, and Hope Pioro, (1996), An Algorithm Competition: First-Order Iterations Versus Newton-Based Techniques, *Journal of Economic Dynamics and Control*, (forthcoming).

Laffargue, J.P.(1990), Résolution d'un Modèle Macroéconomique Avec Anticipations Rationnelles, in *Annales d'Economie et Statistique*, Vol. 17, pp. 97-119.

Le Van C, Schubert K, Boucekine R (1996), How to Get the Blanchard-Kahn Form from a General Linear Rational Expectations Model, *Proceedings of the Second International Conference on Computing in Economics and Finance Geneva, Switzerland, 26-28 June 1996*