Impact Project

Impact Research Centre,
The University of Melbourne,

Baldwin Spencer Building (2nd floor)
PARKVILLE, VIC. 3052 AUSTRALIA

Telephone: (03) 344 7417
Telex: AA 35185 UNIMEL
E-mail:
Fax: (03) 347 7539

(from overseas: 61 3 344 7417)
Telegrams: UNIMELB, PARKVILLE
impact@impact.unimelb.EDU.AU
(from overseas: 61 3 347 7539)

A Fast Algorithm for Solving Rational Expectations Models

by

Peter J. Wilcoxen

Impact Project Research Centre
University of Melbourne

Preliminary Working Paper No. IP-47, April 1990

ISSN 1031 9034

ISBN 0 642 10057 8

Abstract

At the heart of most rational expectations models is a system of first order differential equations describing the behaviour of the model's endogenous variables. The methods commonly used to solve these models, such as multiple shooting and Fair's algorithm, can be very slow to converge. This paper describes a generalization of Fair's technique that is significantly faster while providing equally accurate results.

References

Fair, R.C. (1979) "An Analysis of a Macro-Econometric Model with Rational Expectations in the Bond and Stock Markets", *Econometrica*, 69(4), 539-552.

Fair, R.C. and J.B. Taylor (1983) "Solution and Maximum Likelihood Estimation of Dynamic Nonlinear Rational Expectations Models", *Econometrica*, 51(4), 1169-1185.

Lipton, D., J. Poterba, J. Sachs and L. Summers (1982) "Multiple Shooting in Rational Expectations Models", *Econometrica*, 50(5), 1329-1333.

Wilcoxen, P.J. (1985) "Numerical Methods for Investment Models with Foresight", IMPACT Project *Preliminary Working Paper* No. IP-23, University of Melbourne.

Wilcoxen, P.J. (1988) "The Effects of Environmental Regulation and Energy Prices on U.S. Economic Performance", Ph.D. Thesis, Harvard University.

A Fast Algorithm for Solving Rational Expectations Models

by

Peter J. Wilcoxen

At the heart of most rational expectations models is a system of first order differential equations describing the behaviour of the endogenous variables. Each equation will be accompanied by a boundary condition on one of the variables; some specified as initial values, and others as terminal conditions. If the variables are divided into vectors k and w according to whether the corresponding boundary condition is specified at the initial or terminal time, the problem is to solve:

$$\dot{k}_t = g(w_t, k_t, z_t) \quad , \tag{1}$$

$$\dot{w}_t = f(w_t, k_t, z_t) \quad , \tag{2}$$

given boundary conditions:

$$w_T = \bar{w} \quad , \tag{3}$$

$$k_0 = \bar{k} \quad . \tag{4}$$

In a typical problem, the elements of k are stocks of physical capital, while the elements of w are the values of those stocks. Vector z_t is exogenous. Time 0 is the present, and T is a date in the future. (As noted in Lipton, *et al.* (1982), no additional difficulty is presented by boundary conditions expressed as limits when t approaches infinity.) Since some of the boundary conditions hold at time 0, and others at T, this is known in the differential equations literature as a two point boundary value problem, and it can be very difficult to solve.

The first step in obtaining a numerical solution is to approximate equations (1) and (2) using differences. If equation (1) is then solved for $t-1$ instead of t, the problem can be written:

$$k_t = k_{t-1} + g(w_{t-1}, k_{t-1}, z_{t-1}) \quad , \tag{5}$$

$$w_t = w_{t+1} - f(w_t, k_t, z_t) \quad . \tag{6}$$

This highlights an important point: k_0 is known, so if the path of w were available, equation (5) could be used to calculate k at each point in time. Unfortunately, only the terminal value of w is known from the boundary conditions and solving the problem is not trivial.

Two numerical methods are commonly used by economists to solve two point boundary value problems.[1] The first, originally due to Fair (1979) and later extended by Fair and Taylor (1983)[2], is based on the Gauss-Seidel algorithm. The second is multiple shooting, introduced to economists by Lipton, *et al.* (1982). Both are iterative procedures that work in the following way. A trial vector $\hat{w}$ is used to solve equation (5) forward through time to obtain a path of k, say $\hat{k}$. Then, $\hat{k}$ and $\hat{w}$ are used to check whether equation (6) is satisfied. If it isn't, $\hat{w}$ is revised and the process repeated. The algorithms differ in how much of w is guessed at each iteration and how the revision is conducted. In multiple shooting, w is guessed at a few widely separated points and a complex procedure is used for revising it. Fair's algorithm, on the other hand, requires w for each point in time, but the revision rule is much

1. A third method, known as finite differences, has been used by Wilcoxen (1985). It is particularly well suited to linear (or linearized) models.
2. We will be concerned with Fair and Taylor's type II iteration. Their type III procedure, needed when the terminal condition is unknown, could be used with the algorithm without any difficulty.

Perturbing the second period expectations produces J_{12}. Once J_{11} and J_{12} have been computed, a trial path of expectations must be provided. Inserting that into equation (5) gives the implied paths of the state variables. Then, equation (6) can be used to create "actual" values for the expected variables. Finally, equation (12) shows how to revise the guess, and can be solved by backward substitution.[6]

The algorithm introduced here is faster than either Fair's method or multiple shooting, and is not difficult to implement. Although derived differently, it can be regarded as a generalization of Fair's algorithm that makes use of the intertemporal equations in the model to accelerate convergence. Tests on several problems, including a large general equilibrium model, have shown it to be 2-10 times faster than the original Fair algorithm (Wilcoxen, 1988).

6. In practice, it is sometimes useful to multiply the two Jacobians by damping constants (scalars less than 1) when applying equation (12). This makes the revision process more gradual but helps to keep it monotonic.

determine earlier values of E since the periods are linked by the J_{12} terms in the (I–J) matrix.

Since the below diagonal terms are all zero, the system of equations can be solved easily by backward substitution.[4] The revised expectation for the last period is given by:

$$E_{i+1}^{3} = (I - J_{11})^{-1} \left[A_3 - J_{11} E_i^{3} \right] \quad . \tag{16}$$

Earlier periods are then found by repeated application of the expression:

$$E_{i+1}^{j} = (I - J_{11})^{-1} \left[A_j - J_{11} E_i^{j} - J_{12} E_i^{j+1} + J_{12} E_{i+1}^{j+1} \right] \quad . \tag{17}$$

Since J_{11} is assumed to be constant over periods, it is only necessary to compute $(\mathbf{I}-\mathbf{J}_{11})^{-1}$ once for each iteration of the algorithm. Moreover, for models with a single foresight variable, the expression becomes particularly simple since the matrix inversion is replaced by ordinary division.

As a practical matter, using the algorithm consists of several steps. The first is to obtain the values of J_{11} and J_{12}.[5] One way to do this is to solve the first several periods for an arbitrary (but reasonable) set of expectations. Then, by perturbing the first period expectations one at a time, the elements of J_{11} can be determined.

4. Nonzero terms below the diagonal would mean that solving the system would require matrix inversion or Gaussian elimination. Both of these are much more difficult than backward substitution, so that is why it is convenient to approximate those terms with zeros.
5. If there is one foresight variable, these will be scalars; otherwise, they will be matrices.

simpler. The method proposed below is intermediate between these two. It retains much of the simplicity and ease of use of Fair's procedure, while employing a more sophisticated revision rule that converges much faster.

The algorithm can be derived formally as follows: given a vector valued function **A** that generates actual values from a vector of expectations **E**, find a vector **E** which solves the equation:

$$\mathbf{E} = \mathbf{A}(\mathbf{E}) \quad . \tag{7}$$

In the notation used earlier, **E** would contain the values of w_t at all t from 0 to T and the actuals vector would be computed from the following two equations:

$$k_t = k_{t-1} + g(E_{t-1}, k_{t-1}, z_{t-1}) \quad , \tag{8}$$

$$A_t = E_{t+1} - f(E_t, k_t, z_t) \quad . \tag{9}$$

Equation (7) suggests using a numerical approach similar to Newton's method. Replacing **A** in (7) with a first-order Taylor series expansion of it around vector $\mathbf{E}_i$ produces:

$$\mathbf{A}(\mathbf{E}_{i+1}) \approx \mathbf{A}(\mathbf{E}_i) + \mathbf{J}_i(\mathbf{E}_{i+1} - \mathbf{E}_i) \quad ; \tag{10}$$

where $\mathbf{E}_i$ is the trial vector of expectations at iteration i and $\mathbf{J}_i$ is the Jacobian of **A** evaluated at $\mathbf{E}_i$. Taking $\mathbf{E}_{i+1}$ to be a solution, the left hand side can be replaced to give the following:

$$\mathbf{E}_{i+1} = \mathbf{A}_i + \mathbf{J}_i(\mathbf{E}_{i+1}-\mathbf{E}_i) \quad ; \tag{11}$$

where $\mathbf{A}_i$ is function $\mathbf{A}$ evaluated at $\mathbf{E}_i$. Collecting unknown terms on the left yields the expression below:

$$(\mathbf{I}-\mathbf{J}_i)\mathbf{E}_{i+1} = \mathbf{A}_i - \mathbf{J}_i\mathbf{E}_i \quad . \tag{12}$$

Equation (12) is the basis of the algorithm: it's used as a revision rule to find a new guess, $\mathbf{E}_{i+1}$, from the previous guess and the actuals.

For comparison, Fair's algorithm proceeds by computing $\mathbf{A}(\mathbf{E}_i)$ for a trial solution $\mathbf{E}_i$. If $\mathbf{A}_i$ is not sufficiently close to $\mathbf{E}_i$, a new trial vector is determined as shown:

$$\mathbf{E}_{i+1} = \gamma\mathbf{A}_i + (1-\gamma)\mathbf{E}_i \quad ; \tag{13}$$

where $\mathbf{A}_i = \mathbf{A}(\mathbf{E}_i)$. The term γ is a parameter between zero and 1 which can be adjusted to improve the rate of convergence when the guess is far from the solution. Aside from γ, this is the same as equation (12) with $\mathbf{J}_i$ set to zero. The new algorithm can be thought of as an extension to Fair's method in which the Jacobian is not zero. This makes the revision rule more complex, but since it uses more information about the problem, it takes fewer iterations to converge.

For simple models, it may be possible to derive J_i analytically, but usually it will have to be found numerically. Fortunately, several features of the problem help simplify the task. The first is that the actuals in period t don't depend on expectations beyond period $t+1$. This can be confirmed by differentiating equation (6), and means

that the Jacobian is almost lower triangular. Only one element to the right of the diagonal will be nonzero in each row.

To describe the other simplifications, it will help to use a simple model in scalar notation. Suppose the problem has one expected variable, three periods, and can be written as follows, where the subscript k on $\mathbf{J}$ has been eliminated for clarity:

$$\begin{bmatrix} 1-J_{11} & -J_{12} & -J_{13} \\ -J_{21} & 1-J_{22} & -J_{23} \\ -J_{31} & -J_{32} & 1-J_{33} \end{bmatrix} \begin{bmatrix} E_{i+1}^1 \\ E_{i+1}^2 \\ E_{i+1}^3 \end{bmatrix} = \begin{bmatrix} A_1 \\ A_2 \\ A_3 \end{bmatrix} - \begin{bmatrix} J_{11} & J_{12} & J_{13} \\ J_{21} & J_{22} & J_{23} \\ J_{31} & J_{32} & J_{33} \end{bmatrix} \begin{bmatrix} E_i^1 \\ E_i^2 \\ E_i^3 \end{bmatrix} \quad . \tag{14}$$

As noted above, J_{13} is zero. A further simplification is to set the derivatives of the actuals with respect to past expectations to zero. Unlike the treatment of J_{13}, however, this is only an approximation: the actuals will depend somewhat on state variables which will have been influenced by earlier expectations.[3] One final approximation is to assume that the remaining partials are the same across periods, so $J_{11}=J_{22}$, and so on. All this produces the system below:

$$\begin{bmatrix} 1-J_{11} & -J_{12} & 0 \\ 0 & 1-J_{11} & -J_{12} \\ 0 & 0 & 1-J_{11} \end{bmatrix} \begin{bmatrix} E_{i+1}^1 \\ E_{i+1}^2 \\ E_{i+1}^3 \end{bmatrix} = \begin{bmatrix} A_1 \\ A_2 \\ A_3 \end{bmatrix} - \begin{bmatrix} J_{11} & J_{12} & 0 \\ 0 & J_{11} & J_{12} \\ 0 & 0 & J_{11} \end{bmatrix} \begin{bmatrix} E_i^1 \\ E_i^2 \\ E_i^3 \end{bmatrix} \quad . \tag{15}$$

Finally, if the terminal value of E is known, as it often is, the equation for the last actual can be dropped and the final element of E set directly. This also helps

3. This assumption is not essential to the algorithm. If the below-diagonal terms are likely to be large, they can be calculated and used appropriately. Setting them to zero does, however, provide a computational benefit that will be described below.