A Simple Second-Order Solution Method for Dynamic General Equilibrium Models

Notes on MATLAB codes

Alan Sutherland
September 2003

The main solution algorithm is contained in three MATLAB m-files: `first_order`, `lambdas` and `second_order`. These m-files take as input numerical coefficient matrices in a format which is described below. `first_order`, `lambdas` and `second_order` can be used directly on any numerical input matrices which are in the correct format, but, as an aid to constructing input matrices, there are three further m-files (`load_model`, `load_parameters` and `s2n`) which read a model in symbolic form (stored in a text file) and generate the required numerical coefficient matrices. `load_model`, `load_parameters` and `s2n` make use of the Matlab Symbolic Math Toolbox. There are a number of further m-files which generate deterministic impulse responses, create output tables with headings and check results.

`first_order` and `second_order` make use of the QZ algorithm for solving linear dynamic systems (as described in Klein (2000) and Sims (1996)) . This algorithm has the advantage that the system of dynamic equations can be augmented with contemporaneous relationships. All equations of the model (both dynamic and contemporaneous) can therefore be solved as a single unified system.

Model format:
The variables of the model are arranged in a vector *Z* which is partitioned as follows:

$$Z = \begin{bmatrix} Z_1 \\ Z_2 \\ Z_3 \end{bmatrix}$$

where Z_1 is a $n_1 \times 1$ vector of predetermined endogenous variables, Z_2 is a $n_2 \times 1$ vector of exogenous autoregressive forcing variables and Z_3 is a $n_3 \times 1$ vector of all other variables (including forward-looking variables). Define $n = n_1 + n_2 + n_3$.

The equations of the model are arranged in the following matrix form:

$$A_1\begin{bmatrix} Z_{1,t+1} \\ Z_{2,t+1} \\ E_t\left[Z_{3,t+1}\right] \end{bmatrix} + B_1\begin{bmatrix} E_0\left[Z_{1,t+1}\right] \\ E_0\left[Z_{2,t+1}\right] \\ E_0\left[Z_{3,t+1}\right] \end{bmatrix} = A_2\begin{bmatrix} Z_{1,t} \\ Z_{2,t} \\ Z_{3,t} \end{bmatrix} + B_2\begin{bmatrix} E_0\left[Z_{1,t}\right] \\ E_0\left[Z_{2,t}\right] \\ E_0\left[Z_{3,t}\right] \end{bmatrix} + A_3\Lambda_t + A_4\varepsilon_{t+1} \qquad (1)$$

where Λ is a vector of second-order terms in the variables of the model - i.e. terms in the squares and cross products of the elements of Z_t - and ε is a vector of iid shocks. (Note that the structure in (1) is somewhat more general than presented in Sutherland (2002) and there are some small differences in notation.)

Exogenous forcing processes must be of the form $\xi_{t+1} = \delta\xi_t + \varepsilon_{\xi,t+1}$ where the iid shocks only enter the model via the exogenous forcing processes with one iid shock per forcing process (which enters with a unit coefficient). The iid shocks can be correlated.

It is assumed that the leads and lags in Λ_t are no greater than unity - thus Λ_t only contains terms in the elements of $Z'_{t-1}Z_{t-1}$, Z'_tZ_t and $Z'_{t+1}Z_{t+1}$. There are two situations where it may be necessary to augment the variables and equations of a model in order to conform to this format.

(i) An element of Λ_t contains a cross product between an element of Z_t and an element of Z_{t-1} or Z_{t+1}. In this case it is necessary to add a dummy variable to Z_t and an extra equation to bring the two parts of the cross product into Z_t.

(ii) An element of Λ_t contains a (square or cross-product) term in a one-period-ahead forecast of an element of Z_t, i.e. a term of the form $E_t\left[z_{j,t+1}\right] \times E_t\left[z_{k,t+1}\right]$. In this case it is necessary to add a dummy variable to Z_t and an extra equation to bring the one-period-ahead forecast into Z_t.

The first-order system embedded in (1) is given by

$$A_1\begin{bmatrix} Z_{1,t+1} \\ Z_{2,t+1} \\ E_t\left[Z_{3,t+1}\right] \end{bmatrix} = A_2\begin{bmatrix} Z_{1,t} \\ Z_{2,t} \\ Z_{3,t} \end{bmatrix} + A_4\varepsilon_{t+1} \qquad (2)$$

This system is used to generate the conditional second moments of Z_t.

By taking time-zero expectations of (1) the following system is obtained

$$\left(A_1+B_1\right)\begin{bmatrix} E_0\left[Z_{1,t+1}\right] \\ E_0\left[Z_{2,t+1}\right] \\ E_0\left[Z_{3,t+1}\right] \end{bmatrix} = \left(A_2+B_2\right)\begin{bmatrix} E_0\left[Z_{1,t}\right] \\ E_0\left[Z_{2,t}\right] \\ E_0\left[Z_{3,t}\right] \end{bmatrix} + A_3 E_0\left[\Lambda_t\right] \tag{3}$$

This system is used to generated a second-order solution for the first moments of Z_t. Notice that the matrices B_1 and B_2 allow for the possibility that the dynamics of first moments may be different from the dynamics of responses to shocks (for instance because policy may respond differently to anticipated and unanticipated events).

MATLAB functions:

The second-order solution is generated with the followings sequence of MATLAB commands:

```
VC = first_order(A1,A2,A4,SIGMA,ns);
LAMBDA = lambdas(SC,VC);
SOL = second_order(A1,A2,A3,B1,B2,LAMBDA,ns);
```

By default `first_order` generates conditional second moments for 400 periods, `lambdas` generates values for Λ_t for 400 periods and `second_order` generates first moments for 200 periods.

The three functions are now described in more detail:

`first_order`:

`first_order(A1,A2,A4,SIGMA,ns)` solves the first-order system (2) and returns the conditional second moments of Z_t. The following input is required:

1. The coefficient matrices from (2) - `A1,A2,A4`.
2. The variance-covariance matrix of the iid shocks - `SIGMA`.
3. The total number of (endogenous and exogenous) predetermined variables - `ns`.

Two further optional arguments may be provided: `hr1` is the number of time periods for which solutions for first moments will be generated and `hr2` is the number of additional

time periods for which conditional second moments are required. By default `hr1=hr2=200` so `first_order` generates conditional second moments for 400 periods.

`lambdas:`

`lambdas(SC,VC)` uses conditional second moments to generate a time series for Λ_t. The following input is required:

1. The coefficients of the second-order terms in Λ. These are coded in a matrix `SC` which has a special format which is described below.
2. The conditional second moments of Z_t (as generated by `first_order`).

`hr1` and `hr2` may be included as optional arguments. (`hr1` and `hr2` must be included with the same values as given in `first_order`).

The format of `SC` requires some description. Note that element *i* of Λ_t can be represented in matrix form by

$$\lambda_{i,t} = Z'_{t-1}R_{i,-1}Z_{t-1} + Z'_t R_{i,0} Z_t + Z'_{t+1}R_{i,1}Z_{t+1}$$

This is a compact format for algebraic representation but it is highly inefficient format from a computational point of view because (in most models) the vast majority of the elements of the *R* matrices will be zero. For this reason the coefficients of the second-order terms are passed to the function `lambdas` using a special compact format. To understand this format first note that each second-order term can be written as a scalar summation as follows:

$$\lambda_{i,t} = \sum_{s=-1}^{1}\sum_{j=1}^{n}\sum_{k=j}^{n} d_{j,k} r_{i,s,j,k} z_{j,t+s} z_{k,t+s} \qquad \text{where } d_{j,k} = \begin{cases} 1 \text{ if k} = j \\ 2 \text{ if } k > j \end{cases}$$

where $z_{i,t}$ is element *i* of *Z* at time *t* and $r_{i,s,j,k}$ is element *(j, k)* of matrix $R_{i,s}$. Each non-zero term in this summation is coded into a row in `SC`. So, for instance, a term of the form $d_{j,k} r_{i,s,j,k} z_{j,t+s} z_{k,t+s}$ which appears in λ_i is coded into a row of `SC` in the form

`(i,s,j,k,` $d_{j,k} r_{i,s,j,k}$ `)`.

As an illustration suppose there are two elements in Λ of the following form

$$\lambda_{t,1} = r_{1,0,1,1} z_{1,t}^2 + 2 r_{1,0,3,4} z_{3,t} z_{4,t} + r_{1,-1,6,6} z_{6,t-1}^2$$
$$\lambda_{t,2} = r_{2,1,8,8} z_{8,t+1}^2$$

If λ_1 and λ_2 are the only elements of Λ then SC is given by:

$$\mathtt{SC} = \begin{bmatrix} 1 & 0 & 1 & 1 & r_{1,0,1,1} \\ 1 & 0 & 3 & 4 & 2r_{1,0,3,4} \\ 1 & -1 & 6 & 6 & r_{1,-1,6,6} \\ 2 & 1 & 8 & 8 & r_{2,1,8,8} \end{bmatrix}$$

second_order:

second_order(A1,A2,A3,B1,B2,LAMBDA,ns) uses the time series for Λ_t generated by lambdas to generate the solution to the second-order system. The following input is required:

1. The coefficient matrices from (3) - A1,A2,A3,B1,B2.
2. The time series for Λ_t - LAMBDA.
3. The total number of (endogenous and exogenous) predetermined variables - ns.

hr1 and hr2 may be included as optional arguments. (hr1 and hr2 must be included with the same values as given in first_order and lambdas).

Model input in symbolic form:

If the equations of a model are stored in text file *model-filename*, with parameters stored in text file *parameter-filename*, the model can be loaded into MATLAB and converted into numerical form using the following set of commands:

```
[a1,a2,a3,a4,b1,b2,sc,Z,ns,ER,LR] = load_model(model-filename);
[prmn,prmv,SIGMA] = load_parameters(parameter-filename);
[A1,A2,A3,A4,B1,B2,SC] = s2n(a1,a2,a3,a4,b1,b2,sc,prmn,prmv);
```

Annotated example input files are provided as a formatting guide.

`load_model(`*`model-filename`*`)` loads the model equations and definitions of the second-order terms and converts them into symbolic matrix form. The output arguments `ER` and `LR` provide a check that the model has been correctly formatted and that it has been correctly converted into matrix form. A non-zero element of `ER` indicates an error in processing the corresponding model equation and a non-zero element of `LR` indicates an error in processing the corresponding second-order term.

`load_parameters(`*`parameter-filename`*`)` loads the parameter names and values.

`s2n(a1,a2,a3,a4,b1,b2,sc,prmn,prmv)` converts the symbolic matrices to numeric matrices using the supplied parameter values.

Note that calls to the Symbolic Math Toolbox can be very slow. Consequently execution times for `load_model`, `load_parameters` and `s2n` can be quite long, particularly for large models. For instance, a model with 50 equations can take up to one minute to load and convert into numeric form on a P4-2GHz machine. (But once the model is in numeric form `first_order`, `lambdas` and `second_order` execute in less than one second.)

References

Klein, Paul (2000) "Using the Generalised Schur Form to Solve a Multivariate Linear Rational Expectations Model" *Journal of Economic Dynamics and Control*, 24, 1405-1423.

Sims, Christopher (1996) "Solving Linear Rational Expectation Models", unpublished manuscript, Princeton University.

Sutherland, Alan (2002) "A Simple Second-Order Solution Method for Dynamic General Equilibrium Models" CEPR Discussion Paper No 3554.