Extracting Rational Expectations Model Structural Matrices from Dynare

Callum Jones[*]

New York University

In these notes I discuss how to extract the structural matrices of a model from its Dynare implementation. This can be useful for many applications. I illustrate one example here – implementing a ZLB algorithm. The ZLB algorithm is described in detail in 'Unanticipated Shocks and Forward Guidance at the Zero Lower Bound.'

Contents

1 Linear rational expectations model

Consider a rational expectations model $x_t = \Psi(x_{t-1}, \mathbb{E}_t x_{t+1}, w_t)$. Linearize the model around a non-stochastic steady state to get:

$$\mathbf{A}x_t = \mathbf{B}x_{t-1} + \mathbf{D}\mathbb{E}_t x_{t+1} + \mathbf{E}w_t. \tag{1}$$

The constant is included in x_t but can be written separately. In an economy where all agents know the regime and expectations are formed under that regime, the solution is a reduced-form VAR:

$$x_t = \mathbf{F}x_{t-1} + \mathbf{G}w_t.$$

[*]Department of Economics, New York University. Email: `callum.jones@nyu.edu`. All remaining errors are mine. Date: January 26, 2016.

where **F** and **G** are conformable matrices which are functions of the structural matrices **A**, **B**, **D** and **E**. As in Binder and Peseran (1995), **F** is solved from the quadratic expression:

$$\mathbf{F} = [\mathbf{A} - \mathbf{DF}]^{-1} \mathbf{B}.$$

With **F** in hand, we compute $\mathbf{G} = [\mathbf{A} - \mathbf{DF}]^{-1} \mathbf{E}$. The reduced form can be found by other solution concepts like the method of Sims (2002).

2 Dynare

In Dynare `.mod` files, a model's structural equations are written explicitly. For example, consider a three equation New Keynesian model with the following structural equations:

$$\hat{x}_t = E_t\hat{x}_{t+1} - (\hat{r}_t - E_t\hat{\pi}_{t+1})$$

$$\hat{\pi}_t = \beta E_t\hat{\pi}_{t+1} + \psi\hat{x}_t$$

$$\hat{r}_t = \hat{r}_{t-1} + \rho_\pi\hat{\pi}_t + \rho_g\hat{g}_t + \rho_x\hat{x}_t.$$

The linear equations block of the Dynare `.mod` file reads:

```
model (linear) ;
    x  = x(+1) - ( r - pi(+1) ) ;
    pi = beta * pi(+1) + psi * x ;
    r  = r(-1) + rho_pi * pi + rho_g * g + rho_x * x ;
end ;
```

Dynare outputs **F** and **G** and stores the result in the structure `M_` (see my Dynare notes for more details). However, it does not give the structural matrices **A**, **B**, **D** and **E** directly. In many cases it is useful to have these in hand.

3 The `dyn_to_str` function

We have to rebuild the structural matrices from the Dynare output. In particular, we use the Jacobian outputted by Dynare. Why the Jacobian? From the Dynare manual:

> When computing the Jacobian of the dynamic model, the order of the endogenous variables in the columns is stored in M_.lead_lag_incidence. The rows of this matrix represent time periods: the first row denotes a lagged (time t-1) variable, the second row a contemporaneous (time t) variable, and the third row a leaded (time t+1) variable. The columns of the matrix represent the endogenous variables in their order of

declaration. A zero in the matrix means that this endogenous does not appear in the model in this time period. The value in the M_.lead_lag_incidence matrix corresponds to the column of that variable in the Jacobian of the dynamic model. Example: Let the second declared variable be c and the (3,2) entry of M_.lead_lag_incidence be 15. Then the 15th column of the Jacobian is the derivative with respect to c(+1).

The function `dyn_to_str` also computes the reduced form matrices from the extracted structural matrices. They should be the same as the Matlab solution. Comparing the reduced form matrices of `dyn_to_str` and the Dynare reduced form matrices:

```
F(:,oo_.dr.state_var) is equivalent to oo_.dr.ghx(oo_.dr.inv_order_var,:)
G is equivalent to G2 where
    - G2=oo_.dr.ghu*sqrt(M_.Sigma_e) ;
    - G2(oo_.dr.inv_order_var,:) ;
```

4 Illustrating with ZLB algorithm

I show here why it is useful to have the structural matrices in an example application to the ZLB. It is taken from the paper 'Unanticipated Shocks and Forward Guidance at the Zero Lower Bound', available at my website `wp.nyu.edu/callum`.

4.1 The algorithm

The steps of the algorithm are:

0. Linearize the model around the non-stochastic steady state, ignoring the ZLB.
1. For each period t:
   (a) Solve for the path $\{x_\tau\}_{\tau=t}^{T}$ with T large, using the solution of the linearized economy from step (0), given w_t and the initial vector of variables x_{t-1}, and assuming no future uncertainty. This gives a path for the nominal interest rate, $\mathbf{i}_t^k = \{i_\tau^k\}_{\tau=t}^{T}$.
   (b) Examine the path $\mathbf{i}_t^k$. If $\mathbf{i}_t^k \geq 0$, then the ZLB does not bind, and move to step (2). If $\mathbf{i}_t^k < 0$, then move to step (1c).
   (c) For the *first* time period where $\mathbf{i}_t^k < 0$, set the nominal interest rate in that period to zero. This changes the anticipated structure of the economy. Under this new structure, calculate the path of all variables, including the new path for the nominal interest rate $\mathbf{i}_t^{k+1} = \{i_\tau^{k+1}\}_{\tau=t}^{T}$.

   Iterate on steps 2 and 3 until convergence of $\mathbf{i}_t^{k+1}$ and $\mathbf{i}_t^k$.
2. Increment t. The initial vector of variables becomes x_t, which was solved for in step 1. Draw a new vector of unanticipated shocks w_{t+1} and return to step 1.

4.2 Details of each step

At the following steps:

0. Write the n equations of the linearized structural model at t as:

$$\mathbf{A}x_t = \mathbf{C} + \mathbf{B}x_{t-1} + \mathbf{D}\mathbb{E}_t x_{t+1} + \mathbf{E}w_t, \tag{SM}$$

where x_t is a $n \times 1$ vector of state and jump variables and w_t is a $l \times 1$ vector of exogenous variables. Use standard methods to obtain the reduced form:

$$x_t = \mathbf{J} + \mathbf{F}x_{t-1} + \mathbf{G}w_t. \tag{RF}$$

1. For each period t:

   (a) Using (RF), obtain the path $\{x_\tau\}_{\tau=t}^{T}$ given w_t. Set T to be large. Assume $\{w_\tau\}_{\tau=t+1}^{T} = 0$ (no future uncertainty), so that:

$$\begin{aligned} x_t &= \mathbf{J} + \mathbf{F}x_{t-1} + \mathbf{G}w_t \\ x_{t+1} &= \mathbf{J} + \mathbf{F}x_t \\ &\vdots \\ x_T &= \mathbf{J} + \mathbf{F}x_{T-1} \end{aligned}$$

   This step gives a path $\mathbf{i}_t = \{i_\tau\}_{\tau=t}^{T}$.

   (b) Examine the path $\{i_\tau\}_{\tau=t}^{T}$

   - if $i_\tau \geq 0$ for all $t \leq \tau < T$, accept $\{x_\tau\}_{\tau=t}^{T}$. The i_t path does not violate ZLB today or in future.
   - if $i_\tau < 0$ for any $t \leq \tau < T$, move to step (1c).

   (c) Update the path of $\{i_\tau\}_{\tau=t}^{T}$ for the ZLB. For the first time period t^* where $i_{t^*} < 0$, set $i_{t^*} = 0$. The model system at t^* therefore becomes:

$$\mathbf{A}^* x_{t^*} = \mathbf{C}^* + \mathbf{B}^* x_{t^*-1} + \mathbf{D}^* \mathbb{E}_{t^*} x_{t^*+1} + \mathbf{E}^* w_{t^*},$$

   Compute the new path $\{i_\tau\}_{\tau=t}^{T}$. This involves computing $\{x_\tau\}_{\tau=t}^{t^*}$ and $\{x_\tau\}_{\tau=t^*+1}^{T}$. At t^*, $\mathbb{E}_{t^*} x_{t^*+1}$ is computed using the the reduced form solution (RF) and $w_{t^*+1} = 0$. This expresses x_{t^*} as a function of x_{t^*-1}. Proceeding in this way with the correct structural matrices (either ZLB $*$ or no ZLB at each time period), compute the path $\{i_\tau\}_{\tau=t}^{T}$.

   A convenient way to compute the new path $\{i_\tau\}_{\tau=t}^{T}$ is to form the time varying matrices

$\{\mathbf{J}_\tau, \mathbf{F}_\tau, \mathbf{G}_\tau\}_{\tau=t}^{T}$ which satisfy the recursion:

$$\begin{aligned}
\mathbf{F}_t &= [\mathbf{A}_t - \mathbf{D}_t\mathbf{F}_{t+1}]^{-1}\mathbf{B}_t \\
\mathbf{J}_t &= [\mathbf{A}_t - \mathbf{D}_t\mathbf{F}_{t+1}]^{-1}(\mathbf{C}_t + \mathbf{D}_t\mathbf{J}_{t+1}) \\
\mathbf{G}_t &= [\mathbf{A}_t - \mathbf{D}_t\mathbf{F}_{t+1}]^{-1}\mathbf{E}_t,
\end{aligned}$$

with the final set of reduced form matrices for the recursion being the non-ZLB matrices **J**, **F**, **G** from (RF).

These time-varying matrices are then used to compute the path by calculating $x_\tau = \mathbf{J}_\tau + \mathbf{F}_\tau x_{\tau-1} + \mathbf{G}_\tau w_\tau$. For more details on this particular recursion, see Kulish and Pagan (2012).

4.3 Code implementing method

The first thing to do is run Dynare on the model without the ZLB.

```
model_name = 'modelQ' ;
eval(['dynare ', model_name, '.mod noclearall']) ;
```

Next, extract the structural matrices using `dyn_to_str` and store the output in the structure `SET`.

```
dyn_in.M_  = M_ ;
dyn_in.oo_ = oo_ ;

out = dyn_to_str(dyn_in) ;

SET.mats.Q = out.mats.Q ; Q = SET.mats.Q ;
SET.mats.G = out.mats.G ; G = SET.mats.G ;

SET.str_mats.A = out.mats.A ;
SET.str_mats.B = out.mats.B ;
SET.str_mats.D = out.mats.D ;
SET.str_mats.E = out.mats.E ;
SET.str_mats.D2 = out.mats.D2 ;
SET.str_mats.Gamma = out.mats.Gamma ;
```

Next, we need to create the starred structural matrices corresponding to the ZLB regime. The first set of code sets the ZLB algorithm's parameters.

```
SET.pos_of_i = SET.variable.i ; % Position of interest rate in .mod
SET.tr_row   = 9 ;              % Row of policy rule in .mod
SET.maxchk   = 200 ;            % Check ZLB binding out 200 periods
SET.stoch    = 1 ;              % Shocks each period? For IRF, same as 0
SET.zlb_val  = -log(M_.params(SET.param_names.RSS)) ; % i at ZLB
```

Next, we construct the starred matrices by manipulating the equation corresponding to the Taylor rule.

```
mat_i_f_zlb = SET.str_mats ;

mat_i_f_zlb.A(SET.tr_row,:) = 0 ;
mat_i_f_zlb.B(SET.tr_row,:) = 0 ;
mat_i_f_zlb.D(SET.tr_row,:) = 0 ;
mat_i_f_zlb.A(SET.tr_row,SET.pos_of_i) = 1 ;
mat_i_f_zlb.A(SET.tr_row,end) = -SET.zlb_val ; % Constant
```

Then we store the ZLB starred and no-ZLB non-starred regimes in SET for useful input into the ZLB function.

```
SET.mat_init    = SET.str_mats ; % Structural matrices not at ZLB
SET.mat_i_f_zlb = mat_i_f_zlb ;  % Structural matrices at ZLB
SET.mat_fin     = SET.str_mats ; % Structural matrices after ZLB
```

In running the ZLB algorithm, we simply initialize the first period vector of endogenous variables to the steady-state values, sample some shocks and compute the path given those shocks.

```
for t = 2:SET.horizon
    [y_zlb(:,t), zlb_f(t), z_f_T(t)] = zlb(SET, y_zlb(:,t-1), et(:,t)) ;
end
```

The inputs to the `zlb` function include last period's state vector, the vector of shocks and the SET structure, which contains the structural matrices computed above.Figure 1 plots output of the method for endogenous variables across demand shocks.

A description of `zlb` is left to a complementary set of notes, but the details here illustrate how the structural matrices can be used in practice.

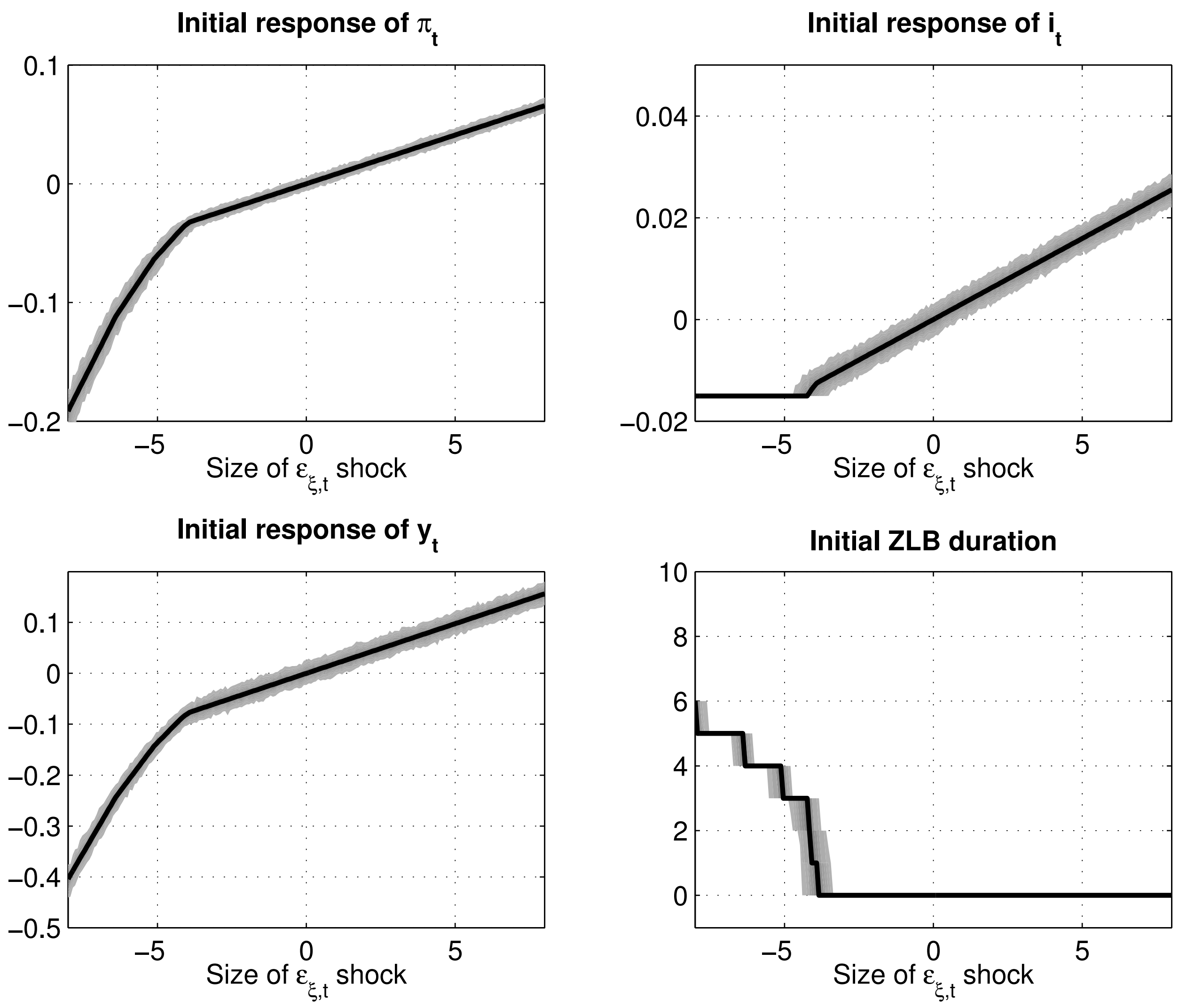

Figure 1: **Initial values of variables across the size of the demand shock**