Materials 12b - A general solution method for the learning model

-

The goal is to have a flexible, automated method so one can tweak the model and avoid errors.
See Notes 11 Dec 2019

Laura Gáti

December 11, 2019

1 Model equations and goal

$$x_t = -\sigma i_t + \hat{\mathbb{E}}_t \sum_{T=t}^{\infty} \beta^{T-t} \big((1-\beta)x_{T+1} - \sigma(\beta i_{T+1} - \pi_{T+1}) + \sigma r_T^n\big) \tag{1}$$

$$\pi_t = \kappa x_t + \hat{\mathbb{E}}_t \sum_{T=t}^{\infty} (\alpha\beta)^{T-t} \big(\kappa\alpha\beta x_{T+1} + (1-\alpha)\beta\pi_{T+1} + u_T\big) \tag{2}$$

$$i_t = \psi_\pi \pi_t + \psi_x x_t + \rho i_{t-1} + \bar{i}_t \tag{3}$$

Goal: obtain endogenous stuff as a function of expectations and states:

$$z_t = \begin{bmatrix} \pi_t \\ x_t \\ i_t \end{bmatrix} = A_a f_a + A_b f_b + A_s s_t \tag{4}$$

where I already have expectations f_a, f_b and the state vector can vary by model, but in this default case it is

$$s_t = \begin{bmatrix} r_t^n \\ \bar{i}_t \\ u_t \\ i_{t-1} \end{bmatrix} \tag{5}$$

That is, we want the matrices A_a, A_b and A_s.

2 Step 1 - introduce LH expectations for observables and states

$$x_t = -\sigma i_t + \begin{bmatrix}\sigma, & 1-\beta, & -\sigma\beta\end{bmatrix} f_b + \sigma \begin{bmatrix}1 & 0 & 0 & 0\end{bmatrix} (I_{nx} - \beta h_x)^{-1} s_t \tag{6}$$

$$\pi_t = \kappa x_t + \begin{bmatrix}(1-\alpha)\beta, & \kappa\alpha\beta, & 0\end{bmatrix} f_a + \begin{bmatrix}0 & 0 & 1 & 0\end{bmatrix} (I_{nx} - \alpha\beta h_x)^{-1} s_t \tag{7}$$

$$i_t = \psi_\pi \pi_t + \psi_x x_t + \begin{bmatrix}0 & 1 & 0 & \rho\end{bmatrix} s_t \tag{8}$$

$$\Leftrightarrow$$

$$x_t = -\sigma i_t + c_{x,b} f_b + c_{x,s} s_t \tag{9}$$

$$\pi_t = \kappa x_t + c_{\pi,a} f_a + c_{\pi,s} s_t \tag{10}$$

$$i_t = \psi_\pi \pi_t + \psi_x x_t + c_{i,s} s_t \tag{11}$$

where

$$c_{x,b} = \begin{bmatrix}\sigma, & 1-\beta, & -\sigma\beta\end{bmatrix} \tag{12}$$

$$c_{x,s} = \sigma \begin{bmatrix}1 & 0 & 0 & 0\end{bmatrix} (I_{nx} - \beta h_x)^{-1} \tag{13}$$

$$c_{\pi,a} = \begin{bmatrix}(1-\alpha)\beta, & \kappa\alpha\beta, & 0\end{bmatrix} \tag{14}$$

$$c_{\pi,s} = \begin{bmatrix}0 & 0 & 1 & 0\end{bmatrix} (I_{nx} - \alpha\beta h_x)^{-1} \tag{15}$$

$$c_{i,s} = \begin{bmatrix}0 & 1 & 0 & \rho\end{bmatrix}$$

3 Step 2 - solve for observables as a function of cs, expectations and states

Now the cool thing is you can unleash Mathematica (`materials12.nb`, bottom) to solve (9)-(11) as a function of the c-matrices. The result will take the following form:

$$x_t = g_{x,a} f_a + g_{x,b} f_b + g_{x,s} s_t \tag{16}$$

$$\pi_t = g_{\pi,a} f_a + g_{\pi,b} f_b + g_{\pi,s} s_t \tag{17}$$

$$i_t = g_{i,a} f_a + g_{i,b} f_b + g_{i,s} s_t \tag{18}$$

$$\tag{19}$$

where Mathematica will output the g-matrices, which can be copied directly into Matlab (`matrices_A_intrate_smoothing2.m`).

The A-matrices will just be a stacking of the g-vectors:

$$\underbrace{A_a}_{ny \times ny} = \begin{pmatrix} g_{\pi,a} \\ g_{x,a} \\ g_{i,a} \end{pmatrix} \quad \underbrace{A_b}_{ny \times ny} = \begin{pmatrix} g_{\pi,b} \\ g_{x,b} \\ g_{i,b} \end{pmatrix} \quad \underbrace{A_s}_{ny \times nx} = \begin{pmatrix} g_{\pi,s} \\ g_{x,s} \\ g_{i,s} \end{pmatrix} \tag{20}$$