ADAPTIVE LEARNING IN PRACTICE: TOOLBOX MANUAL*

Eva Carceles-Poveda, SUNY Stony Brook†
Chryssi Giannitsarou, University of Cambridge and CEPR‡

March 9, 2006

*We thank Seppo Honkapohja for helpful suggestions. For comments, we are grateful to conference and seminar participants of SITE 2004, CEF 2004, EEA-ESEM 2004 and Heriot-Watt University. This project has been funded by the ESRC award RES-000-23-1152.

†Department of Economics, State University of New York, Stony Brook, NY-11733-4384, USA. *E-mail:* ecarcelespov@notes.cc.sunysb.edu. *Web:* http://ms.cc.sunysb.edu/~ecarcelespov.

‡Faculty of Economics, Austin Robinson Building, Sidgwick Avenue, Cambridge, CB3 9DD *E-mail*: cg349@cam.ac.uk. *Web*: http://www.econ.cam.ac.uk/faculty/Giannitsarou/index.html

1. DESCRIPTION OF THE TOOLBOX

The present manual provides a detailed description of the computing toolbox that accompanies the paper for simulating adaptive learning in the context of forward looking reduced form models. The general procedure for implementing adaptive learning is summarized in table 1. Apart from the files mentioned in the table, several other files are contained in the toolbox. These are tmap.m, vmap.m, errors1.txt, get_RGD.m, get_SIGMA.m, learning_ls_rep.m and learning_sg_rep.m. The first two are Matlab functions that compute the T and V maps defined in section 4 of the paper; the third contains the default realizations of the shocks for generating the initial values; the file get_RGD.m, generates data for the initial training period if the RGD initialization method is used; the file get_SIGMA.m recovers the asymptotic covariance matrix of the OLS estimator if the DIS initialization method is used; the last two are used to calculate average statistics with many replications. Next, we give a detailed description of each step.

Step 1. Derivation of the reduced form and $\gamma-$coefficients. The first step of the general procedure involves the calculation of the reduced form coefficients, the steady state, and the coefficients of the laws of motion for the endogenous variables that are different from the endogenous state k_t. While this can be done with paper and pencil, we provide model specific files implementing the necessary calculations for our examples using Mathematica.

The derivation of the reduced form coefficients a_1, a_2 and b for all the examples is explained in detail in section two of the paper. In essence, the implementation of this step using Mathematica involves (i) declaring the system of steady state and log-linearized equations, (ii) substituting the relevant constraints in the expectational equation that determines the law of motion of k_t to eliminate all the variables except from k and z, and (iii) collecting the terms to obtain the expressions for the coefficients a_1, a_2 and b that satisfy the reduced form. Note that $a_2 = 0$ for the purely forward looking reduced form.

Recall that if q_t is an endogenous variable of interest, its log-linear law of motion can be written as

$$q_t = \gamma_{q1} k_t + \gamma_{q2} k_{t-1} + \gamma_{q3} z_t + \gamma_{q4} E_t k_{t+1}$$

where $\gamma_{q2} = 0$ for the purely forward looking reduced form. To obtain expressions for these coefficients using Mathematica, we simply have to (iv) substitute the relevant constraints in the log-linearized equation that determines each endogenous variable q_t to eliminate all the variables except from k and z, and (v) collect the terms to obtain expressions for the γ-coefficients that satisfy the previous law of motion.

Step	Description	File
1.	Get: a. Reduced form coefficients b. Steady state values c. Coefficients γ of ALM for all the variables	(example_x.nb)
2.	Main code: Declare: a. variable names b. shock parameters c. steady states and initial x d. a_1, a_2, b. Then call steps 3-5 and 7-9	(example_x.m)
3.	Find how many endogenous and exogenous states there are	(get_STATES.m)
4.	Find REE, check stationarity and E-stability	(solution.m)
5.	Define learning options: a. Default or random shock b. Number of periods c. Plot of ϕ d. Projection Facility d. Learning algorithm: RLS, SG, CG e. Initialization: RGD, AH, DIS Then call step 6	(learning.m)
6.	Do learning and get laws of motion for (k_t, z_t)	(learning_ls.m or learning_sg.m)
7.	Load $\gamma-$coefficients and get laws of motion with AL and RE	(gammasx.txt) (laws_re.m) (laws_al.m)
7.	Do plots for all variables	(do_plots.m)
8.	Get statistics for all variables	(do_statistics.m)

Table 1: Procedure for Numerical Implementation.

Finally, once the expressions for the reduced form and the γ-coefficients have been obtained, we have to (vi) define the parameter values and (vii) calculate the steady state values. Using these, we can (viii) get the numerical values for a_1, a_2, b and γ that will be inserted directly in the main Matlab code corresponding to step 2.

The codes implementing steps (i)-(viii) for our examples are example_1.nb, example_2.nb and example_3.nb. In the first example, the endogenous variables of interest are output y_t and consumption c_t. Furthermore, the second example also includes the two factor prices r_t and w_t, the hours worked n_t, and the government expenditures g_t.

Step 2. Main Matlab code. A main program calling the necessary routines to implement the different steps discussed below is provided for the examples discussed in the text under the name example_x.m for $x = \{1, 2, 3\}$. The first part of the program, which is the one that needs to be modified by the user, declares (i) the names of the endogenous variables of interest starting with the endogenous state variable k_t, (ii) the names for the state variables starting from the exogenous state variable x_t, (iii) the two parameters of the autoregressive shock ρ and σ, (iv) the steady state values of the endogenous variables of interest following the same order as the variable names, (v) the initial condition for the vector x_t, and (vi) the coefficients of the reduced form. Moreover, the main code loads the txt-file with coefficients γ for the rest of the endogenous variables. In addition, the second part of the program calls the routines implementing steps 3-5 and 7-9, which are explained below.

Step 3. Find how many endogenous and exogenous states there are. After declaring the names for the endogenous variables of interest and the state variables, the routine get_STATES.m finds how many endogenous and exogenous state variables there are. This will be used to identify if the reduced form of the model has a lag or if it is purely forward looking.

Step 4. Find the REE solution. Given the reduced form coefficients, step three involves calculating the REE solutions $\bar{\phi} = (\bar{\phi}_k, \bar{\phi}_z)$ using the in section three of the paper, and then determining a unique stationary and E-stable REE solution of the model.

This is implemented by the routine solution.m. For the reduced form with the lag, the program calculates the two MSV solutions and determines their stationarity by checking whether $\left|\bar{\phi}_k^i\right| < 1$ for $i = 1, 2$. If none of the two solutions is stationary, the programme exits. If there is a unique stationary solution, the programme checks if it is E-stable using the stability conditions and it exits otherwise. Furthermore, if there are two stationary MSV solutions, the programme selects the one that is E-stable. If both or none are E-stable, the programme exits. For the purely forward looking reduced form, the program calculates the unique solution and checks if it is E-stable using the stability conditions and it exists otherwise.

At the end of the code, the selected REE and its second moment matrix are declared for further use. It can be verified that, for all our examples, there exists a unique stationary solution that satisfies the E-stability conditions for all reasonable parameter values.

Steps 5-6. Learning. The next step involves simulating the evolution of x_t under rational expectations and learning. For a general model that satisfies our reduced forms, this is implemented by calling the file learning.m, where the user is asked to choose from several options. In particular, the user can choose (a) to use the default realizations of shocks (important for comparisons when doing sensitivity analysis) or a random realization, (b) the number of periods T for the simulations, (c) to make plots of the evolution of the learning coefficients ϕ_t, (d) to impose the projection facility for the reduced form with the lag, (e) the learning algorithm and (f) the method for initializing the algorithm.

Once the main options are set, the program calls the Matlab file learning_ls.m or learning_sg.m, which simulate x_t under rational expectations and under learning. The file learning_ls.m simulates x_t under RLS and CG-RLS, whereas learning_sg.m simulates x_t under SG and CG-SG.

If RLS or CG-RLS are selected, the first part of learning_ls.m determines the initial conditions as described in section 5 of the paper, depending on the selection of the user, who can decide to initialize with RGD, AH or DIS methods. If the first method is selected, the programme finds the minimum t_0 for which the both the invertibility condition of R_{t_0} and the stationarity condition of ϕ_{t_0} are satisfied, and then gives the user the option to increase t_0 as desired. The file get_RDG.m generates the data for the training period. If SG or CG-SG are selected, the initialization method AH is used.

With the initial conditions set, the program simulates the actual law of motion for x_t. Finally, when the code learning_ls.m (or learning_sg.m) is completed, the programme returns to learning.m and reports how many times the projection facility has been activated. If selected, it shows in which periods it happened.

Step 7. Laws of motion. The next step involves obtaining simulated time series for the other endogenous variables of interest. This is implemented by calling the Matlab file laws_al.m and laws_re.m, which calculate time series for the endogenous variables under adaptive learning and rational expectations respectively. Before calling the programs, we have to load txt-files containing the $\gamma-$elasticities of the endogenous variables calculated as described in step 1.

Assume first that the log-linearized equation that determines q_t does not contain expectational terms, in which case we can use the coefficients γ_{q_1}, γ_{q_2} and γ_{q_3} of q_t with respect to

k_t, k_{t-1} and z_t calculated in in step 1. In particular, using the fact that $T_2(\phi_{t-1}) = \rho V(\phi_{t-1})$, the actual law of motion of k_t is given by

$$k_t = T_1(\phi_{t-1})k_{t-1} + V(\phi_{t-1})z_t$$

For the reduced form with a lag, the law of motion of q_t can be rewritten as

$$q_t^{AL} = [\gamma_{q_1} T_1(\phi_{t-1}) + \gamma_{q_2}]k_{t-1} + [\gamma_{q_1} V(\phi_{t-1}) + \gamma_{q_3}]z_t$$

This is the equation used by the program laws_al.m to calculate the laws of motion under learning. Similarly, the program laws_re.m uses the same law of motion with the RE elasticities $\bar{\phi}$, i.e.

$$y_t^{RE} = [\gamma_{q_1} T_1(\bar{\phi}) + \gamma_{q_2}]k_{t-1} + [\gamma_{q_1} V(\bar{\phi}) + \gamma_{q_3}]z_t$$

If a log-linearized equation contains a lead, we can then also substitute for

$$\begin{aligned} E_t k_{t+1} &= \phi_{k,t-1} k_t + \phi_{z,t-1} z_t \\ &= \phi_{k,t-1} \left[T_1(\phi_{t-1})k_{t-1} + V(\phi_{t-1})z_t \right] + \phi_{z,t-1} z_t \end{aligned}$$

and obtain

$$\begin{aligned} q_t &= \left[\gamma_1 T_1(\phi_{t-1}) + \gamma_2 + \gamma_4 \phi_{k,t-1} T_1(\phi_{t-1}) \right] k_{t-1} \\ &\quad + \left[\gamma_1 V(\phi_{t-1}) + \gamma_3 + \gamma_4 \phi_{k,t-1} V(\phi_{t-1}) + \gamma_4 \phi_{z,t-1} \right] z_t \end{aligned}$$

Similar equations can be obtained for the purely forward looking reduced form.

Steps 8-9. Plots and Statistics. The last two steps involve plotting the time series and calculating statistics for the simulated variables. This is implemented by calling the Matlab files do_plots.m and do_statistics.m. In the figures, the series generated by RE are always depicted by a dotted line, while the ones generated using learning are depicted by a solid line. The code statistics.m calculates relative deviations and correlations. The user has the option of calculating HP filtered statistics, or statistics for the variables in logs or levels. Further, the user is also given the option of calculating the statistics for one or many replications. If many replications are selected, the files learning_ls_rep.m and learning_sg_rep.m are used.