

Learning to Optimize

George W. Evans
University of Oregon

Bruce McGough
Oregon State University

March 1, 2009

Abstract

How does a boundedly rational optimizing agent make decisions? Can such an agent learn to behave optimally? We address these questions in a standard regulator environment. Our behavioral primitive is anchored to the shadow price of the state vector. The regulator forecasts the value of an additional unit of the state tomorrow, and uses this forecast to choose her control. The value of the control, together with the agent's forecast of the tomorrow's shadow price, are then used to compute a proxy for the unobserved shadow price of today's state vector. This proxy provides a new data point which the agent uses to update her forecasting model. We find conditions sufficient to guarantee that, over time, the regulator learns to optimize. We also embed this type of boundedly rational optimizing behavior in a simple DSGE model and compare the results with those obtained from embedding other learning mechanisms.

JEL Classifications: E52; E31; D83; D84

Key Words: Learning, Optimization, Bellman Systems

Preliminary

1 Introduction

A central paradigm of modern macroeconomics is the need for micro-foundations. Macroeconomists construct their models by aggregating the behavior of individual agents who are assumed “rational” in two important ways: they form forecasts optimally; and, given these forecasts, they make choices by maximizing their objective. Together with simple market structures and, sometimes, institutional frictions, it is this notion of rationality that identifies a micro-founded model.

While assuming rationality is at the heart of much economic theory, the implicit sophistication required of agents, both as forecasters and as decision theorists, is substantial: they must be able to form expectations conditional on the true distributions of the endogenous variables in the economy; and they must be able to make choices – i.e. solve infinite horizon programming problems – given these expectations. The criticism that the ability to make optimal forecasts requires an unrealistic level of sophistication has been leveled repeatedly; and, in response to this criticism, the literature on “bounded rationality” was developed. Boundedly rational agents are not assumed to know the true distributions of the endogenous variables; instead, they have forecasting models that they use to form expectations. These agents update their forecasting models as new data become available, and through this updating process, the dynamics of the associated economy can be explored. In particular, the asymptotic behavior of the economy can be analyzed, and if the economy converges in some natural sense to a rational expectations equilibrium, then we may conclude that agents in the economy are able to learn to forecast optimally. In this way, the learning literature has provided a response to the criticism that rational expectations is unrealistic. The original work on learning in macroeconomics was done by Bray (1982), Bray and Savin (1986); for a comprehensive treatment, see Evans and Honkapohja (2001).

While the learning literature focuses primarily on whether by using estimated models, agents can learn to be forecast optimally, the literature has largely neglected the link between forecasts and agent level decision making. Instead, the standard procedure as applied to a DSGE model is as follows: assume rationality on the part of agents when obtaining conditions capturing optimizing behavior; aggregate and impose market conditions; simplify the associated sequence of equilibrium restrictions; and linearize the restrictions to obtain a reduced form system of linear expectational difference equations. Only then is bounded rationality imposed by assuming the expectations operator in the expectational difference equations is no longer the rational expectations operator, but instead it is an operator capturing the boundedly rational forecasting behavior of the agents. Notice the disconnect between the derivation of the reduced form system and the imposition of bounded rationality: agents are assumed to behave as if they were optimal forecasters and optimal decision makers when deriving the reduced form system, and only then are they assumed boundedly rational. We call this type of learning implementation “reduced form learning.”

We address this disconnect by simply reversing the order of analysis: we assume agents are boundedly rational before aggregation and market clearing are imposed, that is, before the reduced form equations are obtained. This order reversal exposes an additional subtlety involved when incorporating bounded rationality at the micro-level: boundedly rational agents are assumed to have limited sophistication as forecasters, but are assumed to be highly sophisticated as decision makers; in particular, they are assumed to understand how, given their beliefs, to solve stochastic dynamic

programming problems. We find this discontinuity in sophistication unsatisfactory, particularly because it is internal to the agents.

To address this discontinuity we define the notion of *bounded optimality*. We imagine our agents facing a sequence of decision problems in an uncertain environment: not only is there uncertainty in that the environment is inherently stochastic, but also, our agents do not fully understand the conditional distributions of the variables requiring forecasts. One option when modeling agent decisions in this type of environment is to assume agents are Bayesian; and, given their priors, that they are able to fully solve their dynamic programming problems; however, we feel this level of sophistication is extreme, and instead, we prefer to model our agents as relying on decidedly simpler behavior. Informally, we assume that each day our agents act as if they face a two period optimization problem: they think of the first period as “today” and the second period as “the future,” and use one period forecasts of shadow prices to measure the trade-off between choices today and the impact of these choices on the future. We call our implementation of bounded optimality *shadow price learning* (SP-learning).

Our notion of bounded optimality is inexorably linked to bounded rationality: agents in our economy as not assumed to fully understand the conditional distributions of the economy’s variables, or, in the context of an individual’s optimization problem, the conditional distributions of the state variables. Instead, consistent with the learning literature, we provide our agents with forecasting models, which they re-estimate as new data become available. Our agents use these estimated models to make one period forecasts, and then use these one period forecasts to make control decisions.

We find our learning mechanism appealing for a number of reasons: it requires only simple econometric modeling and thus is consistent with the learning literature; it assumes agents make only one period ahead forecasts instead of establishing priors over the distributions of all future endogenous variables; and it imposes only that agents make decisions based on these one period ahead forecasts, rather than requiring agents to solve a dynamic programming problem with parameter uncertainty. Finally, notice SP-learning postulates that, fundamentally, agents make decisions by facing prices – a hallmark of economics.

We begin our analysis of bounded optimality, in association with bounded rationality, by first addressing whether the behavior makes sense from an agent’s perspective. To this end, we back off the DSGE environment, and instead consider a standard quadratic regulator’s problem. After recalling the derivation of the optimal policy rule, we carefully specify the behavioral primitives of our boundedly optimal regulator. We then show, under quite general conditions, that the policy rule employed by our boundedly optimal regulator converges to the optimal policy rule: following our simple behavioral primitives, our regulator learns to optimize.

We are not the first to carefully consider the link between agents' forecasts and their associated decisions; as mentioned above, Bayesians take precisely this approach, though with a high level of imposed sophistication. Further, researchers in the learning literature have noted the limitations of imposing bounded rationality directly on the reduced form system of a DSGE model. Evans and Honkapohja (2006) introduce Euler equation learning to address this concern: these authors take the Euler equation of a representative agent as the behavioral primitive and assume agents make decisions based on the boundedly rational forecasts required by the Euler equation.¹ They find, in a variety of models, that Euler equation learning imparts the same implied dynamics as reduced form learning, and, in this way, Euler equation learning can be used to justify reduced form learning.

There is a close connection between Euler equation learning and SP-learning, and we explore this connection in detail in Section 3. We define an n^{th} -order Euler equation system as a sequence of first order conditions establishing relationships between current variables and their forecasts n -periods in the future. We then obtain two simple sets of well-known conditions establishing the existence of first order Euler equation systems. Turning to bounded optimality, we show that if either set of conditions is satisfied, then Euler equation learning and SP-learning are generically equivalent; however, when higher order Euler equations are required to capture the programming problem's first order conditions, SP-learning and Euler equation learning may well differ. The intuition for their difference is quite simple: Euler equation learning requires fewer perceived parameters in the agent's forecasting rule, and therefore, demands that the agent has more structural knowledge: in order to behave like an Euler equation learner, the an SP-learner must understand the relationships between the states' shadow prices, and further, he must use these relationships to impose restrictions on his perceived parameters.

Having found that bounded optimality is reasonable from an agent's perspective, in that he can expect to eventually behave optimally, we embed shadow price learning into a simple DSGE model consistent with our quadratic regulator environment. We consider a Robinson Crusoe economy, with quadratic preferences and linear technology: see Hansen and Sargent (2005) for many examples of these types of economies, including the model we use. Our only innovation is to include a double lag in production, thus providing the simplest possible example of a multivariate model for which no first order Euler equation system exist. We use the Crusoe economy to walk carefully through the boundedly optimal behavior displayed by our agent, thus providing examples of, and intuition for the behavioral assumptions made in Section 2.1. Also, exploiting the double-lagged production function, we compute convergence rates under SP-learning and Euler equation learning, and find, as predicted, that they are equivalent only in case the second lag coefficient vanishes.

¹See also Evans, Honkapohja, and Mitra (2003).

The Crusoe economy that we investigate is special not only because we impose a linear-quadratic environment, but also because there is only one agent. The application of SP-learning to a generic linearized DSGE model raises a number of interesting and subtle issues including working with linearized behavioral conditions and carefully distinguishing between individual and aggregate variables. In a companion paper, Evans and McGough (2009), we modify shadow price learning to apply to linearized DSGE models, and, within the context of an RBC model with habit persistence, we carefully address these and other issues. Further, in that paper, we explore in detail the relationships between SP-learning, Euler equation learning (and its variants) and infinite horizon learning, which is another agent-level learning mechanism, emphasized by Preston (2005).

This paper is organized as follows: in Section 2 we investigate the regulator’s problem. We specify the behavioral primitives of bounded optimality and provide conditions under which a boundedly optimal regulator will learn to optimize. Section 3 provides a general comparison of SP-learning and Euler equation learning. Section 4 introduces shadow price learning into a simple DSGE model; we characterize individual agents’ decisions as based on their boundedly rational forecasts, and analyze associated learning dynamics. Section 5 concludes, and all proofs are in the Appendix, Section 6.

2 Learning to optimize

We begin by specifying the programming problem of interest. We focus on the behavior of a quadratic regulator facing a linear constraint to exploit certainty equivalence and to allow for parametric analysis.² It is of considerable interest to analyze a general regulator’s problem, and we are currently investigating extending our results to more general settings using non-parametric methods. The specification of our quadratic problem is taken from Hansen and Sargent (2005); for additional reading, we recommend Stokey and Lucas Jr. (1989), and Bertsekas (1987).

2.1 The quadratic regulator

The “sequence problem” is to determine a sequence of controls u_t that solve

$$\begin{aligned} \max \quad & -E_0 \sum \beta^t (x_t' R x_t + u_t' Q u_t + 2x_t' W u_t) \\ \text{s.t.} \quad & x_{t+1} = A x_t + B u_t + C \varepsilon_{t+1}, \end{aligned} \tag{1}$$

with x_0 taken as given. Here ε_t is a zero mean i.i.d. process. Under well-known conditions, which will be discussed in detail below, this problem has a unique solution,

²Also, many applied DSGE models are ultimately linearized.

and the sequence of controls are determined by $u_t = -Fx_t$ for an appropriate matrix F . This matrix may be obtained by analyzing the associated Bellman functional equation:

$$V(x) = \max_u - (x'Rx + u'Qu + 2x'Wu + \beta EV(Ax + Bu + C\varepsilon)).$$

By guessing that V has a quadratic functional form, it is not difficult to show that

$$\begin{aligned} V(x) &= -x'Px - \frac{\beta}{1-\beta} \text{tr}PCC'\sigma^2 \\ F &= -(Q + \beta B'PB)^{-1} (\beta B'PA + W'), \end{aligned} \quad (2)$$

where, again, under appropriate assumptions, P is the unique symmetric positive semi-definite matrix satisfying the Riccati equation

$$P = R + \beta A'PA - (\beta A'PB + W) (Q + \beta B'PB)^{-1} (\beta B'PA + W'). \quad (3)$$

Solving the Riccati equation is not possible analytically; however, a variety of numerical methods are available.

Conditions sufficient to guarantee the problem (1) has a unique solution and that the Riccati equation has a unique positive semi-definite solution are most easily stated by transforming the problem to eliminate the state-control interaction in the objective: see Sargent and Hansen (2005), Chapter 8 for many details. Set $\hat{A} = \beta^{\frac{1}{2}} (A - BQ^{-1}W')$, $\hat{B} = \beta^{\frac{1}{2}} B$, and $DD' = \hat{R}$ where $\hat{R} = R - WQ^{-1}W'$. The problem becomes

$$\begin{aligned} \max \quad & -E_0 \sum \left(\hat{x}'_t \hat{R} \hat{x}_t + u'_t Q u_t \right) \\ \text{s.t.} \quad & \hat{x}_{t+1} = \hat{A} \hat{x}_t + \hat{B} u_t + \beta^{\frac{t+1}{2}} C \varepsilon_{t+1}, \end{aligned} \quad (4)$$

To place restrictions on the matrices identifying the problem (4), a few definitions are needed. Let

- A matrix is stable if its eigenvalues have modulus less than one.
- The matrix pair (\hat{A}, \hat{B}) is stabilizable if there exists a matrix K so that $\hat{A} + \hat{B}K$ is stable.
- A matrix pair (\hat{A}, D) is detectable if the eigenvalues of \hat{A} that are not detectable by D' have modulus smaller than one. Thus if y is a non-zero eigenvector of \hat{A} associated to the eigenvalue λ , and if $D'y = 0$ so that D' doesn't detect this eigenvalue, then $|\lambda| < 1$.

With these definitions at hand, we assume

LQ.1: The matrix \hat{R} is symmetric positive semi-definite and the matrix Q is symmetric positive definite.

LQ.2: The system (\hat{A}, \hat{B}) is stabilizable.

LQ.3: The system (\hat{A}, D) is detectable.

This list represents a standard set of assumptions sufficient to guarantee a well-behaved problem. LQ.1 imparts the appropriate convexity assumptions on the objective and LQ.2 says that it is possible to find a set of controls driving the state to zero in the deterministic problem. The need for LQ.3 is slightly more subtle: if (\hat{A}, D) is detectable then the control path must be chosen to counter dynamics in the explosive eigenspaces of \hat{A} . To illustrate, suppose z is an eigenvector of A with associated eigenvalue λ , suppose that $|\lambda| > 1$, and finally assume that $x_0 = z$. If the control path is not chosen to mitigate the explosive dynamics in the eigenspace associated to λ then the state vector will diverge in norm. Furthermore, because (\hat{A}, D) is detectable, we know that $D'z \neq 0$. Taken together, these observations imply that an explosive state is suboptimal:

$$-x_t' \hat{R} x_t = -\lambda^{2t} z' D D' z = -(|\lambda|^t |D' z|)^2 \rightarrow -\infty.$$

Sargent and Hansen (2005) put it more concisely (and eloquently): If (\hat{A}, \hat{B}) is stabilizable then it is feasible to stabilize the state vector; if (\hat{A}, D) is detectable then it is desirable to stabilize the state vector.

Theorem 1 *Under assumptions LQ.1 – LQ.3, the Riccati equation (3) has a unique positive semi-definite solution, P^* , and iteration of the Riccati equation yields convergence to P^* if initialized at any positive semi-definite matrix P_0 . Also, there is a unique sequence of controls solving (1), and they are given by $u_t = F x_t$, where F is determined by (2).*

2.2 Bounded optimality

For an agent to solve the programming problem (1) as described above, he must understand the quadratic nature of his value function as captured by the matrix P^* , he must know the relationship of this matrix to the Riccati equation, he must be aware that iteration on the Riccati equation provides convergence to P^* , and finally, he must know how to deduce the optimal control path given P^* . Furthermore, this behavior is predicated upon the assumption that he knows the conditional means of the state variables, that is, he knows A and B .

We modify the primitives identifying agent behavior, first by imposing bounded rationality and then by assuming bounded optimality. Our agent is not assumed to

know the state variables' conditional means: he must estimate A and B . Our agent is also not assumed to know how to solve his programming problem: he does not know Theorem 1. Instead, he uses a simple forecasting model to estimate the value of a unit of state tomorrow, and then he uses this forecast, together with his estimate of the transition equation, to determine his control today. Then, he based on his control choice and his forecast of the value of a unit of state tomorrow, he re-estimates the value of a unit of state today. This provides him new data to update his state-value forecasting model.

To facilitate intuition for our learning mechanism, we reconsider the above problem using a Lagrange multiplier formulation. The Lagrangian is given by

$$\mathcal{L} = E_0 \sum \beta^t (-x_t' R x_t - u_t' Q u_t - 2x_t' W u_t + \lambda_t' (A x_{t-1} + B u_{t-1} + C \varepsilon_t - x_t))$$

As usual, λ_t may be interpreted as the shadow price of the state vector x_t along the optimal path. The first order conditions provide

$$\begin{aligned} \mathcal{L}_{x_t} &= 0 \Rightarrow \lambda_t' = -2x_t' R - 2u_t' W' + \beta E_t \lambda_{t+1}' A \\ \mathcal{L}_{u_t} &= 0 \Rightarrow 0 = -2u_t' Q - 2x_t' W + \beta E_t \lambda_{t+1}' B. \end{aligned}$$

Transposing and combining with the transition equation yields the following dynamic system:

$$\lambda_t = -2R x_t - 2W u_t + \beta A' E_t \lambda_{t+1} \tag{5}$$

$$0 = -2W' x_t - 2Q u_t + \beta B' E_t \lambda_{t+1} \tag{6}$$

$$x_{t+1} = A x_t + B u_t + C \varepsilon_t. \tag{7}$$

This system, together with transversality, identifies the unique solution to (1). It also provides intuitive behavioral restrictions on which we base our notion of bounded optimality.

We now marry the assumption from the learning literature that agents make boundedly rational forecasts with a list of behavioral assumptions characterizing the decisions agents make given these forecasts; and, we do so in a manner that we feel imparts a level of sophistication consistent with bounded rationality. Much of the learning literature centers on equilibrium dynamics implied by one-step-ahead boundedly rational forecasts;³ we adopt and expand on this notion by developing assumptions consistent with the following intuition: agents make one-step-ahead forecasts and agents know how to solve a two period optimization problem based on their forecasts. Formalizing this intuition, we make the following assumptions:

1. Agents know their individual instantaneous return function, that is, they know Q , R , and W ;

³There are notable exceptions, including infinite horizon learning: see Preston (2005)

2. Agents know the form of the transition law and estimate the coefficient matrices: let A_t and B_t be the associated estimates;
3. Conditional on their their perceived value of an additional unit of x tomorrow, agents know how to choose their control today;
4. Conditional on their perceived value of an additional unit of x tomorrow, agents know how to compute the value of an additional unit of x today.

Assumption one seems quite natural: if agents are to make informed decisions about a certain vector of quantities u , they should at least be able to understand the direct impact of these decisions. Assumption two is standard in the learning literature: agents are not certain about the evolution of the state vector. Assumptions three and four require some explanation.

Let λ_t^* be the unobserved shadow price of x *along the realized path of x and u* . Do not think of λ^* as directly related to λ ; indeed λ is the vector of shadow prices of x *along the optimal path of x and u* and agents are not (necessarily) interested in this value. Let $\hat{E}_t \lambda_{t+1}^*$ be the agent's time t forecast of the time $t+1$ value of an additional increment of the state x . Assumption three says that given $\hat{E}_t \lambda_{t+1}^*$, agents know how to choose u_t , that is, they know how to solve the associated two period problem. And how is this choice made? The agents simply contemplates an incremental decrease du_i in u_i and equates marginal loss with marginal benefit. If r is the rate function then the marginal loss is $r_{u_i} du_i$. To compute the marginal gain, he must estimate the effect of du_i on the whole state vector tomorrow. This effect is determined by $B_{it} du_i$, where B_{it} is the time t estimate of the i^{th} -column of B . To weigh this effect against the loss obtained in time t , he must then compute its inner product with the expected price vector, and discount. Thus

$$r_{u_i} du_i = \beta \hat{E}_t (\lambda_{it+1}^*)' B_{it} du_i.$$

Stacking, and imposing our linear quadratic set-up gives the bounded rationality equivalent to (6):

$$0 = -2W' x_t - 2Q u_t + \beta B_t' \hat{E}_t \lambda_{t+1}^*. \quad (8)$$

Equation (8) operationalizes assumption 3.

To update their shadow price forecasting model, agents need an estimate of the unobserved value λ_t^* . Let $\hat{E}_t \lambda_t^*$ be agents' time t estimate of λ_t^* . Assumption four says that given $\hat{E}_t \lambda_{t+1}^*$, agents know how to compute $\hat{E}_t \lambda_t^*$. And how is this estimate computed? Agents simply contemplate an incremental increase in x and evaluate the benefit. Again, if r is the rate function then the benefit of dx_i is given by

$$\hat{E}_t \lambda_t^* dx_i = \left(r_{x_i} + \beta \hat{E}_t (\lambda_{it+1}^*)' A_{it} \right) dx_i.$$

Stacking, and imposing our linear quadratic set-up yields the bounded rationality equivalent to (5):

$$\hat{E}_t \lambda_t^* = -2R x_t - 2W u_t + \beta A_t' \hat{E}_t \lambda_{t+1}^*. \quad (9)$$

Equation (9) operationalizes assumption 4.

Assumption 3, as captured by (8), lies at the heart of bounded optimality: it provides that agents make one-step-ahead forecasts of shadow prices and make decisions today based on those forecasted prices, just as they would if solving a two period problem. Assumption 4, as captured by (9), provides the mechanism by which agents assess their forecast of prices: agents use the forecast of prices at time $t + 1$ and their control decision at time t to re-estimate the value of time t state; in this way, our boundedly optimal agents keep track of their forecasting performance. Below, we allow agents to exploit this re-estimation by updating their shadow price forecasting model. We call boundedly optimal behavior, as captured by assumptions 1 – 4, *shadow price learning*.

To close the model we must specify the shadow price forecasting model, that is, the way agents form $\hat{E}_t \lambda_{t+1}^*$. Along the optimal path it is not difficult to show that $\lambda_t = -2P^* x_t$, and so it is quite natural to impose a forecasting model of this functional form form. Therefore, we assume that, at time t agents believe

$$\lambda_t^* = H_t x_t + \mu_t \quad (10)$$

for some $n \times n$ matrix H_t and error term μ_t .

Given the perceived value of H_t , we may use (8) to determine the value chosen by agents for their control:

$$\begin{aligned} u_t &= (2Q - \beta B_t' H_t B_t)^{-1} (\beta B_t' H_t A_t - 2W') x_t \\ &\equiv F(H_t, A_t, B_t) x_t, \end{aligned} \quad (11)$$

where we recall that Q is positive definite and so invertible; the invertibility of $2Q - \beta B_t' H_t B_t$ for appropriately chosen H_t follows from the fact that the fixed point P^* of the Riccati equation is positive semi-definite, so that $Q + \beta B_t' P^* B_t$ is invertible, and H_t will remain near $-2P^*$.

Given the value of H_t and the value of the control chosen by agents, we may use (9) and (10) to compute agents' perceived shadow price of the state, that is, to compute $\hat{E}_t \lambda_t^*$. We find

$$\begin{aligned} \hat{E}_t \lambda_t^* &= (-2R - 2WF(H_t, A_t, B_t) + \beta A_t' H_t (A_t + B_t F(H_t, A_t, B_t))) x_t \\ &\equiv \hat{T}(H_t, A_t, B_t) x_t. \end{aligned} \quad (12)$$

Give the estimates A_t and B_t , and given the forecasting model (10), equation (11) indicates how agents make their time t decision and equation (12) indicates how

agents assess their previous forecasts. Notice that given his control choice u_t , the agent may numerically compute $\hat{E}_t \lambda_{t+1}^*$, and then use

$$\hat{E}_t \lambda_t^* = -2R x_t - 2W u_t + \beta A_t' \hat{E}_t \lambda_{t+1}^*$$

to numerically compute $\hat{E}_t \lambda_t^*$. In particular, it is not necessary to assume, nor do we assume that our agent knows the \hat{T} -map. We discuss the subtleties of this assumption in the context of a simple DSGE model in Section 4.

We now provide recursive algorithms that allow agents to update their estimates of the transition function and to use their assessments to updating their shadow price forecasting model. To capture recursive updating, we model our agents as least squares learners. Using the data available, agents run OLS regressions and make forecasts and choices based on the estimated models. To estimate the transition equation, and thus obtain the estimates A_t and B_t , agents regress x_{t-1} on x_{t-2}, u_{t-2} , using data $\{x_{t-1}, u_{t-1}, \dots, x_0, u_0\}$.⁴ To estimate the shadow price forecasting model at time t , and thus obtain the estimate H_t , we assume agents use $\hat{E}_t \lambda_{t-1}^*$ as a proxy for λ_t^* : agents regress $\hat{E}_t \lambda_{t-1}^*$ on x_{t-1} using data $\{x_{t-1}, \dots, x_0, \hat{E}_t \lambda_{t-1}^*, \dots, \hat{E}_0 \lambda_0^*\}$.

We may describe the evolution of the estimate of A_t, B_t and H_t over time using RLS. The following dynamic system, written in recursive causal ordering, captures the evolution of agent behavior under bounded optimality.

$$\begin{aligned} R_t &= R_{t-1} + \frac{1}{t} (x_{t-1} x_{t-1}' - R_{t-1}) \\ H_t &= H_{t-1} + \frac{1}{t} R_{t-1}^{-1} x_{t-1} (E_{t-1}^* \lambda_{t-1}^* - H_{t-1} x_{t-1})' \\ \hat{R}_t &= \hat{R}_{t-1} + \frac{1}{t} \left(\begin{pmatrix} x_{t-2} \\ u_{t-2} \end{pmatrix} (x_{t-2}', u_{t-2}') - \hat{R}_{t-1} \right) \\ \begin{pmatrix} \hat{A}_t \\ \hat{B}_t \end{pmatrix} &= \begin{pmatrix} \hat{A}_{t-1} \\ \hat{B}_{t-1} \end{pmatrix} + \frac{1}{t} \hat{R}_t^{-1} \begin{pmatrix} x_{t-2} \\ u_{t-2} \end{pmatrix} \left(x_{t-1} - \begin{pmatrix} \hat{A}_{t-1} & \hat{B}_{t-1} \end{pmatrix} \begin{pmatrix} x_{t-2} \\ u_{t-2} \end{pmatrix} \right)' \\ x_t &= A x_{t-1} + B u_{t-1} + C \varepsilon_t \\ u_t &= F(H_t, \hat{A}_t', \hat{B}_t') x_t \\ E_t^* \lambda_t^* &= \hat{T}(H_t, \hat{A}_t', \hat{B}_t') x_t \end{aligned} \tag{13}$$

Proposition 2 *If LQ.1 – LQ.3 are satisfied then there exists a neighborhood of $(-2P^*, A, B)$ so that if the initial conditions are chosen within this neighborhood and if learning algorithm (13) is augmented with a projection facility then (H_t, A_t, B_t) converges to $(-2P^*, A, B)$ almost surely.*

⁴As is standard in the learning literature, when analyzing real time learning, agents are not assumed to use current data to form current estimates as this avoids technical difficulties with the recursive formulation of the estimators. There are alternatives: see Marcat and Sargent (1989) for details.

See the Appendix for the proof, including a more careful statement of the proposition, a construction of the relevant neighborhood, and a discussion of the “projection facility,” which essentially prevents the estimates from wandering too far away from the fixed point. A detailed discussion of real time learning in general and projection facilities in particular is provided by Evans and Honkopolhja (2001). We conclude that under quite general conditions, our simple notion of boundedly optimal behavior is asymptotically optimal, that is, shadow price learners learn to optimize.

The heart of the proof of Proposition (2) lies in understanding $D\hat{T}_H$; in fact, the proof hinges on showing that the ordinary differential equation

$$\frac{dH}{d\tau} = \hat{T}(H, A, B) - H$$

is Lyapunov stable. In this way, asymptotic optimality is governed by an E-stability differential equation entirely analogous to the standard results in the theory of least squares learning. An interesting twist here, though, is that \hat{T} is not a T-map in the usual sense (which is why we do not denote it T): it does not map the perceived law of motion to the actual law of motion because the λ_t^* is unobserved.

3 Euler equation learning

As mentioned in the introduction, we are not the first to consider issues surrounding agent-level learning: Euler equation learning and infinite horizon learning have been explored within the context of particular DSGE models. In this section, we compare our shadow price learning mechanism to Euler equation learning using the quadratic regulator as our laboratory; because infinite horizon learning, at least as developed thus far, applies specifically to consumption problems, we conduct its examination in our companion paper.⁵

3.1 What is an Euler equation?

To detail Euler equation learning, we must first specify what we mean by an Euler equation. While there are many references to, and constructions of Euler equations in the literature, we are unable to find a source that offers a precise definition. Some reflection reveals why: whereas construction of an Euler equation within a particular

⁵Infinite horizon learning asserts that the agent’s behavioral primitive includes the lifetime budget constraint, which specifies that at any time t the present value of consumption equal (not be more than) current savings plus the present value of income flow. While this notion of a lifetime budget constraint applies most naturally to problems of agent consumption (say, government or consumer), nothing in principle prevents it from being applied in more general settings, provided the transition equation is linear. We explore this possibility in the companion paper.

model usually requires only simple variational arguments, defining Euler equations in the context of general models that include complex interactions between variations in controls today and states n periods in the future, is quite complicated. In our companion paper, Evans and McGough (2009), we offer a general definition and explore issues of existence. Because these details would distract from our current goal of providing a simple comparison of Euler equation learning and SP-learning, we avoid the general, precise definition here and instead focus on examples germane to special cases of the regulator’s problem.

The problem of interest is given by

$$V(x_t) = \max_{s \geq 0} E_t \sum \beta^s r(x_{t+s}, z_{t+s}, u_{t+s}) \quad (14)$$

$$\text{s.t. } u_{t+s} \in \Gamma(x_{t+s})$$

$$x_{t+s+1} = g(x_{t+s}, z_{t+s}, u_{t+s}, \varepsilon_{t+s+1}) \quad (15)$$

$$z_{t+s+1} = h(z_{t+s}, \eta_{t+s+1}). \quad (16)$$

Here $u_{t+s} \in \mathbb{R}^m$ is the control, $\Gamma(x_{t+s})$ is the set of feasible controls, $x_{t+s} \in \mathbb{R}^n$ is the vector of *endogenous* state variables, that is, states that depend in part on the control, and $z_{t+s} \in \mathbb{R}^k$ is the vector of *exogenous* state variables.

As is standard in the literature, we think of an “Euler equation” as a system of first order conditions that must be satisfied along the optimal path.⁶ Typically Euler equations may be constructed using a simple variational technique: assume the optimal control path has been identified and consider a small change du_t in the control at time t ; then, determine a finite sequence of small changes in future controls, du_{t+k} for $k = 1, \dots, n$, together with the associated expected changes in the endogenous states, dx_{t+k} for $k = 1, \dots, n$, so that $dx_{t+n+1} = 0$. Since the path is assumed optimal, the small variation $\{du_{t+k}, dx_{t+k}\}_{k=0}^n$ must not change the regulator’s expected return: $dV(x_t) = 0$.⁷ The Euler equation system is obtained by eliminating the differentials from the equation implied by $dV(x_t) = 0$. Notice that while the derivatives encoding the Euler equation system will condition on the exogenous state vector z_t , and while the transition equation (16) will be used to form expectations, z_t plays no explicit role in the construction of the Euler equation.

As a simple example, consider an endowment problem with habit persistence:

$$\max_{t \geq 0} E_t \sum \beta^t u(c_t - \gamma c_{t-1})$$

$$\text{s.t. } s_t = R_t s_{t-1} - c_t$$

$$R_{t+1} = h(R_t, \eta_{t+1})$$

⁶With particular assumptions on the return and transition functions, together with transversality, these conditions may also be sufficient to guarantee optimality.

⁷Because u_t does not affect x_t , dx_t is taken to be zero.

Here s_0 is the initial stock of a good that grows stochastically at rate R_t , and exogenous income is zero for all times t . For simplicity, set $\gamma = 0$ for now, but we will return to the non-zero case directly. The endogenous state variable is $x_t = s_{t-1}$ and the control is $u_t = c_t$.⁸ To compute the Euler equation system, contemplate a variation $dc_t \neq 0$. Our goal is to choose dc_{t+1} so that $dx_{t+2} = 0$. Now, $dx_{t+2} = ds_{t+1}$, and $s_{t+1} = R_{t+1}s_t - c_{t+1}$; thus, we may choose $dc_{t+1} = R_{t+1}ds_t$, which yields

$$dx_{t+2} = ds_{t+1} = R_{t+1}ds_t - dc_{t+1} = 0.$$

Finally, we may compute the effect of this proposed variational path on the time t value of utility; a necessary conditional for optimality is that this effect be zero. We have

$$\begin{aligned} dV(x_t) = 0 &\implies u'(c_t)dc_t + E_t u'(c_{t+1})dc_{t+1} = 0 \\ &\implies u'(c_t)dc_t + E_t u'(c_{t+1})R_{t+1}ds_t = 0 \\ &\implies u'(c_t)dc_t - E_t u'(c_{t+1})R_{t+1}dc_t = 0 \end{aligned} \tag{17}$$

$$\tag{18}$$

where the last line uses $ds_t = -dc_t$. Notice that (17) is the usual Euler equation.

In the previous example, it was possible to choose du_{t+1} so that $dx_{t+2} = 0$, and this resulted in an Euler equation with only one lead: to satisfy the Euler equation, agents are required only to forecast one period ahead. Because some models require multiple leads, we offer the following terminology: an Euler equation has order N if it predicates dependence on time t expectations of time $t + N$ variables.⁹ To illustrate, consider again the above example, but now assume $\gamma > 0$. As before, the control is $u_t = c_t$, but now the endogenous state vector must be expanded: $x_t = (s_{t-1}, c_{t-1})'$. Again, contemplate a small change in control, $dc_t \neq 0$. For a first order Euler equation to exist, we must choose dc_{t+1} so that $dx_{t+2} = 0$. Now, $dx_{t+2} = 0$ implies that $ds_{t+1} = 0$ and $dc_{t+1} = 0$. But then

$$ds_{t+1} = R_{t+1}ds_t - dc_{t+1} = -R_{t+1}dc_t = 0,$$

which contradicts our assumption that $dc_t \neq 0$.

On the other hand, a second order Euler equation does exist. Now we require that $dx_{t+3} = 0$, which means $ds_{t+2} = dc_{t+2} = 0$. Since $ds_{t+2} = R_{t+2}ds_{t+1} - dc_{t+2}$, we need $ds_{t+1} = 0$. Thus choose $dc_{t+1} = R_{t+1}ds_t$, as above. We have $dV(x_t) = 0 \implies$

$$u'(t)dc_t - \beta\gamma E_t u'(t+1)dc_t = -\beta E_t u'(t+1)R_{t+1}ds_t + \beta^2\gamma E_t u'(t+2)R_{t+1}ds_t = 0$$

where $u'(t) = u'(c_t - \gamma c_{t-1})$. Again, using the transition equation to determine that $ds_t = -dc_t$, we obtain the usual Euler equation.

⁸Alternatively, we could choose s_t to be the control, and substitute out consumption. In this case, the variation contemplated is different, but the Euler equation is the same.

⁹In general, even N^{th} -order Euler equations may not exist: see Evans and McGough (2009).

To provide sufficient conditions guaranteeing the existence of first order Euler equations, we require a little more notation. Let $r_*(s)$ is the partial of r with respect to $*$ evaluated at (x_s, z_s, u_s) , and $g_*(s)$ is the partial of g with respect to $*$ evaluated at $(x_s, z_s, u_s, \varepsilon_{s+1})$. Consider a small change in the controls u_t and u_{t+1} . A first order Euler equation exists provided that given du_t , we can choose du_{t+1} so that $dx_{t+2} = 0$, where

$$dx_{t+2} = g_x(t+1)g_u(t)du_t + g_u(t+1)du_{t+1}. \quad (19)$$

This leads to the following two simple cases.

Case 1: $\dim u \geq \dim x, \det g_u(t) \neq 0$

By trivially expanding the state vector, we may assume that $\dim u = \dim x$. Since $g_u(t+1)$ is invertible, we may choose

$$du_{t+1} = -g_u(t+1)^{-1}g_x(t+1)g_u(t)du_t.$$

The resulting Euler equation has the form

$$r_u(t) + \beta (r_x(t+1) - r_u(t+1)g_u(t+1)^{-1}g_x(t+1)) g_u(t) = 0. \quad (20)$$

Case 2: $g_x = 0$

Since $g_x = 0$, it follows that $dx_{t+2} = g_u(t+1)g_u(t)du_t + g_u(t+1)du_{t+1}$. Thus we may choose $du_{t+1} = 0$. We obtain the usual Euler equation

$$r_u(t) + \beta E_t (r_x(t+1)g_u(t)) = 0.$$

See, for example, Ljungqvist and Sargent (2006).

3.2 Euler equation learning

We now consider Euler equation learning in the context of the quadratic regulator. To fix ideas, consider a non-stochastic quadratic regulator who faces a linear constraint satisfying Case 2 above: $A = 0$. The regulator's problem is

$$\max \quad -E_t \sum_{t \geq 0} \beta^t (x_t' R x_t + u_t' Q u_t + 2x_t' W u_t) \quad (21)$$

$$\text{s.t.} \quad x_{t+1} = B u_t, \quad (22)$$

and the associated Euler equation is given by

$$Qu_t + W'x_t = -\beta(B'RE_t x_{t+1} + B'WE_t u_{t+1}). \quad (23)$$

To implement Euler equation learning, we back off the assumption that our regulator knows how to solve the system given by the transition equation and (23); instead, we follow Evans, Honkapohja, and Mitra (2003) and take (23) as the behavioral primitive: agents form boundedly rational forecasts of x_{t+1} and u_{t+1} , and make their time t control decision to meet their perceived Euler equation. For simplicity, we assume the agent knows the transition matrix B , though he could just as well estimate it.

The agent is required to forecast his own future control decision, and we provide him a forecasting model that is functionally consistent with optimal behavior: PLM $u_t = Fx_t$. The agent computes

$$E_t x_{t+1} = Bu_t, \quad \text{and} \quad E_t u_{t+1} = FE_t x_{t+1},$$

which yields the control decision

$$u_t = -(Q + \beta B'RB + \beta B'W'FB)^{-1} W'x_t.$$

Finally, agents update their forecast of future behavior by regressing the control on the state. This updating process results in a recursive algorithm analogous to (13), which identifies the agent's behavior over time.

The stability of this recursive algorithm may be analyzed via the T-map, which is given by

$$T_F(F) = -(Q + \beta B'RB + \beta B'W'FB)^{-1} W'.$$

The associated matrix differential

$$dT_F = \beta (Q + \beta B'RB + \beta B'W'FB)^{-1} B'W (dF) B (Q + \beta B'RB + \beta B'W'FB)^{-1} W'.$$

Vectorizing, and studying the eigenvalues at the optimal matrix F provides stability results.

We now ask whether Euler equation learning and shadow price learning are equivalent in some natural sense. The T-maps are not the same – often they act on spaces of different dimensions – and so a precise notion of equivalence is somewhat subtle. We will say that learning mechanisms are equivalent if they predict the same rate of convergence as determined by the E-stability differential equation.¹⁰

Consider agent behavior under SP-learning. We provide agents the usual PLM: $\lambda = Hx$. The T-map may be computed by our previous work, and is given by

$$T_H(H) = -2R + 4W(2Q - \beta B'HB)^{-1}W',$$

¹⁰The convergence rate is governed by the largest real part of the T-map derivatives' eigenvalues.

with associated matrix differential

$$dT_H = 4\beta W (2Q - \beta B'HB)^{-1} B' (dH) B (2Q - \beta B'HB)^{-1} W'.$$

At the fixed points, we have that

$$2(2Q - \beta B'HB) = Q + \beta B'RB + \beta B'W'FB \equiv D.$$

Therefore, the only difference in the differentials dT_F and dT_H is ordering:

$$\begin{aligned} dT_F &= \beta DB'W (dF) BDW' \\ dT_H &= \beta WDB' (dH) BDW' \end{aligned}$$

Because for square conformable matrices S and T we have that $\text{eig}(ST) = \text{eig}(TS)$, it follows SP-learning and Euler equation learning are equivalent:

$$\max(\text{Re}(\text{eig}(WDB' \otimes DB'W))) = \max(\text{Re}(\text{eig}(WDB' \otimes WDB'))) \quad (24)$$

This equivalence generalizes so that SP-learning and Euler equation learning are equivalent whenever the conditions sufficient for first order Euler equations to exist, as specified in Case 1 and Case 2, are satisfied:

Theorem 3 *Consider the following quadratic regulator problem:*

$$\begin{aligned} \max \quad & -E_0 \sum \beta^t \left(\begin{pmatrix} x'_t & z'_t \end{pmatrix} R \begin{pmatrix} x_t \\ z_t \end{pmatrix} + u'_t Q u_t + 2 \begin{pmatrix} x'_t & z'_t \end{pmatrix} W u_t \right) \\ \text{s.t.} \quad & x_{t+1} = B u_t + C z_t + \varepsilon_{t+1}, \\ & z_{t+1} = \rho z_t + \eta_t \end{aligned}$$

In this case, Euler equation learning and SP learning are equivalent.

The proof is contained in the Appendix.

Conjecture 4 *Consider the following quadratic regulator problem:*

$$\begin{aligned} \max \quad & -E_0 \sum \beta^t \left(\begin{pmatrix} x'_t & z'_t \end{pmatrix} R \begin{pmatrix} x_t \\ z_t \end{pmatrix} + u'_t Q u_t + 2 \begin{pmatrix} x'_t & z'_t \end{pmatrix} W u_t \right) \\ \text{s.t.} \quad & x_{t+1} = A x_t + B u_t + C z_t + \varepsilon_{t+1}, \\ & z_{t+1} = \rho z_t + \eta_t \end{aligned}$$

If $\dim(x) = \dim(u)$ and $\det(B) \neq 0$, then Euler equation learning and SP learning are equivalent.

Euler equation learning and SP learning are not, in general, equivalent: when higher order Euler equations are needed to capture the model’s first order conditions, then they will generically differ. The intuition for their difference is straightforward. Suppose $\dim(u) = 1$. The PLMs are given as follows:

$$\begin{aligned} \text{SP PLM:} \quad & \lambda_t = Hx_t \\ \text{EL PLM:} \quad & u_t = Fx_t \end{aligned}$$

If x is also univariate then H and F are both scalars: in this sense the PLMs require and capture the same amount of information. On the other hand, if x is bivariate,¹¹ then $\lambda \in \mathbb{R}^2$, and H is 2×2 , whereas F is 2×1 : in this case SP learners have twice as many perceived parameters as their Euler equation counterparts. Said differently, the SP-PLM requires less information than the EL PLM: for the SP and EL PLMs to be equivalent, the SP-learner must understand the structural relationship between the states’ shadow prices, and he must incorporate this relationship by imposing restrictions on his perceived parameters.

4 SP-learning in a DSGE model

By Proposition (2), an individual with quadratic preferences and facing a linear constraint can learn to make optimal choices provided he makes boundedly rational forecasts and uses boundedly optimal behavior. To further investigate and gain insight into the behavior imparted by SP-learning, we turn to a simple DSGE model.

The application of Proposition (2) to a generic DSGE model raises a number of interesting and subtle issues. Specifically,

- The DSGE models prevalent in the literature typically specify non-quadratic objective functions and non-linear transition systems; therefore our result can not be applied directly.¹²
- The reduced form equations identifying the DSGE model’s equilibrium are often linearized before learning mechanisms are applied.
- Even in representative agent models, incorporating agent-level-learning appropriately requires carefully distinguishing between individual and aggregate variables.

In Evans and McGough (2009), we modify shadow price learning to apply to linearized DSGE models, and, within the context of an RBC model with habit persistence, we

¹¹For this case to be interesting, $A \neq 0$. Otherwise, x_{1t} is proportional to x_{2t} and thus redundant.

¹²We are currently working to modify our result to include more general specifications of the regulator’s problem.

carefully address the issues raised above. Further, in that paper, we explore in detail the relationships between SP-learning, Euler equation learning (and its variants) and infinite horizon learning. We avoid these issues in the current paper by considering a Robinson Crusoe economy with quadratic preferences and linear technology, as in Sargent and Hansen (2009). In Section 4.3 below, we do consider shadow price learning in a linearized version of the Crusoe economy with more general preferences.

4.1 A Robinson Crusoe economy

A narrative approach may facilitate intuition. Thus, imagine Robinson Crusoe, a middle class Brit, finding himself marooned on a tropical island. An organized young man, he quickly takes stock of his surroundings. He finds that he faces the following problem:

$$\max \quad -\frac{1}{2}E \sum_{t \geq 0} \beta^t (c_t - b_t)^2 \quad (25)$$

$$\text{s.t.} \quad y_t = A_1 s_{t-1} + A_2 s_{t-2} + z_t \quad (26)$$

$$s_t = y_t - c_t$$

$$b_t - b^* + \Delta(b_{t-1} - b^*) + \varepsilon_t$$

$$z_t = \rho z_{t-1} + \eta_t$$

Here y_t is fruit and c_t is consumption of fruit. Equation (26) is Bob's production function – he can either plant the fruit or eat it, seeds and all – and the double lag captures the production differences between young and old fruit trees. Note that s_t is the quantity of fruit planted in time t ; thus it is also the quantity of new trees in $t+1$ and the quantity of old trees in $t+2$. Finally, z_t is a productivity shock (think weather) and b_t is stochastic bliss: see Sargent and Hansen (2009) for further discussion of this economy as well as many other examples of economies governed by quadratic objectives and linear transitions.¹³

When he is first marooned, Bob doesn't know if there is a cyclic weather pattern; but he thinks that if last year was dry this year might be dry as well. Good with numbers, Bob decides to estimate this possible correlation using RLS. Bob also learns that the fruit rots immediately (but still grows if you plant it), so he must make his consumption decision (as a function of known values, such as the number of trees, weather, etc.) before the trees' yield is realized. This requires estimating the production function, which he also does using RLS.¹⁴ Finally, Bob contemplates

¹³The only novelty in our economy is the presence of a double lag in production. The double lag is a mechanism to expose the difference between Euler equation learning and SP-learning. Other mechanisms, such as the incorporation of habit persistence in the quadratic objective, yield similar results.

¹⁴Having the fruit rot immediately implies that Bob must forecast current output, and therefore

how much fruit to eat. He decides that his consumption choice should depend on the value of future fruit trees forgone. He concludes that the value of an additional tree tomorrow will depend (linearly) on how many trees there are, and makes a reasoned guess about this dependence. Given this guess, Bob estimates the value of an additional tree tomorrow, and chooses how much fruit to eat today.

Belly full, Bob pauses to reflect on his decisions. Bob realizes his consumption choice depended in part on his estimate about the value of additional trees tomorrow, and that perhaps he should revisit this estimate. He decides that the best way to do this is to contemplate the value of an additional tree today. Bob realizes that an additional young tree today would provide additional young trees tomorrow (if he planted the young tree's fruit) and an old tree tomorrow, and that an additional old tree today would provide young trees tomorrow (if he planted the old tree's fruit). Using his estimate of the value of additional trees tomorrow, Bob estimates the value of an additional young tree and an additional old tree today. He then uses these estimates to re-evaluate his guess about the dependence of tree-value on tree-stock. Exhausted by his efforts, Bob falls sound asleep. He should sleep well: Proposition 2 tells us that by following this simple procedure, Bob will learn to optimally exploit his island paradise.

This simple narrative describes the behavior of our boundedly optimal agent. It also points a subtle behavioral assumption that is more easily examined by adding precision to the narrative. To avoid unnecessary complication, set $\Delta = 0$, $\varepsilon_t = 0$, $z_t = 0$, and $\eta_t = 0$. The simplified problem becomes

$$\begin{aligned} \max \quad & -\frac{1}{2} E \sum_{t \geq 0} \beta^t (c_t - b^*)^2 \\ \text{s.t.} \quad & y_t = A_1 s_{t-1} + A_2 s_{t-2} \\ & s_t = y_t - c_t \end{aligned} \tag{27}$$

Let λ_{1t}^* be the time t value of an additional new tree in time t and λ_{2t}^* the time t value of an additional old tree in time t .¹⁵ Bob guesses that λ_{it}^* depends on s_{t-1} and s_{t-2} :

$$\lambda_{it}^* = a_i + b_i s_{t-1} + d_i s_{t-2}. \tag{28}$$

He then forecasts λ_{it+1}^* :

$$E_t \lambda_{it+1}^* = a_i + b_i E_t s_t + d_i s_{t-1}. \tag{29}$$

estimate the production coefficients A_i , to make his consumption decision. Estimates of A_i are not required for the consumption decision if Bob can condition on output. In this case, estimates of A_i are still needed to update the estimates of the shadow prices.

¹⁵The state vector in this model is three dimensional, but the shadow price corresponding to the constant term plays no role in the control or updating behavior. The same point holds for any exogenous state variable.

Because he must choose consumption, and therefore savings, before output is realized, Bob estimates the production function and finds

$$E_t s_t = A_{1t-1} s_{t-1} + A_{2t-1} s_{t-2} - c_t, \quad (30)$$

where A_t is obtained by regressing s_t on $(s_{t-1}, s_{t-2})'$. He concludes that

$$E_t \lambda_{it+1}^* = a_i + (b_i A_{1t-1} + d_i) s_{t-1} + b_i A_{2t-1} s_{t-2} - b_i c_t, \quad (31)$$

which, he notes, depends on his consumption choice today.

Now Bob contemplates his consumption decision. By increasing consumption by dc , Bob gains $-(c_t - b^*)dc$ and loses $\beta E_t \lambda_{1t+1}^* dc$. Bob equates marginal gain with marginal loss, and solves for consumption. With c_t in hand, he also obtains *numerical values* for $E_t \lambda_{it+1}^*$ via (31).

Finally, Bob revisits his parameter guesses a_i, b_i , and d_i . He first thinks about the benefit of an additional new tree today ($ds_{t-1} = 1$): The fruits could be saved to produce A_{1t} new trees tomorrow, plus he gets an additional old tree tomorrow. He concludes

$$E_t \lambda_{1t}^* = \beta A_{1t} E_t \lambda_{1t+1}^* + \beta E_t \lambda_{2t+1}^*. \quad (32)$$

He then thinks about the benefit of an additional old tree today ($ds_{t-2} = 1$): the fruits could be saved to produce A_{2t} new trees tomorrow. Thus

$$E_t \lambda_{2t}^* = \beta A_{2t} E_t \lambda_{1t+1}^*. \quad (33)$$

Because Bob has numerical values for $E_t \lambda_{it+1}^*$, (32) and (33), together with the estimates A_{it} , generate numerical values for the updated shadow price perceptions. Bob may then use these data to form new estimates of his parameter guesses a_i, b_i , and d_i .

This precise implementation of the narrative above highlights our view of Bob's behavior: he estimates forecasting models, makes decisions, and collects new data to update his models. The implementation also reveals the subtle behavioral assumption alluded to above: we think of our agent as forming a forecasting model for shadow prices, and determining consumption behavior; then, based on this consumption behavior, our agent obtains numerical estimates of future shadow prices, and uses these estimates to revise his estimates of current shadow prices. Alternatively, one could assume Bob recognizes the impact of his shadow price forecasting model, and of the perceived parameters in particular, on his current shadow price revisions: one could assume Bob knows and understands the \hat{T} -map. Under this interpretation, it might further seem natural for Bob to search for a forecasting model that is consistent with the way shadow prices are subsequently revised: Bob could simply seek a fixed point of the \hat{T} -map. We view this alternative behavioral assumption as strong, and somewhat unnatural for two reasons. First, understanding the importance of, searching

for, and finding a fixed point to the \hat{T} -map is equivalent to full optimality given the perceived transition equations. This violates a principal assumption that our agent, as a decision maker, has limited sophistication. We prefer to assume our agent does not even explicitly recognize the existence of a \hat{T} -map. But even if he did know the \hat{T} -map, would he recognize that a fixed point is what's wanted to ensure optimal behavior? Why would the agent think such a fixed point even exists? And if it did exist, how would the agent find it? Recognition that a fixed point is important, exists, and is computable is precisely the knowledge afforded those who study dynamic programming; our assumption is that our agent does not have this knowledge, even implicitly.

Our second reason for assuming Bob does not seek a fixed point to the \hat{T} -map relates to the above observation that obtaining such a fixed point is equivalent to full optimality given the “perceived transition equations.” However, if the perceived coefficients A_{it} are far from the true coefficients, A_i , it is not clear that the behavior dictated by a fixed point to the \hat{T} -map is superior to the behavior we assume. Given that computation is unambiguously costly, it makes more sense to us that Bob not iterate on the \hat{T} -map for fear that he might make choices based on magnified errors.

4.2 Comparing learning mechanisms in a Crusoe economy

The simplified model (27) provides a nice laboratory to compare and contrast SP-learning with Euler equation learning. We assume that our agent knows the true values of A_i : the second part of Proposition 2 shows this assumption is innocuous. Shadow price learning has been detailed in the previous section: the agent has PLM (28), and using this PLM, he forecasts future shadow prices: see (29). These forecasts yield his consumption decision

$$c_t = \phi_1(a_i, b_i, d_i) + \phi_3(a_i, b_i, d_i)s_{t-1} + \phi_3(a_i, b_i, d_i)s_{t-2},$$

which he uses to compute shadow price forecasts via (31). We may use these forecasts, together with equations (32) and (33) to determine \hat{T} -map.

To obtain the Euler equation, we proceed as in Section (3.1): we use simple variational techniques to identify the sequence of first order conditions, which, when coupled with the transversality condition, are sufficient to guarantee optimality. The Euler equation is given by

$$c_t = \Psi + \beta A_1 E_t c_{t+1} + \beta^2 A_2 E_t c_{t+2}, \tag{34}$$

where $\Psi = b^*(1 - \beta A_1 - \beta^2 A_2)$. As described in Section 3, Euler equation learning is implemented by taking (34) as the behavioral primitive. The agent is assumed to forecast his future consumption behavior and then choose consumption today based

on these forecasts. The agent is assumed to form forecasts using a PLM that is functionally consistent with optimal behavior:

$$c_t = a + bs_{t-1} + ds_{t-2}.$$

Using this forecasting model and the transition equation

$$s_t = A_1s_{t-1} + A_2s_{t-2} - c_t,$$

the agent behaves so as to satisfy (34). This behavior can then be used to identify the associated T-map.

Our interest here is to compare shadow price learning and Euler equation learning. As in the previous section, we will say that the learning mechanisms are equivalent if they indicate the same speed of convergence. If $A_2 = 0$ then the agent's problem has a one dimensional control and a two dimensional state, with one dimension corresponding to a constant: by Proposition 4, shadow price learning and Euler equation learning will be equivalent in this case; however, for $A_1 > 0$, the endogenous state's dimension becomes two, and the equivalence may break down, as is evidenced by Figure 1.

Figure 1 Here

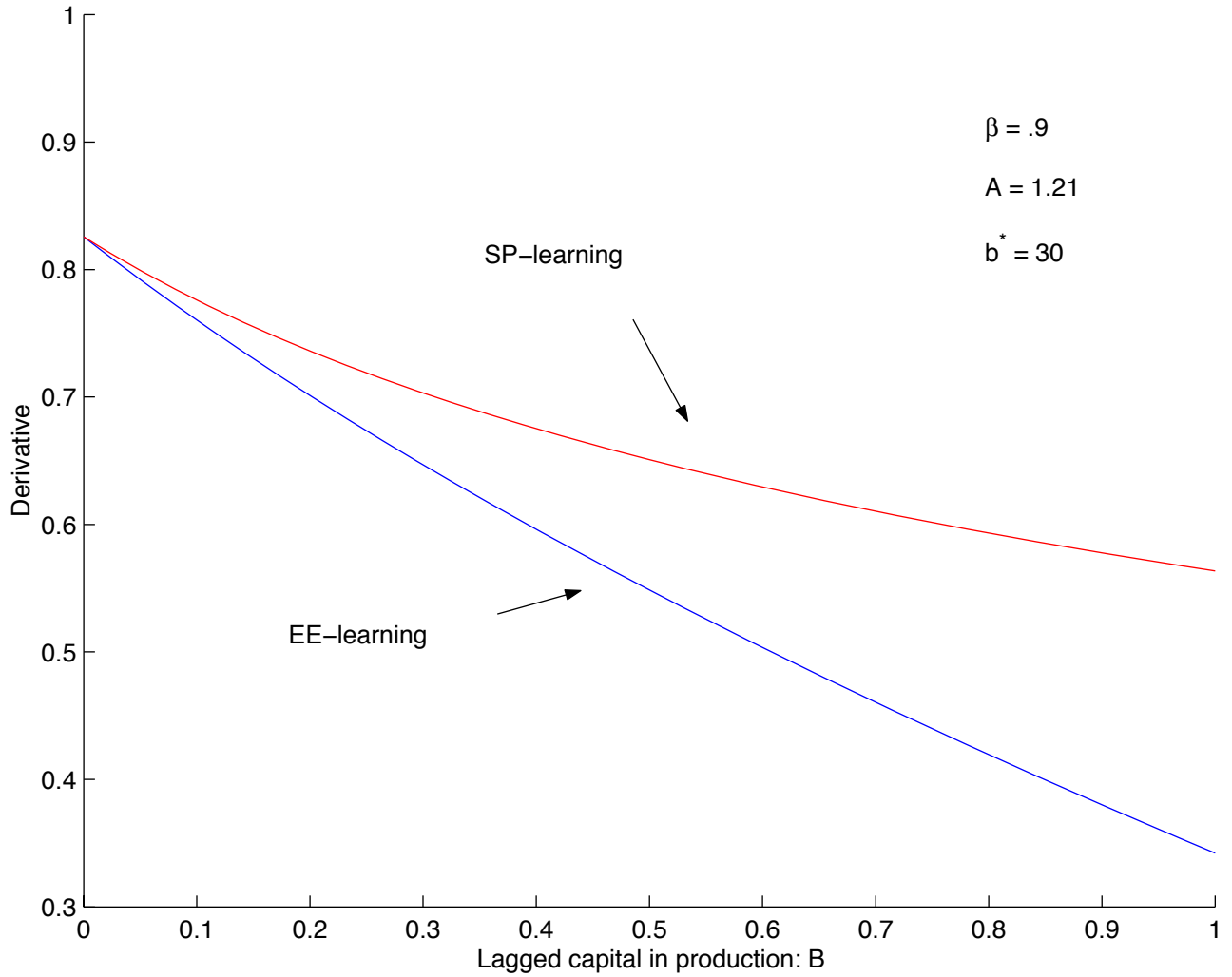
The intuition for the inequivalence of shadow price learning and Euler equation learning is precisely as indicated in Section (3.2): shadow price learning recognizes the two endogenous states and requires that the agent learn the relationship between their prices; Euler equation learning assumes agents understand the link between the two shadow prices and can exploit this link to simplify the forecasting and decision problem.

4.3 Shadow price learning in a linearized model

Proposition 2 applies only to economies identified by quadratic objectives and linear constraints; however, the behavioral assumptions governing shadow price learning are easily adapted to more general settings, provided that we abandon the requirement that agents learn to behave optimally, and instead adopt the more lenient requirement that they learn to behave optimally, up to first order. Consider, for example, the simple Crusoe economy identified above, but modified to incorporate a more general utility function:

$$\begin{aligned} \max \quad & E \sum_{t \geq 0} u(c_t) \\ \text{s.t.} \quad & y_t = A_1s_{t-1} + A_2s_{t-2} \\ & s_t = y_t - c_t \end{aligned}$$

Shadow price learning vs Euler equation learning, LQRC economy



Under SP-learning, our agent chooses his consumption today given his forecast of the benefit of a new tree tomorrow. The fully optimal, non-linear equation capturing this behavior is given by

$$u'(c_t) = \beta E_t \lambda_{1t+1}^*. \quad (35)$$

Since our goal is to assess whether an agent can learn to make decisions based on the first order approximation to fully optimal behavior, we assume that given his forecasts, our agent makes his consumption decision to meet the linearization of (35). Assuming $u' > 0$ and $u'' < 0$ requires us to take $\beta A_1 + \beta^2 A_2 = 1$, so that a steady state capturing optimal behavior exists. The equation governing consumption choice becomes

$$c_t = -\frac{1}{\sigma} E_t \lambda_{1t+1}^*, \quad (36)$$

where σ is the usual measure of risk aversion, and variables are now in proportional deviation from steady state. The linearized versions of equations (32) and (33) are

$$E_t \lambda_{1t}^* = \beta A_{1t} E_t \lambda_{1t+1}^* + \beta^2 A_{2t} E_t \lambda_{2t+1}^* \quad (37)$$

$$E_t \lambda_{2t}^* = E_t \lambda_{1t+1}^*. \quad (38)$$

By simply providing agents with PLMs of the form (28), equations (36) – (38) can be used to determine the relevant \hat{T} -map, and to analyze stability under learning. We note that, as expected, in case $A_2 = 0$, SP-learning and Euler equation learning are equivalent in this linearized model, but if $A_2 > 0$, they may differ: see our companion paper, Evans and McGough (2009), for a detailed exploration of SP-learning in linearized models.

5 Conclusion

The prominent role played by the micro-foundations in modern macroeconomic theory has directed researchers to intensely scrutinize the assumption of rationality – an assumption on which these micro-foundations fundamentally rest; and, some researchers have mounted criticism that the implied level of sophistication demanded of agents in these micro-founded models is unrealistically high. Rationality on the part of agents consists of two central behavioral primitives: that agents are optimal forecasters; and that agents make optimal decisions given these forecasts. While the learning literature has successfully defended the optimal forecasting ability of agents by showing that agents may learn the economy’s rational expectations equilibrium, and thereby learn to forecast optimally, the way in which agents make decisions while learning to forecast has been given much less attention.

In this paper, we formalize the connection between boundedly rational forecasts and agents’ choices by introducing the notion of bounded optimality. Our agents follow simple behavioral primitives: they use econometric models to forecast one-period

ahead shadow prices; and they make control decisions today based on the trade-off implied by these forecasted prices. We call this learning mechanism shadow price learning. We find our learning mechanism appealing for a number of reasons: it requires only simple econometric modeling and thus is consistent with the learning literature; it assumes agents make only one period ahead forecasts instead of establishing priors over the distributions of all future endogenous variables; and it imposes only that agents make decisions based on these one period ahead forecasts, rather than requiring agents to solve a dynamic programming problem with parameter uncertainty.

Investigation of SP-learning reveals that it is behaviorally consistent at the agent level: by following our simply behavioral assumptions, a regulator facing a standard programming problem will learn to optimize. Further, we find that SP-learning is related to, but distinct from Euler equation learning, and in higher dimensions, less informationally demanding. Finally, we find that SP-learning embeds naturally in a simple DSGE models, and can be generalized to model behavior in linearized economies.

While SP-learning is the only agent level learning known to be behaviorally consistent, it may well be that under the primitives, such as dictated by Euler equation learning, or infinite horizon learning, regulators can, in general, also learn to optimize. In this case, the research on agent level learning raises interesting empirical questions: Are any of the learning mechanisms consistent with the data? If so, can the data be used to select among the learning mechanisms? What about planning horizons? Do the data tell us agents look forward only one period, or do they make longer forecasts? These and other related questions are the topics of future research.

Additional theoretical questions arise from our research as well. Section 4 shows that for the DSGE model under investigation that while both SP-learning and Euler equation learning impart stability on the model's unique REE, the learning dynamics differed. Also, Preston (2005) finds examples where Euler equation learning and infinite horizon learning provide different stability conditions. It would be useful to characterize as formally as possible the different agent-level learning mechanisms provide the same stability results. Whether this can be done in a general context rather than always be a model specific question is unknown.

6 Appendix: proof of Proposition 2

The proof is completed in two main steps. first, we use the theory of stochastic recursive algorithms to show that the asymptotic behavior of our system is governed by the Lyapunov stability of the differential system

$$\frac{dH}{d\tau} = \hat{T}(H, A, B) - H.$$

The second step involves showing that the real part of the eigenvalues of $D\hat{T}_H$ are less than unity.

Recall the dynamic system under consideration:

$$R_t = R_{t-1} + \frac{1}{t} (x_{t-1}x'_{t-1} - R_{t-1}) \quad (39)$$

$$H_t = H_{t-1} + \frac{1}{t} R_{t-1}^{-1} x_{t-1} (E_{t-1}^* \lambda_{t-1}^* - H_{t-1} x_{t-1})'$$

$$\hat{R}_t = \hat{R}_{t-1} + \frac{1}{t} \left(\begin{pmatrix} x_{t-2} \\ u_{t-2} \end{pmatrix} (x'_{t-2}, u'_{t-2}) - \hat{R}_{t-1} \right) \quad (40)$$

$$\begin{pmatrix} \hat{A}_t \\ \hat{B}_t \end{pmatrix} = \begin{pmatrix} \hat{A}_{t-1} \\ \hat{B}_{t-1} \end{pmatrix} + \frac{1}{t} \hat{R}_t^{-1} \begin{pmatrix} x_{t-2} \\ u_{t-2} \end{pmatrix} \left(x_{t-1} - \begin{pmatrix} \hat{A}_{t-1} & \hat{B}_{t-1} \end{pmatrix} \begin{pmatrix} x_{t-2} \\ u_{t-2} \end{pmatrix} \right)'$$

$$x_t = Ax_{t-1} + Bu_{t-1} + C\varepsilon_t$$

$$u_t = F(H_t, \hat{A}'_t, \hat{B}'_t)x_t$$

$$E_t^* \lambda_t^* = \hat{T}(H_t, \hat{A}'_t, \hat{B}'_t)x_t \quad (41)$$

To apply the theory of stochastic recursive algorithms, we must place our system in the following form:

$$\theta_t = \frac{1}{t} H(\theta_{t-1}, X_t) + \frac{1}{t^2} \rho_t(\theta_{t-1}, X_t) \quad (42)$$

$$X_t = A(\theta_{t-1})X_{t-1} + B(\theta_{t-1})\eta_t. \quad (43)$$

Here $\theta \in \mathbb{R}^M$ for some M . For extensive details on the asymptotic theory of recursive algorithms such as this, see Chapter 6 of Evans and Honkapohja (2001).

Notice that $R_t, \hat{R}_t, \hat{A}'_t$ and \hat{B}'_t are matrices and θ is a column vector. Therefore, we will need to identify the space of matrices with \mathbb{R}^M for appropriate M . Let $M(n, m)$ be the space of real $n \times m$ matrices and let

$$\text{vec} : M(n, m) \rightarrow \mathbb{R}^{nm}$$

be the usual “vec” operator. Then vec is a vector space isomorphism. Also, if $f : M(n, m) \rightarrow M(p, q)$ then define $f_v : \mathbb{R}^{mn} \rightarrow \mathbb{R}^{pq}$ by $f_v = \text{vec} \circ f \circ \text{vec}^{-1}$.

Next, we must deal with a standard timing issue that arises in real time learning environments. Write $S_{t-1} = R_t$ and $\hat{S}_{t-1} = \hat{R}_t$. Then

$$S_t = S_{t-1} + \frac{1}{t} (x_t x'_t - S_{t-1}) + \frac{1}{t^2} \left(-\frac{t}{1+t} \right) (x_t x'_t - S_{t-1}),$$

and similarly for \hat{S} . With these new variables we may define $\phi_t = (\hat{A}'_t, \hat{B}'_t)'$, and

$$\theta_t = \begin{pmatrix} \text{vec}(S_t) \\ \text{vec}(H_t) \\ \text{vec}(\hat{S}_t) \\ \text{vec}(\phi_t) \end{pmatrix} \quad \text{and} \quad X_t = \begin{pmatrix} x_t \\ x_{t-1} \\ u_{t-1} \\ x_{t-2} \\ u_{t-2} \\ \varepsilon_{t-1} \end{pmatrix}.$$

We may now construct the function

$$H : \mathbb{R}^{n^2} \oplus \mathbb{R}^{n^2} \oplus \mathbb{R}^{(n+m)^2} \oplus \mathbb{R}^{n(n+m)} \rightarrow \mathbb{R}^{n^2} \oplus \mathbb{R}^{n^2} \oplus \mathbb{R}^{(n+m)^2} \oplus \mathbb{R}^{n(n+m)}$$

component-wise as follows:

$$\begin{aligned} H^1(\theta_{t-1}, X_t) &= \text{vec}(x_t x'_t - S_{t-1}) \\ H^2(\theta_{t-1}, X_t) &= \text{vec} \left(S_{t-1}^{-1} x_{t-1} \left((\hat{T}(H_{t-1}, \hat{A}'_{t-1}, \hat{B}'_{t-1}) - H_{t-1}) x_{t-1} \right)' \right) \\ H^3(\theta_{t-1}, X_t) &= \text{vec} \left(\begin{pmatrix} x_{t-1} \\ u_{t-1} \end{pmatrix} (x'_{t-1}, u'_{t-1}) - \hat{S}_{t-1} \right) \\ H^4(\theta_{t-1}, X_t) &= \text{vec} \left(\hat{S}_{t-1}^{-1} \begin{pmatrix} x_{t-2} \\ u_{t-2} \end{pmatrix} \left(((A, B) - (\hat{A}'_{t-1}, \hat{B}'_{t-1})) \begin{pmatrix} x_{t-2} \\ u_{t-2} \end{pmatrix} + C\varepsilon_{t-1} \right)' \right) \end{aligned}$$

Letting $\eta_t = (\varepsilon_t, \varepsilon_{t-1})'$ it is straightforward to find a matrix $A(\theta_{t-1})$ and a matrix B so that (43) is satisfied. Finally, to complete the process of rewriting our system in the canonical form (42) we simply define ρ_t appropriately to capture the second order terms in the recursions for S and \hat{S} .

Let $M = 2n^2 + (n+m)^2 + n(n+m)$. The theory of stochastic recursive algorithms tells us to fix $\theta \in U \subset \mathbb{R}^M$ (where U is an open set to be defined below) and consider the function $h : \mathbb{R}^M \rightarrow \mathbb{R}^M$ defined by

$$h(\theta) = \lim_{t \rightarrow \infty} EH(\theta, X_t).$$

Notice that $h(\theta)$ captures the long run average behavior of the function H for given θ . In learning algorithms, this function H usually reflects a forecast error, and so we may think of $h(\theta)$ as the long run average forecast error obtained when using a forecasting model based on the perceived parameters θ .

Now fix H near $-2P^*$ and \hat{A} and \hat{B} near A' and B' respectively. Recall that $F(-2P^*, A, B)$ stabilized the matrix pair (A, B) , so that the eigenvalues of $A + BF(-2P^*, A, B)$ are inside the unit circle. By continuity, the eigenvalues of $\hat{A}' + \hat{B}'F(-2P^*, \hat{A}', \hat{B}')$ will be inside the unit circle as well. Thus, for H near $-2P^*$

and \hat{A} and \hat{B} near A' and B' , the process (x_t, u_t) is asymptotically stationary. Let $M_1(H, \hat{A}', \hat{B}') = \lim E x_t x_t'$ and

$$M_2(H, \hat{A}', \hat{B}') = \lim E \begin{pmatrix} x_t \\ u_t \end{pmatrix} \begin{pmatrix} x_t' & u_t' \end{pmatrix}.$$

Set

$$\theta^* = \begin{pmatrix} \text{vec}(M_1(-2P^*, A, B)) \\ \text{vec}(-2P^*) \\ \text{vec}(M_1(-2P^*, A, B)) \\ \text{vec}((A, B)') \end{pmatrix}$$

Then the above argument shows that there is an open set U with $\theta^* \in U$ so that $\theta \in U$ implies

$$h^1(\theta) = \text{vec}(M_1(\theta) - S) \tag{44}$$

$$h^2(\theta) = \text{vec}\left(S^{-1}M_1(\theta)\left(\hat{T}(H, \hat{A}', \hat{B}') - H\right)\right) \tag{45}$$

$$h^3(\theta) = \text{vec}\left(M_2(\theta) - \hat{S}\right) \tag{46}$$

$$h^4(\theta) = \text{vec}\left(\hat{S}^{-1}M_2(\theta)\left(\begin{pmatrix} A' \\ B' \end{pmatrix} - \begin{pmatrix} \hat{A} \\ \hat{B} \end{pmatrix}\right)\right) \tag{47}$$

where $M_i(\theta) = M_i(H, \hat{A}, \hat{B})$ and the H , \hat{A} and \hat{B} come from the components of θ .

Having computed $h(\theta)$, we next analyze the differential equation $\dot{\theta} = h(\theta)$. Notice that $h(\theta^*) = 0$, so that θ^* is a fixed point of this differential equation. The theory of stochastic recursive algorithms tell us that under certain conditions, if θ^* is a Lyapunov stable fixed point, then our learning algorithm will converge to it almost surely.

The determination of Lyapunov stability for the system $\dot{\theta} = h(\theta)$ involves simply computing the derivative of h and studying its eigenvalues: if the real parts of these eigenvalues are negative then the fixed point is Lyapunov stable. Computing the derivative of h looks somewhat daunting at first, particularly since we do not know the functional forms of $M_i(\theta)$; however, computation is made easy by observing that the terms multiplying the $M_i(\theta)$ in equation (45) and (47) are zero when evaluated at θ^* so that, by the product rule, the associated derivatives are zero. The resulting block diagonal form of the derivative of h yields repeated eigenvalues that are -1 and the eigenvalues of $\partial h^2 / \partial \text{vec}(H)'$.

Let $S(n, n) \subset M(n, n)$ be the vector space of symmetric matrices and let $D \subset S(n, n)$ be the symmetric negative definite matrices. Let $T : M(n, n) \rightarrow M(n, n)$ by $T(H) = \hat{T}(H, A, B)$, when defined. Notice that T is defined on D and that $T(D) \subset D$. Furthermore, the $T(S) \subset S$ which implies that if $H_0 \in S$ then $H_t \in S$. Since $-2P^* \in D$ and D is an open subset of S , it suffices to restrict the stability analysis of h^2 to D . We have the following result:

Lemma 5 *If LQ.1 – LQ.3 are satisfied then, as a function from D to D , the eigenvalues of $DT_v((\text{vec}(-2P^*))$ have modulus less than one.*

Proving this lemma is the second part of the proof of Proposition 2, and is found at the end of the Appendix. Notice that

$$DT_v((\text{vec}(-2P^*)) - I_{n^2} = \partial h^2 / \partial \text{vec}(H)' \Big|_{-2P^*}$$

so that if the eigenvalues of $DT_v((\text{vec}(-2P^*))$ have modulus less than one then the eigenvalues of $\partial h^2 / \partial \text{vec}(H)' \Big|_{-2P^*}$ have negative real parts.

To complete the proof of Proposition 2, we apply the theory of stochastic recursive algorithms to our system: for details, see chapter 6 of Evans and Honkapohja (2001), and here we follow their notation. Because θ^* is Lyapunov stable there exists an open set D contained in the basin of attraction of the ode $\dot{\theta} = h(\theta)$, and associated Lyapunov function $U : D \rightarrow \mathbb{R}_+$. For $C > 0$, set

$$K(c) = \{\theta \in \mathbb{R}^M : U(\theta) \leq c\}.$$

There is a collection of regularity conditions on H , ρ_t , and $A(\theta)$ that must be met: see p. 123 – 125 of Evans and Honkapohja (2001). Checking that these conditions are met on some set $W = \text{int}K(c)$ for some $c > 0$ is routine: we omit detail, but they are available from the authors upon request.

We now create the projection facility: Choose $0 < c_1 < c_2$ so that $K(c_2) \subset W$. Let $\hat{\theta}_t \in K(c_1)$. Define a new recursive algorithm for θ_t as follows:

$$\theta_t = \begin{cases} \hat{\theta}_t = \frac{1}{t}H(\theta_{t-1}, X_t) + \frac{1}{t^2}\rho_t(\theta_{t-1}, X_t) & \text{if } \hat{\theta}_t \in K(c_2) \\ \bar{\theta}_t & \text{if } \hat{\theta}_t \notin K(c_2) \end{cases} \quad (48)$$

In view of Lemma 5, we have the following corollary, which is a restatement of Proposition 2:

Corollary 6 *There exists an open set U with $\theta^* \in U$ so that if $\theta_0 \in U$ then $\theta_t \rightarrow \theta^*$ almost surely.*

This corollary is proved by choosing $U \subset D$ and so that the map \hat{T} is well-defined, that is, $2Q - \beta\hat{B}H\hat{B}'$ is invertible (which is straightforward because the collection of invertible matrices is open), and finally, so that $F(H, \hat{A}', \hat{B}')$ stabilizes (A, B) . We may then appeal to Corollary 6.8 on page 136 of Evans and Honkapohja, noting that because ε_t has bounded support, provided F stabilizes (A, B) , the random vector X_t is almost surely uniformly bounded.

Proof of Lemma 5. Let \hat{D} be the the set of $n \times n$ matrices so that $\det(Q + B'PB) \neq 0$ and set $G : \hat{D} \rightarrow \hat{D}$ by

$$G(P) = R + A'PA - (B'PA + W)'(Q + B'PB)^{-1}(B'PA + W).$$

Notice that $G(P)$ is simply the RHS of the Riccati equation. Note also that for H in D , $T(H) = -2G(-\frac{1}{2}H)$, so that $DT(-2P^*) = DG(P^*)$. Therefore, we study the eigenvalues of $DG_v(\text{vec}(P^*))$.

Let $A_1 = \beta^{1/2}(A - BQ^{-1}W')$ and $B_1 = \beta^{1/2}BQ^{-\frac{1}{2}}$, where $Q^{-\frac{1}{2}}$ is the usual square root of the positive definite matrix Q . It is straightforward to check that (A_1, B_1) is stabilizable and that (A_1, D) is detectable. From Theorem 13.5.2 in Lancaster and Rodman (1995), the eigenvalues of $A_1 - B_1(I_n + B_1'P^*B_1)^{-1}B_1'PA_1$ have modulus less than one. Now notice that $A_1 - B_1(I_n + B_1'P^*B_1)^{-1}B_1'PA_1$

$$\begin{aligned} &= A - BQ^{-1}W - BQ^{-\frac{1}{2}} \left(I_n + Q^{-\frac{1}{2}}B'P^*BQ^{-\frac{1}{2}} \right)^{-1} Q^{-\frac{1}{2}}B'P(A - BQ^{-1}W) \\ &= A - BQ^{-1}W - B(Q + B'P^*B)^{-1}B'P(A - BQ^{-1}W) \\ &= A - B(Q + B'P^*B)^{-1}B'P^*A - B(I - (Q + B'P^*B)^{-1}B'P^*B)Q^{-1}W \\ &= A - B(Q + B'P^*B)^{-1}B'P^*A - B(Q + B'P^*B)^{-1}W \\ &= A - B(Q + B'P^*B)^{-1}(B'P^*A + W). \end{aligned}$$

Set $\hat{A} = A - B(Q + B'P^*B)^{-1}(B'P^*A + W)$. We will now show that $DG_v(\text{vec}(P^*)) = \hat{A}' \otimes \hat{A}'$, which, by the above argument, will complete the proof.

To compute $DG_v(\text{vec}(P^*))$, we compute the Frechet derivative DG of G . Recall that if B is a Banach space and $F : B \rightarrow B$ then F is Frechet differentiable at $x \in B$ if there exists a linear map $DF(x) : B \rightarrow B$ so that

$$\lim_{\|h\| \rightarrow 0} \frac{\|F(x+h) - F(x) - DF(x)(h)\|}{\|h\|} = 0.$$

If this limit exists then $DF(x)$ is unique and is called the Frechet derivative of F at x . Now notice that if $B = M(n, n)$ then, exploiting that for appropriate matrix norm, the vec operator is an isometric isomorphism, we have

$$\begin{aligned} &\lim_{\|h\| \rightarrow 0, h \in \mathbb{R}^{n^2}} \frac{\|F_v(\text{vec}(x) + h) - F_v(\text{vec}(x)) - \text{vec} \circ DF(x) \circ \text{vec}^{-1}(h)\|}{\|h\|} \\ &= \lim_{\|h\| \rightarrow 0, h \in \mathbb{R}^{n^2}} \frac{\|\text{vec}^{-1} \circ F_v(\text{vec}(x) + h) - \text{vec}^{-1} \circ F_v(\text{vec}(x)) - DF(x) \circ \text{vec}^{-1}(h)\|}{\|h\|} \\ &= \lim_{\|h\| \rightarrow 0, h \in B} \frac{\|F(x + \text{vec}(h)) - F(x) - DF(x)(h)\|}{\|h\|} = 0. \end{aligned}$$

We conclude that $DG_v(\text{vec}(P^*))(v) = \text{vec} \circ DG(P^*) \circ \text{vec}^{-1}(v)$.

Some well-known results of Frechet derivatives will be useful.

1. If $f, g : S(n, n) \rightarrow S(n, n)$ are Frechet differentiable at $x \in S(n, n)$ the

$$D(fg)(x)(v) = Df(x)(v)g(x) + f(x)Dg(x)(v).$$

2. If $\det f(x) \neq 0$ then f^{-1} exists (locally) and

$$Df^{-1}(x)(v) = -f(x)^{-1}Df(x)(v)f(x)^{-1}.$$

3. $Df'(x)(v) = (Df(x)(v))'$.

Now set

$$f_1 : S(n, n) \rightarrow S(n, n) \text{ by } f_1(P) = W' + B'PA$$

$$f_2 : S(n, n) \rightarrow S(n, n) \text{ by } f_2(P) = Q + B'PB$$

Then

$$G(P) = R + A'PA - f_1(P)'f_2(P)^{-1}f_1(P).$$

Then, using the results of Frechet derivatives listed above, for $v \in S(n, n)$, we have that $DG(P^*)(v)$

$$\begin{aligned} &= A'vA - (Df_1(P^*)(v))' f_2^{-1}(P^*)f_1(P^*) - f_1(P^*)'f_2^{-1}(P^*)Df_1(P^*)(v) \\ &+ f_1(P^*)'f_2(P^*)^{-1}Df_2(P^*)(v)f_2(P^*)^{-1}f_1(P^*) \\ &= A'vA - A'vB(Q + B'PB)^{-1}(W' + B'PA) - (W' + B'PA)'(Q + B'PB)^{-1}B'vA \\ &+ (W' + B'PA)'(Q + B'PB)^{-1}B'vB(Q + B'PB)^{-1}(W' + B'PA) \\ &= \hat{A}'v\hat{A}. \end{aligned}$$

Finally, we get that

$$DG_v(\text{vec}(P^*))(x) = \text{vec} \circ DG(P^*) \circ \text{vec}^{-1}(x) = \text{vec} \left(\hat{A}'\text{vec}^{-1}(x)\hat{A} \right) = \left(\hat{A}' \otimes \hat{A}' \right) (x),$$

so that the $\text{eig}(DG_v(\text{vec}(P^*))) = \text{eig} \left(\hat{A}' \otimes \hat{A}' \right)$, which completes the proof.

References

- BERTSEKAS, D. (1987): *Dynamic Programming*. Prentice-Hall INC, Englewood Cliffs, NJ.
- BRAY, M. (1982): “Learning, Estimation, and the Stability of Rational Expectations Equilibria,” *Journal of Economic Theory*, 26, 318–339.
- BRAY, M., AND N. SAVIN (1986): “Rational Expectations Equilibria, Learning, and Model Specification,” *Econometrica*, 54, 1129–1160.
- EVANS, G. W., AND S. HONKAPOHJA (2001): *Learning and Expectations in Macroeconomics*. Princeton University Press, Princeton, New Jersey.
- (2006): “Monetary Policy, Expectations and Commitment,” *Scandinavian Journal of Economics*, 108, 15–38.
- EVANS, G. W., S. HONKAPOHJA, AND K. MITRA (2003): “Notes on Agents’ Behavioral Rules under Adaptive Learning and Recent Studies of Monetary Policy,” mimeo,.
- LANCASTER, P., AND L. RODMAN (1995): *Algebraic Riccati Equations*. Oxford University Press, Oxford, UK.
- MARCEY, A., AND T. J. SARGENT (1989): “Convergence of Least-Squares Learning Mechanisms in Self-Referential Linear Stochastic Models,” *Journal of Economic Theory*, 48, 337–368.
- PRESTON, B. (2005): “Learning about Monetary Policy Rules when Long-Horizon Forecasts Matter,” *International Journal of Central Banking*, 1.
- STOKEY, N., AND R. E. LUCAS JR. (1989): *Recursive Methods in Economic Dynamics*. Harvard University Press, Cambridge, Mass.