Learning to Dynamically Optimize from Individual Experience

Nathan M. Palmer

June 20, 2016

Abstract

Sims (1980) famously stated that non-rational behavior is a "wilderness:" there is only one way for economic agents to be "rational" but infinitely many ways to be non-rational. While this may be strictly true, some paths through the wilderness are more familiar than others. I expand upon a growing economic literature that uses tools from reinforcement learning and approximate dynamic programming to impose bounded rationality in intertemporal choice problems. This paper introduces a framework for implementing bounded rationality in the canonical household intertemporal consumption-savings problem. Agents employ a novel learning-to-optimize algorithm: individual experience is used to approximate to a value function, and this value function is used to determine a regret-minimizing policy function. Preliminary results from numerical simulation demonstrate that learning approaches rational behavior as experience increases. Even when experience is limited, however, agents follow behavior which appears to them as best conditional on their information. Importantly, usage of the canonical optimal control framework allows this model to be directly compared to optimal solutions in welfare terms. Methods of estimation, a key future extension, are discussed as well.

1 Introduction

Sims (1980) famously stated that non-rational behavior is a "wilderness:" there is only one way for economic agents to be "rational" but infinitely many ways to be non-rational. While this may be strictly true, some paths through the wilderness are more familiar than others. Specifically, the economics literature has long employed the tools of optimal control, and particularly dynamic programming, to model agents as optimizing in dynamic and uncertain environments. This optimizing behavior, particularly combined with rational expectations, imposes extensive limits on the size and detail of models which may be built to explain economic phenomena. This creates a tension: if structurally complex models are important for describing economic phenomena, agents cannot be given rational expectations with traditional tools. They may be given various ad-hoc behavior, but as Sims (1980) notes, the space of possible ad-hoc decision rules is infinitely wide.

Consider the housing model of Geanakoplos et al. (2012), "Getting At Systemic Risk via an Agent-Based Model of the Housing Market." The authors create a large-scale computational model of the Washington, D.C. metropolitan area housing market for the years 1997-2009, leading up to and through the financial crisis. The goal was to examine whether rich household-level heterogeneity and non-price considerations (such as leverage) could have a significant impact on model dynamics. Labor and mortgage markets in the model are partial equilibrium and calibrated to historical data, while the housing market obtains a matching equilibrium each period. The primary decision-maker is the household, who makes up both sides of the housing market. Households are highly heterogeneous and their individual decision problem is non-trivial.

Implementing a rational expectations equilibrium in this model is computationally intractable. Instead agents in the model are provided with rules of thumb derived from theoretical considerations, with specific parameters fit to data on household behavior. Households incorporate changing price expectations in their policy function but other key parameters are fixed, such as the marginal propensity to consume. The reason is simply practical: solving for rational expectations is intractable, so what should be done instead? A fixed policy which allows for dynamic expectations seems like a reasonable baseline. An improvement might be a method which allows agents to update their microeconomic policy function as their experience unfolds. I offers a first step in this direction by considering such learning for a foundational consumption-savings problem.

This paper introduces "Regret Learning," a middle ground between optimization and ad hoc behavior for the canonical consumption-savings problem under uncertainty. Agents are endowed with traditional preferences and seek to maximize a traditional objective function. It is assume that agents cannot solve the maximization problem directly. In addition they are restricted to a class of linear consumption functions discussed in Allen and Carroll (2001) which make sequential learning straightforward and intuitive. An agent employs a consumption function[1] for a time and then uses the experience gained to learn parameters for a new consumption function. Learning is structured such that under *enough* experience, the learned consumption functions are distributed close, in welfare terms, to the true optimal consumption function which maximizes their objective. Even when their experience is short, however, they are always moving in a direction that appears best to them conditional on their own experience.

This work is explicitly intended to provide a theoretical foundation for non-optimal agent behavior in both agent-based models and in more traditional macroeconomic models. I particularly want to capture the idea that economic agents would *like* to behave optimally, but they cannot for any number of reasons – perhaps they simply do not know how, or they do not have enough information, or the optimal solution is simply intractable because the problem is too complex. I provide a rigorous description of agent behavior which is "stumbling towards the optimum;" with enough experience and time they can obtain near-optimal rules, but even without extensive time they settle into a stable distribution of rules near the optimal behavior. By using the same optimization framework as traditional models, I can directly calculate the welfare costs of this non-optimal behavior with respect to the optimal behavior.

On a technical level this learning is modeled after the most recent developments from the mathematical dynamic programming literature. These new methods are referred to as "approximate dynamic programming" (ADP) in applied mathematics and "reinforcement learning" (RL) in the computer science literatures. These methods generalize dynamic programming and apply it in situations where an objective function can be formulated but little else is known about the properties of the problem (for example laws of motion, distributions of shocks, or even the shape of the utility function). Although tremendous literature has been generated in other fields, these methods have only been slowly adopted by economics. One impediment to adoption is that the simplest of these methods are often formulated for discrete state and choice spaces, while many foundational problems in macroeconomics and finance require continuous states and choice. Continuous versions of ADP and RL methods exist but often introduce many new parameters with little intuitive interpretation. Regret Learning, introduced in this paper, handles continuous state and choice spaces in a natural way as agents learn about their income shocks, state space, and value functions non-parametrically (described in detail below).

One surprising result of this work is that learning a near-optimal rule can be accomplished with explicitly

[1] I use "policy function" and "consumption function" interchangeably.

biased estimates derived from a single agent's experience. As described in Section 4.3, it is straightforward to formulate a simple Monte Carlo estimator for the optimization problem faced by an agent. Allen and Carroll (2001), however, show that without a highly efficient search or learning method, it can take millions of periods to distinguish near-optimal rules consistently. Regret learning demonstrates that a highly efficient search method does in fact exist, almost exactly as Allen and Carroll (2001) hypothesize, and it exists because the Bellman policy update step in approximate dynamic programming can make significant progress (in expected utility terms) even when using a biased value function, as might be obtained from a single agent's draw of experience.

The rest of the paper is organized as follows. Section 2 reviews related literature. Section 3 describes the agent problem, the approximate form of the consumption function which this solution method will use, and welfare costs of approximation. Section 4 briefly reviews policy iteration before describing the regret learning algorithm. Section 5 uses numerical simulation to demonstrate dynamic statistical properties of regret learning in terms of welfare distance from the optimal solution. Section concludes and discusses and next steps.

2 Related Literature

There is a tremendous literature on "learning" in economics, much too broad to be reviewed here. The specific type of learning which this paper addresses is finding the solution to a dynamic optimization problem. Learning about intertemporal choice poses a particular challenge: obtaining enough information to estimate value and policy functions accurately requires many draws of time series. Since the time series are generated by agent experience, it can be quite difficult to obtain the number of draws needed. Lettau and Uhlig (1999) is one of the earliest such efforts; the authors use a classifier system (an early form of reinforcement learning) which chooses between an optimal rule and a version of the spendthrift ("consume everything") rule. Their model learns the non-optimal rule under a wide range of conditions, and the authors caution readers that households in practice may not learn optimal rules for similar reasons. Interestingly, Başçı and Orhan (2000) revisit Lettau and Uhlig (1999) results and find that agents are once again able to distinguish the optimal rule from non-optimal rules when "trembling hand" experimentation is allowed.

Allen and Carroll (2001) lays out the basic consumption-savings problem under uncertainty, and is the closest to the work done in this paper. Agents in their model use brute force to explore a set of possible consumption functions, selecting the base consumption function by estimating value functions from their own experience. Their work produces a positive and negative result: agents can learn a near-optimal consumption function from enough experience, but enough experience must be in the hundreds of thousands or millions of periods. Since their model is parameterized such that a period is a year, this is very far from appearing practical for use in any model, either traditional or agent-based. Regret learning is closest to the value estimation conducted in Allen and Carroll (2001), but adds a step in which policy functions are learned are well. Howitt and Özak (2014) and Özak (2014) address the same problem introduced by Allen and Carroll (2001), but employ learning based on the Euler equation. Their agents must be somewhat more sophisticated than those of Allen and Carroll (2001), specifically in the sense of being able to solve envelope conditions of their optimization problem, but the payoff is much faster learning, closer to 50-100 periods.

2.1 Learning Dynamic Optimization in Economics

Tesfatsion and Judd (2006) provides an excellent outline of early uses of reinforcement learning in economics. More recently, a handful of authors have explored reinforcement learning applied to intertemporal consumption-savings problems. Allen and Carroll (2001) use a clever parameterization of the consumption function for a simple consumption-savings problem and employ a Monte Carlo learning estimator to chose the best policy on a fixed grid of policies. They find that (a) agents can find a near-optimal rule given enough time, but (b) enough time can be millions of periods, rendering the learning not useful for practical time parameterizations. They suggest that one solution to this may be social learning, which is explored further in Palmer (2012). Howitt and Özak (2014) and Özak (2014) address the same consumption problem as Allen and Carroll (2001), but using a clever form of policy-gradient learning which utilizes marginal utilities to update a consumption function. Policy-gradient learning is a form of reinforcement learning which attempts hill-climbing on an appropriately specified value function surface; see Sutton and Barto (1998) for an intuitive overview. As noted above, Howitt and Özak (2014) and Özak (2014) obtain agents who can find a near-optimal rule quickly but must know more about the structure of the problem than agents of Allen and Carroll (2001).

The above models are largely cast in a partial-equilibrium framework, in which agents actions do not affect the aggregate states of their models. One of the main reasons that learning-to-optimize is interesting is that it may influence aggregate dynamics. One of the original papers related to this is Krusell and Smith (1996), which predates their famous Krusell and Smith Jr (1998) paper. In Krusell and Smith (1996), they examine the general equilibrium effects of agents choosing between using the true optimal solution and various rules of thumb, when the true optimal solution has some small cost. They find that very small costs to optimization, less than a tenth of a percent of per-period consumption, can cause agents to choose to use rules of thumb, which in turn change the statistical dynamics of the aggregate system in non-trivial ways. Evans and McGough (2014) address a similar problem as that of Allen and Carroll (2001), but in a general equilibrium context. Like Howitt and Özak (2014), their agents must know something about the first-order conditions of their optimization problem, and use these to learn about the shadow process of their choices. As with Krusell and Smith (1996), they find that the aggregate dynamics, particularly the transition dynamics, are greatly affected by learning.

An alternative application of learning-to-optimize is finding optimal solutions when traditional techniques are intractable. Examples of this include Hull (2012) and Jirnyi and Lepetyuk (2011). Hull (2012) constructs an overlapping generation model with 60 rolling cohorts of agents, a housing market, housing and non-housing production sectors, a financial intermediary sector, and a central bank. He solves this for optimal agent behavior and explores a number of policy questions. Jirnyi and Lepetyuk (2011) exactly solve a traditional Krusell and Smith Jr (1998) macroeconomic model with aggregate uncertainty, without needing to rely on abbreviations of agent information sets or of aggregate dynamics. While the author finds this approach impressive, I am interested in the additional dynamics which may be introduced by bounded rationality via learning. Yıldızoğlu et al. (2014) explicitly examines the same problem as Allen and Carroll (2001) and Howitt and Özak (2014), but Yıldızoğlu et al. (2014) use a neural network as part of their learning scheme and their agents successfully learn the optimal rule. This is an example of what might be called "artificial intelligence" learning, of which there is an extensive literature. Sargent (1993) outlines much of the early literature in this sub-field.

Recently, Gabaix (2014) developed a sparsity-based dynamic programming model which seeks to capture the idea that agents do not re-evaluate their behavior unless they are prompted by "big enough" events in

their world. In his model these "big" events cause agents to re-optimize to find new behavior. This could be thought of as something of an (s,S) trigger which forces re-optimization. The current form of regret learning exogenously imposes when an agent will re-optimize from experience; future versions will ideally incorporate a similar (s,S) trigger.

In the experimental literature, a number of authors have examined the role that learning plays in dynamic optimization problems. This includes Ballinger et al. (2003), Brown et al. (2009), and Carbone and Duffy (2014). Results of experiments have been mixed. In Chua and Camerer (2011), for example agents could find the optimal solution, but only after many "lifetimes." Ballinger et al. (2003) use multiple overlapping "generations" of subjects, to simulate what learning from predecessors may look like. They find that all their agents – including the best-performing third wave – do not get very close to learning the optimal rule. Brown et al. (2009) find that agents learn a near-optimal solution individually withing roughly four "life cycles," or roughly two when there is social learning.

Houser et al. (2004) sets up a difficult to solve intertemporal optimization problem and estimates the number of types of learners which appear in experimental laboratory data. They find three distinct and clearly identified types of learners, which they label "near rational," "fatalist," and "confused." This is a striking result, which sheds light on potential confounding factors in the previous experiments (different fractions of learning-types may not have been controlled) and motivates the search for agents which can learn a near-optimal rule from experience. This paper can be understood as examining one potential path by which the "near rational" learners found above may be modeled rigorously for quantitative macroeconomic and financial models.

The algorithms used by both Allen and Carroll (2001) and Howitt and Özak (2014) are examples of what is known as "reinforcement learning" in the computer science literature, or "approximate dynamic programming" in the operations research and applied mathematics literature. Bertsekas and Tsitsiklis (1996) is one of the earliest texts outlining foundational theory which ties online optimization methods to dynamic programming. More recently, Bertsekas (2012), Bertsekas (2013), and Powell (2007) explicitly outline the relationship between these approximation methods and traditional dynamic programming. Finally, Sutton and Barto (1998) outlines much of the foundational results and literature from the computer science perspective.

In the computer science literature, different formulations of "regret" are used to judge how well an online reinforcement learning algorithm accomplishes a task of learning-to-optimize. Some definitions and theoretical results related to "regret" and "regret bounds" can be found in a number of papers in this literature, including Jaksch et al. (2010), Auer et al. (2002), and Mannor and Tsitsiklis (2004). I use the term "regret" to describe a different process than the one used in these literatures. Namely, I use it to describe the way in which agents in reflect on their own past experience and make better choices (in a utility-maximizing sense) once they have learned the value of their previous choices. Regardless, there are many ideas in this branch of literature which appear ripe for economic applications.

3 The Consumer Problem

The learning-to-optimize behavior I discuss will be explicitly applied to a stationary, infinite-horizon dynamic optimization problem. Electronic Appendix A outlines an extended finite-horizon problem which can be transformed into the infinite-horizon problem by adding a simple Poisson probability of death each period, providing a simple justification for using an infinite-horizon problem as a learning target. Additionally, both Gourinchas and Parker (2002) and Cagetti (2003) note that households in the data appear to act as though

they are solving infinite-horizon problems until around age 45-55, at which point they apparently realize that retirement is fast approaching and begin to save as though they are solving a finite-horizon problem as in Appendix A).

3.1 Infinite Horizon Consumer Problem

The basic household consumption-savings problem under uncertainty can be stated as follows. Households have an uncertain income and may either save money at a risk-free rate of return or spend it immediately on consumption. Consumption produces rewards for the household via a utility function; this utility function is used to create a lifetime objective function. Savings produce rewards only in so far as they represent the ability of the household to eventually consume in the future.

The objective is to maximize total expected lifetime utility. The household problem, then, can be stated as follows:

$$\begin{aligned} \max_{\{c_t\}_{t=0}^{\infty}} \; & \mathbb{E}_0 \left[\sum_{t=0}^{\infty} \beta^t u(c_t) \right] \\ & s.t. \\ m_{t+1} &= \mathsf{R}(m_t - c_t) + y_{t+1} \;\; \text{the law of motion,} \\ c_t &\geq 0 \\ m_t &\geq 0 \\ & m_0 \; given. \end{aligned} \tag{1}$$

Where:

- y_t is a random income shock each period, normalized and distributed as an IID lognormal random variable with mean 1 and $\sigma_y = 0.2$,
- c_t is consumption in period t,
- m_t is "cash-on-hand," the total **m**onetary resources under control of the household after interest accrues on savings and wages y_t are paid,
- $a_t \equiv (m_t - c_t)$ is savings,
- $R = 1.03$ is deterministic, risk-free return on savings every period,
- $\beta = 0.95$ is the discount factor, and
- $u(.)$ is a Constant Relative Risk Aversion (CRRA) utility function with risk aversion $\rho = 3$.

The calibrated parameter values are summarized in Table (1). Utility takes the CRRA form:

$$u(c) = \frac{c^(1-\rho)}{1-\rho}.$$

Carroll (1997) and Carroll (2001b) provide excellent background for the usage of this functional form for this problem (and related difficulties).

Table 1: Parameters and Sources

Parameter	Value	Description	Source
β	0.95	Geometric discount factor	Allen and Carroll (2001)
ρ	3.0	Risk aversion	Allen and Carroll (2001)
σ_y	0.2	Shock to income	Carroll (1992)
R	1.03	Return factor, savings	Modeler choice

The solution to the discounted dynamic optimization problem (1) is the infinite sequence of consumption choices $\{c_t\}_{t=0}^{\infty}$ which maximizes the problem's expected discounted utility stream. Regularity conditions on both $u(.)$ and the law of motion $m_{t+1} = \mathsf{R}(m_t - c_t) + y_{t+1}$ guarantee that a solution exists for this problem, and in addition, the optimal consumption vector $\{c_t^*\}_{t=0}^{\infty}$ can be produced by an optimal policy (consumption) function $c_t^* = c^*(m_t)$ which is a function of the state space. Thus the apparently intractable problem of choosing an infinite-length vector becomes a simpler problem of choosing a function which maps an infinite stream of states provided by the law of motion into a stream of consumption choices.

If I denote the optimal consumption policy function c^* then the optimal value function $v^*(m)$ is simply the value of starting in state $m = m_0$ in the expected discounted infinite sum in problem (1):

$$\begin{aligned} v^*(m) &= \mathbb{E}_0 \left[\sum_{t=0}^{\infty} \beta^t u(c^*(m_t)) \right] \\ &s.t. \\ m_{t+1} &= \mathsf{R}(m_t - c_t) + y_{t+1} \\ c_t, m_t &\geq 0; \;\; m = m_0 \; given. \end{aligned} \tag{2}$$

Note that the above relationship holds not only for the optimal consumption function c^*, but also any arbitrary consumption function such that the appropriate Blackwell Sufficiency Conditions hold for the problem.[2] Let an arbitrary consumption function be parameterized by a set of parameters, which I will collectively denote θ, and define such a consumption function as c^θ. Then under the appropriate conditions, the associated value function can be derived from c^θ as follows:

$$\begin{aligned} v^\theta(m_0) &= \mathbb{E}_0 \left[\sum_{t=0}^{\infty} \beta^t u(c^\theta(m_t)) \right] \\ &s.t. \\ m_{t+1} &= \mathsf{R}(m_t - c_t) + y_{t+1} \\ c_t, m_t &\geq 0; \;\; m_0 \; given. \end{aligned} \tag{3}$$

As discussed below, given an arbitrary value function v^{arb_1}, I can derive an associated policy function c^{arb_1}, and given an arbitrary policy function c^{arb_1} I can derive an associated value function v^{arb_2}. However this process will *not* produce $v^{arb_1} = v^{arb_2}$ unless I have discovered the optimal policy and value functions.[3]

[2] See Bertsekas (2012) for a full discussion of the requirements on the structure of the problem for there to exist a unique fixed-point value and policy function solution. Unless otherwise noted, the assumptions will be made for all proceeding discussion.

[3] Or rather, $|v^{arb_1} - v^{arb_2}| \leq \delta$ for some tolerance δ.

This property is key for the policy iteration dynamic programming solution and will provide key intuition for the regret learning algorithm to be defined below.

When the solution as described above exists, the problem can be re-written in Bellman form:

$$
\begin{aligned}
v(m_t) = \max_{c_t} \; & u(c_t) + \beta \mathbb{E}_t \left[v(m_{t+1}) | m_t, c_t \right] \\
& s.t. \\
m_{t+1} = & \mathsf{R}(m_t - c_t) + y_{t+1} \\
c_t \geq & \, 0 \\
m_t \geq & \, 0 \\
& m_0 \; given.
\end{aligned}
\tag{4}
$$

The following section will discuss when the solution to this problem exists. (See Carroll (2012b) for a lengthy discussion of when the solution for this problem exists.)

This form breaks down the consumer problem into a much simpler form: the agent is simply trading off between the utility of consumption today, denoted by $u(c_t)$, and the expected future lifetime utility of the consequence of the choice today, denoted by $\beta \mathbb{E}_t \left[v(m_{t+1} \right]$.

3.2 Buffer Stock Solution and Approximate Policy Function

Under mild conditions and the parameterization of this problem,[4] the optimal consumption function takes the form:

$$c^*(m_t) = E[y_t] + g(m_t - \bar{m}^*),$$

where g is a nonlinear function resulting from optimizing problem (1) and $\bar{m}^*$ is the target buffer stock savings level for liquid wealth which exists under my conditions. I abuse notation slightly by denoting both the consumption in period t as a function of the cash-on-hand state variable m_t: $c_t = c^*(m_t)$ in the case of the optimal consumption function, or $c_t = c^{\theta}(m_t)$ in the case of an arbitrary approximate consumption function parameterized by θ, to be described further below.

When a consumer experiences a shock which pushes m_t away from $\bar{m}^*$ he/she will consume such that in expectation next period, $E\left[m_{t+1}\right]$ will move towards $\bar{m}^*$. As noted in Carroll (2012b), the exact form of the function g is highly nonlinear and difficult to describe analytically. Allen and Carroll (2001) propose a simple piecewise linear approximation to this function which has extremely low welfare cost and an intuitive parameterization, which will be used for the learning algorithm outlined in this paper. A more extensive discussion of the properties of this approximation can be found in Allen and Carroll (2001), Özak (2014), and Palmer (2012). A first-order Taylor approximation taken around $\bar{m}^*$ gives us a linear consumption function with an intuitive interpretation:

[4]The condition in this version of the model is $\mathsf{R}\beta < 1$. Intuitively, this is a statement about the "patience" level of the consumer versus potential growth in savings. See Carroll and Samwick (1997) and Carroll (2012b) for detailed discussion regarding this condition.

$$\widetilde{c}^{\theta}(m_t) = E[y_t] + \kappa\left[m_t - \bar{m}\right], \tag{5}$$

where $\theta \equiv (\kappa, \bar{m})$ and the tilde "$\widetilde{c}$" denotes that this is an intermediate step to the final form c^{θ}. κ is the marginal propensity to consume out of wealth[5], and $\bar{m}$ has the same interpretation as before as the buffer-stock savings target. To obtain the final piecewise linear consumption-function form, I simply impose the liquidity constraint as follows:

$$c^{\theta}(m_t) = \begin{cases} m_t & if\ \widetilde{c}^{\theta}(m_t) \geq m_t \\ \widetilde{c}^{\theta}(m_t) & otherwise. \end{cases} \tag{6}$$

This complete restriction on borrowing is imposed here for simplicity of exposition. However note that the imposition of the borrowing constraint is not as restrictive as it may first seem. Carroll et al. (1992) outlines evidence for consumers facing a low but non-zero probability of a zero-income shock occurring at the annual frequency, which differs from the distribution of shocks typically faced by the consumer. Any rational consumer who faces such a process would self-impose a borrowing constraint each period alive and with a positive probability of a zero-income even each period this collapses to a complete liquidity constraint used here. Even if the consumer did not rationally self-impose this constraint, a rational lender may do so. Regardless, a straightforward extension is to implement a borrowing constraint which is linear in m_t.

3.3 Welfare Cost of Approximate Solutions

As noted in the discussion following expression (3), for an approximate policy function c^{θ} I can obtain an associated value function v^{θ}. If I have also solved for the optimal value function v^{*} for this problem, I can calculate a measure of welfare cost implied by following the approximate policy function c^{θ}. Following Allen and Carroll (2001), I call this value a "sacrifice value." For a consumer following the optimal consumption rule, the sacrifice value represents the maximum amount the consumer would be willing to pay to to avoid switching permanently to the non-optimal rule. For a parameter θ, denote this value ϵ^{θ} and derive it as follows:

$$\begin{aligned} v^{*}(m - \epsilon^{\theta}) &= v^{\theta}(m)\ \ \forall_m \\ \Leftrightarrow m - \epsilon^{\theta} &= v^{*\,-1}(v^{\theta}(m))\ \ \forall_m \\ \Rightarrow \epsilon^{\theta}(m) &= m - v^{*\,-1}(v^{\theta}(m))\ \ \forall_m, \end{aligned}$$

and, using the ergodic distribution F_m of m under the optimal policy c^{*}[6], calculate the expected sacrifice value as:

[5] To see this, take the derivative of $c^{\theta}(m_t)$ with respect to m_t.

[6] See Carroll (2001a) for a discussion of the ergodic distribution of cash-on-hand following an optimal buffer-stock rule, as well as the methodology used to generate the ergodic distribution.

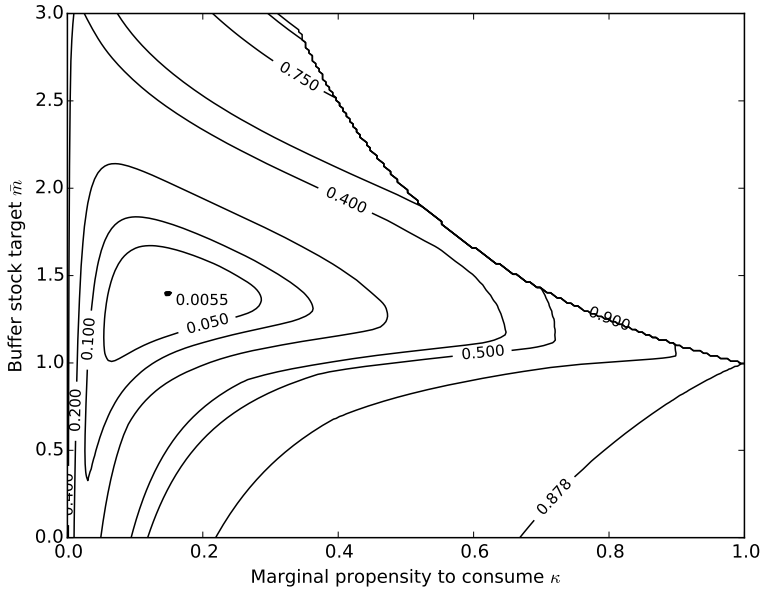


Figure 1: Sacrifice Values for Approximate Consumption Functions

$$\bar{\epsilon}^{\theta} = \int_{\bar{q}}^{\infty} \epsilon^{\theta} dF_m.$$

I use $\bar{\epsilon}^{\theta}$ to identify the expected sacrifice value for any given approximate rule c^{θ}. This provides an explicit way to compare non-optimal rules with the optimal solution, allowing us to map the effectiveness of learning and rules directly to more traditional methodology.

Since $\bar{\epsilon}^{\theta}$ can be calculated for any approximate function c^{θ}, it is simple to construct the surface of welfare costs over any given range of consumption function parameters. Figure (1) displays the contour plot of this surface for $\bar{m} \in [0,\ 3]$ along the y-axis and $\kappa \in [0,\ 1]$ along the x-axis. This should be read like a contour map – each line denotes approximate consumption rules which have equal sacrifice values. A few things can be quickly observed. The c^{θ} function with minimal sacrifice value – that is, the c^{θ} which is closest to c^{*} in utility terms – occurs at $\bar{m} \approx 1.4$, $\kappa \approx 0.15$ and has a sacrifice value of $\bar{\epsilon}^{\theta} = 0.0055$. I will denote the best approximate consumption function $c^{\theta *}$. That is slightly more than one half of one percent of expected annual income, very close to the true optimal rule. Since each line has the associated sacrifice value value printed on it, the inner-most ring around the minimal sacrifice point indicates a sacrifice value of 0.05, or 5 percent of expected annual income. The point which represents the "consume everything" consumption rule is at (1.0, 1.0); the sacrifice contour which intersects here has a value of 0.878.

The large "empty" region in the upper right-hand portion of Figure (1) is due to the fact that the parameters in this area produce consumption functions which tell a consumer to consume zero for a non-trivial number of m-values. For example, Figure (2) displays one such consumption function, with $\bar{m} = 2.5$ and $\kappa = 0.8$. It is clear that for some values of cash-on-hand, about $m \leq 1.25$, the consumption function tells the consumer to eat nothing. This is a particular problem for the consumer, because the CRRA utility function used here goes to infinity as consumption goes to zero: $u(c) \to \infty$ as $c \to 0$. Thus the boundary

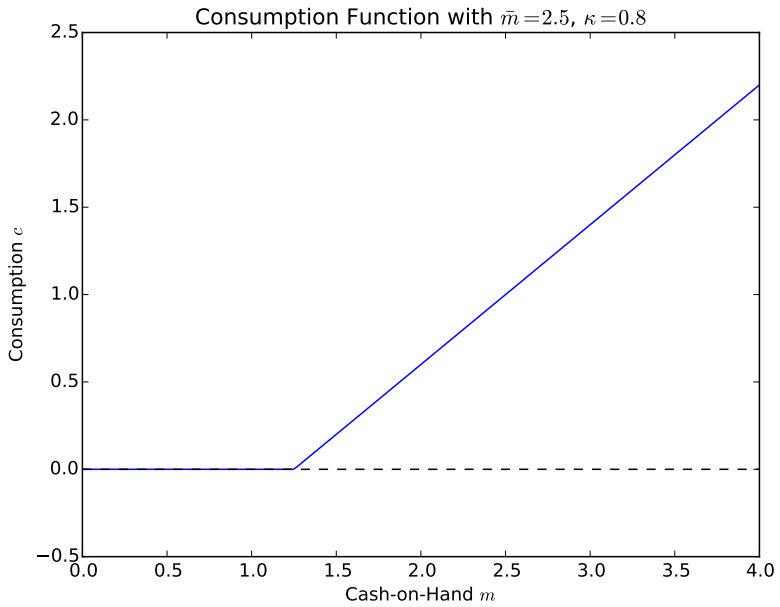


Figure 2: Example of a "Bad" Consumption Function

of the empty region in Figure (1) is simply those rules which would force an agent to consume zero over a non-trivial portion of the state space.

The reason for the small welfare cost of the best approximate consumption rule $c^{\theta *}$ can be quickly seen when examining the function against the true nonlinear optimal rule c^*. These are both displayed in Figure (3). Also displayed are the 5^{th} and 95^{th} cutoffs for the ergodic distribution of cash-on-hand m when following the optimal policy c^*, denoted F_m. Between the 5^{th} and 95^{th} percentiles, the best approximate policy is extremely close to the true optimal policy. Thus, for about 90% of the time, an agent using the approximate optimal rule $c^{\theta *}$ will be extremely close to the true optimal rule in welfare terms.

There is actually a deeper question at hand when one asks the welfare cost of a learning algorithm. The sacrifice value noted above is a value that assumes an agent follows the non-optimal rule for the rest of their lives – this is what the value function represents, and the value functions are used to calculate the sacrifice value. This is an important and reasonable first pass. However in learning agents are likely not using the same rule for the rest of their lives. The value function associated with the *entire learning process* is almost certainly different form that of following a specific rule. This "meta" value function of learning may not even be monotone, which is a requirement for measuring unique sacrifice values. At the intuitive level, if it takes a non-trivial set of periods in a finite life to try another policy, the opportunity cost of agent time is the best policy the agent has seen so far – presumably the current one. How does the agent decide when experimenting with a new policy is not worth it? There is a risk aversion question buried in this decision which has not been explored extensively in the economic literature. This is a question has been understood for a long time in the reinforcement learning literature. In that literature this question is known as the "exploitation/exploration trade-off," and is most cleanly illustrated by a set of simple problems called "bandit" problems. See Sutton and Barto (1998) and Auer et al. (2002) for an excellent description of both the exploration/exploitation trade-off and its illustration in bandit problems. I do not deal with this deeper question of the welfare costs of learning here, instead leaving this to future work. I will simply note that

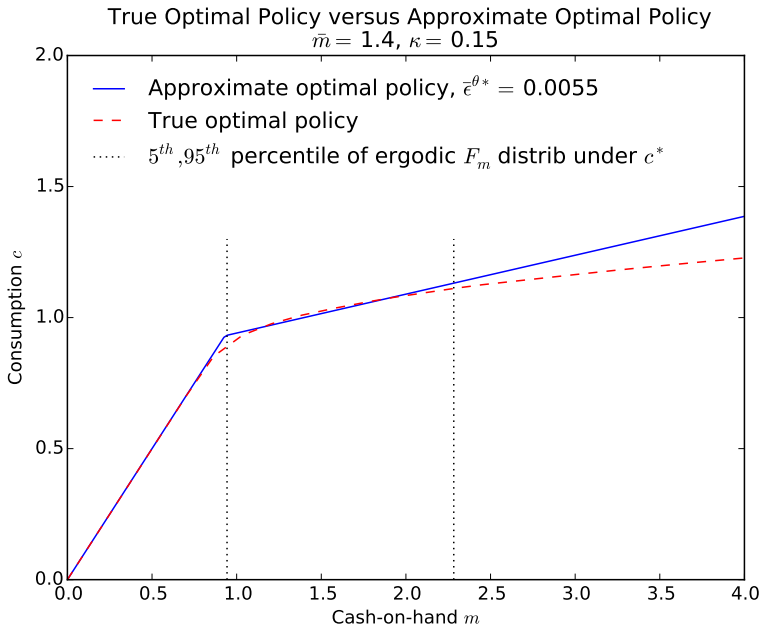


Figure 3: True vs Approximate Optimal Consumption Functions

the exploration / exploitation trade-off is a key fundamental question as soon as one introduces agents who learn policy functions by costly trial and error.

4 Solution Method

Regret learning is a form of policy iteration which replaces key steps in the traditional solution method with approximations constructed from a single agent's experience. In doing so, I impose on the agent the need to *optimize over time*. Rather than optimizing instantaneous each period, the regret learning agent will regularly improve their consumption function conditional on their most recent experience. With enough experience and time they can obtain near-optimal solutions, but even without extensive time the regret learning agent will settle into a stable distribution of rules near the optimal behavior.

I review a specific implementation of regret learning in this section, which uses the piecewise linear consumption function described in Section 3.2, and non-parametric beliefs about the distribution of shocks and states in their problem. I use a nonparametric approach because I want to show that an agent can still learn even when they have no priors at all about the stochastic elements of their system. The solution method is quite flexible, however. In principle any number of parametric functional forms could be used for the consumption function, and Bayesian learning about the distribution of shocks and states could replace nonparametric beliefs. Such extensions will be discussed in the conclusion.

Because regret learning is an approximation to the steps taken in policy iteration from traditional dynamic programming, I first briefly review policy iteration and an extension in Section . Following that I review the inherent difficulty of learning to optimize from experience in Section 4.3. Finally I outline regret learning itself in Section 4.4.

4.1 Traditional Dynamic Programming Solution

The traditional method of finding the optimal policy and value functions for a dynamic stochastic optimization problem such as (4) in dynamic programming[7]. There are two major practical methods of applying dynamic programming: value iteration, which constructs a sequence of value functions which converge to the optimal value function (from which an optimal policy function can be obtained) by iteratively applying the mapping (13) to an arbitrary initial value function, and policy iteration, which iteratively applies a two-step process: from an initial arbitrary policy function, find the associated value function, and from that value function, find the associated policy function. This leap-frog algorithm produces a sequence of policy functions and a sequence of value functions, both of which converge to their optimal counterparts. There are a number of extensions to both value and policy iteration; I will discuss one particular extension to policy iteration which will provide intuition and motivation for an aspect of regret learning below.

4.1.1 Requisite Notation

Before proceeding I define some necessary notation. If the readers is comfortable with the concepts behind dynamic programming and policy iteration in particular, he/she may skip straight to section (4.2) below, and reference back here as needed. These follow the notation of Bertsekas (2012), which defines the mathematical context of these ideas fully and rigorously and derives detailed multi-step proofs for convergence of policy iteration and optimistic policy iteration under appropriate conditions. The reader is encouraged to reference Bertsekas (2012) if any clarification is needed.

Define the transition function f as the function which maps state m_t, choice c_t, and shock y_{t+1} to the next-period state m_{t+1}. Then the right-hand side of the Bellman equation in problem (4) prior to applying the optimization step may be defined as:

$$H(m, c, v) = u(c) + \beta \mathbb{E}\left[v(f(m, c, y))\right]. \tag{7}$$

To avoid notational overload on the letter "c" when discussing policy iteration, I will replace the "c" function with a more broadly defined "μ" function, following the notation of Bertsekas (2012). After finishing discussion of policy iteration I will switch back to the specific notation of c^θ which pertains to the approximate parameterization of my particular problem. The policy (consumption) function μ maps the state m to a consumption choice: $c_t = \mu(m_t)$, and I define $\mathcal{C}(m)$ as the set of acceptable values implied by the law of motion:

$$\mathcal{C}(m) \equiv \left\{ c \;\; s.t. \;\; 0 \leq \; c \leq \left(\frac{y' + \bar{q}}{\mathsf{R}} + m \right) \right\}. \tag{8}$$

This is a general statement of the acceptable choice set $\mathcal{C}(m)$; note that it includes the next-period income shock y' and the borrowing constraint $\bar{q}$. If I impose the assumed zero borrowing limit and minimum possible income shock, I arrive at the simplified borrowing constraint which I employ in this paper:

$$\mathcal{C}(m) \equiv \{c \;\; s.t. \;\; 0 \leq c \leq m\}. \tag{9}$$

[7] See Bertsekas (2012) for a thorough discussion of this topic.

I can now define two mappings which take an arbitrary value function v and map it into a new value function. Given a policy μ[8] and a value function v, define the policy-specific mapping T_μ as:

$$(T_\mu v)(m) = H(m, \mu(m), v) \ \ \forall m. \tag{10}$$

I can also define the optimal mapping T which is independent of a particular policy function and once again maps a value function v into a new value function:

$$(Tv)(m) = \max_{c \in \mathcal{C}(m)} H(m, c, v) \ \ \forall m. \tag{11}$$

As can be seen, both mappings T and T_μ produce new value functions denoted Tv and $T_\mu v$, respectively. As above I denote the optimal policy and value functions as c^* and v^*, respectively; I can now re-write problem (4) as:

$$v^*(m_t) = (Tv^*)(m_t) = \max_{c_t \in \mathcal{C}(m_t)} H(m_t, c_t, v^*) \ \ \forall m. \tag{12}$$

This is the familiar result that the optimal value function v^* is the fixed point of the optimal mapping Tv^*, which may be expressed succinctly as

$$v^* = Tv^*, \tag{13}$$

where $\forall_m$ is assumed for simplification of notation. This result also follows from the fact that T is a contraction mapping on a complete space; as such a sequence constructed by applying the mapping T to an arbitrary initial value function v_0, k times will produce the sequence $\left\{T^k v_0\right\}$, which converges to the optimal fixed point value function v^*:

$$\lim_{k \to \infty} T^k v_0 = v^*. \tag{14}$$

Given the optimal value function v^* I can construct the expression $H(m, \mu m, v*)$ for any policy function μ; then the optimal policy function is simply the function μ^* which solves:

$$\begin{aligned} H(m, \mu^*(m), v^*) &= \max_{c_t \in \mathcal{C}(m)} H(m, c, v^*) \\ \Rightarrow \mu^*(m) &= \operatorname*{argmax}_{c_t \in \mathcal{C}(m)} H(m, c, v^*) \quad \forall_m. \end{aligned} \tag{15}$$

The definition of μ^* in equations (15) can be succinctly expressed as

[8] As noted in the appendix, I assume this policy is consistent with $\mathcal{C}(m)$; that is, $\mu(m) \in \mathcal{C}(m) \ \forall \ m$.

$$T_{\mu^*} v^* = T v^*. \tag{16}$$

Note that the relationship in (15) can be constructed for *any* arbitrary value function w, not only for the optimal value function v^*. This produces a not-necessarily-optimal policy function μ^w which is associated with the arbitrary function w:

$$\begin{aligned} H(m, \mu^w(m), w) &= \max_{c_t \in \mathcal{C}(m)} H(m, c, w) \\ \Rightarrow \mu^w(m) &= \underset{c_t \in \mathcal{C}(m)}{\operatorname{argmax}} \; H(m, c, w) \quad \forall_m. \end{aligned} \tag{17}$$

Like T, T_μ is also a contraction mapping on a complete space[9] and like T each has a unique fixed point, denoted v^μ. That is, for a given policy μ there is a unique fixed point value function v^μ which fulfills,

$$v^\mu = T_\mu v^\mu, \tag{18}$$

which can be found be repeatedly applying the mapping T_μ to an arbitrary initial value function to produce a convergent sequence $\left\{T_\mu^k v_0\right\}$ such that:

$$\lim_{k \to \infty} T_\mu^k v_0 = v^\mu. \tag{19}$$

Again, this follows from the contraction mapping properties of T and T_μ in these contexts. Equations (18) and (19) provide a way to construct the fixed points associated with the optimal mapping T and policy-specific mapping T_μ.

4.2 Policy Iteration and Optimistic Policy Iteration

Policy iteration is an intuitive process which "ratchets" the value function associated with an arbitrary initial policy function to the optimal value and policy functions. I can define the policy iteration algorithm using equations (18) and (16) from above. Start with an arbitrary initial policy function μ^0 then iteratively apply the following two steps:

4.2.1 Policy Iteration (PI) Algorithm

- Step (1) **Policy Evaluation**: Given policy μ^k, find the unique value function v_{μ^k} which is the fixed point of T_{μ^k} as in equation (18):
$$v_{\mu^k} = T_{\mu^k} v_{\mu^k}.$$
- Step (2) **Policy Improvement**: Given value function v_{μ^k}, obtain the new policy μ^{k+1} which is optimal with respect to v_{μ^k}. I can state the definition of μ^{k+1} explicitly as the "greedy" solution with respect to

[9] See Bertsekas (2012) for the requisite assumptions and proofs for both statements.

$H(x, u, v_{\mu^k})$, as $\mu^{k+1}(x) = \text{argmax}_{c \in \mathcal{C}(x)} \; H(x, c, v_{\mu^k}) \; \forall \; x \in X$, and I can express the implicit definition of μ^{k+1} succinctly as:

$$T_{\mu^{k+1}} v_{\mu^k} = T v_{\mu^k}$$

Repeat this process until the distance between $v_{\mu^{k+1}}$ and v_{μ^k} falls within some acceptable tolerance, or in the case of a finite state and choice space, $v_{\mu^{k+1}} = v_{\mu^k}$.

The key step is the policy improvement step: defining the next policy function as optimal with respect to the previous value function. In a technical sense this acts as a sort of "utility ratchet" which guarantees that the next value function iterate will be ever closer to the true value function.

The following section outlines a variation of policy iteration from dynamic programming called "optimistic policy iteration", a version of policy iteration which employs only partial fixed-point iterations on the policy improvement step. Remarkably, executing the policy evaluation step incompletely *still guarantees convergence.* As noted in Ljungqvist and Sargent (2012)[10], policy iteration and optimistic policy iteration are often much faster than the equivalent value iteration solution for the same problem. As described in Bertsekas (2012), the reason for this falls out of the proofs of convergence: the value functions associated with each step in the or both policy iteration and optimistic policy iteration algorithms are bounded *between* those of the value iteration algorithm and the true optimal value function. That is, only in the *worst* case are policy iteration and optimistic policy iteration are as far from the true value function as is value iteration. I turn to that result briefly now.

4.2.2 Optimistic Policy Iteration (OPI) Algorithm

The optimistic policy iteration algorithm is extremely similar, only slightly modifying the policy evaluation step. The steps occur in the opposite order and notation differs slightly.

Start with an arbitrary initial value function v_0, then iteratively apply the following two steps:

- Step 1 - **Policy Improvement**: Obtain policy μ^k which solves $H(x, \mu^k(x), v_k) = \max_{c \in \mathcal{C}(x)} H(x, c, J_k)$. That is, given v_k, find the policy μ^k consistent with

$$T_{\mu^k} v_k = T v_k.$$

- Step 2 - **Optimistic Policy Evaluation**: Given value function v^k and policy μ_k, execute the optimistic value function update to obtain v_{k+1}, which results from the application of mapping T_{μ^k} to v^k m_k times. That is:

$$v_{k+1} = T_{\mu^k}^{m_k} v_k.$$

Repeat the process at step 1 using the new value function v_k until convergence.

Note that as with policy iteration above, the policy improvement step can be satisfied by constructing μ^k as the "greedy" policy with respect to $H(x, c, v_k)$:

[10] Ljungqvist and Sargent (2012) use the term "modified policy iteration." I prefer the Bertsekas (2013) term "optimistic policy iteration" and use it here.

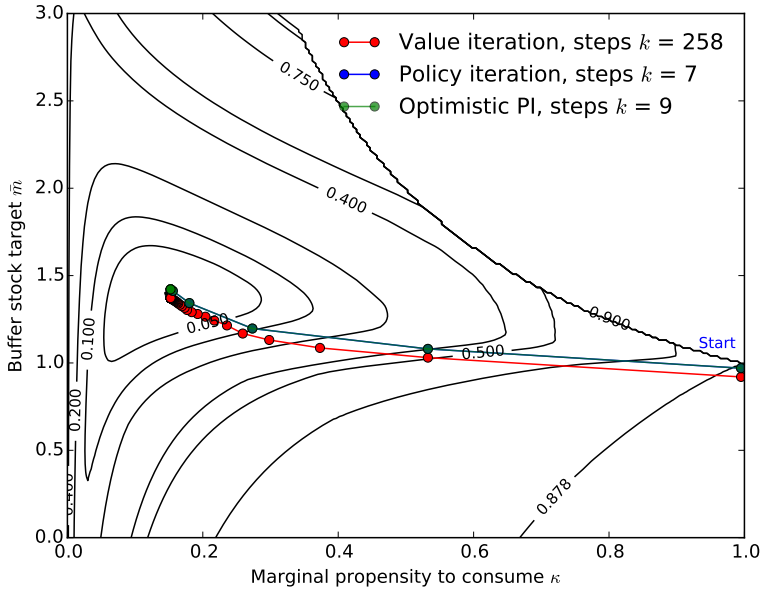


Figure 4: Steps to Converge for Policy Iteration, Value Iteration, and Optimistic Policy Iteration on Welfare Surface

$$\mu^k(x) = \text{argmax}\, c \in \mathcal{C}(x)\ H(x, c, v_k)\ \forall\ x \in X.$$

Optimistic policy iteration algorithm differs from policy iteration in the policy evaluation step: instead of iterating to the fixed point of mapping (18), only a finite number of mappings, m_k, are applied to the value function at the beginning of each policy evaluation step k. Surprisingly, this process still produces a sequence of value and policy functions which converge to the optimal functions. As with policy iteration, the optimistic policy value function created at each step k is squeezed to the optimal value function by the equivalent steps in value iteration – once again value iteration is a worst-case bound on convergence.

Figures (4), (5) and (6) compare policy iteration, optimistic policy iteration, and value iteration. In many ways optimistic policy iteration falls between policy and value iteration at each extreme. When $\{m_k\} = 1\ \forall\ k$, optimistic policy iteration acts like value iteration; when$\{m_k\} \to \infty\ \forall\ k$, optimistic policy iteration acts like vanilla policy iteration.

As an illustration, Figure (4) demonstrates the number of algorithm steps k required to find the optimal solution, for the same tolerance value, for policy iteration, value iteration, and optimistic policy iteration. The path taken by each algorithm starts at the "consume everything" policy function and is projected onto the welfare surface discussed earlier. For optimistic policy iteration I choose $\{m_k\} = 40\ \forall\ k$ mappings for each policy evaluation stage, for reasons which will be made clear below. The value iteration steps are offset slightly for visual clarity ($\bar{m} - 0.05$).

Vanilla policy iteration takes $k = 7$ steps to meet the chosen convergence criteria, while value iteration requires 258. Optimistic policy iteration with $m_k = 40$ mappings at each policy evaluation stage only requires $k = 9$ steps to terminate, and each step is nearly identical to the policy iteration steps.

Figure (5) shows how many steps optimistic policy iteration would require to terminate at the optimal solution as we vary m_k from 1 to 130 on the x-axis. At $m_k = 1$ optimistic policy iteration takes exactly as

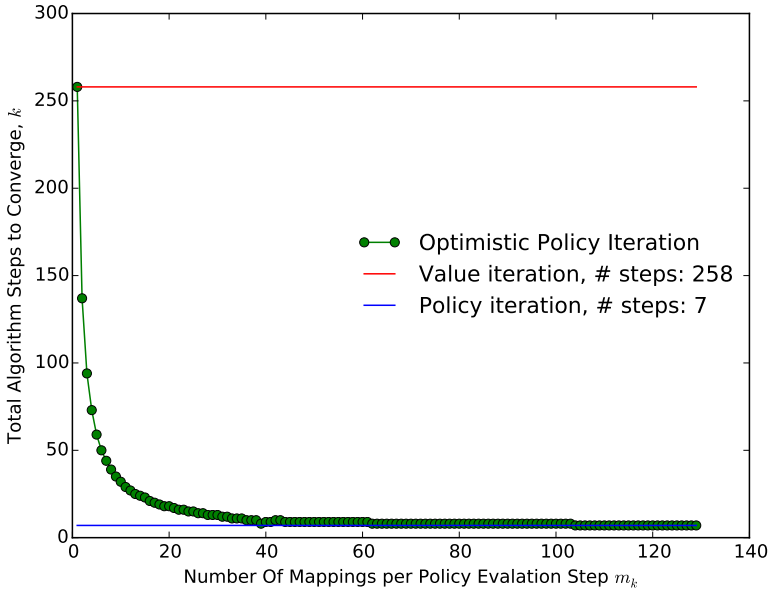


Figure 5: Steps to Converge for Policy Iteration, Value Iteration, and Optimistic Policy Iteration as m_k Varies

many steps as value iteration, represented by the red flat line. This descends steadily until around $m_k = 104$, at which point optimistic policy iteration terminates in the same number of steps, $k = 7$, as vanilla policy iteration in the blue line. (At $m_k = 10,000$, not shown, this still holds true.)

When m_k is small, each step of optimistic policy iteration executes quickly – so quickly, in fact, that even a large number of k-steps may execute faster than a few steps of vanilla policy iteration. Figure (6) demonstrates this tradeoff by plotting the total clock time on the y-axis against number of mappings per step, m_k, on the x-axis. Value iteration, the red line, is clearly above vanilla policy iteration, the blue line. We can see that around $m_k = 7$ the clock time to find a solution by optimistic policy iteration falls below the clock time for vanilla policy iteration and stay there the remainder of the plot. The minimum occurs around $m_k = 40$.

This variation on policy iteration is highlighted for two reasons: first, to show that variations on policy iteration which do not fully estimate v^θ at each step can none-the-less converge to the optimal solution (an intuitive motivation for the value estimation step of regret learning), and second, as an outline for future extensions of regret learning along lines similar to optimistic policy iteration, to be discussed further in the conclusion.

4.3 Inherent Difficulty of Learning to Optimize from Experience

Before diving into the details of the regret learning, I take a moment to think about the difficulties facing an agent attempting to form these estimations from experience. As noted by Allen and Carroll (2001) and Sutton and Barto (1998), the simplest way to approximate the value function associated with a given policy is via Monte Carlo estimation of the sum in expression (3), repeated here for convenience:

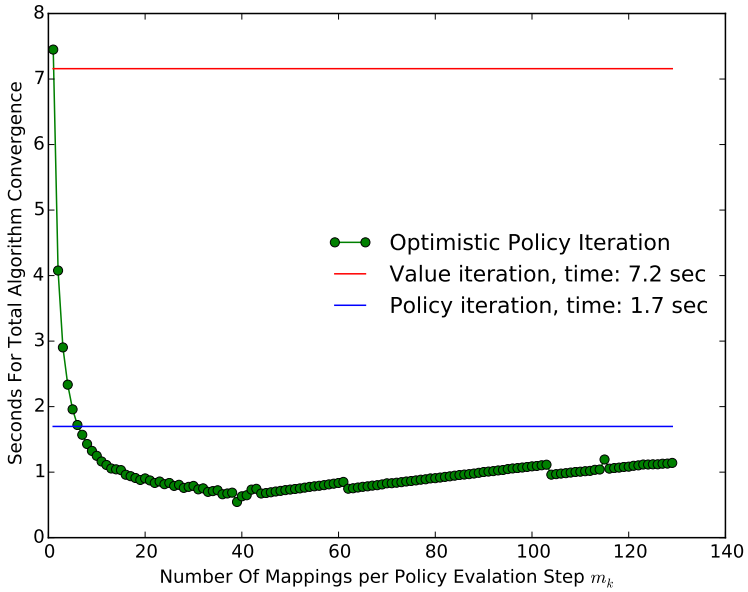


Figure 6: Clock Time to Converge for Policy Iteration, Value Iteration, and Optimistic Policy Iteration as m_k Varies

$$
\begin{aligned}
v^{\theta}(m_0) &= \mathbb{E}_0\left[\sum_{t=0}^{\infty}\beta^t u(c^{\theta}(m_t))\right] \\
&s.t. \\
m_{t+1} &= \mathsf{R}(m_t - c_t) + y_{t+1} \\
c_t, m_t &\geq 0; \quad m_0 \; given.
\end{aligned}
\tag{20}
$$

As luck would have it, agents have access to a data-generating process for the y_t process: their own income experience.[11]

However there are two immediate problems with agents using own experience to estimate v^{θ} as a step in policy iteration. First, agents clearly cannot estimate $v^{\theta}(m)$ for infinite periods – this would preclude them from ever advancing past the first policy evaluation step to the first policy improvement step. Second, agents cannot possibly hope to form an unbiased estimator of the expectation in problem (20) in a traditional Monte Carlo sense. To see why, first consider what an ideal method of moments Monte Carlo estimator based on agent experience would look like.

Denote the entire income experience of an agent i as

$$\vec{y} = [y_0, y_1, ..., y_t, ...] \subset \mathbb{R}^{\infty},$$

and denote a k^{th} subset of this income experience, which I will call an "episode," as

[11] In the simplest version of regret learning presented here, I restrict agents to only have access to their own income streams – they can learn nothing from other agents who may experience similar shocks.

$$\vec{y}_k = \left[y_{t_k}, y_{t_k+1}, ..., y_{t_{k+1}-1} \right],$$

where a subset of equidistantly-spaced time indices $\tau = \{t_k\}_{k=0}^{\infty},\ k \in \mathbb{N}$ define the borders of subsets of time experience and $D \equiv (t_{k+1} - t_k)\ \forall_k$. Form a "single stream" estimate of v^θ in expression (20) using one of these k^{th} subsets:

$$\begin{aligned} w_k^\theta(m) &= \sum_{t=0}^{D-1} \beta^t u(c^\theta(m_t)), \\ &s.t. \\ m_{t+1} &= \mathsf{R}(m_t - c_t) + y_{t+1} \\ c_t, m_t &\geq 0; \quad m_0 = m\ given. \end{aligned} \tag{21}$$

To form a method of moments estimate of the expectation in expression (20), ideally the agent would be able to repeat this experience independently many times, for many different starting values of m – say, K times for every $m \in M$. Then the agent i could form the following Monte Carlo estimator for v^{theta}:

$$\bar{w}^\theta(m) = \frac{1}{D} \sum_{k=0}^{K} w_k^\theta(m_0)\ \ \forall_m \in M. \tag{22}$$

Note that there are three dimensions which the agent needs to send to infinity to have the $\bar{w}^\theta$ expression be a consistent and unbiased estimator for $v^\theta(m)$. It is clear that I need both $D \to \infty$ and $K \to \infty$ – that is, I need the *length of each episode* to go to infinity, as well as the *number of episodes per state* m to go to infinity. In addition, since $m \in \mathbb{R}$, I need the number of starting values of m, N_m, to go to infinity.

The problem here is somewhat analogous to the "curse of dimensionality" in traditional dynamic programming, only here it is a "curse of temporality" – the agent who uses own experience as a Monte Carlo data generating process simply *does not have enough time* to form a "good" estimate of the value function v^θ. Finally, *even if* the agent had the time to do this, there is an additional difficulty. To achieve a consistent and unbiased Monte Carlo estimator of $v^\theta(m)$ I need some iid properties. I need the initial m_0 values at which each episodic estimate of $w_k^\theta(m_0)$ is estimated to be iid.[12] I also need the income episodes themselves, $\vec{y}_k$, to be iid in k; while this property is fulfilled for my current problem it will not be fulfilled in general.

Importantly, all of these considerations are only for the first step in the first iteration of policy iteration. If I want to execute policy iteration using agent experience as the data generating process, I must be able to repeat the above steps *indefinitely* until the sequence of policy and value functions converge. The curse of temporality for estimating value functions from an agent's experience appears unavoidably intractable.

There is hope, however; each of the points above is not as insurmountable as they first appear. First, optimistic policy iteration tells us that the exact estimation of v^θ is not needed at each iteration of the policy iteration algorithm – in fact the fixed point calculation in step 1 can be curtailed far short of achieving the estimate of v^θ and convergence is still assured! The key here is that the policy improvement step "ratchets" the next-period policy towards the optimal policy in utility terms, conditional on the current value function,

[12] Ideally the initial m values are distributed as the ergodic distribution of m under the rule θ, presuming that distribution exists

regardless of what that current value function is.

The first-pass regret-learning agents presented here will rely on a similar mechanism. Instead of the partial fixed-point calculation of optimistic policy iteration seen in equation (4.2.2) for OPI, regret learning agents will implement a noisy estimation of the fixed point value calculation in vanilla PI, equation (4.2.2), following a process similar to the one described in equation (21).

In order to implement the second policy improvement step, equation (4.2.1), the agent will need to estimate the value function for the entire state space of m. Fortunately the agent will not need to implement estimation (4.2.2) for an infinite number of points to obtain an effective representation of the function $v^{\theta}(m)\ \forall m$; instead the agent will simply estimate a version of this for a grid of points over the state space, and interpolate between these points. As noted in Pál and Stachurski (2013) and Stachurski (2009), it is known that a non-expansive function approximation, such as linear interpolation or nearest neighbor interpolation, will preserve the contraction properties of the T mapping, and thus can preserve convergence properties as well.[13] The agent using regret learning will use a form of linear approximation which is constructed to be convergent in expectation to the function v^{θ}, employing the law of total probability. This form of approximation is motivated by the observation that human decision-makers often divide the state-space of dynamic optimization problems into discrete "chunks" to make the problem tractable, and then proceed with solution Vanderbilt (2013). Informally, this "chunking" process can be seen in many economic papers, macroeconomic papers in particular, wherein the dynamic process for, say, the employment matching technology, or house prices, is divided into a discrete set of states, usually an odd number, and usually given labels such as "Good, Average, Bad" (for three states) or "Good, Medium-Good, Average, Medium-Bad, Bad".[14]

4.4 The Regret Learning Solution

Consider the agent problem presented in expression (1), restated here for convenience:

$$\begin{aligned}
&\max_{\{c_t\}_{t=0}^{\infty}} \mathbb{E}_0 \left[\sum_{t=0}^{\infty} \beta^t u(c_t) \right] \\
&\qquad s.t. \\
&m_{t+1} = \mathsf{R}(m_t - c_t) + y_{t+1} \ \text{ the law of motion,} \\
&c_t \geq 0 \\
&m_t \geq 0 \\
&\qquad m_0 \ given.
\end{aligned}$$

An agent i begins with an initial policy $c^{\theta_{k=0}}$. Although I will refer to agent i, all subscripts denoting agent i are dropped from the following discussion to reduce notational clutter. The agent experiences an indefinitely long income stream $\vec{y}$, subdivided into D-length episodes, each episode denoted by the subscript k. For example, the first $k = 1$ episode would be denoted $\vec{y}_k \equiv [y_0, y_1, ..., y_t, ..., y_{D-1}]$. For the duration of the episode, the agent simply follows the current consumption policy and records a rolling sum of discounted

[13] This formal result has been demonstrated in practice for some time. As noted in Carroll (2012a) however, one must be very cautious about errors growing outside of the approximation grid defined by the researcher. The interpolation procedure proposed here in part seeks to address these concerns via an endogenous selection of the grid such that the highest-probability observations as determined from experience occur in exactly the most well-approximated locations.

[14] See for example the aggregate shock process to income in Krusell and Smith Jr (1998).

utility of the consumption experienced. At the end of the D-length episode k, the agent has formed a rough approximation, $\hat{v}^{\theta_k}$, to the true value function v^{θ} associated with c^{θ}.

Using this newly generated value function $\hat{v}^{\theta_k}$, which the agent did not have at the beginning of the episode, the agent can look back on the shocks experienced and ask, "if I knew then what I know now" – namely the $\hat{v}^{\theta_k}$ function – "what optimizing consumption choices *should* I have made?" This provides the agent with a set of choices that *would* have been optimal under the $\hat{v}^{\theta_k}$ value function. Call these consumption choices the "regret choices" – what the agent *should* have done, if they'd only known what they know now. The agent then finds the consumption function of the form in equation (6) which is closest to the regret choices in a sum of squared errors sense. This produces the next consumption function to employ, $c^{\theta_{k+1}}$, and the process is repeated.

Thus in a specific, technical sense, the agent is using "regret" – the act of looking back on past choices and determining what they should have done given knowledge learned from experience – to ascertain the next best course of action. As will be described in the Section (5), this can provide agents with a convergence to a distribution around the optimal solution.

The following sections describe four key elements needed to implement regret learning's experience-based version of policy iteration. These are:

- approximating the distribution of shocks $\mathcal{F}_y$ and state space under the current policy $\mathcal{F}_m^{\theta}$,
- $\mathbb{E}_t\left[v^{\theta}(m_{t+1})|m_t, c_t\right]$,
- identifying regret choices, and
- updating the consumption function by minimizing regret.

These will each be discussed in turn in the following sections, before a complete definition of regret learning is provided.

4.5 Approximating State and Shock Distributions

Consider a single episode of experience indexed by k. Recall that the income experienced during this episode is denoted

$$\vec{y}_k = \left[y_{t_k}, y_{t_k+1}, ..., y_{t_{k+1}-1}\right].$$

The corresponding states visited during this episode resulting from the shocks and application of the consumption function are likewise denoted

$$\vec{m}_k = \left[m_{t_k}, m_{t_k+1}, ..., m_{t_{k+1}-1}\right].$$

Denote the cumulative density function of the state variables m under a particular consumption function c^{θ} (and under a particular income process) as $\mathcal{F}_m^{\theta}$, and denote its inverse, the quantile function, as $\mathcal{Q}_m^{\theta} \equiv \mathcal{F}_m^{-1,\,\theta}$. Denote the empirical counterparts to both of these, constructed with sample size D, as $\hat{\mathcal{F}}_{m,D}^{\theta}$, and $\hat{\mathcal{Q}}_{m,D}^{\theta}$, respectively. Both empirical counterparts converge to the true functions as $D \to \infty$.[15] The empirical income shock distribution, $\hat{\mathcal{F}}_{y,D}$, is likewise calculated using current-episode data.

[15] While the empirical cumulative density function (ECDF) has a unique form, there are a number of way to construct its inverse, the empirical quantile function, due to the step-function nature of the ECDF. I employ the median-unbiased quantile function estimator recommended by Hyndman and Fan (1996).

Importantly, this non-parametric approach uses data from the current episode only. All information is forgotten in a technical sense after updating the consumption function, as will be shown. Obvious extensions are modeling these via Bayesian learning about each, using the posterior distribution from previous episodes as the prior for the current episode.

4.6 Approximating the Conditional Expectation

The following section outlines the approximation of the conditional expectation $\mathbb{E}_t\left[v^\theta(m_{t+1})|m_t, c_t\right]$ using agent experience. This is a detailed section, divided into a number of substeps. The key intuition is that the agent will use the law of total probability to approximate the conditional expectation using a *fixed value function* over a set of "bins" that represent the state space, and a variable *conditional probability* that each "bin" will be experienced, conditional on choices made. The agent will use their very limited experience to approximate the following expression, which will allow them to consider "what they might have done differently" when updating their policy function:

$$\mathbb{E}_t\left[v^\theta(m_{t+1})|m_t, c_t\right] = \sum_{B\in\mathcal{B}} \mathbb{E}\left[v^\theta(m_{t+1})|m_{t+1}\in B\right] \times \mathsf{prob}(m_{t+1}\in B|m_t, c_t).$$

As a notational shorthand, define the following:

$$\begin{aligned}\mathbb{E}\left[v^\theta(B)\right] &\equiv \mathbb{E}\left[v^\theta(m_{t+1})|m_{t+1}\in B\right],\\ \mathsf{prob}(B|m_t, c_t) &\equiv \mathsf{prob}(m_{t+1}\in B|m_t, c_t).\end{aligned}$$

Now expression (4.6) can be stated succinctly as:

$$\mathbb{E}_t\left[v^\theta(m_{t+1})|m_t, c_t\right] = \mathbb{E}\left[v^\theta(B)\right] \times \mathsf{prob}(B|m_t, c_t).$$

The regret learning agent will approximate the two elements of the right hand side of this expression to obtain an approximation to $\mathbb{E}_t\left[v^\theta(m_{t+1})|m_t, c_t\right]$, which will be used to determine regret choices.

The three subsections that follow will outline the steps taken to do this:

- creating the partition $\mathcal{B}$ over the state space,
- creating an estimate of the expected value of each "bin" B, $\mathbb{E}\left[v^\theta(B)\right]$, and
- calculating the conditional probability of each "bin" B occurring, $\mathsf{prob}(B|m_t, c_t)$.

4.6.1 Determining the State-Space Partition $\mathcal{B}$

A key element to regret learning is determining a equiprobable partition of the state space at the end of an episode k of agent experience. Because the state space is the real line, this amounts to choosing a set of partition boundaries,

$$b_k \equiv [b_{k,0}, b_{k,1}, ..., b_{k,n}, ..., b_{k,N}] \subset \mathbb{R}^N, b_{k,n} \leq b_{k,n+1}\ for\ n = 0, 1, ..., N,$$

such that each subset of the line is contiguous. For simplicity, call each subset of the partition a "bin;" a partitioning might be coarse (few bins) or fine (many bins). Note that N is the number of bins in the partition.

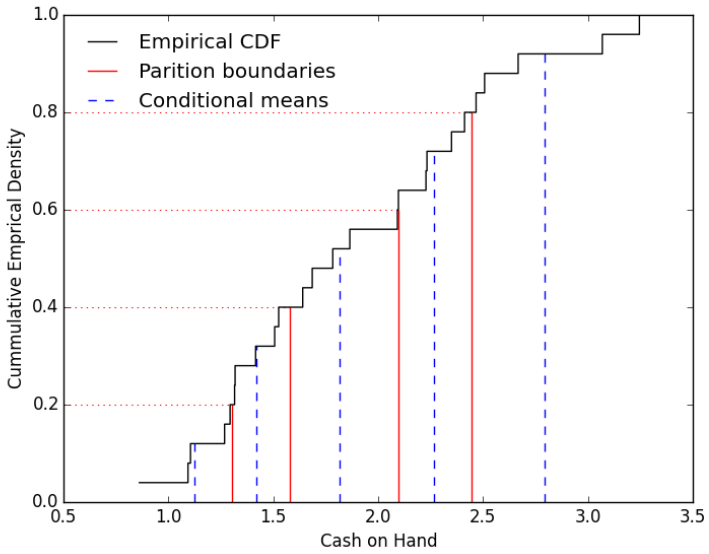


Figure 7: Partitioning the m Space: 5 bins, $D = 25$ learning periods

A coarse partitioning might be motivated by the idea that human behavior, in attempting to solve complex optimization problems, appears to involve "chunking" of the problem space into self-similar groups to offer more efficient solution of these approximate problems.[16]

The points chosen to represent each partition endogenously reflect the experience the agent has had, for better or for worse. If the agent encounters an unusual set of income shocks, the partition of the state space will endogenously reflect this. This is achieved via the simple equiprobable partitioning of the state space based on the agent's experience. Importantly, this partition is construct from experience in such a way that it will converge to an appropriate object as learning time goes to infinity.

Recall the empirical quantile function described in Section : $\hat{\mathcal{Q}}^{\theta}_{m,D}$. Given N, the number of subsets in the partition, simply find the partition boundaries for episode k as follows:

$$b_k \equiv [b_{k,0}, b_{k,1}, ..., b_{k,n}, ..., b_{k,N}], \text{with elements defined,}$$
$$b_{k,n} = \hat{\mathcal{Q}}^{\theta}_{m,D}\left(\frac{n}{N}\right), \ for\ n = 0, 1, ..., N.$$

Figure (7) provides a visualization of this process for a hypothetical agent who has experienced 25 states in an episode ($D = 25$) and who has 5 coarse bins to summarize experience ($N = 5$). The solid red vertical lines represent the boundaries of the bins, B_k; the dashed blue lines indicate the conditional means of each bin, which will be defined as $\hat{M}^k$. The dotted red lines demonstrate that the space has indeed been partitioned into sections with equal probability of occur according to the empirical quantile function $\hat{\mathcal{Q}}^{\theta}_{m,D}$. (Recall that the empirical quantile function algorithm used is median-unbiased.)

Denote the bins defined by the boundaries b_k as follows:

[16]This approach is taken in approximate dynamic programming and reinforcement learning in computer science; see Powell (2007) and Sutton and Barto (1998) for discussion and references, and see (Vanderbilt, 2013) for a particularly interesting intersection of approximate dynamic programming and real-world human problem-solving behavior.

$$B_{k,n} = [b_{k,n-1}, b_{k,n}] \, for \; n = 1, 2, ..., N,$$

and the set of all these bins as:

$$\mathcal{B}_k = \{B_{k,n} = [b_{k,n-1}, b_{k,n}] \, for \; n = 1, 2, ..., N\},$$

where the k is often dropped for notational convenience.

For convenience, denote the finite sets which are defined by $B_{k,n}$ and $\vec{m}_k$ as

$$\begin{aligned} \mathcal{A}_k &\equiv [A_{k,1}, ..., A_{k,n}, ..., A_{k,N}], \text{ where,} \\ A_{k,n} &\equiv \{m \; s.t. \; m \in \vec{m}_k, m \in B_{k,n}]. \end{aligned}$$

Because of the discrete nature of the empirical density used, the probability that each bin occurs is not exactly $\frac{1}{N}$ but rather,

$$\begin{aligned} p_k = [p_{k,1}, ..., p_{k,n}, ...p_{k,N}], \;\; & where \\ p_{k,n} &= \frac{\#A_{k,n}}{D}. \end{aligned}$$

The operator # denotes the number of elements in set $A_{k,n}$.

The agent will use this partitioning to form an experienced-based estimate of the value function. To do so, the agent will need to identify specific points to represent the bins such that nice statistical properties are maintained. The straightforward answer is to define the grid $\hat{M}^k$ as the empirical conditional expected value of each bin $A_{k,n}$:

$$\begin{aligned} \hat{M}^k &= [M_{k,1}, ..., M_{k,n}, ..., M_{k,N}], \text{ where,} \\ M_{k,n} &= \frac{1}{\#A_{k,n}} \sum_{m \in A_{k,n}} m. \end{aligned}$$

where each $M_{k,n}$ converges to the conditional expectation $\mathbb{E}[m \mid m \in A_{k,n}]$ as $D \to \infty$.

As noted, Figure (7) provides a visualization of this process for a hypothetical agent who has experienced 25 states in an episode ($D = 25$) and who has 5 coarse bins to summarize experience ($N = 5$).

Note that, by construction,

$$\frac{1}{N} \sum_{m \in \vec{m}_k} m = M_{k,n} \bullet p_{k,n}.$$

These will be used together to form the conditional expectation of the value function in the next sections.

4.6.2 Approximating the Expected Value of Being in a Bin $\mathbb{E}\left[v^{\theta}(B)\right]$

Recall a single "episode" k of income experience for agent i, with episode length D as described above:

$$\vec{y}_k = \left[y_{t_k}, y_{t_k+1}, ..., y_{t_{k+1}-1}\right].$$

Assume that the agent would like to execute the estimator of v^{θ_k} presented in expression (21) using this income experience $\vec{y}_k$ for a grid of points in the state space. The agent does not calculate this estimate for just any set of points in the state space, but rather uses the grid defined by $\hat{M}^k$.

The agent is in fact attempting to calculate the empirical equivalent to:

$$\mathbb{E}\left[v^{\theta}(B)\right],$$

where I will denote the empirical equivalent:

$$\hat{v}^{\theta}(m \mid B) \equiv \hat{\mathbb{E}}\left[v^{\theta}(B)\right].$$

Executing the single-stream estimate for each point in the grid is now straightforward. The agent simply maintains a vector of state variables, $\vec{m}^{k}_{N,t}$, which is of length N and initially set to the grid for episode k:

$$\vec{m}^{k}_{N,0} \equiv \hat{M}^{k}.$$

The N subscript acts to remind us that this particular vector has dimensions $N \times 1$, that is, it is the length of the grid on which I am estimating the value function, instead of $D \times 1$, which is the time-series length of the current episode k.

This vector of m-values is then updated following the law of motion implied by the current policy function c^{θ} and the shocks $\vec{y}_k$

$$\begin{aligned}\vec{m}^{k}_{N,t+1} &= \mathsf{R}(\vec{m}^{k}_{N,t} - c^{\theta}(\vec{m}^{k}_{N,t})) + y_{t+1},\\ y_{t+1} &\in \vec{y}_k.\end{aligned}$$

The agent also maintains a rolling summation of discounted utility experienced following policy $c^{theta}(.)$. Let $\vec{w}^{k,\theta}_{t}$ denote the vector of value function estimates, updated similarly each period:

$$\begin{aligned}\vec{w}^{k,\theta}_{t+1} &= \vec{w}^{k,\theta}_{t} + beta^{t} u(c^{\theta}(\vec{m}^{k}_{N,t}))\\ \vec{w}^{k,\theta}_{0} &= u(c^{\theta}(\vec{m}^{k}_{0})),\end{aligned}$$

which produces the final vector

$$\vec{w}^{\theta}_{k} \equiv \vec{w}_{t_{k+1}-1}.$$

This final vector $\vec{w}^{\theta}_{k}$ has been constructed to represent $\hat{v}^{\theta}(m \mid B) \forall\ B \in \mathcal{B}$. Importantly, because of the nature of its construction in equations (4.6.2), the monotonicity of the original utility function is preserved as the estimation progresses. As discussed in the appendix, monotonicity plays a key role in the convergence properties of policy iteration, and I strive to maintain the monotonicity of elements of the regret-learning value function whenever possible.

This process is outlined in Figures (8) and (9) for an agent with $N = 5$ and $D = 25$. The red line on top of Figure (8) represents a set of income shocks. The green dashed line represents the experienced consumption following the current consumption function, and the blue line below represents the resulting states that follow from the shocks, consumption, and initial state. The agent uses the experienced m states to form a partition over the state space; the midpoints of this partition, $\hat{M}^{k}$, is shown as the blue triangles with dotted blue lines on the left y-axis.

Having determined this partition, the agent then uses the *realized* income shocks and the midpoints $\mathbb{M}^{k}$ to construct the "speculative" experience – this is what "would have occurred" had the agent simply started at one of the midpoints of their state-space experience, but relived the same shocks as before.

Finally, the "single stream" value function is calculated for each point in $\hat{M}^{k}$ – this is the calculation $\vec{w}^{\theta}_{k}$.

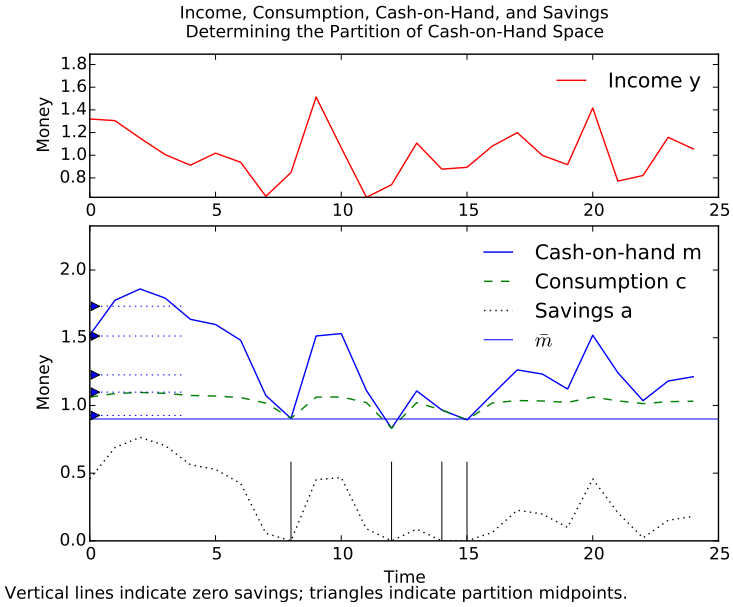


Figure 8: Agent Learning a Partition From Experience, $N = 5$ bins, $D = 25$ periods

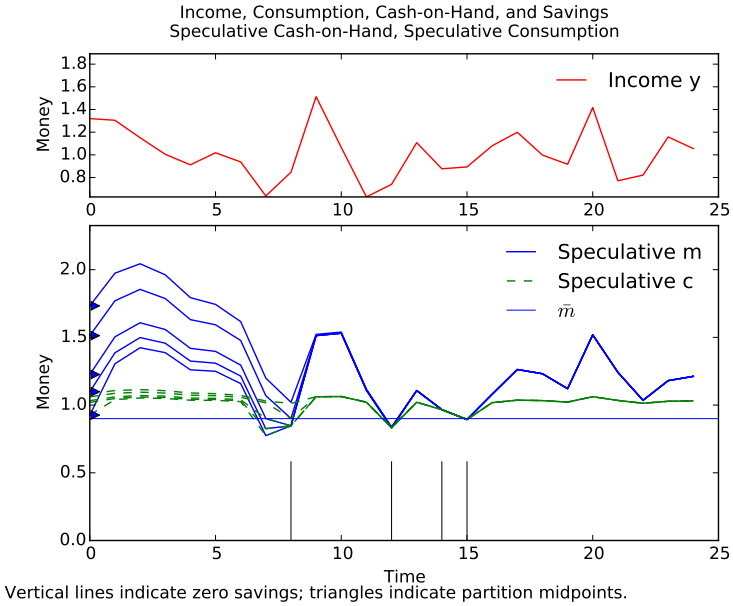


Figure 9: Agent Learning a Partition From Experience, $N = 5$ bins, $D = 25$ periods

Note that this conditional value function vector can be dotted with the unconditional empirical probabilities of each bin occurring, to arrive at an unconditional point estimate of the expected value of following the consumption function $\hat{c}^{\theta}$. This will be important for combining this individual-level learning of this paper with the social learning suggested in Palmer (2012). This important advancement will be discussed further in the conclusion.

4.6.3 Forming Conditional Distribution $\Pr(m_{t+1} \in B_{k,n} \mid c_t, m_t)$

Recall the partition boundaries for this episode k:

$$\begin{aligned} b_k &\equiv [b_{k,0}, b_{k,1}, ..., b_{k,n}, ..., b_{k,N}], \text{with elements defined,} \\ b_{k,n} &= \hat{\mathcal{Q}}^{\theta}_{m,D}(\frac{n}{N}), \end{aligned}$$

and recall that these boundaries denote the bins:

$$B_{k,n} = [b_{k,n-1}, b_{k,n}) \, for \; n = 1, 2, ..., N.$$

Recall that $\hat{\mathcal{F}}_{y,D}$ denotes the empirical cumulative density function of the income shock constructed with sample size D. Denote the empirical density constructed with data from a specific episode k as $\hat{\mathcal{F}}^{k}_{y,D}$.

For each bin $B_{k,n}$, denote the probability that *next-period* cash-on-hand m_{t+1} will fall in that bin, conditional on current-period cash-on-hand m_t, as:

$$q(B_{k,n} \mid c_t, m_t) \equiv \Pr(m_{t+1} \in B_{k,n} \mid m_t).$$

Define this as follows. Consider the problem for a single bin:

$$B_{k,n} = [b_{k,n-1}, b_{k,n}),$$

and recall the law of motion for m under the consumption choice c_t is

$$m_{t+1} = \mathsf{R}(m_t - c_t) + y_{t+1}.$$

Then,

$$\begin{aligned} \Pr(m_t + 1 \in B_{k,n} \mid m_t) &\equiv \Pr(b_{k,n-1} \leq m_{t+1} < b_{k,n}) \\ &= \mathcal{F}^{\theta}_{m}(b_{k,n} \mid m_t) - \mathcal{F}^{\theta}_{m}(b_{k,n-1} b_{k,n} \mid m_t). \end{aligned}$$

Note that:

$$\begin{aligned} \mathcal{F}^{\theta}_{m}(b_{k,n} \mid m_t) &= \Pr(\mathsf{R}(m_t - c_t) + y_{t+1} \leq b_{k,n}) \\ &= \Pr(y_{t+1} \leq b_{k,n} - \mathsf{R}(m_t - c_t)) \\ &= \mathcal{F}_{y}(b_{k,n} - \mathsf{R}(m_t - c_t)), \end{aligned}$$

Define $z_n(c_t \mid m_t) \equiv b_{k,n} - \mathsf{R}(m_t - c_t)$. Now,

$$
\begin{aligned}
q(B_{k,n} \mid c_t, m_t) &\equiv \Pr(m_{t+1} \in B_{k,n} \mid c_t, m_t) \\
&= \Pr(b_{k,n-1} \leq m_{t+1} < b_{k,n}) \\
&= \mathcal{F}_m^\theta(b_{k,n} \mid m_t) - \mathcal{F}_m^\theta(b_{k,n-1} b_{k,n} \mid m_t) \\
&= \mathcal{F}_y(z_n(c_t \mid m_t)) - \mathcal{F}_y(z_{n-1}(c_t \mid m_t)) \\
&\approx \hat{\mathcal{F}}_{y,D}(z_n(c_t \mid m_t)) - \hat{\mathcal{F}}_{y,D}(z_{n-1}(c_t \mid m_t)).
\end{aligned}
$$

Thus I can construct the probability that m_{t+1} falls into the bin $B_{k,n}$ as

$$
q(B_{k,n} \mid c_t, m_t) = \mathcal{F}_y(z_n(c_t \mid m_t)) - \mathcal{F}_y(z_{n-1}(c_t \mid m_t)),
$$

and the empirical equivalent estimated from experience as,

$$
\hat{q}(m_{t+1} \in B_{k,n} \mid c_t, m_t) = \hat{\mathcal{F}}_{y,D}(z_n(c_t \mid m_t)) - \hat{\mathcal{F}}_{y,D}(z_{n-1}(c_t \mid m_t)),
$$

a straightforward calculation. I will abbreviate this expression

$$
\hat{q}(B_{k,n} \mid c_t, m_t)
$$

for notational convenience. Denote the full partitioned probability mass function for episode k, which uses $\hat{\mathcal{F}}_{y,D}^k$, as

$$
\widehat{P}_m^k(c_t \mid m_t) = \left[\hat{q}(B_{k,1} \mid c_t, m_t), ..., \hat{q}(B_{k,n} \mid c_t, m_t), ..., \hat{q}(B_{k,N} \mid c_t, m_t)\right].
$$

The notation "$(c_t \mid m_t)$" is intended to communicate the idea that m_t will given by nature, while c_t is chosen. This distinction will become apparent in the next section.

Next, construct the empirical, learned-from-experience version of expression (7) above:

$$
\begin{aligned}
\hat{H}(m, c, \hat{v}^\theta) &= u(c) + \beta(\hat{v}^\theta(m \mid B_{k,n}) \bullet \widehat{P}_m^k(c \mid m)) \\
&= u(c) + \beta \sum_{n=1}^{N} \left(\hat{v}^\theta(m \mid B_{k,n}) \times \hat{q}(B_{k,n}\right) \\
&= u(c) + \beta \sum_{n=1}^{N} \left(\left[\hat{E}\right]\left[v^\theta(m) \mid m \in B_{k,n}\right] \times \hat{q}(B_{k,n}\right).
\end{aligned}
$$

The goal of this expression is to come as close as possible to constructing an empirical expression for $\mathbb{E}\left\{v^\theta(m_{t+1}) \mid c_t, m_t\right\}$ using an empirical equivalent to the law of total probability. This provides a rigorous framework for expressing the idea that the agent learns imperfectly about both the value of states as well as the dynamics of their environment from experience.

I am now ready to formalize "regret" and "learning from regret."

4.7 Identifying Regret Choices: What an Agent "Should Have Done"

Once an agent has experienced D periods in an episode k, determined the state-space partition $\hat{M}^k$ and formed a conditional value function estimate $\hat{v}^\theta(m \mid b_{k,n})$ for that episode, it's time to look back and think about what consumption choices the agent *should have made* conditional on $\hat{v}^\theta(m \mid b_{k,n})$.

Recall that the agent's experience in episode k is formalized in the following two vectors,

$$\vec{y}_k = \left[y_{t_k}, y_{t_k+1}, ..., y_{t_{k+1}-1}\right], \text{ income shocks, and}$$
$$\vec{m}_k = \left[m_{t_k}, m_{t_k+1}, ..., m_{t_{k+1}-1}\right], \text{ cash-on-hand states from following } c^\theta.$$

For each period t in episode k, the agent determines the regret choice $\check{c}_t$:

$$\check{c}_t = \text{argmax}\, c \in [0, m_t]\ \hat{H}(m_t, c, \hat{v}^\theta),$$

where the full vector of these choices for episode k is defined,

$$\vec{\check{c}}_k \equiv \left[\check{c}_{t_k}, \check{c}_{t_k+1}, ..., \check{c}_{t_{k+1}-1}\right].$$

Note that each of the $\check{c}_{t_k}$ values directly corresponds to a m_{t_k} value. Now with the regret choices in hand, the agent simply needs to find the consumption function which is closest to these values in a mean squared errors sense.

4.8 Improving c^θ by Minimizing Regret

Once the regret choices $\vec{\check{c}}_k$ are in hand for episode k, the agent needs to determine the policy function to use in the next period, $k+1$. Recall the structure of the consumption function the agent is using, parameterized by $\theta = (\kappa, \bar{m})$:

$$\widetilde{c}^\theta(m_t) = E[y_t] + \kappa\left[m_t - \bar{m}\right], \text{ and}$$
$$c^\theta(m_t) = \begin{cases} m_t & if\ \widetilde{c}^\theta(m_t) \geq m_t \\ \widetilde{c}^\theta(m_t) & otherwise. \end{cases}$$

where as before κ is the marginal propensity to consume out of wealth, $\bar{m}$ is the buffer-stock savings target, and the piecewise linear form of c^θ imposes the liquidity constraint. The problem facing the consumer is to find the parameters $\hat{\theta} = (\hat{\kappa}, \hat{\bar{m}})$ such that the sum of squared errors between the parametric portion of the consumption function applied to the experienced states, $\widetilde{c}^{\hat{\theta}}(\vec{m}_k)$, and the regret choices at each of those states, $\vec{\check{c}}_k$, is minimized.[17] Mathematically, the agent wants to solve:

$$\hat{\theta} = \underset{\theta}{\text{argmin}} \sum_{t=t_k}^{t_{k+1}-1} \left(\widetilde{c}^{\hat{\theta}}(m_t) - \check{c}_t\right)^2,$$

where each $m_t \in \vec{m}_k$ and $\check{c}_t \in \vec{\check{c}}_k$ as defined in the above section. Implementing this error minimization for the $\widetilde{c}^\theta$ portion of the consumption function via OLS is straightforward; run the OLS estimate on:

$$\vec{\check{c}}_k = \alpha_0 + \alpha_1 \vec{m}_k$$

to obtain estimates of the constant α_0 and slope α_1, respectively, and simply back out the $(\hat{\kappa}_k, \hat{\bar{m}}_k)$ values as

[17] The decision whether to use $\widetilde{c}^\theta$ versus c^θ to minimize distance to the regret choices is simply due to computational convenience and reduction in programming error. Future versions will include a restricted regression form which imposes the liquidity constraint directly. Initial experiments with versions of both forms indicates that the outcomes of either approach are extremely similar.

$$\hat{\kappa}_k = \alpha_1, \text{ and}$$
$$\hat{\bar{m}}_k = \frac{E[y] - \alpha_0}{\hat{\kappa}_k},$$

which can be seen by simply rearranging the definition of $\widetilde{c}^{\theta}$:

$$\begin{aligned}\widetilde{c}^{\theta}(m) &= E[y] + \kappa\left[m - \bar{m}\right], \\ &= (E[y] - \kappa\bar{m}) + (\kappa m) \\ &= \alpha_0 + \alpha_1.\end{aligned}$$

Note that technically the value $E[y_t]$ is a parameter of the consumption function in equation (4.8), unless the agent knows this value perfectly. Since the agent must learn everything from experience, the agent learns this value as well. Instead of estimating $E[y_t]$ in the same way that $\hat{\theta}$ is estimated, the agent simply "calibrates" $E[y_t]$:

$$E[y] = \frac{1}{D}\sum_{y \in \vec{y}_k} y.$$

The resulting consumption function can be denoted $c^{\theta_{k+1}}$, and is set as the consumption function for the following episode:

$$\widetilde{c}^{\theta_{k+1}}(m_t) = E[y_t] + \kappa_k\left[m_t - \bar{m}_k\right], \text{ and}$$
$$c^{\theta_{k+1}}(m_t) = \begin{cases} m_t & if\ \widetilde{c}^{\theta_{k+1}}(m_t) \geq m_t \\ \widetilde{c}^{\theta_{k+1}}(m_t) & otherwise. \end{cases}$$

4.9 The Regret Learning Algorithm

I can now finally concisely outline the regret learning algorithm. Recall the two steps of policy iteration, policy evaluation (4.2.1) and policy improvement (4.2.1), respectively, repeated here for convenience:

$$v_{\mu^k} = T_{\mu^k} v_{\mu^k}, \text{ policy evaluation, and,}$$
$$T_{\mu^{k+1}} v_{\mu^k} = T v_{\mu^k} \text{ policy improvement.}$$

Regret learning also has a policy evaluation step and a policy improvement step. These rely on the approximations discussed in the previous sections, which empirically approximate various components of both steps above using experience. They are as follows.

Start with an arbitrary initial consumption function c_0^{θ}, then iteratively apply the following two steps for D-length episodes of experience, each episode denoted k:

Regret Learning:

- Step (1) **Policy Evaluation**: Given policy c^{θ_k}, estimate v^{θ_k} from experience as described in Section (4.6.2), as a noisy, single-stream, curvature retaining Monte Carlo estimate of v^{θ}.
- Step (2) **Policy Improvement**: Given the conditional value function v^{θ_k}, find the regret choices $\vec{\check{c}}_k$ as described in Section (4.7). Using these regret choices, find the new policy function $c_{k+1}^{\theta'}$ which minimizes the regret choices in a mean-square errors sense, as described in Section (4.8).

These steps are repeated indefinitely.

4.10 Visualizing Regret Learning

This section presents two visualizations for regret learning in action. First, Figures (10) through (24) display a single run for an individual agent with $N = 11$ bins and $D = 25$ learning periods. Figure (25) examines the progress of a different learning experience through the welfare surface for the same parameters, $N = 11$ bins and $D = 25$. All of these plots examine single experiences of regret learning for illustrative purposes. The following Section 5 will examine distributional properties of this process numerically.

The sequence of Figures (10) through (24) trace out a typical path through the consumption space. The dashed blue line represents the agent's previous-episode consumption function c^{θ_k}. The blue stars represent the regret choices identified above, and the solid blue line represents the new regret-minimizing consumption function $c^{\theta'}_{k+1}$ which minimizes distance to the regret choices. The red dashed line represents the true optimal consumption function. The vertical dashed lines in each consumption plot outline the 5^{th} and 95^{th} percentiles of the ergodic distribution of cash-on-hand , m, following the true optimal rule. These are included to provide a rough indication of how "out of the ordinary" a set of consumption experiences may be with respect to the target optimal policy. When experiences stray outside these boundaries, the agent tends to learn a poor consumption function. The caption for each plot includes the welfare change for the dashed blue line to the solid blue line – as before, lower is better.

In the first Figure (10), the dashed blue line is the 45^* line – the initial spendthrift consumption function, c^{θ}_0. The sacrifice value of this consumption function is quite high, 0.87, or roughly 90% of expected annual income. In the following consumption figure, the dashed blue line is the 45^* line – the initial spendthrift consumption function, c^{θ}_0 . The blue stars represent the regret choices – the choices the agent *wishes* he/she made, after following the spendthrift rule and then the estimated value function is not shown in any plots). The solid blue line is the first improved policy function $c^{\theta'}_1$ which the agent learns from these regret choices – it is already bending towards the optimal policy.

As indicated in the caption, the welfare cost of the new consumption function improved significantly, from 0.87 to 0.38. In the following Figure (11) the stars once again indicate the regret choices which are consistent with the value function estimated from the experience under the $c^{\theta'}_1$ function. Note that the regret values are concentrated around the lower 5^{th} percentile – when the agent minimizes regret through these choices, the resulting consumption function, the solid blue line, visually moves *slightly away* from the true optimal policy. As a result, this episode is less impressive for agent learning. The agent actually retreats slightly in utility terms; the learned function has a sacrifice value of ~0.54.

There is no need for despair, however. The next updated consumption function, the solid blue line in Figure (12), is moving back on track. Figures (13), (14) and (15) show the agent making solid progress to a consumption function with a 7% sacrifice value.

Continue this examination from (15) through the end of the agents recorded experience in Figure (24). Conveniently these last figures demonstrate the agent being deceived by experience one final time by a set of regret experiences on the low end of experience – see the consumption update in Figure (24).

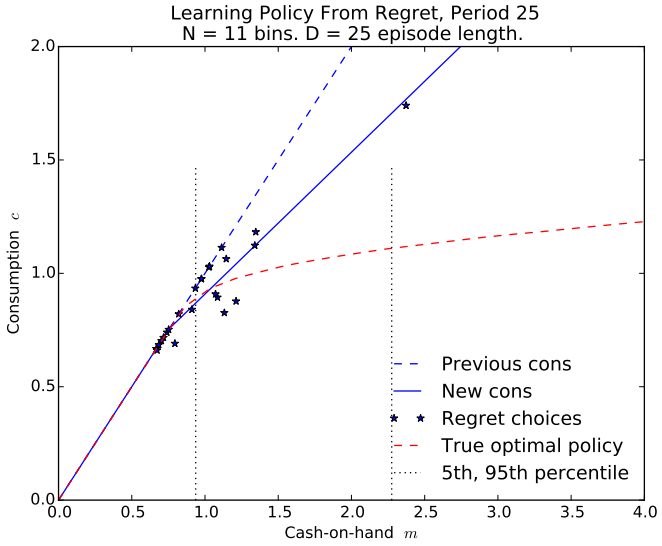


Figure 10: Learned c^θ Functions, $\bar{\epsilon} = 0.87 \rightarrow 0.38$

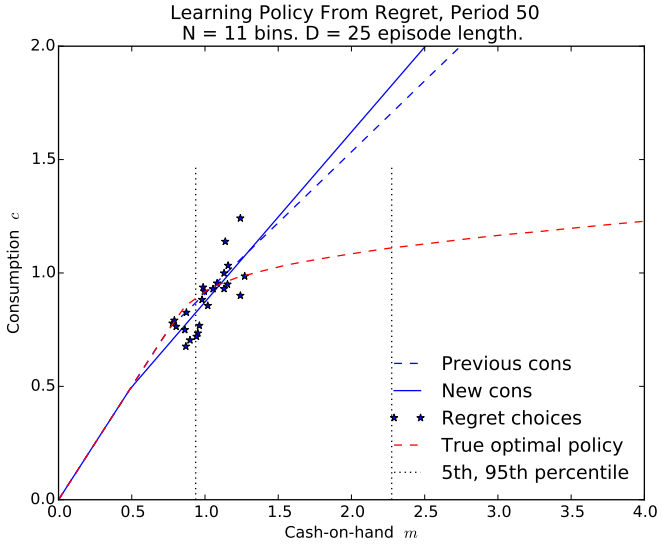


Figure 11: Learned c^θ Functions, $\bar{\epsilon} = 0.38 \rightarrow 0.54$

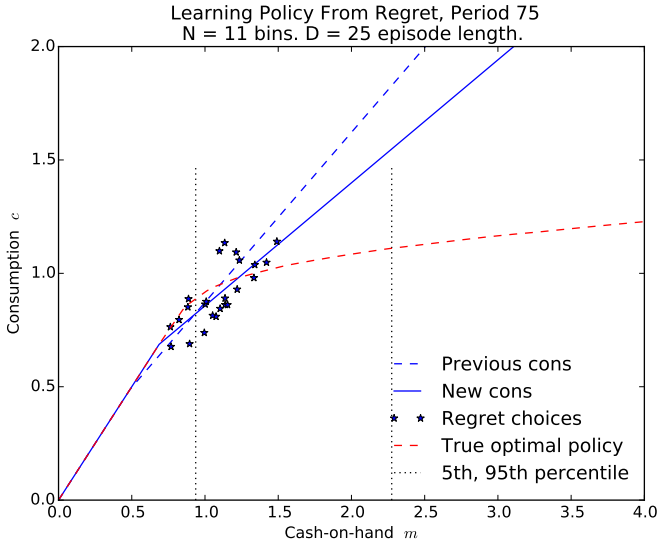


Figure 12: Learned c^θ Functions, $\bar{\epsilon} = 0.54 \rightarrow 0.27$

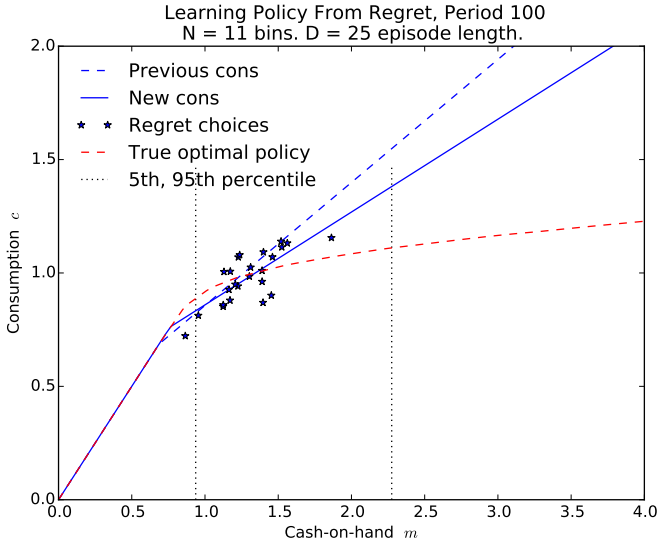


Figure 13: Learned c^θ Functions, $\bar{\epsilon} = 0.27 \rightarrow 0.14$

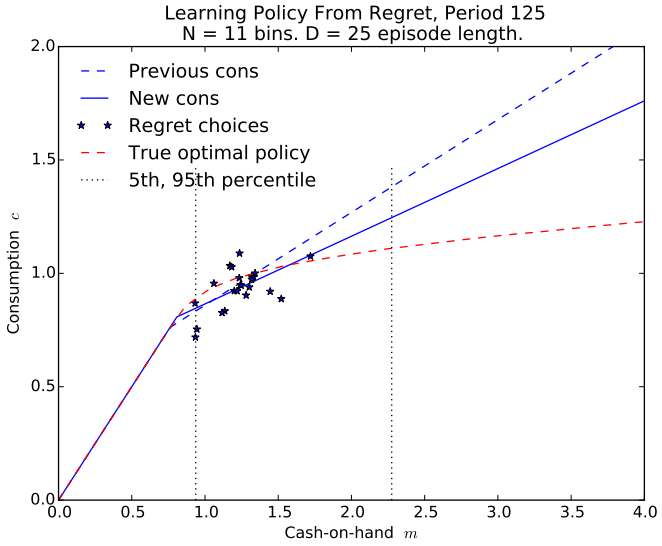


Figure 14: Learned c^{θ} Functions, $\bar{\epsilon} = 0.14 \rightarrow 0.07$

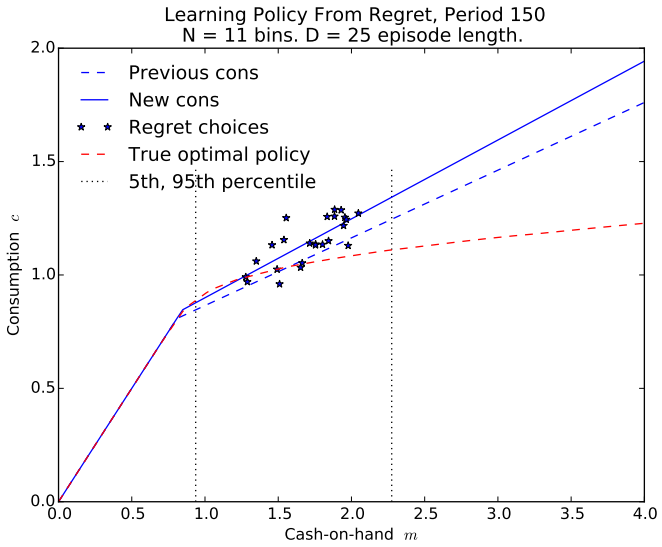


Figure 15: Learned c^{θ} Functions, $\bar{\epsilon} = 0.07 \rightarrow 0.09$

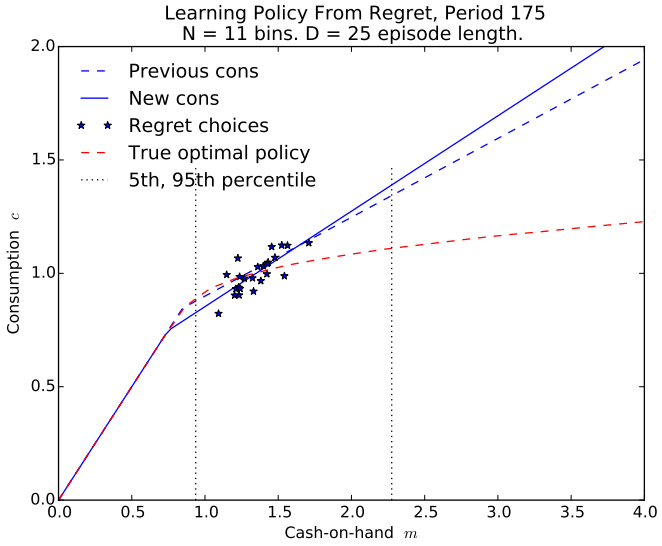


Figure 16: Learned c^θ Functions, $\bar{\epsilon} = 0.09 \rightarrow 0.15$

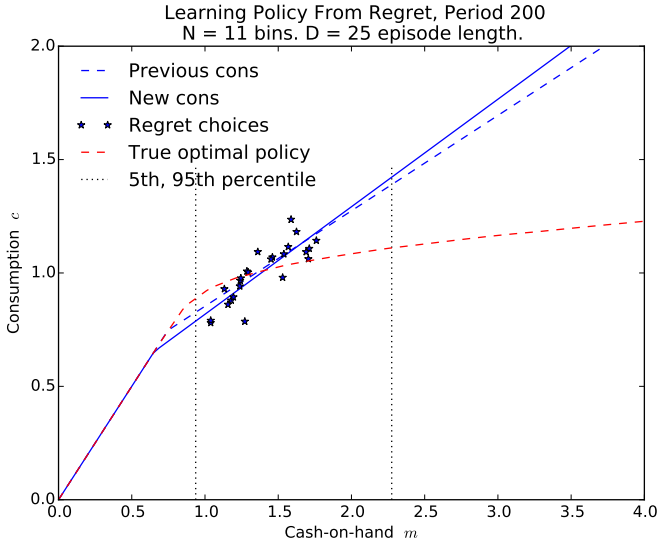


Figure 17: Learned c^θ Functions, $\bar{\epsilon} = 0.15 \rightarrow 0.22$

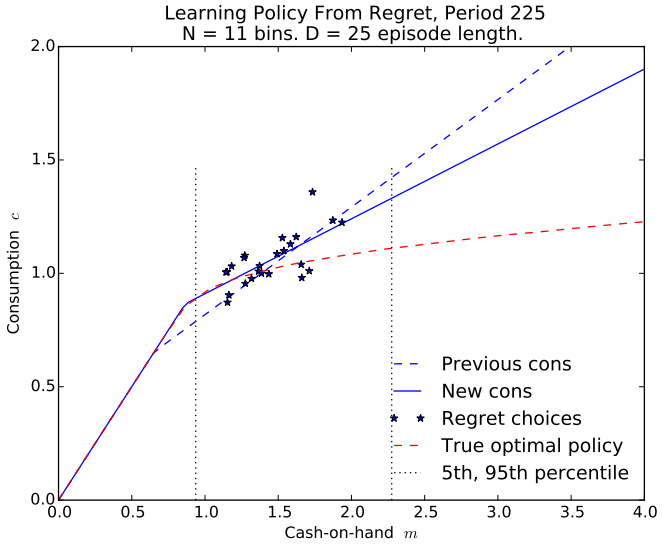


Figure 18: Learned c^θ Functions, $\bar{\epsilon} = 0.22 \rightarrow 0.09$

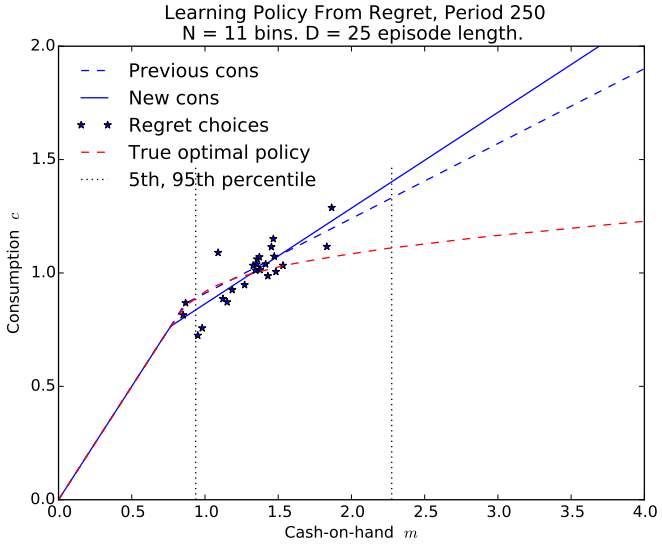


Figure 19: Learned c^θ Functions, $\bar{\epsilon} = 0.09 \rightarrow 0.15$

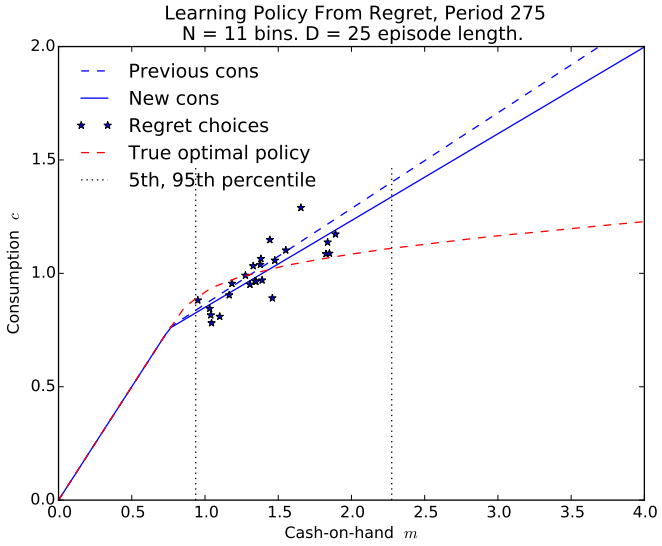


Figure 20: Learned c^{θ} Functions, $\bar{\epsilon} = 0.15 \rightarrow 0.13$

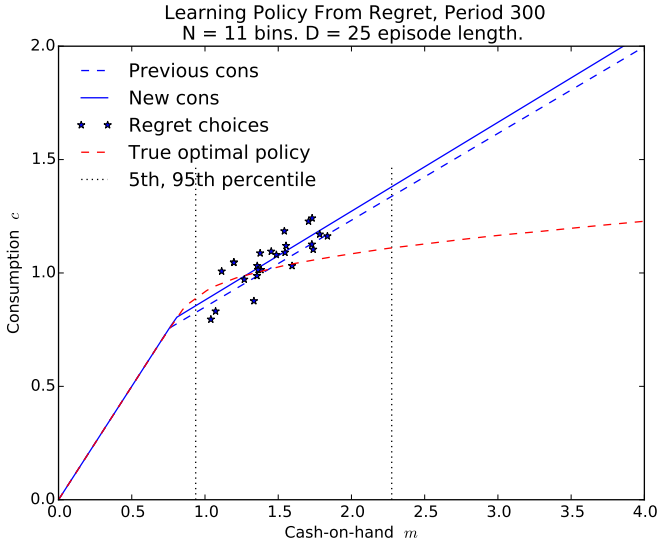


Figure 21: Learned c^{θ} Functions, $\bar{\epsilon} = 0.13 \rightarrow 0.12$

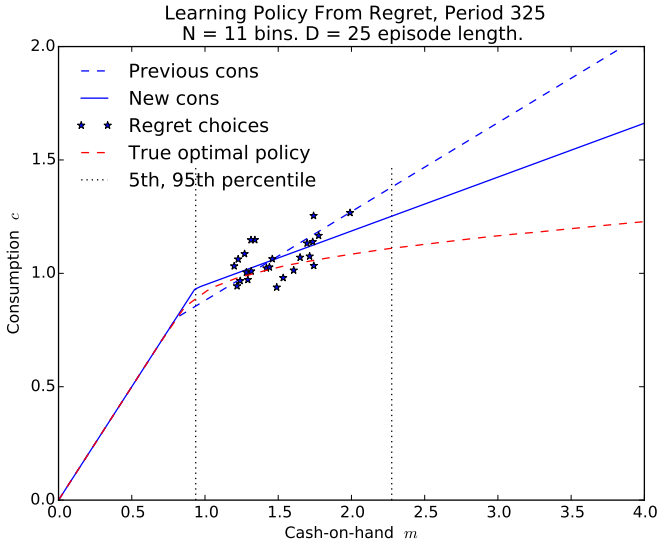


Figure 22: Learned c^θ Functions, $\bar{\epsilon} = 0.12 \rightarrow 0.06$

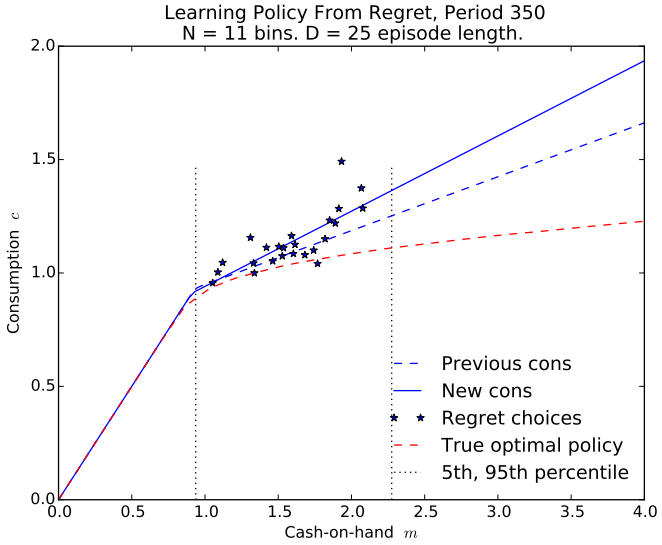


Figure 23: Learned c^θ Functions, $\bar{\epsilon} = 0.06 \rightarrow 0.13$

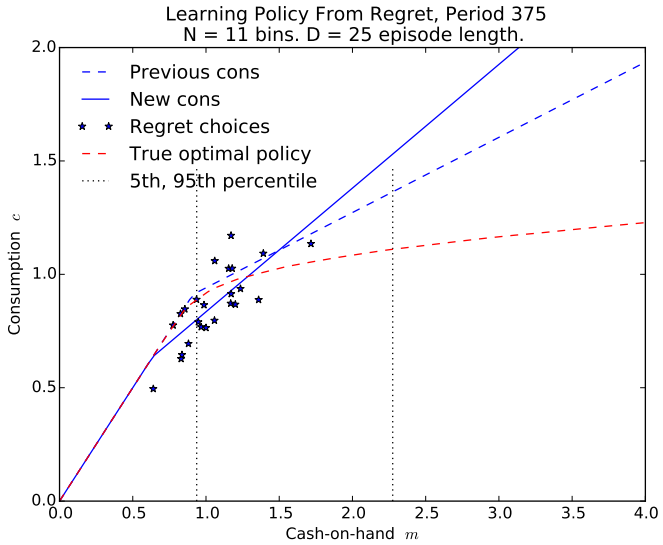


Figure 24: Learned c^{θ} Functions, $\bar{\epsilon} = 0.13 \rightarrow 0.28$

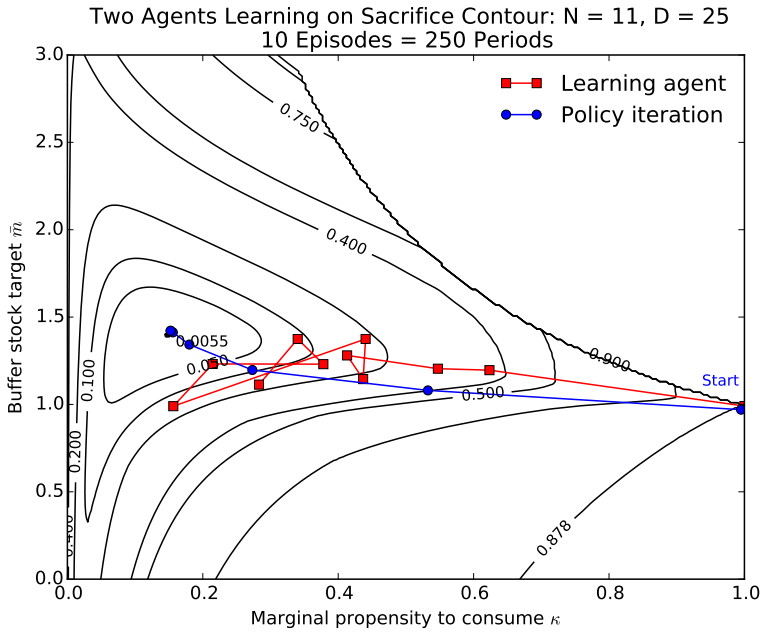


Figure 25: Learning from Regret: N = 11, D = 25

We can directly compare the experience of an agent to the policy iteration solution in Figure (25). Here we can see that the learning process appears to parallel the policy iteration solution for about three iterations, before setting into a "cloud" of update steps just to the "southeast" of the optimal solution. If we were to run this for a large number of different sets of shocks and observe the resulting distribution on this surface, we would find agents settling into a fairly stable distribution *in welfare terms* which would encompass this single experience. Observing such a distribution is not intuitive on this contour surface; Section 5 will examine distributional properties of this process in a more intuitive framework.

One final important note: agents are restricted from learning consumption functions with either an unattainable $\bar{m}$ buffer-stock savings targets (namely, a buffer-stock target outside the borrowing constraint) or a negative slope on the MPC, κ.[18] Without this restriction, an agent may occasionally learn a rule with either property, with the consequence that the agents takes a large step away from the optimal rule, essentially "starting over" the learning process in utility term. Both restrictions on $\bar{m}$ and κ seem to be common-sense, and in addition both are supported by theoretical restrictions on these parameters.

5 Results

This section uses numerical simulation to demonstrate dynamic statistical properties of regret learning in terms of welfare distance from the optimal solution. A few broad questions to answer are:

- Can regret learning attain a near-optimal policy?
- How long does this take, and what does it look like?
- How is behavior affected by the parameters D and N?

Using the welfare measure defined in Section 3.3,

To address these questions, 1,000 agents were simulated for 10,000 periods for a variety of combinations of the N, D parameters: the number of bins used to model the following period, and the number of periods used to estimate a value function before updating, respectively. The agent's *behavioral* parameters, risk aversion (ρ), discounting (β), and variance of shocks to income (σ_y), are calibrated to common values estimated from microeconomic data. The specific parameters used are expressed in Table (1). Future work will examine a sweep across a grid of these parameters.

The results of the simulation experiment are largely positive. To examine something of a worst-case scenario, agents start with a spendthrift "consume everything" consumption function. This has a sacrifice value of roughly 0.9, which can be interpreted as equal to a one-time payment of ~90% of expected annual income. Within one learning episode, this sacrifice value can be cut in half, and within three to five episodes it can be cut to less than a tenth of the original sacrifice value. These improvements are only partially permanent, however. As I will demonstrate below, agents spend a short time improving permanently over their initial, worst-case spendthrift consumption functions, but then spend the rest of their lives buffeted around a well-defined distribution of distances from the optimal rule. The reasons for agent indecision about their learning rules is intuitive – they forget their distant past as they continue to live, and their most recent set of experiences can deceive them about the effectiveness of a consumption rule. As the researchers I a aware of this deception, but to the agent, acting on their conditional and limited information, it appears to be the best choice. I argue that this is a feature, not a bug – while agents can be forced very close to

[18] When the improved consumption function has either property the agent considers a convex combination of the previous three consumption function parameters; if this is still not acceptable, the agent simply reverts to the previous parameters.

Table 2: Parameter Sweep Values

Plots in Figures (26) through (36)

	Number of Bins, N						
Length of learning episode D	3	5	7	11	25	55	95
13 periods	x	x	x	x			
24	x	x	x	x			
48	x	x	x	x			
72	x	x	x	x			
101	x	x	x	x	x	x	x
201	x	x	x	x	x	x	x
301	x	x	x	x	x	x	x

the optimal solution, it is intuitively appealing to have a rigorous model of agent learning, which strives for optimal behavior (and in fact explicitly has optimal behavior as its target) but none-the-less falls prey to whatever sets of shocks the agent has most recently experienced. After examining the characteristics of this behavior I will discuss the particular structural reasons which drive it, and outline extensions which can extend these results.

5.1 Aggregate Regret-Learning Behavior

I examine agent behavior first, outlining distributional effects before discussing the persistence of position in the results distribution. I will primarily focus on the differences in behavior which derive from different N and D values. For a given (N, D) parameter pair, agents converge to a distribution of distances from the optimal rule fairly quickly, within 5-10 episodes (recall that each episode is of length D). This is displayed in Figures (26) through (36), which show the time series of sacrifice values from start of a simulation through a total of 10,000 periods.

The figures are organized by N-value, the number of bins the agents use to create the value function estimate. For each N value, the values for a number of lengths of learning period, denoted D, are displayed.

For $N \in [3, 5, 7, 11]$, there is a figure displaying the distribution of sacrifice values $\bar{\epsilon}^{\theta}$ over time for $D \in [13, 24, 48, 72]$ and another figure displaying the distribution of $\bar{\epsilon}^{\theta}$ over time for $D \in [101, 201, 301]$. For $N \in [25, 55, 95]$, there are only figures for $D \in [101, 201, 301]$.[19]

The choice of $N \in [3, 5, 7, 11]$ is to demonstrate the high gains agents experience for increasing N even a little. The choice of $N \in [25, 55, 95]$ demonstrates that these gains quickly start to tail off as N increases, if D is being held constant.

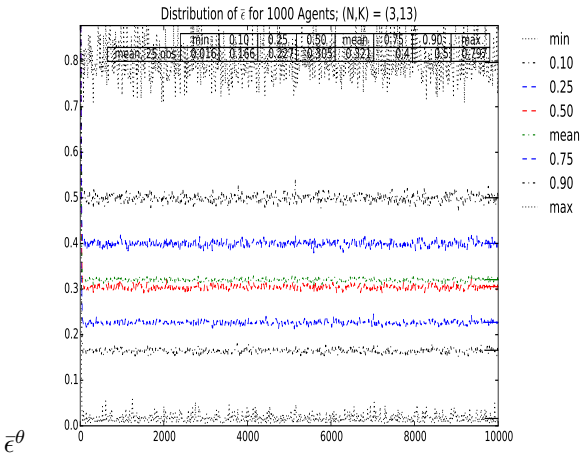
The choices for $D \in [13, 24, 48, 72]$ correspond to roughly the decade after late teens, the two decades after late teens, four decades after late teens, and lastly an entire lifetime after late teens.

Finally, the choices for $D \in [101, 201, 301]$ are again to demonstrate how the marginal gains from an increase in learning lengths drop off as D increases for a fixed N.

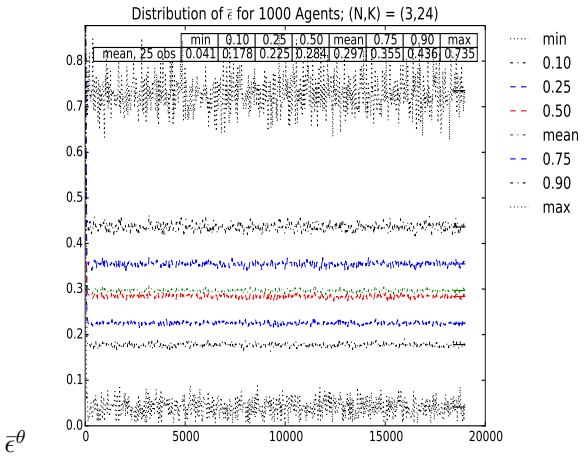
The parameters used for plots (26) through (36) are summarized in Table (2).

[19]Note that $N < D$ in all circumstances; if this is not the case, agents cannot populate their probability mass function in expression (4.6.3).

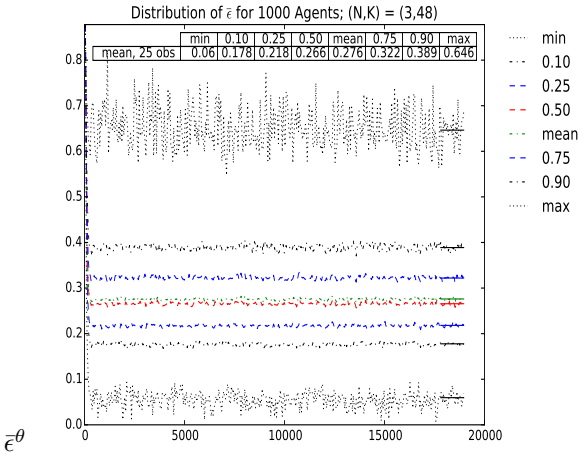
Distribution of $\bar{\epsilon}^{\theta}$ for N = 3, D = 13



Distribution of $\bar{\epsilon}^{\theta}$ for N = 3, D = 24



Distribution of $\bar{\epsilon}^{\theta}$ for N = 3, D = 48



Distribution of $\bar{\epsilon}^{\theta}$ for N = 3, D = 72

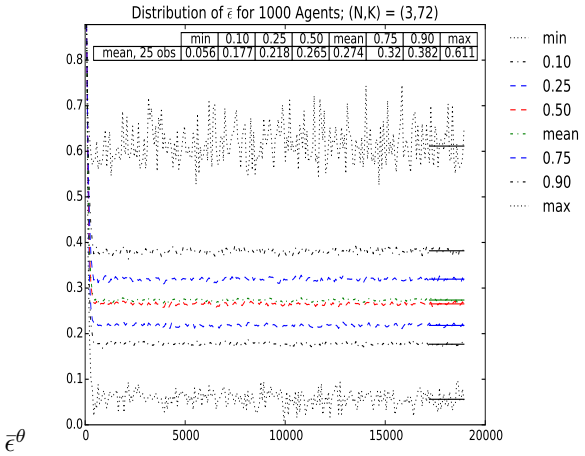
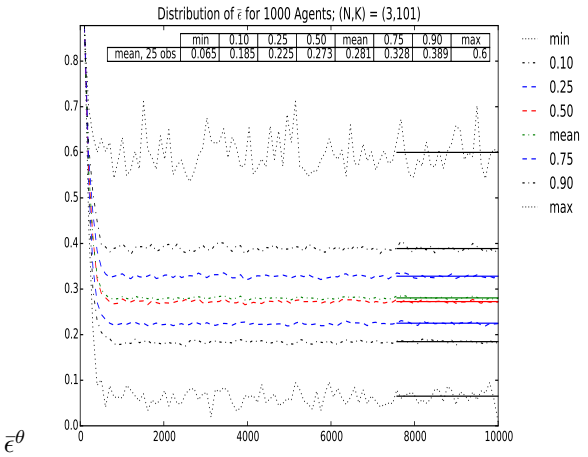
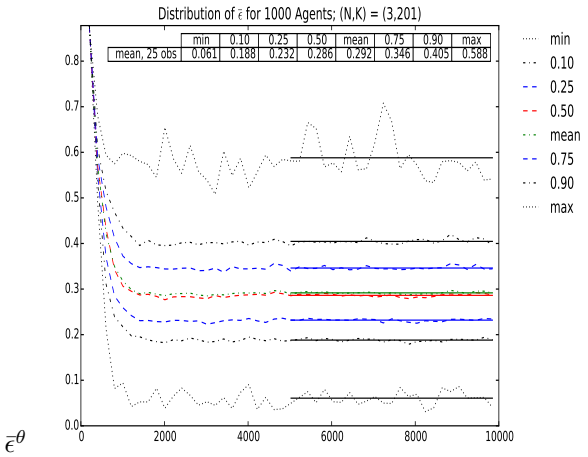


Figure 26: Distribution of $\bar{\epsilon}^{\theta}$ Over Time, N = 3, D = 13, 24, 48, 72

N = 3, D = 101



N = 3, D = 201



N = 3, D = 301

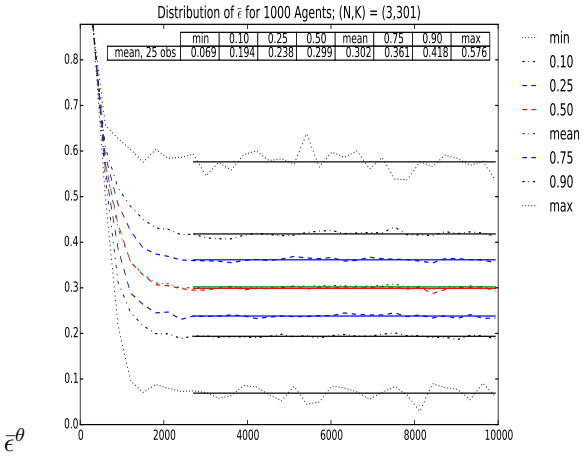


Figure 27: Distribution of $\bar{\epsilon}^{\theta}$ Over Time, N = 3, D = 101, 201, 301

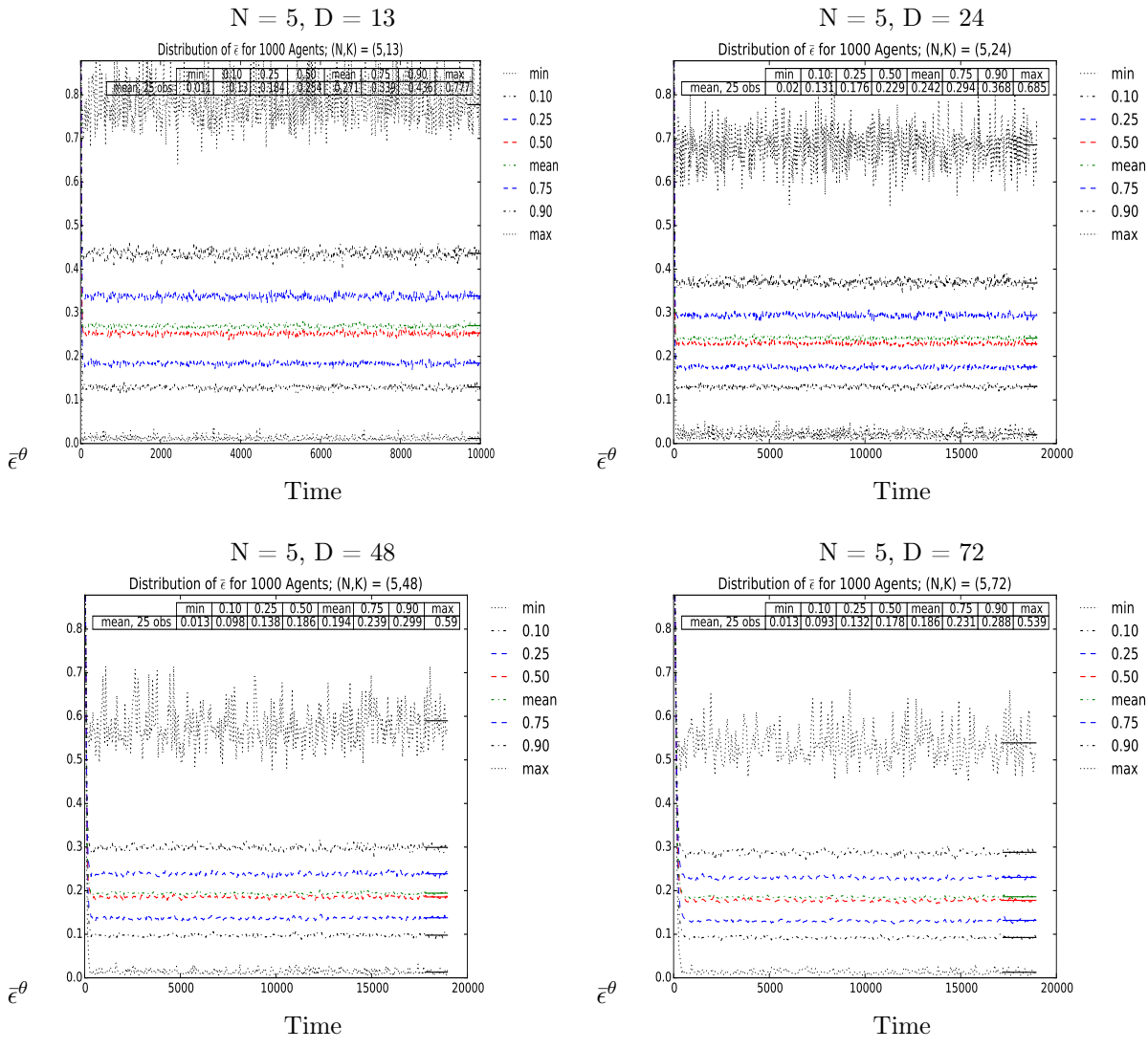


Figure 28: Distribution of $\bar{\epsilon}^{\theta}$ Over Time, N = 5, D = 13, 24, 48, 72

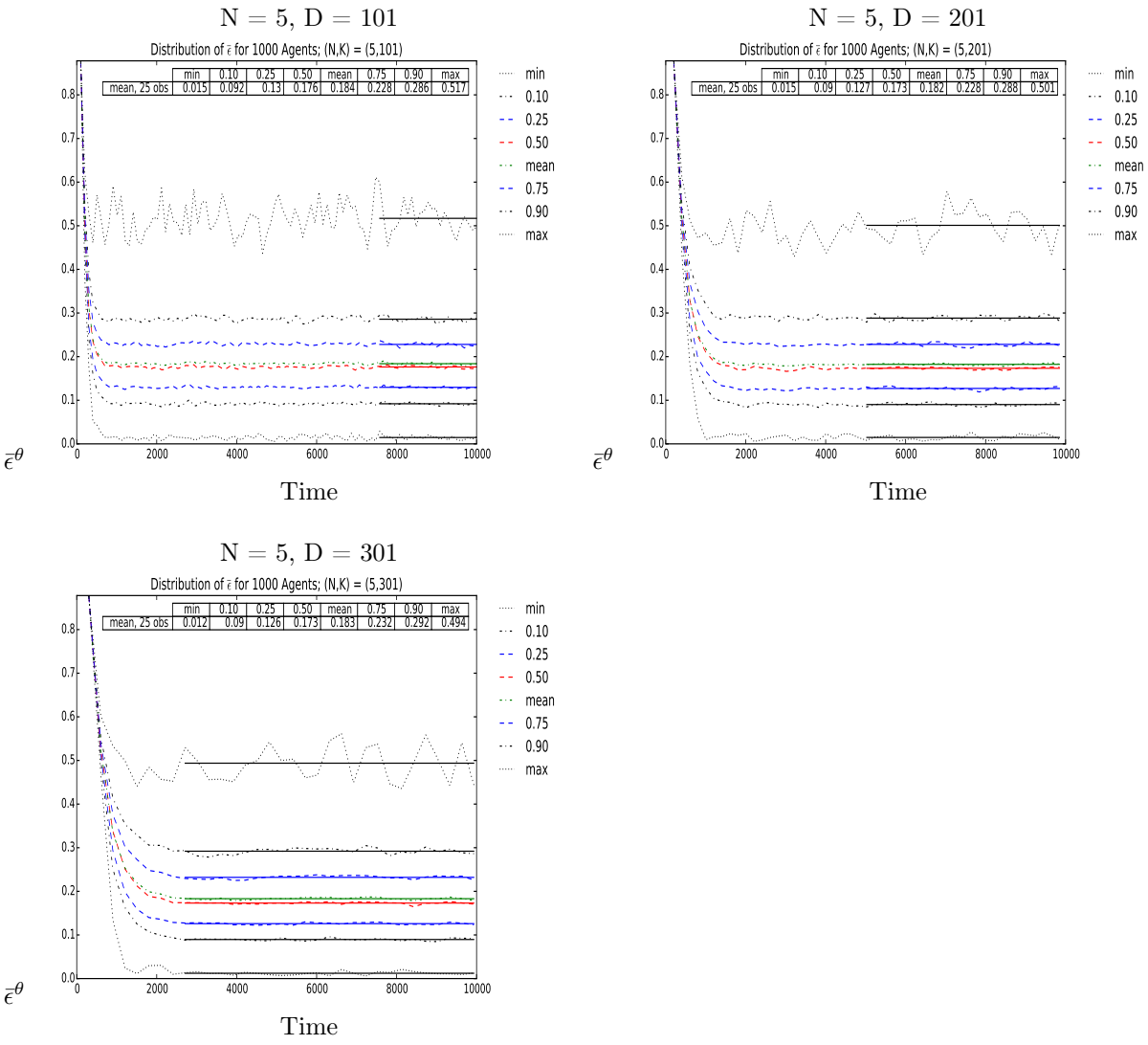


Figure 29: Distribution of $\bar{\epsilon}^{\theta}$ Over Time, N = 5, D = 101, 201, 301

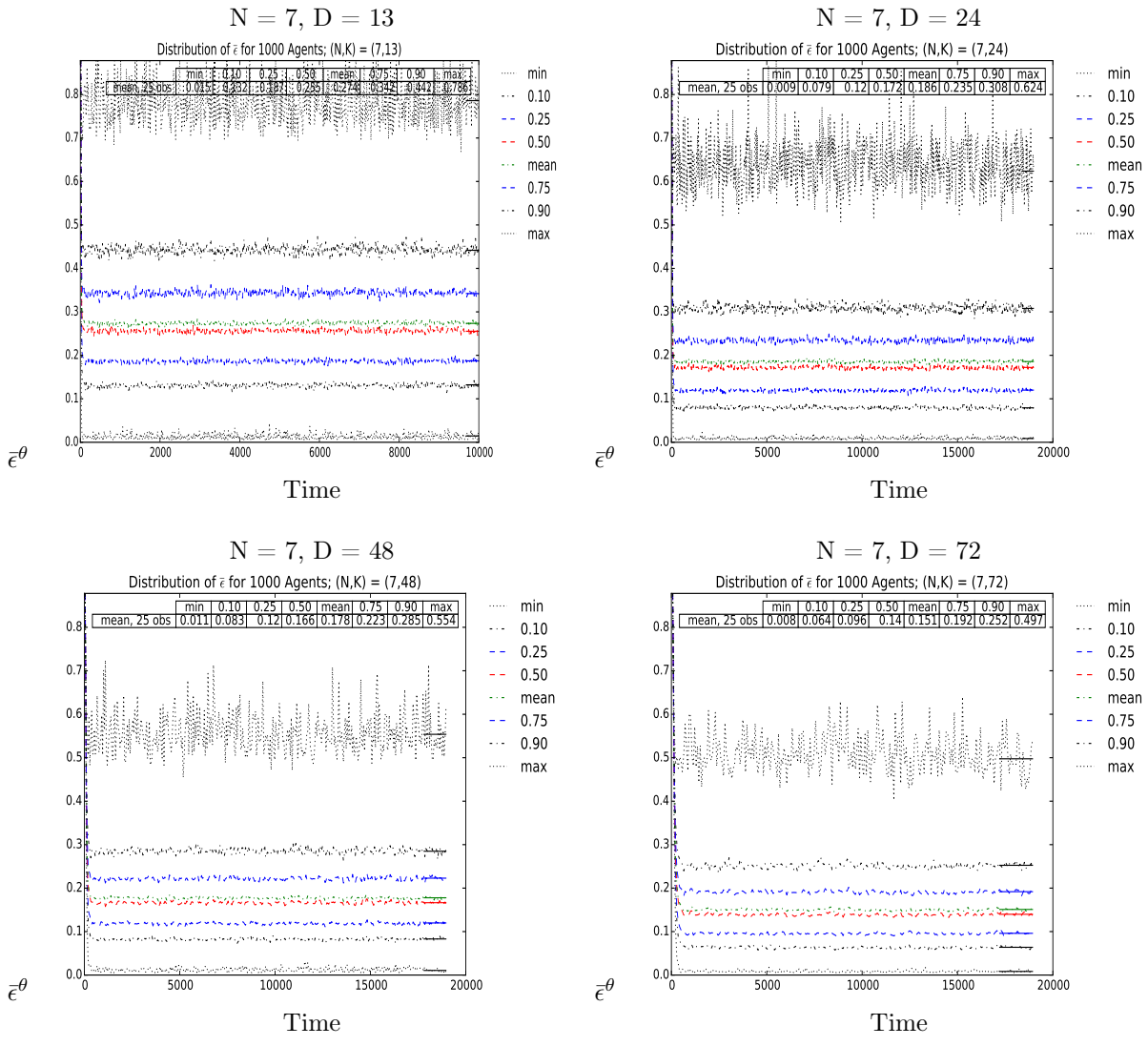
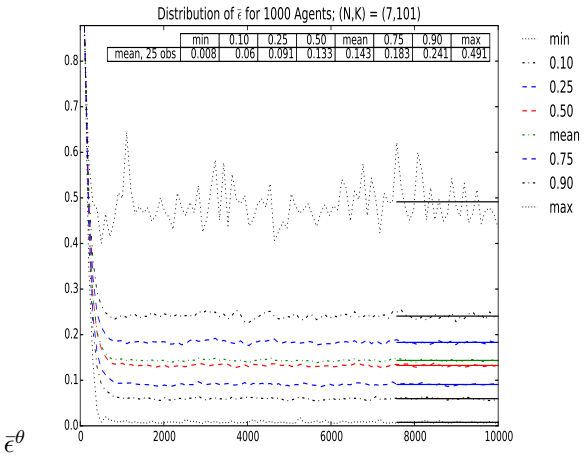
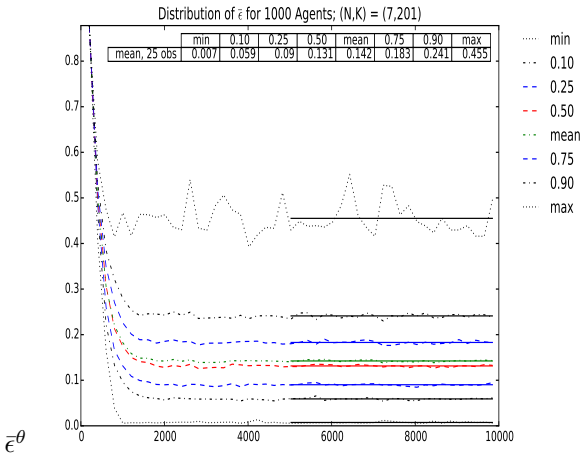


Figure 30: Distribution of $\bar{\epsilon}^{\theta}$ Over Time, N = 7, D = 13, 24, 48, 72

N = 7, D = 101



N = 7, D = 201



N = 7, D = 301

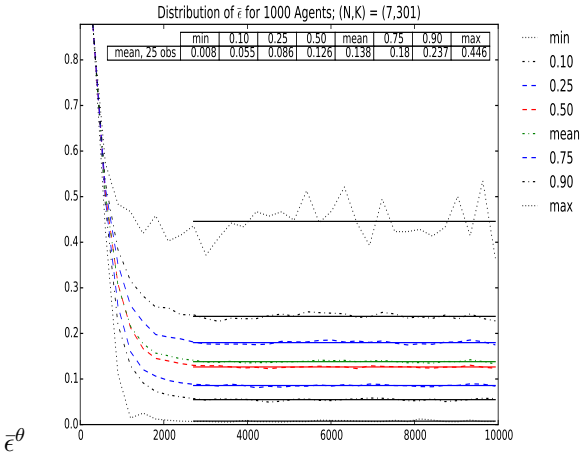


Figure 31: Distribution of $\bar{\epsilon}^{\theta}$ Over Time, N = 7, D = 101, 201, 301

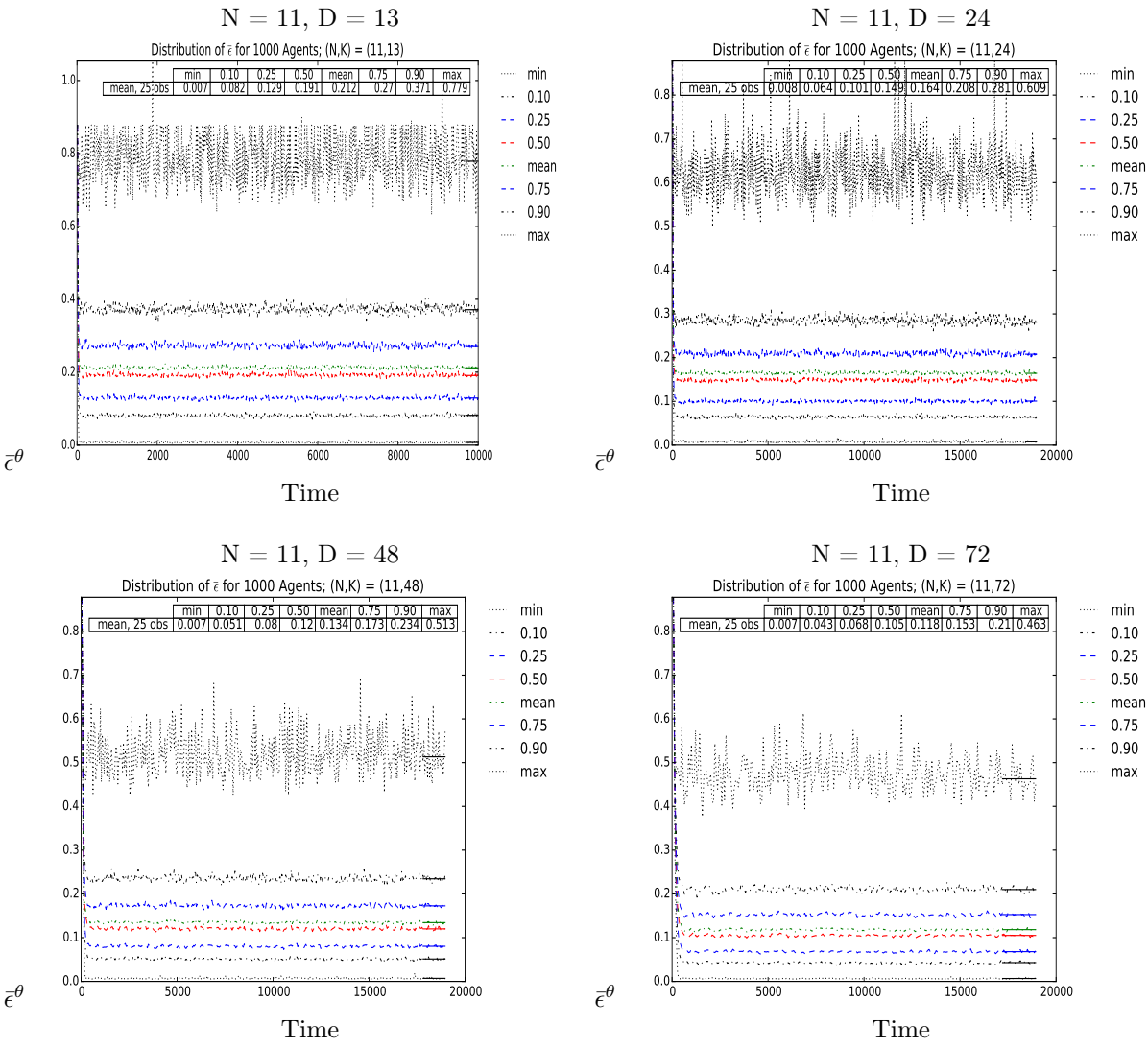
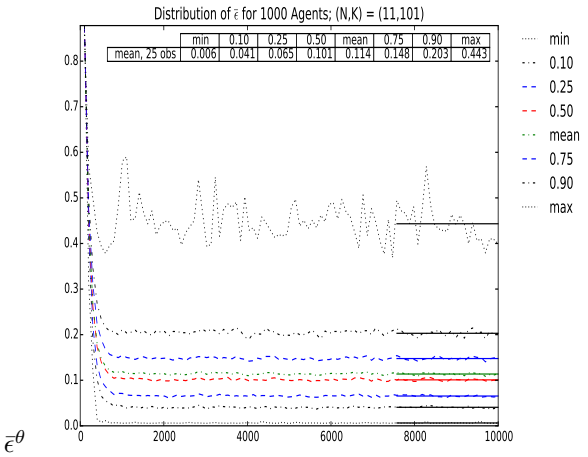
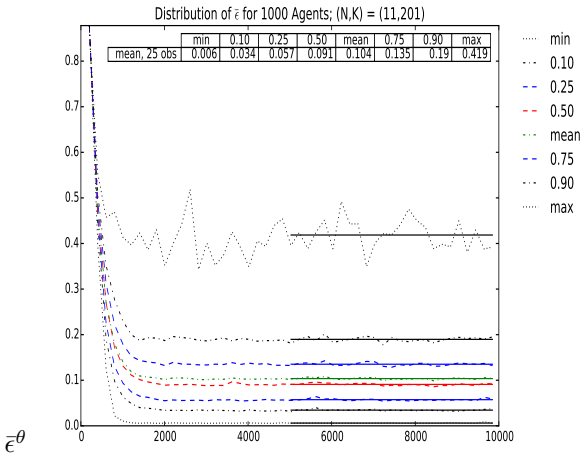


Figure 32: Distribution of $\bar{\epsilon}^{\theta}$ Over Time, N = 11, D = 13, 24, 48, 72

N = 11, D = 101



N = 11, D = 201



N = 11, D = 301

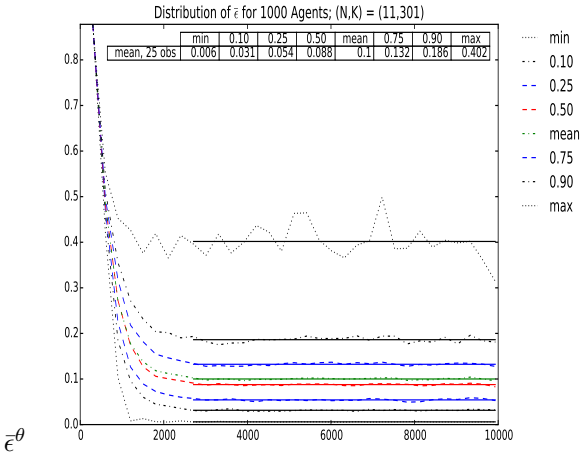
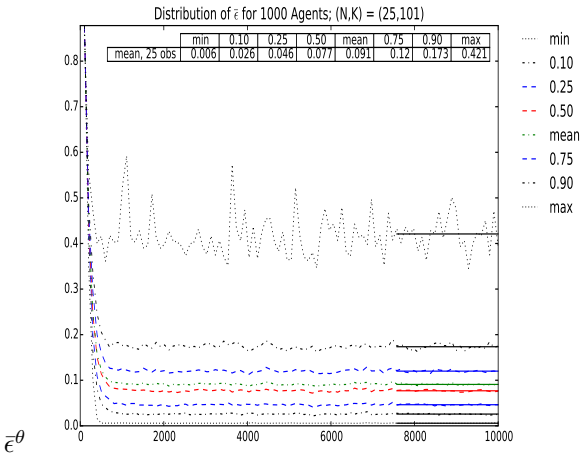
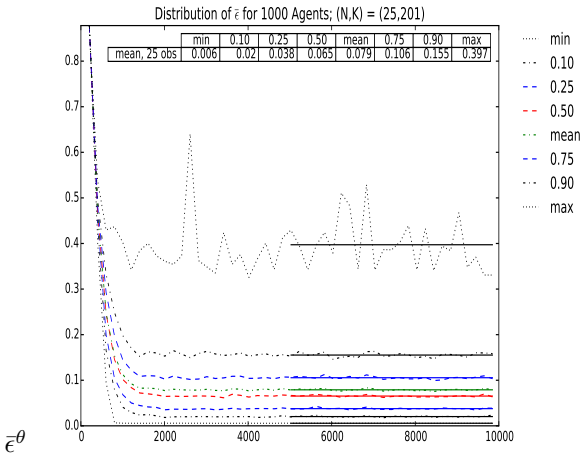


Figure 33: Distribution of $\bar{\epsilon}^{\theta}$ Over Time, N = 11, D = 101, 201, 301

N = 25, D = 101



N = 25, D = 201



N = 25, D = 301

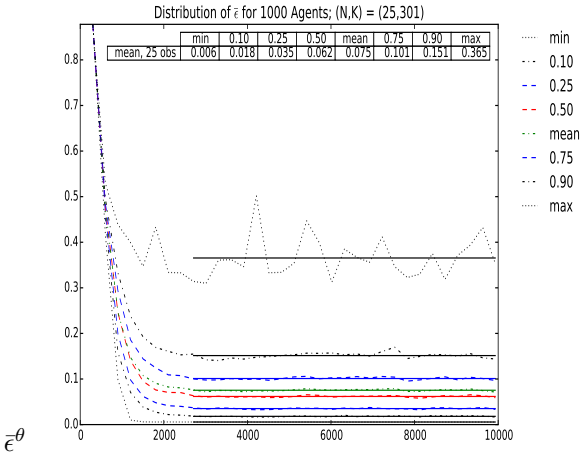
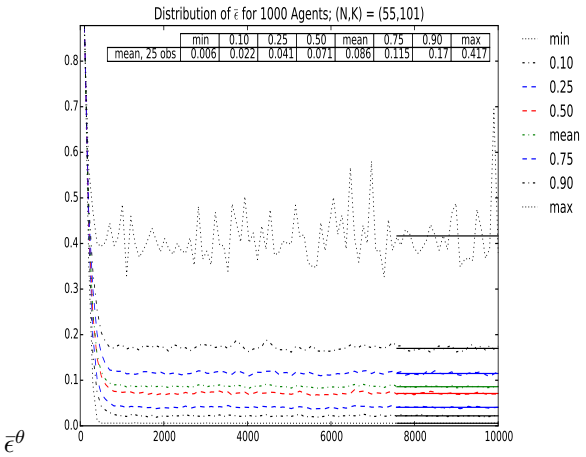
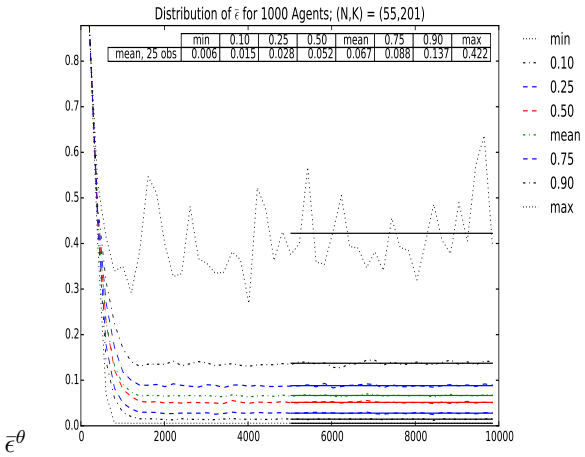


Figure 34: Distribution of $\bar{\epsilon}^{\theta}$ Over Time, N = 25, D = 101, 201, 301

N = 55, D = 101



N = 55, D = 201



N = 55, D = 301

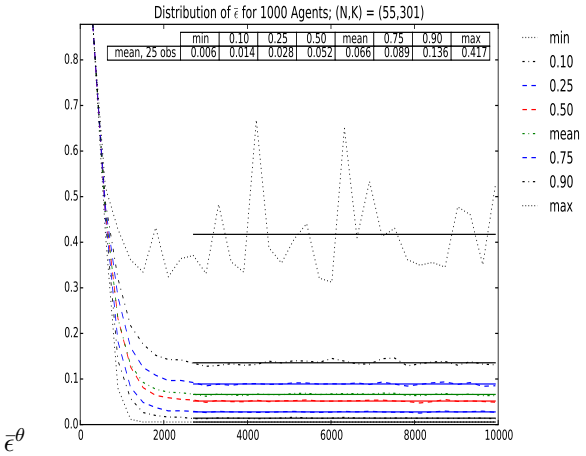


Figure 35: Distribution of $\bar{\epsilon}^{\theta}$ Over Time, N = 55, D = 101, 201, 301

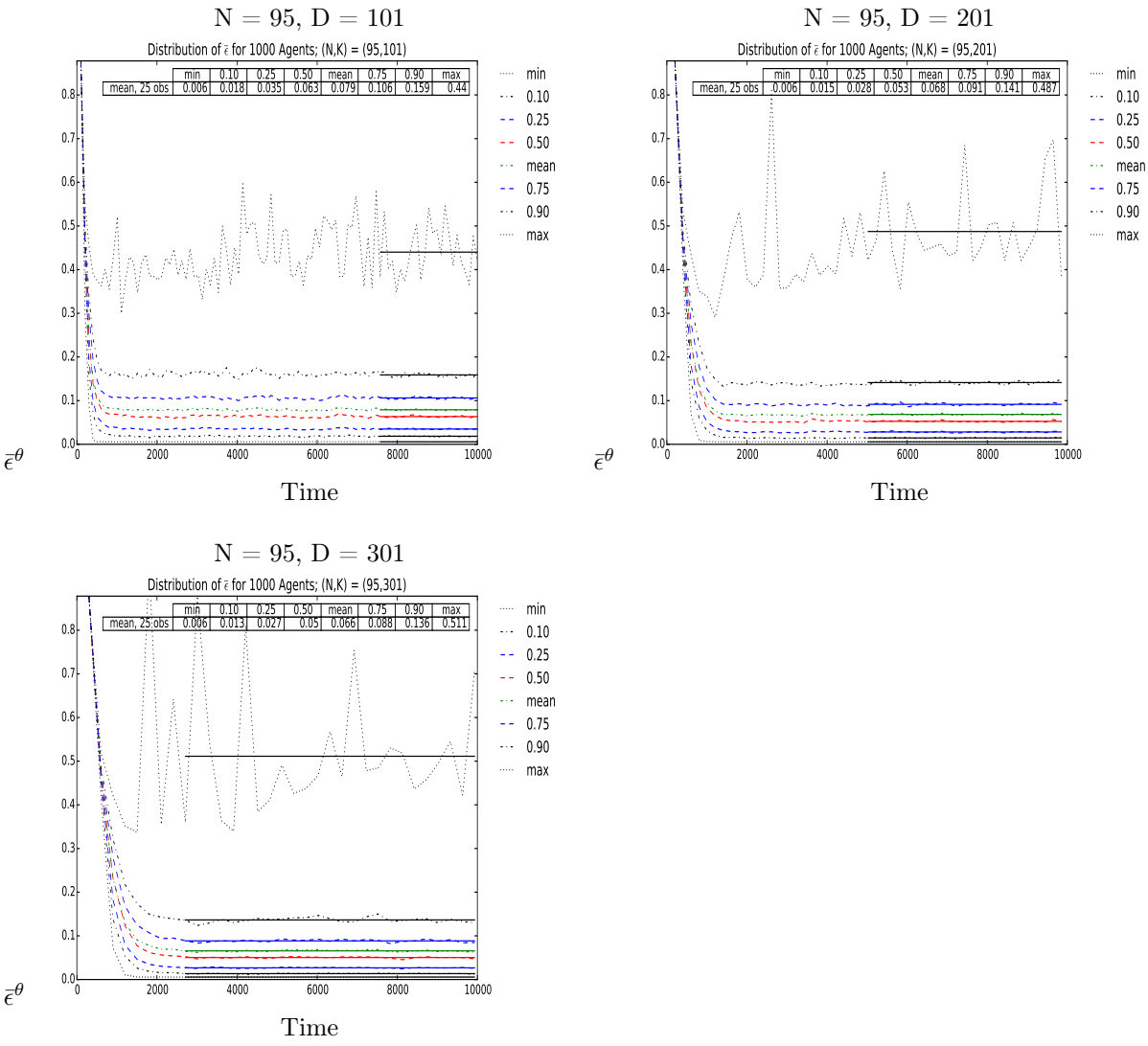


Figure 36: Distribution of $\bar{\epsilon}^{\theta}$ Over Time, N = 95, D = 101, 201, 301

Each plot in Figures (26) through (36) display the average of the distributional characteristics – min, median, mean , max, and 10^{th}, 25^{th}, 75^{th}, and 90^{th} percentiles – for the final 25 periods. There are displayed in each plot, and additionally are recored in agonizing detail in Tables (3) and (4).

Somewhat easier on the eyes, the mean, median, 90^{th} and 10^{th} percentiles rows from Tables (3) and (4) are displayed in Figures (37) through (40), along with the $90^{th} - 10^{th}$ inter-percentile range, a measure of distribution dispersion, displayed in Figure (41).

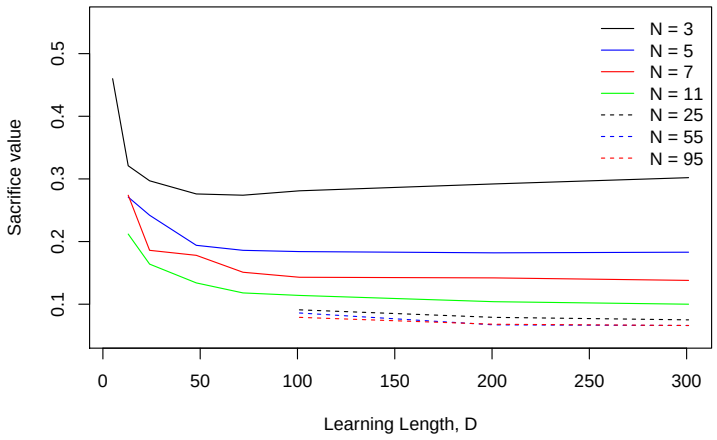


Figure 37: Mean of $\bar{\epsilon}^{\theta}$ Distribution for all N x D values

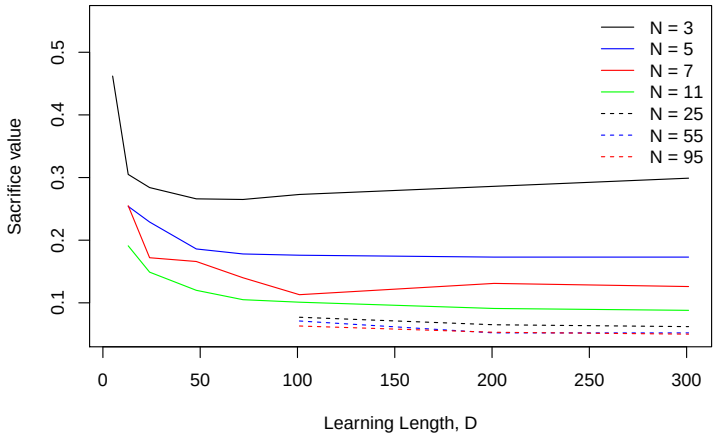


Figure 38: Median of $\tilde{\epsilon}^{\theta}$ Distribution for all N x D values

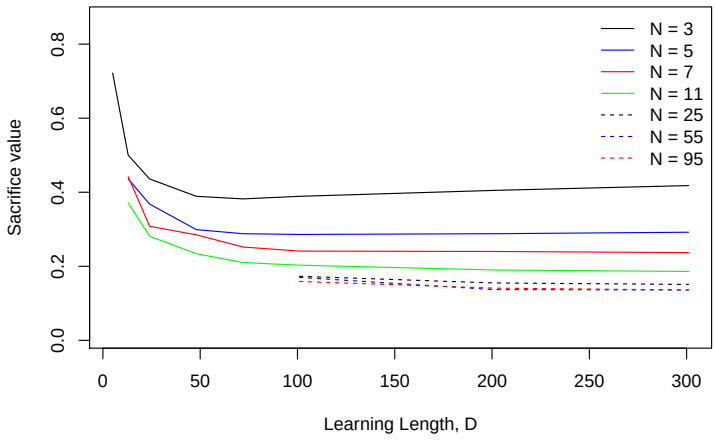


Figure 39: 90^{th} Percentile of $\bar{\epsilon}^{\theta}$ Distribution for all N x D values

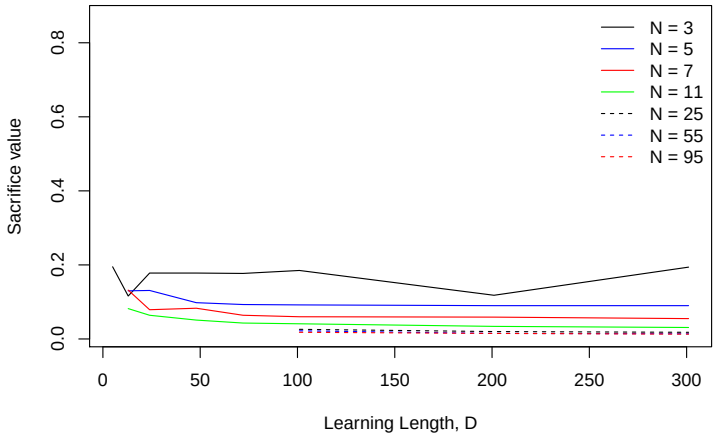


Figure 40: 10^{th} Percentile of $\bar{\epsilon}^{\theta}$ Distribution for all N x D values

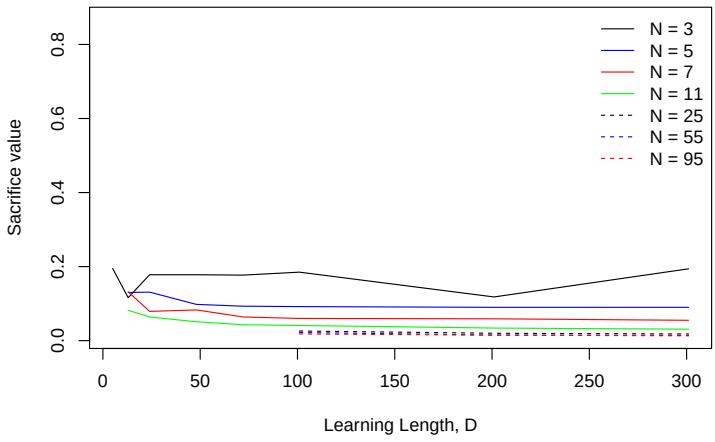


Figure 41: $90^{th} - 10^{th}$ Inter-Percentile Range for $\bar{\epsilon}^{\theta}$ Distribution for all N x D values

Table 3: Distribution of $\bar{\epsilon^{\theta}}$ Over Time, N = 3,5,7,11, D = 13, 24, 48, 72, 101, 201, 301

N=3	5	13	24	48	72	101	201	301
min	0.008	0.016	0.041	0.06	0.056	0.065	0.061	0.069
0.1	0.195	0.116	0.178	0.178	0.177	0.185	0.118	0.194
0.25	0.313	0.227	0.225	0.218	0.218	0.225	0.232	0.238
0.5	0.462	0.305	0.284	0.266	0.265	0.273	0.286	0.299
mean	0.46	0.321	0.297	0.276	0.274	0.281	0.292	0.302
0.75	0.608	0.400	0.355	0.322	0.032	0.328	0.346	0.361
0.9	0.722	0.500	0.436	0.389	0.382	0.389	0.405	0.418
max	0.91	0.797	0.735	0.646	0.611	0.006	0.588	0.576
N=5	5	13	24	48	72	101	201	301
min	-	0.011	0.02	0.013	0.013	0.015	0.015	0.012
0.1	-	0.13	0.131	0.098	0.093	0.092	0.09	0.09
0.25	-	0.184	0.176	0.138	0.132	0.013	0.127	0.126
0.5	-	0.254	0.229	0.186	0.178	0.176	0.173	0.173
mean	-	0.271	0.242	0.194	0.186	0.184	0.182	0.183
0.75	-	0.339	0.294	0.239	0.231	0.228	0.228	0.232
0.9	-	0.436	0.368	0.299	0.288	0.286	0.288	0.292
max	-	0.777	0.685	0.059	0.539	0.517	0.501	0.494
N=7	5	13	24	48	72	101	201	301
min	-	0.015	0.009	0.011	0.008	0.008	0.007	0.008
0.1	-	0.132	0.079	0.083	0.064	0.06	0.059	0.055
0.25	-	0.187	0.12	0.12	0.096	0.091	0.09	0.086
0.5	-	0.255	0.172	0.166	0.14	0.113	0.131	0.126
mean	-	0.274	0.186	0.178	0.151	0.143	0.142	0.138
0.75	-	0.342	0.235	0.223	0.192	0.183	0.183	0.18
0.9	-	0.442	0.308	0.285	0.252	0.241	0.24	0.237
max	-	0.786	0.624	0.554	0.497	0.491	0.455	0.446
N=11	5	13	24	48	72	101	201	301
min	-	0.007	0.008	0.007	0.007	0.006	0.006	0.006
0.1	-	0.082	0.064	0.051	0.043	0.041	0.034	0.031
0.25	-	0.129	0.101	0.08	0.068	0.065	0.057	0.054
0.5	-	0.191	0.149	0.12	0.105	0.101	0.091	0.088
mean	-	0.212	0.164	0.134	0.118	0.114	0.104	0.1
0.75	-	0.27	0.208	0.173	0.153	0.148	0.135	0.132
0.9	-	0.371	0.281	0.234	0.21	0.203	0.19	0.186
max	-	0.779	0.609	0.513	0.463	0.443	0.419	0.402

Table 4: Distribution of $\bar{\epsilon}^{\theta}$ Over Time, N = 25,55,95, D = 101, 201, 301

N=25	5	13	24	48	72	101	201	301
min	-	-	-	-	-	0.006	0.066	0.006
0.1	-	-	-	-	-	0.026	0.02	0.018
0.25	-	-	-	-	-	0.046	0.038	0.035
0.5	-	-	-	-	-	0.077	0.065	0.062
mean	-	-	-	-	-	0.091	0.079	0.075
0.75	-	-	-	-	-	0.12	0.106	0.101
0.9	-	-	-	-	-	0.173	0.155	0.151
max	-	-	-	-	-	0.421	0.397	0.365
N=55	5	13	24	48	72	101	201	301
min	-	-	-	-	-	0.006	0.006	0.006
0.1	-	-	-	-	-	0.022	0.015	0.014
0.25	-	-	-	-	-	0.041	0.028	0.028
0.5	-	-	-	-	-	0.071	0.052	0.052
mean	-	-	-	-	-	0.086	0.067	0.066
0.75	-	-	-	-	-	0.115	0.088	0.089
0.9	-	-	-	-	-	0.17	0.137	0.136
max	-	-	-	-	-	0.417	0.422	0.417
N=95	5	13	24	48	72	101	201	301
min	-	-	-	-	-	0.006	0.006	0.006
0.1	-	-	-	-	-	0.018	0.015	0.013
0.25	-	-	-	-	-	0.035	0.028	0.027
0.5	-	-	-	-	-	0.063	0.53	0.05
mean	-	-	-	-	-	0.079	0.068	0.066
0.75	-	-	-	-	-	0.106	0.091	0.088
0.9	-	-	-	-	-	0.159	0.141	0.136
max	-	-	-	-	-	0.44	0.487	0.511

Table 5: Comparing Results with Previous Literature

Periods Required			
Expected $\bar{\epsilon}^{\theta}$	Allen and Carroll (2001) 400 rules	5 rules*	Regret Learning 3-5 rules
0.27	4,000	50	39-60
0.12	40,000	500	216-360
0.08	80,000	1,000	303-505

There are a number of observations one can take from Figures (26) through (36), Tables (3) and (4), and particularly Figures (37) through (40).

Agents converge to a fairly stable distribution of sacrifice values relatively quickly, well within 5-10 episodes of beginning with a spendthrift "consumer everything" consumption function. This can be difficult to see for low-D Figures, such as Figure (26), but can be seen much more clearly for higher-D values such as Figure (27).

To discuss the effectiveness of regret learning, I focus on the mean and median sacrifice values, averaged over the last 25 learning episodes from each simulation.[20] While the mean and median values can be observed in the full distribution plots or tables, these summary values are most easily seen in Figures (37) and (38). Here it is readily clear that for a fixed N value, the mean and median sacrifice value are nearly flat past a learning-length of roughly 75-100 periods – this can literally be seen as the flat portions of the blue, green, and red lines in each plot.[21] In addition, it can also be seen that there are smaller and smaller improvements in learning when increasing N for a fixed D. For example, for $D = 100$, there is a large improvement in moving from $N = 3$ to $N = 5$, a slightly smaller improvement for $N = 5$ to $N = 7$, somewhat smaller again from $N = 7$ to $N = 11$, and even smaller again from $N = 11$ to $N = 25$, a the largest increase in N for the smallest improvement thus seen. From $N = 25$ to $N = 55$, a nearly doubling of the number of bins, the improvement is almost indistinguishable. The same is true for the next near-doubling of N-values, from $N = 55$ to $N = 95$.

For the highest N, D combinations, for example $N = 95, D = 301$, is to possible to push the median sacrifice value to 5%. This can be clearly seen in Table (4). Even the 90^{th} percentile achieves a sacrifice value of ~15% across all episode lengths. This is a more than 80% decrease from the original sacrifice value of 90% under the spendthrift consumption function.

5.2 Compared To Related Literature

A very brief review of the results of Allen and Carroll (2001), Howitt and Özak (2014), Özak (2014), Yıldızoğlu et al. (2014), and others regarding learning the solution to the consumption-under uncertainty problem could be stated as follows. Learning the optimal solution is possible, but there is a tradeoff required between the time needed and agent sophistication. For Allen and Carroll (2001) in particular, the Table (5) sums up an important set of their results.

Agents in Allen and Carroll (2001) conduct a brute-force search over a grid of 400 policy rules in a predefined policy space. They use their experience to learn the value of each rule and then select the best rule after their experience is over. The rule selected by an agent is scored by the welfare cost described above, and this is repeated 100 times so the authors could estimate the distribution of welfare costs associated with

[20] Recall that a learning episode is D periods long – so for example, 25 episodes of $D = 25$ periods each totals to 625 periods.
[21] The slight upward trend for the $N = 3$ plot is unusual, and there is not yet a good explanation for this.

a given time to search.

Table (5) displays a sample of results from Allen and Carroll (2001). For expected welfare costs of $\bar{\epsilon}^{\theta} = 0.12, 0.08, 0.06$ the first column shows the time required for the brute force search over 400 rules. The times are astronomical, as might be expected. Allen and Carroll (2001) speculate that there might someday be a highly efficient search routine which selected rules with the same efficiency; even with this efficiency this would produce the times seen in the hypothetical second column.

The third column demonstrates times associated selected rules from Tables (3) and (4) which match the $\bar{\epsilon}^{\theta}$ values displayed. These times demonstrate that regret learning fulfills the highly efficient search routine that Allen and Carroll (2001) speculated may exist. It reaffirms a difficulty that is still present, however, which is that even though 505 periods is much less than 1,000 periods, both are still well outside the realm of reasonable time frames for agent behavior.

There are a few caveats. First, for these purposes, regret learning is initialized in something of a worst case position, starting with the "consume everything" consumption function. Trials not shown here indicate that when started closer to the optimal rule, regret learning can achieve its stable welfare neighborhood within 1-2 episodes. This is still limited by the lower bound of the length of each episode, which in the table of results above is 13-101 periods.

Second, early results (not discussed here) indicate that when regret learning is coupled with the social learning and more efficient estimators described in Palmer (2012), learning can converge toward the optimal rule very quickly. The key is whether or not the agents are allowed to switch rules before finishing an episode, an option not allowed under the current regret learning setup.

5.3 Persistence of Position in the Welfare Distribution

An important question is whether or not an agent's position in the distribution of sacrifice values is persistent. That is, does an agent learn a relatively bad rule and "get stuck there," or vice versa, learn a relatively good rule and forever reap the associated rewards?

The answer is no – once past the first few episodes, agents move freely throughout their distribution. There are a number of was to examine this: directly estimate an AR process on the time series of each agent's position in the distribution of sacrifice values the persistence in an AR process, or examine the distribution of consecutive periods spent in deciles of the sacrifice distribution, or look at the distribution of longest streaks for all agents. All measurements, however, indicate that there is very little persistence of an agent's *relative* position in the distribution of sacrifice values over time. In fact the distribution of sacrifice values over time appears to be an excellent description of what the agent can expect one period ahead. For example, an agent with $N = 5$ and $D = 24$ has a roughly 10% chance of attaining a sacrifice value ≤ 0.13 in a following episode, and a roughly 90% change of attaining a consumption rule with a sacrifice value ≤ 0.3.

The reason for the lack of persistence in agent experience is straightforward. Regret learning takes a stark approach to all information prior to the learning episode: it is forgotten completely. If the regret learning algorithm described in Section (4.9) (and requisite foundational sections) is examined closely, it can be seen that no information is used from prior episodes when forming both the value function estimate, the regret choices, or the regret-minimizing consumption function. Importantly, the *prior* value functions learned from previous periods are forgotten entirely. This is also departure from the algorithms for policy iteration and optimistic policy iteration, specifically in the policy evaluation step.[22]

[22]It is not, however, complete departure, because much of the mechanism by which policy iteration is shown to converge relies on the monotonicity of T and T_{μ} in transforming v. This monotonicity is largely preserved in the parallel approximating

The reason for this omission is straightforward: if agents are to remember all past information, one must take a stance on how this is incorporated into agent behavior. The single-stream Monte Carlo estimator has no simple way to incorporate previous value functions. A number of different potential approaches each raise important methodological questions. If one were to simply attempt to average in previous experience, even aside from issues raised by the shifting bins, one must take a stance on whether much earlier information should be treated differently than very recent information – that is, one needs to take a stance on the "gains" which may be used.[23] Having agents forget information completely when starting each new episode is a clean, simple first step in exploring the behavior of regret learning.

6 Conclusion and Next Steps

This paper introduces regret learning, a non-parametric learning algorithm applied to the canonical infinite horizon consumption under uncertainty problem with risky income. Regret learning approximates policy iteration using an individual agent's experience. Numerical simulation demonstrates that learning produces in solutions in a stable distribution near the true optimal solution. With enough experience, learned consumption rules are distributed very close to the true optimal solution in welfare terms, but even without extensive time they settle into a stable distribution the optimal behavior.

The goal of regret learning is to capture the simplest possible model of learning in a dynamic environment with minimal new parameters, while maintaining convergence to the optimal solution. It is meant to be easy to understand and immediately implementable for the economist and agent-based modeler who is familiar with dynamic programming. Furthermore, it has been designed from the ground up to be nearly trivial to take to data – this is a natural result of being closely tied to the dynamic programming framework used to solve similar problems in the consumption under uncertainty literature. Future work will employ the Heterogeneous-Agent Resources toolKit (HARK), presented in Carroll and Palmer (2015), to estimate a version of regret learning alongside traditional models.[24]

There are a number of immediate next steps for regret learning, as well as longer term extensions. These include:

- Estimating the model against data to establish a fit for risk aversion, the discount factor, and N and D parameters.
- Implementing variations of the regret learning algorithm:
  - Replacing the fixed D-length learning periods with a dynamic learning period: an (s,S) update trigger or Calvo-style Poisson trigger.
  - Bayesian learning of the shock and state distributions.
  - Regret learning in an optimistic policy iteration framework.
- Use simulation-based model selection, as discussed in Shalizi (2015), to distinguish between different possible variations of the regret-learning framework as suggested above.

steps in regret learning, particularly the single-stream estimation of v discussed in Section (4.6.2). I have yet to show the monotonicity of policy improvement step for regret learning; this is currently underway as as part of future work.

[23] See LeBaron (2012) for an excellent examination of related questions in agent-based models generally, and see Evans and Honkapohja (2001) for a broad discussion of the topic in macroeconomic learning.

[24] The fact that the HARK framework estimates a finite-horizon consumption savings buffer-stock problem does not present a problem; as has been often observed in Cagetti (2003), Carroll and Samwick (1997), and Gourinchas and Parker (2002), households do not seem to accumulate liquid savings for retirement until roughly 45-50 years old. This leaves at least 30 years of data against which the infinite-horizon model above can be estimated without fear of structural mismatch.

- Calculate convergence properties and error bounds.
- Continue to explore the analytical foundations of regent leaning, including application of non-expansive transformations in an approximate policy iteration framework.
- Construct an empirical "horse race" between various versions of the learning algorithms in the small but growing "learning to optimize" literature.

The original motivation for developing regret learning was to provide agents in Palmer (2012) with a robust method of local exploration. This has been archived, and in addition regret learning presents an immediate option to include social learning into the model. Because agents estimate the full distribution and full value function in regret learning, they can communicate their unconditional expectation of the value of using their function, and can thus socially share information as in Palmer (2012) with any agent in their population. Early results (not discussed here) results suggest that learning can occur quite quickly with both social and individual aspects are combined.

Finally, this model is explicitly intended to be used in large-scale agent-based models such as Geanakoplos et al. (2012) or the CRISIS model (Hommes and Iori, 2015), or large rational-expectations macroeconomic and macrofinance models in which rational expectations is intractable. These is a prime example of the appropriate application of regret learning to agent-based models, and the type of application for which it is very well suited.

References

Allen, T. W. and Carroll, C. D. (2001). Individual learning about consumption. *Macroeconomic Dynamics*, 5(02):255–271.

Auer, P., Cesa-Bianchi, N., and Fischer, P. (2002). Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256.

Ballinger, T. P., Palumbo, M. G., and Wilcox, N. T. (2003). Precautionary saving and social learning across generations: an experiment*. *The Economic Journal*, 113(490):920–947.

Başçı, E. and Orhan, M. (2000). Reinforcement learning and dynamic optimization. *Journal of Economic and Social Research*, 2(1):39–57.

Bertsekas, D. P. (2012). *Dynamic Programming and Optimal Control, Vol. II, 4th Edition: Approximate Dynamic Programming*. Athena Scientific.

Bertsekas, D. P. (2013). Abstract dynamic programming. *Athena Scientific, Belmont, MA*.

Bertsekas, D. P. and Tsitsiklis, J. N. (1996). Neuro-dynamic programming (optimization and neural computation series, 3). *Athena Scientific*, 7:15–23.

Brown, A. L., Chua, Z. E., and Camerer, C. (2009). Learning and visceral temptation in dynamic savings experiments. *Quarterly Journal of Economics*, 124(1):197–231.

Cagetti, M. (2003). Wealth accumulation over the life cycle and precautionary savings. *Journal of Business & Economic Statistics*, 21(3):339–353.

Carbone, E. and Duffy, J. (2014). Lifecycle consumption plans, social learning and external habits: Experimental evidence. *Journal of Economic Behavior & Organization*, 106:413–427.

Carroll, C. D. (1997). Buffer-stock saving and the life cycle/permanent income hypothesis*. *The Quarterly journal of economics*, 112(1):1–55.

Carroll, C. D. (2001a). Death to the log-linearized consumption euler equation!(and very poor health to the second-order approximation). *Advances in Macroeconomics*, 1(1).

Carroll, C. D. (2001b). A theory of the consumption function, with and without liquidity constraints (expanded version). *National Bureau of Economic Research*.

Carroll, C. D. (2012a). Solving microeconomic dynamic stochastic optimization problems. *Lecture Notes, The Johns Hopkins University, Department of Economics*.

Carroll, C. D. (2012b). Theoretical foundations of buffer stock saving. *Mimeo, The Johns Hopkins University, Department of Economics*.

Carroll, C. D., Hall, R. E., and Zeldes, S. P. (1992). The buffer-stock theory of saving: Some macroeconomic evidence. *Brookings papers on economic activity*, pages 61–156.

Carroll, C. D., Otsuka, M., and Slacalek, J. (2011a). How large are housing and financial wealth effects? a new approach. *Journal of Money, Credit and Banking*, 43(1):55–79.

Carroll, C. D. and Palmer, N. M. (2015). The heterogeneous-agent compumetrik toolkit: An extensible framework for solving and estimating hetergeneous-agent models. *Working Paper, presented at Computing in Economics and Finance Conference, Society of Computational Economics*.

Carroll, C. D. and Samwick, A. A. (1997). The nature of precautionary wealth. *Journal of monetary Economics*, 40(1):41–71.

Carroll, C. D., Slacalek, J., and Sommer, M. (2011b). International evidence on sticky consumption growth. *Review of Economics and Statistics*, 93(4):1135–1145.

Chua, Z. and Camerer, C. F. (2011). Experiments on intertemporal consumption with habit formation and social learning. *Mimeo, California Institute of Technology, Division of the Humanities and Social Sciences*.

Evans, G. W. and Honkapohja, S. (2001). *Learning and expectations in macroeconomics*. Princeton University Press.

Evans, G. W. and McGough, B. (2014). Learning to optimize. *Mimeo, The University of Oregon, Department of Economics*.

Gabaix, X. (2014). A sparsity-based model of bounded rationality. *The Quarterly Journal of Economics*, 129(4):1661–1710.

Geanakoplos, J., Axtell, R., Farmer, D. J., Howitt, P., Conlee, B., Goldstein, J., Hendrey, M., Palmer, N. M., and Yang, C.-Y. (2012). Getting at systemic risk via an agent-based model of the housing market. *The American Economic Review*, 102(3):53–58.

Gourinchas, P.-O. and Parker, J. A. (2002). Consumption over the life cycle. *Econometrica*, 70(1):47–89.

Hommes, C. and Iori, G. (2015). Introduction special issue crises and complexity. *Journal of Economic Dynamics and Control*, 50:1 – 4. Crises and Complexity Complexity Research Initiative for Systemic InstabilitieS (CRISIS) Workshop 2013.

Houser, D., Keane, M., and McCabe, K. (2004). Behavior in a dynamic decision problem: An analysis of experimental evidence using a bayesian type classification algorithm. *Econometrica*, 72(3):781–822.

Howitt, P. and Özak, Ö. (2014). Adaptive consumption behavior. *Journal of Economic Dynamics and Control*, 39:37–61.

Hughes, N. (2014). Applying reinforcement learning to economic problems. *Working Paper, Australian National University, Department of Economics*.

Hughes, N. (2015). Solving large stochastic games with reinforcement learning. *Working Paper, Australian National University, Department of Economics*.

Hull, I. (2012). Interest rate rules and mortgage default. *Job Market Paper, Boston College, Department of Economics*.

Hyndman, R. J. and Fan, Y. (1996). Sample quantiles in statistical packages. *The American Statistician*, 50(4):361–365.

Jaksch, T., Ortner, R., and Auer, P. (2010). Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11:1563–1600.

Jirnyi, A. and Lepetyuk, V. (2011). A reinforcement learning approach to solving incomplete market models with aggregate uncertainty. *Available at SSRN 1832745*.

Kahneman, D. and Tversky, A. (1979). Prospect theory: An analysis of decision under risk. *Econometrica: Journal of the Econometric Society*, pages 263–291.

Krusell, P. and Smith, A. A. (1996). Rules of thumb in macroeconomic equilibrium a quantitative analysis. *Journal of Economic Dynamics and Control*, 20(4):527–558.

Krusell, P. and Smith Jr, A. (1998). Income and wealth heterogeneity in the macroeconomy. *Journal of Political Economy*, 106(5):867–896.

LeBaron, B. (2012). Heterogeneous gain learning and the dynamics of asset prices. *Journal of Economic Behavior & Organization*, 83(3):424–445.

Lettau, M. and Uhlig, H. (1999). Rules of thumb versus dynamic programming. *American Economic Review*, pages 148–174.

Ljungqvist, L. and Sargent, T. (2012). Recursive macroeconomic theory.

Loomes, G. and Sugden, R. (1982). Regret theory: An alternative theory of rational choice under uncertainty. *The economic journal*, pages 805–824.

Lucas, R. E. (1976). Econometric policy evaluation: A critique. In *Carnegie-Rochester conference series on public policy*, volume 1, pages 19–46. Elsevier.

Mannor, S. and Tsitsiklis, J. N. (2004). The sample complexity of exploration in the multi-armed bandit problem. *The Journal of Machine Learning Research*, 5:623–648.

Mas-Colell, A., Whinston, M. D., and Gibbons, R. (1995). Microeconomic theory.

Özak, Ö. (2014). Optimal consumption under uncertainty, liquidity constraints, and bounded rationality. *Journal of Economic Dynamics and Control*, 39:237–254.

Pál, J. and Stachurski, J. (2013). Fitted value function iteration with probability one contractions. *Journal of Economic Dynamics and Control*, 37(1):251–264.

Palmer, N. M. (2012). Learning to consume: Individual versus social learning. *Working Paper, presented at Computing in Economics and Finance Conference, Society of Computational Economics*.

Powell, W. B. (2007). *Approximate Dynamic Programming: Solving the curses of dimensionality*, volume 703. John Wiley & Sons.

Robbins, H. and Monro, S. (1951). A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407.

Sargent, T. J. (1993). Bounded rationality in macroeconomics: The arne ryde memorial lectures. *OUP Catalogue*.

Schelling, T. C. (2006). *Micromotives and macrobehavior*. WW Norton & Company.

Shalizi, C. R. (2015). Advanced data analysis from an elementary point of view. *Mimeo, Carnegie Mellon University, Department of Statistics*.

Sims, C. A. (1980). Macroeconomics and reality. *Econometrica*, pages 1–48.

Singh, S. P. and Sutton, R. S. (1996). Reinforcement learning with replacing eligibility traces. *Machine learning*, 22(1-3):123–158.

Sinitskaya, E. and Tesfatsion, L. (2014). Macroeconomies as constructively rational games. *Mimeo, Iowa State University, Department of Economics*.

Stachurski, J. (2009). *Economic dynamics: theory and computation*. MIT Press.

Sutton, R. S. and Barto, A. G. (1998). *Reinforcement learning: An introduction*. MIT press Cambridge.

Tesfatsion, L. and Judd, K. L. (2006). *Handbook of computational economics: agent-based computational economics*, volume 2. Elsevier.

Vanderbilt, T. (2013). Unhappy truckers and other algorithmic problems. *Nautilus*. [Online; posted 18-July-2013].

Yıldızoğlu, M., Sénégas, M.-A., Salle, I., and Zumpe, M. (2014). Learning the optimal buffer-stock consumption rule of carroll. *Macroeconomic Dynamics*, 18(04):727–752.