Economic Games as Estimators[*]

Michael Zargham[1,2], Krzysztof Paruch[1], and Jamsheed Shorish[1]

[1] Vienna University of Economics and Business, Vienna, Austria
{michael.zargham,krzysztof.paruch,jamsheed.shorish}@wu.ac.at
[2] BlockScience, Inc, California, USA
zargham@block.science

Abstract. Discrete event games are discrete time dynamical systems whose state transitions are discrete events caused by actions taken by agents within the game. The agents' objectives and associated decision rules need not be known to the game designer in order to impose structure on a game's reachable states. Mechanism design for discrete event games is accomplished by declaring desirable invariant properties and restricting the state transition functions to conserve these properties at every point in time for all admissible actions and for all agents, using techniques familiar from state-feedback control theory. Building upon these connections to control theory, a framework is developed to equip these games with estimation properties of signals which are private to the agents playing the game. Token bonding curves are presented as discrete event games and numerical experiments are used to investigate their signal processing properties with a focus on input-output response dynamics.

Keywords: Estimation · Dynamic Games · Cryptoeconomic Systems.

1 Introduction

Cryptoeconomic systems [26] are data-driven, multiscale, adaptive and dynamic networks with a system-level state available to all agents. These systems use cryptographic tokens as information carriers, allowing for economic activities to emerge on top of a shared distributed ledger technology (DLT) enabled infrastructure such as blockchain. Formally, these economies can be described using a state space representation [23], [31], [32], which allows the encoding of agents, transactions and mechanisms, as well as state transitions resulting from activities within the network. Additional requirements on reachable system states can be imposed using *configuration spaces* [30] to design possible future system trajectories, without assuming agents' decision rules or specifying agents' preference functions.

Understanding the formal structure of cryptoeconomic systems is facilitated using game theory, a mathematical framework that formalizes the dynamics of

[*] Supported by the Research Institute for Cryptoeconomics at WU Vienna in collaboration with BlockScience, Inc.

multi-agent systems within a spectrum of repeated discrete games (see e.g. [7]) on the one side and continuous differential games (e.g. [10]) on the other side. Game theory has been applied to cryptoeconomic systems in a variety of ways. In a DLT protocol layer, the economics of consensus mechanisms [1] and the effects on network security [15] have been studied using game theoretic concepts. Applications of DLT to finance, such as portfolio diversification, have also been studied in [5].

Game-theoretic models are usually based upon a specification of the players, or *agents*, and their preferences, strategy sets and associated payoffs. While standard 'toy' models such as the discrete-time repeated prisoner's dilemma (see e.g. [21]) and the continuous-time conflict models (e.g. [10]) are pedagogically useful, more complicated models are required for greater realism. For example, population games [22] model strategic interactions with a large number of individually negligible agents with an explicitly dynamic model of individual choice defined by the revision protocol of every agent. When stochastic revision opportunities arise, agents are free to change strategies, resulting in a Markov process describing the mean dynamics of the system. Three important classes of population games are potential games [18], supermodular games [25] and stable games [9]. Potential games use a single global potential function to represent all players' incentives to change their strategies, which adds additional structure to the game environment. An equilibrium is guaranteed to exist, and there is a wide array of distributed learning algorithms that guarantee convergence.

Learning in games [28], [8], [6] explores how a process might emerge for convergence to e.g. a Nash equilibrium from various initial conditions. Evolutionary games are similar, focusing on the dynamics of strategy changes within a population [22]. Finally, mean field games are sequential games with a continuum of players, in which players affect their opponents in ways that are insignificant at the individual level but significant when aggregated [12], and evolution takes place according to a dynamical relation [27], [17].

Each of the above modeling paradigms possess different 'encodings' of space and time, agent models, system interaction, and payoffs. But there exists a sufficiently general notion of a game that can subsume most or all of these encodings, by interpreting a game as a *system* evolving over time based on the actions of a group of agents. The game is then akin to the plant of a control system, and the agents as a collection of individual controllers with private state, signals, and objectives. Under this interpretation, the design of the game is the design of the system plant, to be controlled by an a priori unknown set of controllers–the agents–and becomes a formal mechanism design problem. We introduce such an interpretation in this work, defining *discrete event games* as discrete time dynamical systems whose state transitions are discrete events caused by actions taken by agents within the game. In this approach there are observable and provable states of the interpreted system plant, regardless of agent objectives, decision rules, etc. which are in general not known to the designer.

Under this interpretation, changes to the system state caused by agent actions act as samples of their private preferences or private information. This allows one to consider a game as an *estimator* that gathers information over time from a multi-agent system [14]. The game design, then, acts to dynamically estimate useful summary statistics of the underlying distribution of agents, even as that distribution changes over time. If, for example, agent decision-making influences the price of an asset (such as a cryptocurrency token), then it is the price which is estimated by the discrete event game. In a similar fashion, the design pattern of combining discrete event models with agent behavior and system parameter estimation also arises in cyber-physical systems [11].

This work contributes to the existing literature in game theory, market theory and estimation theory. It analyzes economic games played by agents on intentionally shaped sub-spaces of the state space, namely on lower-dimensional manifolds. These manifolds are constructed by the system designers to ensure that all economic activity takes place on a plane specifically shaped for this purpose, to reflect the conditions under which the game is intended to be played. A configuration space ensures that these lower-dimensional manifolds have designed characteristics and conserve invariant properties of the system. This allows the focus to shift to the conditions and states of the game, rather than to the particular behavior, strategies and private preferences of agents (since some properties of the system will stay true regardless of the choices of its participants). The game outcome allows the inference of system-level properties that are revealed by actions taken by the agents, *without knowing further details about their particular preferences.* By doing so, it is aggregated agent behavior that acts as a signal, estimating specific parameters (such as prices, treated in this work).

The creation of the conditions for a digital economic game with enforced state space restrictions described above (and expressed in further detail in [30]) is ensured by the use of DLT, which maintains a tamper-proof universal state layer. Whether a future state is reachable will heavily depend upon the design of the configuration space, which can be restricted using *token bonding curves* to impose invariant properties upon the system and thereby limit possible system state trajectories. System participants will make their decisions based on the utility gains they perceive from evaluations of expected effects (in accordance with their private hidden preferences), but will take into consideration that outcomes will follow global *laws of motion* dictated by the 'rules of the game' encoded in the shape of the space. A market for tokens emerges as a result of economic activity between agents, with the token price as a variable describing one particular global property of the system estimated from individual signals of constituent agents. This observation paves the way for a possible contribution to price theory, treating bonding curves as estimators of market prices.

The paper is structured as follows: Section 2 introduces the notation and definitions required to formally represent discrete event games, configuration spaces and the estimation framework. Section 3 reviews the characteristics of bonding curves within the state space representation, via the configuration space and the representation's mechanisms. It continues with a description of the formal process of global price estimation derived from private signals of the agents. Section 4 then presents dynamic price estimation with open loop agents, derived from numerical results for a specific bonding curve parametrization. Finally, Section 5 concludes and outlines future work.

2 Notation and Definitions

2.1 Discrete Event Games

Consider a system in which *agents* interact within a *network*, the topology of which is specified as part of a global *system.* Agents are decision-making entities that are completely characterized by their *state*, considered as a finite vector of k elements taken from a field. In what follows it is assumed that the field is the usual real number line $\mathbb{R}$, but this may be generalized. The state characterizes the agent insofar as it specifies 'private' information known only that agent.

Agents are indexed by an identifier $a \in \{1, 2, 3, \dots, n < \infty\}$, while time $t \in \mathbb{Z}^{\geq 0}$ is a 'lattice' upon which agent decisions and actions are placed. Time also indexes the flow of information, which impacts the state of the agent. Thus, an agent's state may be summarized by a vector $\hat{x}_{a,t}$. Denote the *agent state space* by $\hat{X}_a \subseteq \mathbb{R}^k$, so that $\forall a$, $\forall t$, $\hat{x}_{a,t} \in \hat{X}_a$.

The *agent-level state* is decentralized but may nonetheless be summarized as

$$(\hat{x}_{1,t}, \hat{x}_{2,t}, \dots, \hat{x}_{n,t}) \in \prod_{a=1}^{n} \hat{X}_a \subseteq \mathbb{R}^{nk}.$$

The network carries its own internal state, the *system-level state.* As the network is assumed to be a finite (probabilistic or deterministic) state machine, the internal state may be given by a finite vector of m elements, with (as in the agent case) elements taken from a field. For simplicity we again assume that the field is identical for all elements and equals $\mathbb{R}$, but the approach (and future research) accommodates arbitrary fields. The system-level state, denoted $\bar{x}_t$, depends upon the information arrival process summarized by time t. The *system-level state space* is then a set $\bar{X} \subseteq \mathbb{R}^m$, so that $\forall t$, $\bar{x}_t \in \bar{X}$.

The *system state* x_t is the state of all agents and the system-level state, i.e.

$$x_t := (\hat{x}_{1,t}, \hat{x}_{2,t}, \dots, \hat{x}_{n,t}, \bar{x}_t) \in X := \prod_{a=1}^{n} \hat{X}_a \times \bar{X} \subseteq \mathbb{R}^{nk} \times \mathbb{R}^m.$$

We refer to X as the system or *global state space.*

The role of the system is to provide an 'institutional framework' within which agents interact, both with each other via their network interaction, and with the system itself as the propagator of network interaction. By 'interaction' we mean that, conditional upon a system state x_t at time t, an agent a may select from a menu of *actions*, representing valid (or "legal") actions that are admissible to the network. For simplicity, we suppose that this menu of actions is represented by a mapping $U(x_t, a)$, which is assumed to return a non-empty set at every point in time and for all agents.[3]

[3] It may be the case that, for agent a,

$$U(x_t, a) \equiv U(\hat{x}_{a,t}, a),$$

If an agent acts at time t, they select an action u_t from $U(x_t, a)$. In Section 3 it will be assumed that every agent possesses a *decision rule* incorporating available information and the admissible set of actions $U(x_t, a)$ at time t. The action u_t does not refer to a particular agent because of the way time operates as a 'lattice' for agent decisions.[4] Formally, we require

Assumption 1 *Time is sufficiently finely granulated to ensure that action collisions do not occur, i.e. all agent actions are ordered by t.*

Assumption 1 means that for every t there is one and only one corresponding action u_t, selected by an agent a from $U(x_t, a)$. While this is a formal requirement for what follows, in practice this assumption is reasonable because the system is a state machine—hence events within the system arrive in discrete steps.[5]

The agent's selection of an action u_t *changes the system state* x_t. The system thus possesses a *mechanism* that, taking an action, transitions the system state (i.e. agent level state and the system level state) to the next lattice point $t + 1$. More formally, such a mechanism may be viewed as a transition function f, with $x_{t+1} = f(x_t, u_t)$. Note that f takes values over all possible agent actions $u_t \in U(x_t, a)$, for every possible x_t and for every a—this may be a consequence of a conservation law, which is discussed further in Section 2.2 below.

Definition 1. *By a **discrete event game** is meant the tuple $(\bar{X}, \{\hat{X}_a\}_{a=1}^{n}, U, f)$, comprised of the agent and system level state sets, the decision mapping for agent actions,[6] and the system state transition function f.*

A *trajectory* is a sequence of system states $\{x_t\}$ created by the repeated selections of actions by agents, in response to the system state (or their private agent state, if the system state is not fully visible to every agent). Without further restrictions it is clear that there are infinitely many possible trajectory realizations *ex ante*, depending upon the richness of the sets underlying the discrete event game. In what follows, we demonstrate that it is possible (and usually desired) for the designer of the system to impose additional structure that will restrict possible trajectory realizations to spaces (such as topological manifolds) that reduce the complexity of the game's resulting dynamical evolution.

i.e. the agent's state space is sufficient to define their actions (this would be the case, for example, in a game where every agent has their own action set, or where every agent conditions only upon their own private information). In what follows we allow for full conditioning on x_t.

[4] Naturally there may still be an implicit unique mapping from u_t to a, as is the case with sending Bitcoin, [31].

[5] While ostensibly this model assumes a strict ordering of actions, this is a consequence of the definition of x_t as a global state and f as a global mechanism. Partial orderings may suffice provided a local state transition depends only upon information provided in the local state; see e.g. [14]

[6] The decision mapping defines an agent's *strategy set*, which is a standard primitive defining a game; see e.g. [7].

2.2 Configuration Spaces

In the system design process one or more quantities of interest are usually *conserved*, i.e. are time-invariant over every possible global system state trajectory. A simple example is a discrete event game in which at time t a finite resource, Y_t, must be allocated across agents. If we suppose that an agent a's local state at t is their allocation of this resource, then resource conservation implies

$$\sum_{a=1}^{n} \hat{x}_{a,t} \equiv Y_t \; \forall t. \tag{1}$$

This is a *restriction* on the attainable combinations of individual agent resources that must respect the allocation restriction.

Over time the relative allocation between agents may change, so that for some or all agents, $\hat{x}_{a,t} \neq \hat{x}_{a,t+1}$. But restriction (1) nevertheless holds at every t. In addition, there may be flows into or out of the system that cause Y_t to change, where the change ΔY_t is allocated to one or more agents. Such flows are common constructs in network routing [29], crypto token allocation (such as within the original Bitcoin protocol, see [19]) and crypto mining games (e.g. [24]), where in the latter case there are conserved flows between agents but injections of new token supply into the system according to predefined monetary policies.

The key implication of resource conservation, such as (1) above, is that it reduces the dimension of allowable system trajectories—generally, there is a reduction of one dimension for each (independent) restriction. The system designer may thus focus upon a smaller space for realizations of the trajectory, called the *configuration space* (see e.g. [30] for an introduction to configuration spaces).

The quantity (or quantities) conserved throughout the dynamical evolution of the system can be expressed by designing *conservation laws*, i.e. real-valued[7] functional (linear or non-linear) relationships $V : X \to \mathbb{R}$ where the quantity to be conserved, such as a global state x_t, satisfies $V(x_t) \equiv \bar{V} \in \mathbb{R} \; \forall t$.

A conservation law so designed may be *enforced* by ensuring that the global state transition mechanism over admissible action sets $U(x_t, a)$, $\forall a$, respects

$$\begin{aligned} &V(x_0) = \bar{V}, \\ &V(x_t) = V(f(x_t; u_t)) \; \forall t, \; \forall u_t. \end{aligned}$$

Selecting a pair (V, f) to implement desired conservation laws is the *mechanism design* problem facing the system designer. It depends crucially upon which conserved quantities are present, as well as upon the requirements defining the kinds of actions agents expect to be able to take. If f characterized a set of actions such as sending cryptocurrency, and V encoded a desired invariant such as conserving that cryptocurrency, then one could derive the necessary admissible function U

[7] Although we focus upon real-valued laws here because of the estimation of continuous real-valued signals, in general finite or even infinite state machines may also characterize conservation laws.

by restricting the domain of f to the preimage of the invariant set. This results in the rule that agent a cannot send tokens exceeding its available balance.

A more general mechanism design problem characterizes the goal of the system using *performance metrics*, which tell the designer—and the participants in the system—which states (or functions of states) are considered desirable. When the game is allocating resources it may do so to the benefit of those agents that move the system state in a direction which improves relative to a performance metric. Framed as such, the network itself may be viewed as an evolutionary optimization algorithm where the agents' local efforts to maximize payouts serve to ascend a potential field characterized by the aforementioned performance metrics.

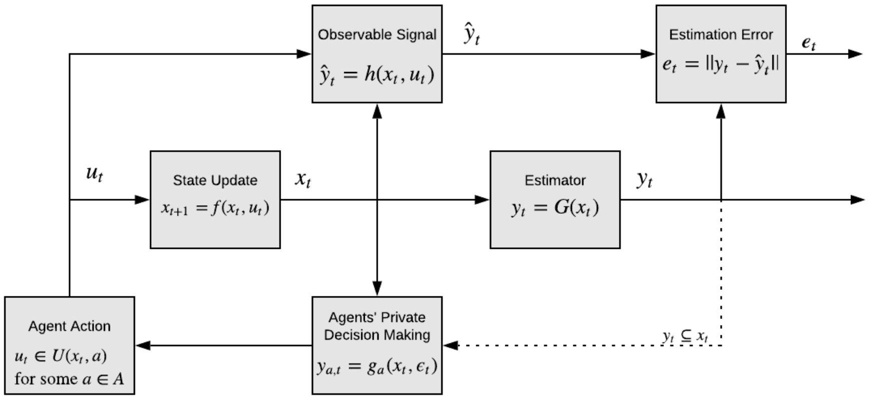


Fig. 1: Block Diagram for a discrete event game with estimation capabilities

2.3 Samples, Signals, and Estimation

In addition to conservation laws, there may be desirable quantities of interest that are generated by the system as a result of agent actions $\{u_t\}$. For example, the buy or sell decisions made according to a token bonding curve will determine a realized price, $\hat{P}_t$. This price is subject to noise from exogenous factors (e.g. market conditions, off-chain supply and demand shocks, etc.), denoted by some process ε_t, and so will be difficult to estimate. In general, we let $\hat{y}_t = h(x_t, u_t)$ represent the realized value, which is a noisy *signal* of the variable of interest. The map h is the *measurement technology*, and may be interpreted as a sensor. See Figure 1 for a block diagram representation of the system.

By implementing a pair (V, f), the system designer creates an *estimate* y_t from noisy samples $\hat{y}_t$, by forcing the trajectory of realized states to admit a mapping G carrying x_t to the estimate y_t. For example, a token bonding curve restricts token demand and supply and in turn generates an estimate P_t of a hidden signal, for which realized price $\hat{P}_t$ is a noisy sample. In general, the sequence of samples $\{\hat{y}_t\}$ depends upon individual agent mappings, which attempt to condition upon ε_t and the system state x_t. This allows the *local representation* of y_t, denoted by $y_{a,t}$, to be expressed in the form $g_a(x_t, \varepsilon_t)$. We interpret the mapping $g_a(x_t, \varepsilon_t)$ as the *representation technology* of the agent. The local representation $y_{a,t}$ may then be used by agent a in decision-making, i.e. there may be a further mapping carrying x_t, $y_{a,t}$ and ε_t (if separately influencing decisions apart from g_a) into a decision u_t from $U(x_t, a)$. Such a mapping, although not articulated in detail here, would be a *decision rule* for the agent.

There may be potential feedback between an agent's action, u_t, and the factor influencing the signal $\hat{y}_t$, i.e. it may be that

$$\frac{\partial \varepsilon_t}{\partial u_t} \neq 0. \tag{2}$$

This is the case for the token bonding curve example presented in Section 3: the signal is the actual realized token price, $\hat{P}_t$, which is determined by buy and sell decisions of the agents, while the spot price P_t is determined from the conservation law V and the same agent trading decisions.

The first-order impact of the agent's action on the noise process ε_t given in (2) implies that an observation of signal $\hat{y}_t$ at t represents a 'draw' or sample from the underlying distribution of local agent representations $y_{a,t}$. In essence, the estimate y_t is the system's 'best guess' of the signal $\hat{y}_t$, accounting for variations in both space (agents) and time. A performance metric that naturally suggests itself is that of an *error* e_t such that

$$e_t = e(y_t, \hat{y}_t) := \|y_t - \hat{y}_t\| \; \forall t.$$

The error specification allows an application of estimation theory to the analysis of the system's stochastic convergence, by 'steering' the system to achieve as low an error as possible. Future research will also focus upon boundary conditions for the error for a variety of signal and estimate specifications.

In practice, the more detailed the foundations of the discrete event game, and its designed implementation of (V, f), the easier it will be to arrive at results with formal bounds in a chosen error metric. The cost, naturally, is the risk that violations of these more detailed modeling assumptions have the potential to undermine conclusions drawn in this fashion. Our analysis in this sense is simply a 'launching point' for a richer modeling paradigm crossing dynamic mechanism design theory with estimation theory (see e.g. [13]).

3 Bonding Curves as Price Estimators

The token bonding curve system, in which a *community token* is managed using bonding curve contracts, fits readily within the framework outlined in Section 2. The system specifies a series of token holdings as outlined in [30], which we summarize here.

Definition 2. *The* ***reserve*** $R_t \in \mathbb{R}_{++}$ *at time* t *is the total quantity of reserve currency tokens bonded to the bonding curve contract.*

The reserve currency is provided by a contract external to the community deploying the bonding curve. This could be the native cryptocurrency or tokenized fiat, such as a stablecoin. At time t each agent a possesses their own holding of the reserve currency, denoted by $r_{a,t} > 0$.

Definition 3. *The* ***supply*** $S_t \in \mathbb{R}_{++}$ *at time* t *is the total quantity of community tokens issued by the bonding curve contract.*

The supply is the total quantity of the community token held by agents. An individual holding $s_{a,t}$ of the supply is part of the local state of agent a at t.

A bonding curve may be characterized using the mechanism design process (V, f)—in particular, it provides the mechanism f and the mappings U guaranteed to preserve V under f. Agents can adjust their token holdings by depositing the reserve currency to mint new community tokens, or burn all or part of their community token holdings to withdraw the reserve currency. Regardless, the supply S_t and reserve R_t always satisfy $V(R_t, S_t) = \text{constant}$. Furthermore, the system's estimate of the token price $P_t = P(R_t, S_t)$ is part of the state.

Definition 4. *The* ***spot price*** $P_t \in \mathbb{R}_{++}$ *at time* t *is the estimate of the value of the community token, in units of* R *per units of* S*.*

Since agents can freely adjust their community token holdings via the bonding curve, the spot price P_t may be interpreted as a dynamic estimate of the value imbued in the token by agents with representation technology $g_a(x_t, \varepsilon)$. The justification for this claim is further borne out by the characterization of the configuration space in Section 3.1. Note that each agent may hold their own (private and potentially exogenous) estimate of the value of the community token, denoted $p_{a,t} = g_a(x_t, \varepsilon)$—this will be discussed further in Section 3.3.

Definition 5. *The* ***system-level state*** *is* $\bar{x}_t := (R_t, S_t, P_t) \in \bar{X} \subset \mathbb{R}^3_{++}$.

We shall see shortly why $\bar{X}$ is a proper subset of $\mathbb{R}^3_{++}$ once token bonding curve and supply conservation laws are taken into consideration.

Each element of the system-level state has an agent-level state counterpart, based upon the agent community token and reserve holdings.

Definition 6. *The* ***agent-level state*** *is* $\hat{x}_{a,t} := (r_{a,t}, s_{a,t}, p_{a,t}) \in \hat{X}_a \subseteq \mathbb{R}^3_{++}$.

In what follows we suppose (although it is not strictly required) that agents can observe the system-level state, but not each other's agent-level states. The system state x_t exists for all t even though it is not globally observable.

Definition 7. *The* ***system state*** *is* $x_t = (\hat{x}_{1,t}, \ldots \hat{x}_{n,t}, \bar{x}_t) \in X \subset \mathbb{R}^{3(n+1)}_{++}$ *and lies in the Cartesian product of the system-level state and the agent-level state.*

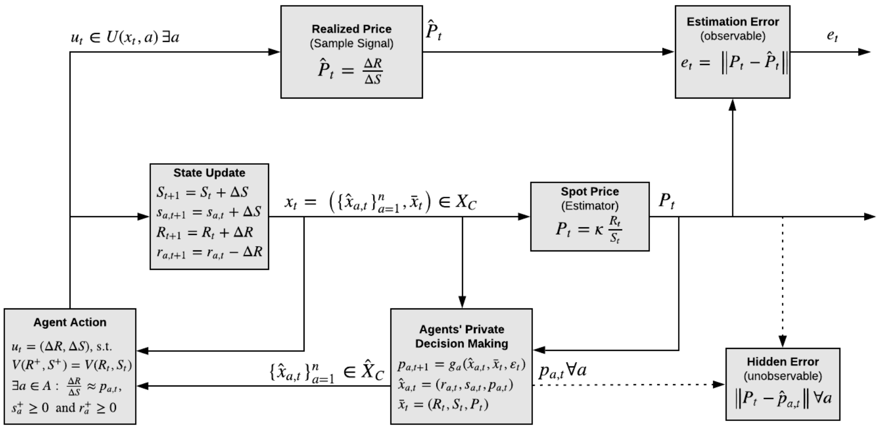

$u_t \in U(x_t, a)\, \exists a$
Realized Price
(Sample Signal)
$\hat{P}_t = \frac{\Delta R}{\Delta S}$
$\hat{P}_t$
Estimation Error
(observable)
$e_t = \|P_t - \hat{P}_t\|$
e_t
State Update
$S_{t+1} = S_t + \Delta S$
$s_{a,t+1} = s_{a,t} + \Delta S$
$R_{t+1} = R_t + \Delta R$
$r_{a,t+1} = r_{a,t} - \Delta R$
$x_t = \left(\{\hat{x}_{a,t}\}_{a=1}^n, \bar{x}_t\right) \in X_C$
Spot Price
(Estimator)
$P_t = \kappa \frac{R_t}{S_t}$
P_t
Agent Action
$u_t = (\Delta R, \Delta S)$, s.t.
$V(R^+, S^+) = V(R_t, S_t)$
$\exists a \in A : \frac{\Delta R}{\Delta S} \approx p_{a,t}$,
$s_a^+ \geq 0$ and $r_a^+ \geq 0$
$\{\hat{x}_{a,t}\}_{a=1}^n \in \hat{X}_C$
Agents' Private
Decision Making
$p_{a,t+1} = g_a(\hat{x}_{a,t}, \bar{x}_t, e_t)$
$\hat{x}_{a,t} = (r_{a,t}, s_{a,t}, p_{a,t})$
$\bar{x}_t = (R_t, S_t, P_t)$
$p_{a,t} \forall a$
Hidden Error
(unobservable)
$\|P_t - \hat{p}_{a,t}\| \forall a$

Fig. 2: Block Diagram for a discrete event game, matching an agent-based model on a bonding curve, including observable and unobservable estimation errors.

3.1 The Configuration Space

The system design incorporates both a mechanism f, which is the token bonding curve mechanism above, and a set of conservation laws V, indicating the design goals that can be formulated as time-invariant quantities of interest. In the case of the bonding curve, a *system level design goal* is to establish diminishing returns for both depositing and withdrawing reserve currency from the bonding curve. This is accomplished by restricting the relationship between R and S:

Definition 8. *The bonding curve* ***conservation function*** *is given by*

$$V(R_t, S_t) := \frac{S_t^\kappa}{R_t} \equiv V_0, \tag{3}$$

where $V_0 = V(R_0, S_0) := \frac{S_0^\kappa}{R_0}$ is a constant defined by initial supply S_0 and initial reserve R_0. Parameter κ is the curvature of the bonding curve.

Given Definition 8, Proposition 1 of [30] allows us to assert that the spot price P_t is completely determined by reserve currency and community token supply holdings and the functional form of V:

$$P_t = P(R_t, S_t) = \left. -\frac{\partial V / \partial S}{\partial V / \partial R} \right|_{(R_t, S_t)}. \tag{4}$$

Definition 9. *The* ***system-level configuration space****, $\bar{\mathbf{X}}_{\mathbf{C}}$ is a 1-manifold, created by applying two one-dimensional restrictions, $V(R_t, S_t) = V_0$ and $P_t = P(R_t, S_t)$ to the three-dimensional state space $\bar{X}$:*

$$\bar{\mathbf{X}}_{\mathbf{C}} := \{\bar{x} = (R_t, S_t, P_t) \in \bar{X} \mid V(R_t, S_t) = V_0,\, P_t = P(R_t, S_t)\} \subset \bar{X}. \tag{5}$$

In addition to the system level design goal, there is also a local conservation restriction. For the community token supply, the total agent holdings at time t cannot exceed the available supply. Letting s_t denote the vector of community tokens held by all agents $(s_{1,t}, \ldots, s_{n,t})$, we have

$$V_S(s_t, S_t) := \sum_{a=1}^{n} s_{a,t} - S_t \equiv 0. \tag{6}$$

Definition 10. *The* ***agent-level configuration space****,* $\hat{\mathbf{X}}_{\mathbf{C}}$ *is a* $(3n-1)$*-manifold, created by enforcing the conservation constraint* $V_S(s_t, S_t) = 0$ *on the* $3n$*-dimensional agent-level state space* $\prod_a \hat{X}_a$*:*

$$\hat{\mathbf{X}}_{\mathbf{C}} := \{\{(r_{a,t}, s_{a_t}, p_{a,t}) \in X_a\}_{a=1}^{n} \mid \sum_{a=1}^{n} s_{a,t} = S_t\} \subset \hat{X}. \tag{7}$$

Note that there is also an inherent asymmetry between the reserve currency and the community token. Community tokens cannot be introduced or removed without doing so through the bonding curve, meaning that community tokens are *internal* to the system. By contrast, the reserve currency can be introduced to or removed from the system without recourse to an internal mechanism—although the reserve currency is assumed to be globally conserved (when considering its holdings outside of the system), it is not locally conserved and is thus *external* to the system. The bonding curve then takes the role of *interface* between the two value systems, with one broader in scope (where the reserve currency originates) and one narrower in scope (where the specialized community token is used).

Definition 11. *The* ***configuration space****,* $\mathbf{X}_{\mathbf{C}}$ *is a* $3n$*-manifold, which is the Cartesian product of the system-level and agent-level configuration spaces.*

$$\mathbf{X}_{\mathbf{C}} := \hat{\mathbf{X}}_{\mathbf{C}} \times \bar{\mathbf{X}}_{\mathbf{C}} \subset \mathbf{X} = \mathbb{R}_{++}^{3n+3} \tag{8}$$

Observe that the configuration space effectively pastes together multiple configuration spaces. In this example, the agent-level and system-level configurations are combined, but this method is more broadly applicable for combining systems in a predictable way. For example, in [30] the bonding curve is augmented with a funding pool as part of the system-level state. While this introduces additional complexity, it is managed through combining the systems in a manner which provably preserves the desired invariant properties.

3.2 Mechanisms

In order to arrive at the laws of motion for the system, it is necessary to characterize the specific bonding curve mechanisms for reserve currency and community token dynamics. In addition, we include a specification of admissible agent actions, to close the feedback mechanism between these actions and the resulting realized signal process. We continue to use [30] as our reference in what follows.

Definition 12. *The **Bond-to-Mint** mechanism takes a system-level state $\bar{x}_t = (R_t, S_t, P_t)$ and an agent a's action, given by a bonded quantity $\Delta R_t := r_{a,t} - r_{a,t+1} \geq 0$ such that $r_{a,t+1} \in \mathbb{R}_{++}$. Quantity ΔR_t is reserve currency transferred to the bonding curve, and returns the state x_{t+1} such that*

$$(R_{t+1}, S_{t+1}, P_{t+1}) = \left(R_t + \Delta R_t,\ \sqrt[\kappa]{V_0(R_t + \Delta R_t)}, \frac{\kappa(R_t + \Delta R_t)}{\sqrt[\kappa]{V_0(R_t + \Delta R_t)}} \right)$$

and the associated updates to the agent-level state $\hat{x}_{a,t}$ are given by,

$$(r_{a,t+1}, s_{a,t+1}, p_{a,t+1}) = \left(r_t - \Delta R_t, s_{a,t} + \sqrt[\kappa]{V_0(R_t + \Delta R_t)}, g_a(x_{t+1}, \varepsilon_t) \right)$$

and $(r_{a',t+1}, s_{a',t+1}, p_{a',t+1}) = (r_{a',t}, s_{a',t}, g_{a'}(x_{t+1}, \varepsilon_t)$ for all agents $a' \neq a$ where g_a is a private mapping for agent a and ε_t is an exogenous signal.

Definition 13. *The **Burn-to-Withdraw** mechanism takes a system-level state $\bar{x}_t = (R_t, S_t, P_t)$ and an agent a's action, given by a burned quantity $\Delta S_t := s_{a,t+1} - s_{a,t} \leq 0$ such that $s_{a,t+1} \in \mathbb{R}_{++}$. Quantity ΔS_t is token supply removed from the system, and results in the state x_{t+1} such that*

$$(R_{t+1}, S_{t+1}, P_{t+1}) = \left(\frac{(S_t + \Delta S_t)^\kappa}{V_0}, S_t + \Delta S_t, \frac{\kappa(S_t + \Delta S_t)^\kappa}{V_0(S_t + \Delta S_t)} \right)$$

and the associated updates to the agent-level state $\hat{x}_{a,t}$ are given by,

$$(r_{a,t+1}, s_{a,t+1}, p_{a,t+1}) = \left(r_{a,t} + R_t - \frac{(S_t + \Delta S_t)^\kappa}{V_0}, s_{a,t} + \Delta S_t, g_a(x_{t+1}, \varepsilon_t) \right)$$

and $(r_{a',t+1}, s_{a',t+1}, p_{a',t+1}) = (r_{a',t}, s_{a',t}, g_{a'}(x_{t+1}, \varepsilon_t)$ for all agents $a' \neq a$ where g_a is a private mapping for agent a and ε_t is an exogenous signal.

Given these mechanisms, an agent's action set $U(x_t, a)$ can be fully determined from both agent-level restrictions and system-level conservation laws:

Definition 14. *An action $u_t := (\Delta R_t, \Delta S_t)$ is **admissible** if at time t:*

$$u_t \in U(x_t, a) := \hat{U}(x_t, a) \cap \bar{U}(x_t)\ \forall t,$$

for agent a, where the agent-level admissibility condition is:

$$\hat{U}(x_t, a) = \{(\Delta R_t, \Delta S_t) \in \mathbb{R}^2 \mid (r_{a,t} - \Delta R_t, s_{a,t} + \Delta S_t) \in \mathbb{R}^2_{++}\}$$

and the the system-level admissibility condition is

$$\begin{aligned}\bar{U}(x_t) = \{(\Delta R_t, \Delta S_t) \in \mathbb{R}^2 \mid (R_t + \Delta R_t, S_t + \Delta S_t) \in \mathbb{R}^2_{++},\\ V(R_t + \Delta R_t, S_t + \Delta S_t) = V(R_t, S_t)\}.\end{aligned}$$

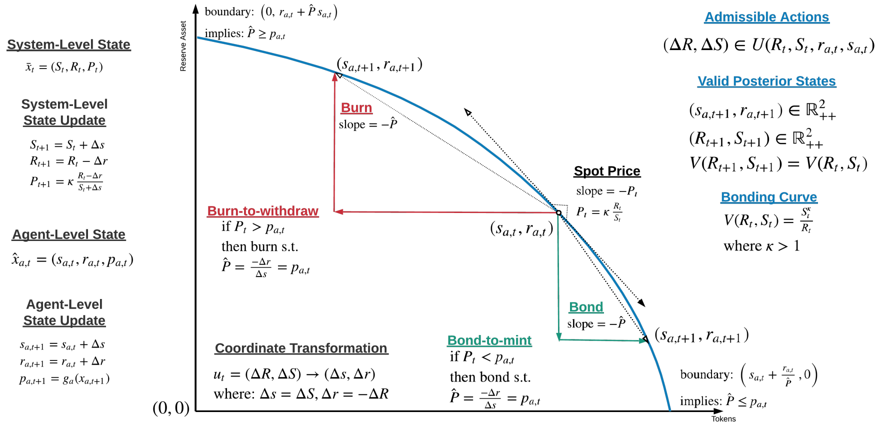

System-Level State
$\bar{x}_t = (S_t, R_t, P_t)$
System-Level State Update
$S_{t+1} = S_t + \Delta s$
$R_{t+1} = R_t - \Delta r$
$P_{t+1} = \kappa \frac{R_t - \Delta r}{S_t + \Delta s}$
Agent-Level State
$\hat{x}_{a,t} = (s_{a,t}, r_{a,t}, p_{a,t})$
Agent-Level State Update
$s_{a,t+1} = s_{a,t} + \Delta s$
$r_{a,t+1} = r_{a,t} + \Delta r$
$p_{a,t+1} = g_a(x_{a,t+1})$
Reserve Asset
boundary: $\left(0,\ r_{a,t} + \hat{P}\, s_{a,t}\right)$
implies: $\hat{P} \geq p_{a,t}$
$(s_{a,t+1}, r_{a,t+1})$
Burn
slope $= -\hat{P}$
Burn-to-withdraw
if $P_t > p_{a,t}$
then burn s.t.
$\hat{P} = \frac{-\Delta r}{\Delta s} = p_{a,t}$
$(s_{a,t}, r_{a,t})$
Spot Price
slope $= -P_t$
$P_t = \kappa \frac{R_t}{S_t}$
Bond
slope $= -\hat{P}$
Bond-to-mint
if $P_t < p_{a,t}$
then bond s.t.
$\hat{P} = \frac{-\Delta r}{\Delta s} = p_{a,t}$
$(s_{a,t+1}, r_{a,t+1})$
Coordinate Transformation
$u_t = (\Delta R, \Delta S) \rightarrow (\Delta s, \Delta r)$
where: $\Delta s = \Delta S, \Delta r = -\Delta R$
$(0, 0)$
boundary: $\left(s_{a,t} + \frac{r_{a,t}}{\hat{P}}, 0\right)$
implies: $\hat{P} \leq p_{a,t}$
Tokens
Admissible Actions
$(\Delta R, \Delta S) \in U(R_t, S_t, r_{a,t}, s_{a,t})$
Valid Posterior States
$(s_{a,t+1}, r_{a,t+1}) \in \mathbb{R}^2_{++}$
$(R_{t+1}, S_{t+1}) \in \mathbb{R}^2_{++}$
$V(R_{t+1}, S_{t+1}) = V(R_t, S_t)$
Bonding Curve
$V(R_t, S_t) = \frac{S_t^\kappa}{R_t}$
where $\kappa > 1$

Fig. 3: An agent's view of their admissible actions given by the bonding curve.

3.3 Price Estimation

The bonding curve system is presented with a sequence of observations associated with the actions $u_t = (\varDelta R, \varDelta S)$, from which sample prices $\hat{P}_t = \frac{\varDelta R}{\varDelta S}$, referred to as *realized prices*, are computed. Due to the restrictions $U(x_t, a)$ on admissible actions, $\hat{P}_t$ is not necessarily a true sample of $p_{a,t} = g_a(x_t, \varepsilon_t)$ for the active agent a. Assuming agent a is acting at time t and that the mapping g_a accounts for any private signals and utility functions (include discounting), the sample price $\hat{P}_t$ may be interpreted as arising from an agent level constrained optimization, for example:

$$u_t = \operatorname*{arg\,min}_{(\varDelta R, \varDelta S) \in U(x_t, a)} \|\varDelta S\, g_{a,t}(x_t, \varepsilon_t) - \varDelta R\|. \tag{9}$$

Due to the dimensional restrictions in the configuration space, the admissible u_t for agent a lies within an open interval embedded in the plane (r_a, s_a), as shown in Figure 3. It suffices for agent a to search this interval for their preferred posterior state, and to choose $u_t = (r_{a,t} - r_{a,t+1}, s_{a,t+1} - s_{a,t})$ accordingly. Whether or not u_t is treated as a strategic action, as in (9), the curvature $\kappa > 1$, implies that every point in the open interval $U(x_t, a)$ is uniquely characterized by the price $\hat{P}_t = \frac{\varDelta R}{\varDelta S}$. Furthermore, the estimator $P_t = G(x_t) = \kappa \frac{R_t}{S_t}$ is a critical point where $\hat{P}_t > P_t$ will always call for burning, and $\hat{P}_t < P_t$ will always call for bonding (see [30], Lemmas 1 and 2). Also from [30], the posterior spot price always decreases for burn actions, and always increases for bond actions. Thus it is guaranteed that the update directions match, that is $(P_{t+1} - P_t)(\hat{P}_t - P_t) \geq 0$ when any agent a takes an action u_t at time t.

Combining this machinery with the assumption that agents act directionally aligned with their preferences $(g_a(x_t, \varepsilon_t) - P_t)(\hat{P}_t - P_t) > 0$, the groundwork is laid for deriving estimation error bounds of the form $\|g_a(x_t, \varepsilon_t) - G(x_t)\| = \|p_{a,t} - P_t\| \leq \xi \|\hat{P}_t - P_t\|\ \forall a \forall t$, with minimal assumptions regarding the agents.

As a first step we proceed next to computational experiments, which test the input-output response dynamics of the bonding curve system; in particular we compare the estimator P_t to sample sequences $\hat{P}_t$ and the associated (unique) actions $u_t \in \bar{U}(x_t)$. Estimation error is given by $e_t = \|\hat{P}_t - P_t\| \forall t$.

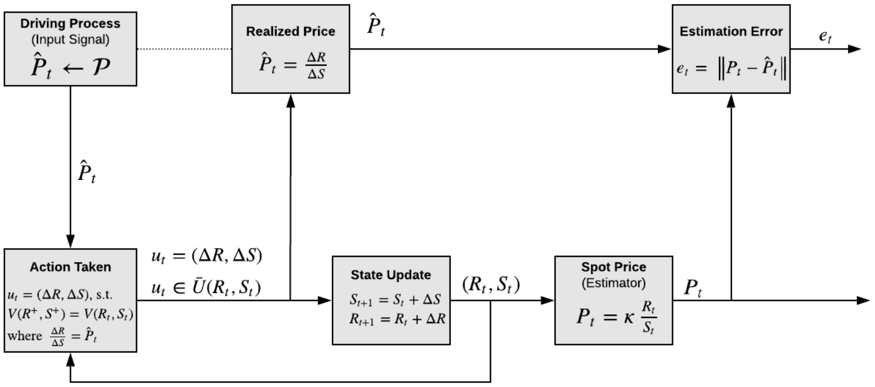

Driving Process
(Input Signal)
$\hat{P}_t \leftarrow \mathcal{P}$
Realized Price
$\hat{P}_t = \frac{\Delta R}{\Delta S}$
$\hat{P}_t$
Estimation Error
$e_t = \|P_t - \hat{P}_t\|$
e_t
$\hat{P}_t$
Action Taken
$u_t = (\Delta R, \Delta S)$, s.t.
$V(R^+, S^+) = V(R_t, S_t)$
where $\frac{\Delta R}{\Delta S} = \hat{P}_t$
$u_t = (\Delta R, \Delta S)$
$u_t \in \bar{U}(R_t, S_t)$
State Update
$S_{t+1} = S_t + \Delta S$
$R_{t+1} = R_t + \Delta R$
(R_t, S_t)
Spot Price
(Estimator)
$P_t = \kappa \frac{R_t}{S_t}$
P_t

Fig. 4: Block Diagram representation of the input-output response dynamics for the bonding curve, viewed as an price estimator.

4 Price Estimator Response Dynamics

Consider a sequence of realized prices $\hat{P}_t$; for any prior system-level state $\bar{x}_t$, there is a unique $u_t = (\Delta R_t, \Delta S_t)$ satisfying $\Delta S_t \hat{P}_t = \Delta R_t$ provided that $\hat{P}_t > R_t/S_t$. The price R_t/S_t, also called the *floor price*, is the ratio of reserve-to-supply, and represents the realized price of liquidating the bonding curve at any time t. It thus acts as a lower bound for all realized prices $\hat{P}_t$. By restricting attention to the sequence $\hat{P}_t$, it is possible to analyze the signal processing properties of the bonding curve using only the system-level trajectory $\bar{x}_t$.

Definition 15. *The* ***driving process*** $\mathcal{P}$ *generates a sequence of price samples* $\hat{P}_t$ *satisfying the condition that* $\hat{P}_t > \frac{R_t}{S_t}$ *for all* t.

The input-output response dynamics of the bonding curve system are constructed by comparing the inputs $\hat{P}_t$ to the outputs P_t for deterministic waveforms as well as non-deterministic input signals $\hat{P}_t$, given a particular characterizations of the bonding curve discrete event game. Figure 4 shows the block diagram for the experimental apparatus to follow.

4.1 Experimental Apparatus

To illustrate the impact on P_t of different driving processes, three deterministic signals are defined: a Square-Wave, a Triangle-Wave and a Sine-Wave. To capture stochastic effects, a Martingale stochastic process is also introduced. These simple signals abstract away from the complexity of possible inputs $\hat{P}_t$, while simultaneously acting as a starting point for the analysis of a broad range of feedback mechanisms caused by modeling agent behavior as endogenous as in agent-based modeling [2]. The deterministic signals can be characterized by wavelength λ, amplitude A and phase ϕ, for time $t \in \{0, \dots 4000\}$, and are described using the functional forms in Table 1.

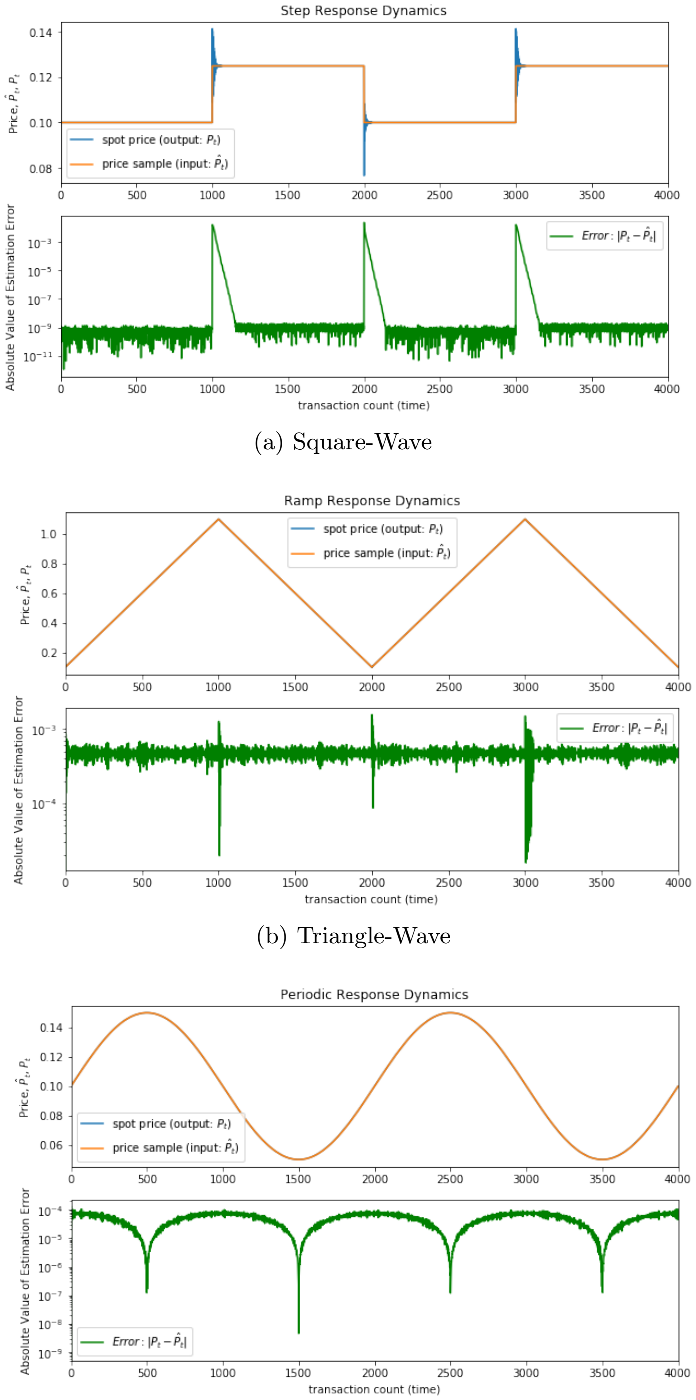
Waveform	**Driving Process** $\mathcal{P}$	**Restriction**
Square-Wave	$\hat{P}_t(t, \lambda, A, \phi) = B + A\,\mathbb{1}_{\{((t-\phi) \mod \lambda)<\lambda/2\}}$	$B > A$
Triangle-Wave	$\hat{P}_t(t, \lambda, A) \quad = B + \frac{2A}{\lambda}\left|((t-\phi) \mod \lambda) - \frac{\lambda}{2}\right| - \frac{2A}{4}$	$B > \frac{2A}{\lambda}$
Sine-Wave	$\hat{P}_t(t, \lambda, A, \phi) = B + A \sin\left(\frac{2\pi t - \phi}{\lambda}\right)$	$B > A$
Random Walk	$\hat{P}_t(t, \mu, \sigma) \quad = (1 + \delta_t)\hat{P}_{t-1}$ where $\delta_t \sim N(\mu, \sigma)$	$\mu = 0$

Table 1: Driving process functional forms for numerical experiments

Our experiments use the bonding curvature parameter $\kappa = 2$ and the system is initialized with a community token supply $S_0 = 1000000$ and reserve currency units $R_0 = 50000$, resulting in an initial price $P_0 = 0.10$ reserve units per token and an invariant $\bar{V} = V_0 = 20000000$. The deterministic driving functions are taken with $\phi = 0$, $B = P_0$ and $\lambda = 2000$. Amplitude A takes values $\frac{P_0}{2}$, $\frac{P_0}{100}$, and $\frac{P_0}{2}$ for the Square-Wave, Triangle-Wave and Sine-Wave, respectively. For the Random Walk, an initial condition $\hat{P}_0 = P_0$ is applied, and the percent change in $\hat{P}_0$ is drawn from a Gaussian distribution with mean $\mu = 0$ and variance $\sigma = 0.05$. Additionally, a 10-run Monte Carlo experiment was executed on the Martingale case for each $\sigma \in \{0.1/2^K | K = 1, \dots 10\}$, totaling 100 experiments.

4.2 Numerical Results

The Square-Wave response in Figure (5a) shows that the step response is tightly tuned, resulting in a large overshoot but remaining stable, oscillating and converging quickly. This behavior is characteristic of a high gain proportional controller. The Triangle-Wave (5b) and Sine-Wave (5c) signals are equally reminiscent of such a controller; the Triangle-Wave exhibits steady state error during the ramp and the Sine-Wave tracks most closely at the peaks and troughs. In the Random Walk case (6a) tracking behavior is observed, but the error radius appears large. Despite the high frequency noise, the error does not appear to accumulate, and does appear to remain within a ball roughly on the order of 3σ, likely an artifact of the random walk whereby $\frac{\Delta\hat{P}_t}{\hat{P}_t} \sim N(0, \sigma)$ for all t. This research suggests that other systems might be designed to exhibit properties of alternative estimators such as a Kalman filter, [20].

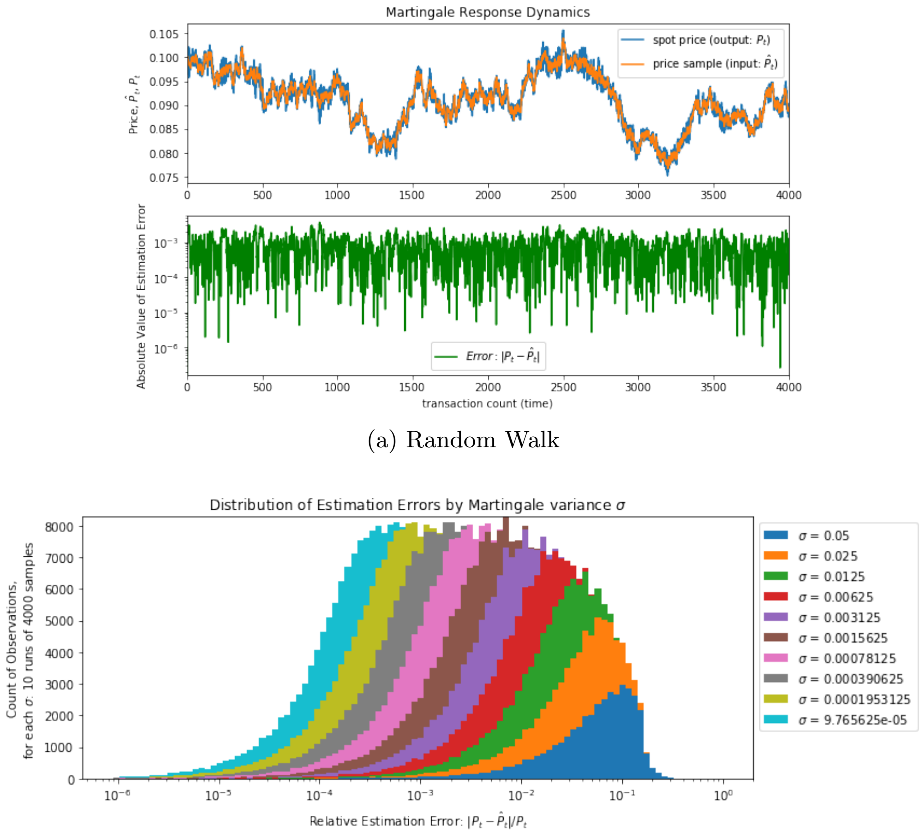


(a) Square-Wave

(b) Triangle-Wave

(c) Sine-Wave

Fig. 5: Summary of response dynamics and estimation error for the discrete case experiments in Section 4.1. Single trajectories are shown for (a) to (c).



(a) Random Walk

(b) Repeated Martingale Param Sweep

Fig. 6: Summary of response dynamics and estimation error for the stochastic case experiments in Section 4.1. Figure (b) shows the distribution of relative estimation errors for various σ-variance Martingales (a).

5 Conclusions & Future Work

The novel equipment provided by dynamic games is the system-level state, and the ability to define and enforce a configuration space that is a proper subset of the state space using mechanism design techniques related to state feedback control. Applying estimation and control-theoretic principles to token bonding curves, analytical groundwork was developed to characterize the relationship between realized prices and spot prices, as well as to posit an estimation bound relating the spot price and the agents' hidden preferences. Numerical experiments demonstrated the signal processing characteristics of the bonding curve, and provide further evidence that expanding the discrete event game machinery and its associated estimation capabilities will facilitate new tools for practical mechanism design, with a focus on cryptoeconomic and cyber-physical systems.

Our ongoing research focuses on the introduction of strategic agents with a variety of assumptions. To align with the canonical notion of a 'rational' agent, one considers solving for a stochastic optimal controller relative to the agent's private objectives and signals, [3]. It is equally reasonable to apply boundedly-rational strategies, as well as to apply various heuristics derived from decision theory, behavioral economics or simply inferred from data. A strength of our framework is the avoidance of 'baking' a model of decision-making into the game framework itself.

In addition to mechanism design, discrete event games may be used for decision policy identification. Provided with a basis for parameterized agent strategies it is plausible to infer the most likely composition of strategies from observed game dynamics using methods from system identification, [16].

In pursuit of this synthesis of theoretical and data-driven models, the *complex adaptive dynamics computer aided design* framework **cadCAD** has been developed as an open source python package, [4]. Our research team uses cadCAD alongside scientific python packages such as numpy, scipy, scikit-learn and networkx. For further information please see the cadCAD documentation accompanying the package.

References

1. Abadi, J., Markus, B.: Blockchain economics (August 2019)
2. Arthur, W.B.: Out-of-equilibrium economics and agent-based modeling. Handbook of computational economics **2**, 1551–1564 (2006)
3. Bertsekas, D.P., Bertsekas, D.P., Bertsekas, D.P., Bertsekas, D.P.: Dynamic programming and optimal control, vol. 2. Athena scientific Belmont, MA (1995)
4. cadCAD.: An open-source python package that assists in the processes of designing, testing and validating complex systems through simulation, https://cadcad.org
5. Caginalp, C., Caginalp, G.: Cryptocurrency equilibria through game theoretic optimization (05 2018)
6. Fudenberg, D., Levine, D.: The Theory of Learning in Games. MIT press (1998)
7. Fudenberg, D., Tirole, J.: Game Theory. MIT Press, Cambridge, MA (1991)
8. Hart, S.: Adaptive heuristics (August 2005)

9. Hofbauer, J., Sandholm, W.: Stable games and their dynamics. Journal of Economic Theory **144**, 1665–1693 (07 2009)
10. Isaacs, R.: Differential games: a mathematical theory with applications to warfare and pursuit, control and optimization. Courier Corporation (1999)
11. Jahandideh, et al.: Hybrid rebeca: Modeling and analyzing of cyber-physical systems. In: Chamberlain, R., Taha, W., Törngren, M. (eds.) Cyber Physical Systems. Model-Based Design. pp. 3–27. Springer International Publishing (2019)
12. Jovanovic, B., Rosenthal, R.: Anonymous sequential games. Journal of Mathematical Economics **17**(1), 77–87 (1988)
13. Kay, S.M.: Fundamentals of Statistical Signal Processing: Estimation Theory. Prentice-Hall, Inc., USA (1993)
14. Kia, S.S., Van Scoy, B., Cortes, J., Freeman, R.A., Lynch, K.M., Martinez, S.: Tutorial on dynamic average consensus: The problem, its applications, and the algorithms. IEEE Control Systems Magazine **39**(3), 40–72 (2019)
15. Liu, Z., Luong, N.C., Wang, W., Niyato, D., Wang, P., Liang, Y.C., Kim, D.I.: A survey on applications of game theory in blockchain (2019)
16. Ljung, L.: System identification. Wiley Encyclopedia of Electrical and Electronics Engineering (2001)
17. Marden, J.R., Shamma, J.S.: Game theory and distributed control. In: Handbook of Game Theory with Economic Applications, vol. 4, pp. 861–899. Elsevier (2015)
18. Monderer, D., Shapley, L.: Potential games. Games and Economic Behavior **14**(1), 124–143 (1996)
19. Nakamoto, S.: Bitcoin: A peer-to-peer electronic cash system (Dec 2008)
20. Olfati-Saber, R.: Distributed kalman filtering for sensor networks. In: 2007 46th IEEE Conference on Decision and Control. pp. 5492–5498 (Dec 2007). https://doi.org/10.1109/CDC.2007.4434303
21. Rapoport, A., Chammah, A.M., Orwant, C.J.: Prisoner's dilemma: A study in conflict and cooperation, vol. 165. University of Michigan press (1965)
22. Sandholm, W.H.: Population games and evolutionary dynamics. MIT press (2010)
23. Shorish, J.: Blockchain state machine representation. SocArXiv (Janu 2018)
24. Singh, R., Dwivedi, A., Srivastava, G., Wiszniewska-Matyszkiel, A., Cheng, X.: A game theoretic analysis of resource mining in blockchain (11 2019)
25. Topkis, D.M.: Supermodularity and Complementarity. Princeton Press (1998)
26. Voshmgir, S., Zargham, M.: Foundations of cryptoeconomic systems. Working paper series, Research Institute for Cryptoeconomics, Vienna (November 2019)
27. Wang, X., Xiao, N., Xie, L., Frazzoli, E., Rus, D.: Discrete-time mean field games in multi-agent systems. In: 2014 13th International Conference on Control Automation Robotics Vision (ICARCV). pp. 711–716 (Dec 2014)
28. Young, H.P.: Strategic Learning and its Limits. Oxford University press (2005)
29. Zargham, M., Ribeiro, A., Ozdaglar, A., Jadbabaie, A.: Accelerated dual descent for network flow optimization. IEEE Transactions on Automatic Control **59**(4), 905–920 (2014)
30. Zargham, M., Shorish, J., Paruch, K.: From curved bonding to configuration spaces. Working paper series, Research Institute for Cryptoeconomics, Vienna (Dec 2019)
31. Zargham, M., Zhang, Z., Preciado, V.: A state-space modeling framework for engineering blockchain-enabled economic systems. In: NECSI ICCS. ArXiv.org (2018)
32. Zhang, Z., Zargham, M., Preciado, V.: On modeling blockchain-enabled economic networks as stochastic dynamical systems. Applied Network Science: Blockchain and Cryptocurrency [forthcoming] (2020)