A Reinforcement Learning Algorithm for Efficient Dynamic Trading Execution in the Presence of Signals

By

Daniel Elkind

B.A. Economics
Princeton University, 2013

SUBMITTED TO THE SLOAN SCHOOL OF MANAGEMENT IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE IN MANAGEMENT RESEARCH

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

SEPTEMBER 2019

Signature redacted

Signature of Author: ____________________________________
Department of Management
August 9, 2019

Signature redacted

Certified by: ____________________________________
Adrien Verdelhan
Associate Professor of Finance
Thesis Supervisor

Signature redacted

Accepted by: ____________________________________
Catherine Tucker
Sloan Distinguished Professor of Management Science
Faculty Chair, MIT Sloan PhD Program

A Reinforcement Learning Algorithm for Efficient Dynamic Trading Execution in the Presence of Signals

By

Daniel Elkind

Submitted to the Sloan School of Management on August 9, 2019 in Partial Fulfillment of the Requirements for the Degree of Master of Science in Management Research

ABSTRACT

This paper focuses the optimal trading execution problem, where a trader seeks to maximize the proceeds from trading a given quantity of shares of a financial asset over a fixed-duration trading period, considering that trading impacts the future trajectory of prices. I propose a reinforcement learning (RL) algorithm to solve this maximization problem. I prove that the algorithm converges to the optimal solution in a large class of settings and point out a useful duality between the Q-learning contraction and the dynamic programming PDE. Using simulations calibrated to historical exchange trading data, I show that (i) the algorithm reproduces the analytical solution for the case of random walk prices with a linear absolute price impact function and (ii) matches the output of classical dynamic programming methods for the case of geometric brownian motion prices with linear relative price impact. In the most relevant case, when a signal containing information about prices is introduced to the environment, traditional computational methods become intractable. My algorithm still finds the optimal execution policy, leading to a statistically and economically meaningful reduction in trading costs.

Thesis Supervisor: Adrien Verdelhan
Title: Associate Professor of Finance

I. Introduction

This paper contributes to a canonical literature on the dynamic execution problem, in which a trader attempts to maximize the proceeds from trading a known quantity (block) of shares of a financial asset over a fixed-duration trading period while subjected to potential market impact of her past trades on future prices of the asset. Though the problem dates back to the seminal papers of Bertsimas & Lo (1998) and Almgren & Chriss (2001), relatively little attention has been paid to the potential for modern machine learning approaches - in particular, reinforcement learning algorithms - to improve the state of the art in this setting.

Reinforcement learning (RL) algorithms are well suited to the task of dynamic execution due to their flexibility to learn complex representations of high-dimensional policy functions in a computationally efficient manner, while updating continuously during the course of trading and therefore bypassing costly batch offline training that plagues traditional dynamic programming (DP) or dual methods. Furthermore, with the level and growth of assets under management held in the quantitative investment industry and the substantial quantity of trading executed through high frequency trading (HFT) firms and other electronic counterparties, trading cost mitigation has become a major focus of academic and industry research. The ability to scalably incorporate asset pricing signals into trading execution systems with relatively little training overhead is an attractive property of a reinforcement learning approach to this application.

I argue that this approach offers a principled methodology to construct optimal trading strategies that is capable of performing efficiently in a classical execution setting while also benefiting from the existence of high-dimensional signals derived from the limit order book (LOB) and/or other alternative risk premia (e.g. stochastic volatility). I derive results prooving the optimality of the RL solution method for several related trading models: I provide a proof of the convergence of Q-learning value iteration for the class of execution problems in question and point out a useful duality between the fixed point of the Q-learning contraction and the optimal solution to the Bellman equation of the dynamic program. I also document simulated experimental results indicating that the RL algorithm is able to reconstruct known analytic solutions, match the output of tabular dynamic programming methods, and significantly outperform those traditional methods in the presence of informative signals about prices.

The literature relevant to the discussion to follow spans several key sets of questions in finance, economics and computer science. This paper primarily contributes to a body of work that studies the dynamic trading execution problem itself from both theoretical and practical perspectives. Whereas the initial papers of Bertsimas & Lo (1998) and Almgren & Chriss (2001) formulated the problem as dynamic programs and derived analytical results related to convergence properties and efficiency gains, more recent work has expanded the scope of potential formulations by deriving solutions for alternative risk criteria, objective functions and constraints on trading (Gatheral & Shied 2011, Huberman & Stanz 2005). Even more recent work has begun to incorporate ideas from the literatures on reinforcement learning into dynamic trading: Hendrix & Wilcox (2014) formulate a theoretical extension of the Algren-Chriss Framework as a reinforcement learning problem, and Feng et al. (2006) perform practical experiments on live trading and exchange data using reinforcement learning algorithms.

This paper also relates to the body of new work focusing on the limit order book (LOB) and how microstructure foundations can effect optimal trading strategies, both in partial and general equilibrium settings. Obizhaeva & Wang (2013) construct a general equilibrium model of supply and demand for trades and study the dynamics of prices and order flow in that environment, while Dreschler et al. (2018) observe an empirical link between liquidity provision and volatility risk, and propose an intermediation mechanism by which those risks might become conflated. Within the domain of LOB and microstructure, there is also focus on the existence of alternative risk premia and their incorporation into optimal trading strategies: Cartea & Jaimungal (2016) focus on how block trades interact with order imbalance on the exchange to drive both temporary and permanent price impact and thereby affect trading, while Lehalle & Neuman (2018) study a more general setting in which potentially high-dimensional Markovian signals must be incorporated into trading strategies.

Finally, this paper contributes to the long history of work on tractable high-dimensional dynamic programming solution methods. Recent work has focused especially on gradient-based parametric methods and approximate dynamic programming for problems in macroeconomics (Brunnermeier & Sannikov 2016), operations research and resource allocation (Bouzaiene et al. 2005), and more general settings (Bhat 2016).

The remainder of the paper will proceed as follows. Section II will develop theoretical foundations of a new algorithm that offers a computationally efficient way to compute quality solutions for the block trading problem in a high frequency environment. It uniquely enables the optimal strategy to incorporate information from potentially high dimensional signals derived from microstructure, fundamental, or other data sources; to overcome the curse of dimensionality that restricts the allowable complexity for existing methods; and to train/update the optimal solution "online" (live during trading) without requiring costly batch retraining.

Section III presents simulated experimental results indicating that canonical known solutions to relatively simplistic variants of the formulation can be replicated using this method: for the well known geometric brownian motion with linear price impact variant of the formulation, key features of the optimal policy function include concavity of the policy surface due to the interaction between the impact term in the stochastic differential equation (SDE) and the zero lower bound (ZLB) on prices, and the increase in that concavity associated with greater market impact or higher asset volatility. Furthermore, cost savings relative to existing solution methods can be achieved when informative signals are introduced to the setting. The results demonstrate that when a known signal with non-trivial correlation to asset prices is introduced to the model, the reinforcement learning framework is better computationally suited to incorporate that information than a traditional DP approach, and thereby achieves significant improvements in execution cost.

In the case of a single asset and single correlated signal in the simulated setting, a correlation coefficient of 20% translates to a 3.25% reduction in trading costs for the counterparty or execution platform. Incorporating a 40% correlated signal translates to an over 8% reduction in trading costs. The improvements in execution performance are both statistically and economically meaningful, and indicate the benefits of the proposed RL algorithm over traditional approaches.

II. Model

A. General setting

In this section, a general dynamic trading execution problem is formulated and several properties are proven for the formulation. Then, solution methods are derived for several increasingly complex special cases of the general model: random walk (RW) price dynamics with linear absolute price impact; geometric brownian motion (GBM) price dynamics with linear relative price impact; and GBM prices with linear relative price impact and a Markovian signal.

Consider a setting in which an agent is tasked with trading Q shares of an asset with price process s_t (residing in the state space $\mathcal{X}$) over the fixed time period $t \in [0, T]$ by specifying the sequence of quantities to trade (i.e. trading strategy, policy) $\{q_t\}_{t=0}^{T}$ drawn from the action space $\mathcal{A}$ $(= \mathcal{R})$. The agent seeks to minimize the expected discounted cost of the entire block transaction, where the cost associated with a single trade is given by the twice continuously differentiable function $u(q_t)$ of the action taken in period t.

In a practical securities market this problem is inherently discrete, in that the agent can only feasibly make a finite number of trades given the physical constraints of execution on the exchange. However, it proves advantageous to formulate the continuous time relaxation of the discrete time problem, where $(\Omega, \mathcal{F}, \mathcal{P})$ is a probability space equipped with the standard filtration $\{\mathcal{F}_t\}_{t=0}^{T}$ and where $p_t : [0, T] \to \mathcal{R}$ and $q_t : [0, T] \to \mathcal{A}$ are interpreted as absolutely continuous processes adapted to $\mathcal{F}_t$ and unbounded on the real line. In continuous time, we can write down the agent's problem $\mathcal{D}$:

$$\begin{aligned}
\min_{\{q_t\}} \qquad & \mathrm{E}_0\Big[\int_0^T u(q_t)dt\Big] \\
\text{subj. to} \qquad & dY_t = \mu_t(Y_t, q_t)dt + \sigma_t(Y_t, q_t)dW_t \\
& dz_t = -q_t dt \\
& z_0 = Q \ , \quad z_T = 0
\end{aligned}$$

The key feature of this formulation is in the specification of the (multidimensional) state process $Y_t = \begin{bmatrix} s_t & Y_t^{(1)}, & \dots \end{bmatrix} \in \mathcal{R}^n$, which is an n-dimensional controlled diffusion process with the target asset price p_t as its first entry, and with the remaining entries determining the dynamics of additional sources of risk that impact prices (e.g. stochastic growth, volatility, etc) or signals (e.g. other asset prices) that may be systematically correlated with p_t. W_t is an n-dimensional standard brownian motion adapted to $\mathcal{F}_t$, while $\mu_t(Y_t, q_t) \in \mathcal{R}^n$ and $\sigma_t(Y_t, q_t) \in \mathcal{R}^{n \times n}$ are the drift and covariance matrices respectively of the controlled diffusion and determine the manner in which the choice of action q_t today impacts the future trajectory of the state. The currently outstanding residual position in the asset at time t is represented by an extra state variable, z_t, whose boundary conditions require that the entire block be executed by the close of the trading period.

We assume that there exists a twice continuously differentiable value function V satisfying the Bellman equation of $\mathcal{D}$:

$$V(Y_t, z_t) = \max_{q_t}\Big\{u(q_t) + \mathrm{E}_t[V(Y_t, z_t)]\Big\} \tag{2.1}$$

with terminal condition $V(Y_T, z_T) = 0$, and seek an associated optimal policy $\{q_t^*(Y_t, z_t)\}_{t=0}^T$ representing the solution to the agent's problem.

THEOREM 2.1 (Dynamic Programming Principle): *Let $\{q_t^*\}$ be an optimal policy for $\mathcal{D}$, and let there exist a value function V in the form of equation 2.1. Then V solves the Bellman equation:*

$$0 = \max_q\Big\{u(q_t) + V_Y\mu_t(Y_t, q_t) - V_z q_t + \frac{1}{2}\operatorname{Tr}\left(\sigma_t(Y_t, q_t)' V_{YY}\sigma_t(Y_t, q_t)\right) + V_t\Big\} \tag{2.2}$$

or, equivalently, satisfies the PDE:

$$\frac{du}{dq_t}(q_t^*) = V_z - V_Y\frac{\delta\mu_t}{\delta q_t}(Y_t, q_t^*) - V_{YY}\sigma_t(Y_t, q_t^*)\frac{\delta\sigma_t}{\delta q_t}(Y_t, q_t^*) \tag{2.3}$$

evaluated at the optimal policy q_t^*. Finally the Bellman equation attains it's maximum at $\{q_t^*\}$ among all admissible policies $\{q_t\} \in [0, T] \times \mathcal{A}$, and the maximal value is zero.

Proof. From the usual stochastic approximation to the Bellman equation 2.1 via Taylor expansion:

$$\begin{aligned}
V(Y_t, z_t) &= \max_q\Big\{u(q_t)dt + \mathrm{E}[V(Y_t + dY_t, z_t + dz_t)]\Big\}\\
&= \max_q\Big\{u(q_t) + \mathrm{E}[V(Y_t, z_t) + V_t + dV]\Big\}\\
&= V(Y_t, z_t) + \max_q\Big\{u(q_t) + V_t + \mathrm{E}[dV]\Big\}dt
\end{aligned}$$

and an application of Ito's lemma:

$$\begin{aligned}
\mathrm{E}[dV] &= \frac{\delta V}{\delta Y_t}(Y_t, z_t)\mu_t(Y_t, q_t) - \frac{\delta V}{\delta z_t}(Y_t, z_t)q_t + \frac{1}{2}\sigma_t(Y_t, q_t)'\frac{\delta^2 V}{\delta Y_t^2}(Y_t, z_t)\sigma_t(Y_t, q_t)\\
&= V_Y\mu_t(Y_t, q_t) - V_z q_t + \frac{1}{2}\operatorname{Tr}\left(\sigma_t(Y_t, q_t)'(V_{YY})\sigma_t(Y_t, q_t)\right)
\end{aligned}$$

we arrive at the HJB equation:

$$0 = \max_q\Big\{u(q_t) + V_Y\mu_t(Y_t, q_t) - V_z q_t + \frac{1}{2}\operatorname{Tr}\left(\sigma_t(Y_t, q_t)'(V_{YY})\sigma_t(Y_t, q_t)\right) + V_t\Big\}$$

Differentiating the argument with respect to q_t and setting the result equal to zero, we arrive at the first order condition for optimality in the form of a PDE:

$$\frac{du}{dq_t}(q_t^*) - V_z + V_Y\frac{\delta\mu_t}{\delta q_t}(Y_t, q_t^*) + \frac{1}{2}\operatorname{Tr}\left(V_{YY}\sigma_t(Y_t, q_t^*)\right)\frac{\delta\sigma_t}{\delta q_t}(Y_t, q_t^*) = 0$$

It is left to demonstrate that equation 2.1 attains its maximum value for the optimal policy $\{q_t^*\}_{t=0}^T := \Pi^*$. Define for all policies $\Pi = \{q_t\} \in [0, T] \times \mathcal{A}$ the Ito process:

$$C_t(\Pi) = \int_0^t u(q_s)ds + \mathrm{E}_t[V(Y_t, z_t)]$$

such that $\mathrm{E}_0[C_T(\Pi)] = \mathrm{E}_0[\int_0^T u(q_t)dt]$ is identical to the objective value of problem $\mathcal{D}$ for policy Π. Since Π^* is optimal for $\mathcal{D}$:

$$\mathrm{E}_0[C_T(\Pi)] \leq \mathrm{E}_0[C_T(\Pi^*)] \implies \mu_C(V, Y, z, \Pi) \leq \mu_C(V, Y, z, \Pi^*)$$

where $\mu_C(V, Y, z, \cdot)$ is the drift term obtained by applying Ito's lemma to C_t (as in the derivation of equation 2.2). The resulting expression is identical to the argument of the RHS of equation 2.2. It follows that the maximizer $\Pi^* = \{q_t^*\}$ of $\mathrm{E}_0[C_T(\Pi)]$ must also be the maximizer of equation 2.1. $\square$

B. *Approximate solutions & the reinforcement learning formulation*

At this point, we have established via the Dynamic Programming Principle a machinery for generating optimal solutions to $\mathcal{D}$ by solving a PDE. If there exists a twice continuously differentiable analytic function V satisfying equation 2.3 exactly, then its partial derivatives can be substituted back into equation 2.3 and the resulting equation can be solved explicitly for $q_t^*(Y_t, z_t)$ in terms of the state variables which, by Theorem 2.1, represents a trajectory along which the Bellman equation attains its optimum. And if this policy function $q_t^*(Y_t, z_t)$ satisfies the conditions of Theorem ??, then it is a verified optimal solution of $\mathcal{D}$.

However, in practice - i.e. for most interesting choices of $\mu_t(Y_t, q_t)$ and $\sigma_t(Y_t, q_t)$ - an exact analytic solution V of equation 2.3 is typically unavailable. In these cases, it is highly desirable to develop a procedure to construct numerical approximations to V with acceptable statistical properties, such that a quality approximate solution to $\mathcal{D}$ can be delivered under practical computational conditions.

Several mature literatures within the umbrella of stochastic dynamic programming develop such approximate methods using various techniques, the most popular of which fall into two broad categories. Quaditure (or "tabular") methods involve discretization of the state and control spaces, followed by delegation of the resulting discrete representation of the PDE as a system of nonlinear equations to an optimization solver, which can be solved to near optimality yielding a discretized, interpolatable representation of the value and/or policy function. The main drawback of these methods is their susceptibility to the curse of dimensionality, as high-dimensional state or policy spaces become difficult or impossible to retain in computer memory (Betts 2009, Judd 1998, Chen et al. 1999). This shortcoming is problematic in our setting, where the set of prices, risks and correlated signals to be incorporated into an optimal trading strategy is likely to be multidimensional.

Alternatively, parametric approximation methods involve imposing local structural assumptions on problematic state equations, the objective function, or the value function itself (or some combination of these) and delegating the parameter estimation problem to nonlinear solvers. Variants of the parametric approximation approach are diverse and include collinear direct methods (Judd 1992, Werbos 1994), local perturbation methods (Judd & Guu 1997), and more recent developments such as pseudospectral methods (Garg et al. 2010) and, most importantly for the method introduced in this section, approximate dynamic programming solutions (Powell 2011, Bertsekas & Tsitsiklis 1996). The latter in particular involves imposing a flexible yet tractable parametric form on the value function itself and solving for the optimal parameters of said representation by gradient-like methods. These techniques share the common trade off between the structural complexity of the parametric representation assumed for the corresponding model object, and the degree of local ill-conditioning that the solution method can bear along the optimal trajectory. As the complexity of parametric representations increase the guaranteed convergence and precision of the global approximation improves, at the expense of greater computational cost, for example in the form of memory required to store local gradients or the time required to solve a sequence of nonlinear systems of equations.

Finally, it should be noted that recent advances have been made in adapting genetic algorithms to the problem of optimal stochastic control, by which "competing" solutions are compared via (certainty equivalent based) heuristic approximations to expected cost in a large-scale simulation framework. Genetic methods have shown promise in fields such as mechanical routing, but have yet to become as widely used for applications in finance, economics, and operations research (Conway 2012).

Consider the approximator $\hat{V}(Y_t, z_t; \hat{\Theta})$ of the true value function $V(Y_t, z_t)$ conditional on $\hat{\Theta} = \{\hat{\theta}^{(j)}\}_j$, a collection of estimators of some true parameter set Θ. Suppose that $\hat{V}$ is infinitely continuously differentiable in all of it's arguments (i.e. all states, parameters, and time). And suppose that we observe the finite sequence of independent, exogenous risk factors $\{W_0, W_1, \ldots, W_T\}$ where $W_t - W_{t-1} \sim N(0, 1_n)$ $\forall t \in [1, T]$.

Take $q^* : \mathcal{R}^n \times \mathcal{R} \to \mathcal{A}$ to be a function mapping states (Y_t, z_t) to optimal actions. For a sequence of estimates $\hat{\Theta}_t$, at each time t we can seek to minimize the loss:

$$\min_{\hat{\Theta}_t} \mathcal{L}(\hat{\Theta}_t; Y_t, z_t) = \mathrm{E}_t\big[\big(V(Y_{t+1}, z_{t+1}) - \hat{V}(Y_{t+1}, z_{t+1}; \hat{\Theta}_t)\big)^2\big]$$

The solution $\hat{\Theta}$ and the corresponding object $\hat{V}(Y_t, z_t; \hat{\Theta})$ represents the optimal approximator among the class parameterized by Θ to the true value function V. We will show that this loss corresponds precisely to the reinforcement learning formulation of $\mathcal{D}$ in discrete time, and that the resulting approximate solution constitutes an optimal policy in accordance with the Dynamic Programming Principle.

We first state a useful contraction mapping theorem for the value function.

LEMMA 2.1: *The true value function V is the fixed point of a contraction operator H mapping value functions $V : \mathcal{X} \times \mathcal{A} \to \mathcal{R}$ to the space of value functions.*

Proof.

$$\begin{aligned}
||HV^{(1)} - HV^{(2)}||_\infty &= \max_{Y,z} \Big| \max_q\{u(q) + \mathrm{E}[V^{(1)}(Y,z)]\} - \max_q\{u(q) + \mathrm{E}[V^{(2)}(Y,z)]\}\Big| \\
&= \max_{Y,z} \Big|\mathrm{E}\big[\max_q\{[V^{(1)}(Y,z)]\} - \max_q\{[V^{(2)}(Y,z)]\}\big]\Big| \\
&\leq \max_{Y,z}\ \mathrm{E}\Big[\Big|\max_q\{[V^{(1)}(Y,z)]\} - \max_q\{[V^{(2)}(Y,z)]\}\Big|\Big] \\
&\leq \max_{Y,z}\ \mathrm{E}\Big[\max_q |V^{(1)}(Y,z) - V^{(2)}(Y,z)|\Big] \\
&= ||V^{(1)} - V^{(2)}||_\infty
\end{aligned}$$

The claim follows from an application of the Contraction Mapping Theorem. □

The fact that the true value function V resides in the space of accumlation points of a contraction mapping is a principal indication that it can be reached by stochastic gradient methods. We therefore proceed by constructing stochastic gradient descent (SGD) iterates with fixed step size λ (the restriction of fixed step size can be relaxed without loss of generality, but we preserve it for the remainder of the section):

$$\begin{aligned}
\hat{\Theta}_{t+1} &= \hat{\Theta}_t - \lambda\ \nabla_\Theta \mathcal{L}(\hat{\Theta}_t; Y_t, z_t) \\
&= \hat{\Theta}_t - 2\lambda\ \big(\hat{V}(Y_t, z_t; \hat{\Theta}_t) - V(Y_t, z_t)\big)\ \nabla_\Theta \hat{V}(Y_t, z_t; \hat{\Theta}_t) \\
&= \hat{\Theta}_t - 2\lambda\ \big(\max_{q_t}\{u(q_t) + \mathrm{E}_t[\hat{V}(Y_{t+1}, z_{t+1}; \hat{\Theta}_t)]\} - \hat{V}(Y_t, z_t; \hat{\Theta}_t)\big)\ \nabla_\Theta \hat{V}(Y_t, z_t; \hat{\Theta}_t)
\end{aligned}$$

The update step is derived in two steps: first, the gradient of the loss function is computed with respect to the parameter in question; then, the true value function is approximated by $\hat{V}$ evaluated in the most recent period, and with the most recent optimal action q_t^*. For the time being, no knowledge of the underlying parametric structure of problem $\mathcal{D}$'s state space is being used, only the definition of the value function operator and the continuous differentiability of $\hat{V}$ in all $\hat{\Theta}$. These updates are also highly computationally efficient due to the fact that they can be performed parameter by parameter and parallelized across GPU hardware with very low memory overhead per iteration.

Using the SGD update rule for $\hat{\Theta}$ we can now recover the classical reinforcement learning value iteration ("Q-learning") update rule by performing the first-order Taylor expansion of $\hat{V}$ in terms of $\hat{\Theta}$, thereby demonstrating an important duality between the dynamic programming and reinforcement learning formulations of $\mathcal{D}$:

$$\begin{aligned}\hat{V}(Y_t, z_t; \hat{\Theta}_{t+1}) &= \hat{V}(Y_t, z_t; \hat{\Theta}_t) + \nabla_\Theta \hat{V}(Y_t, z_t; \hat{\Theta}_t)(\hat{\Theta}_{t+1} - \hat{\Theta}_t) \\ &= \hat{V}(Y_t, z_t; \hat{\Theta}_t) + 2\lambda \nabla_\Theta \hat{V}(Y_t, z_t; \hat{\Theta}_t)^2 \big(\max_{q_t}\{u(q_t) + \mathrm{E}_t[\hat{V}(Y_{t+1}, z_{t+1}; \hat{\Theta}_t)]\} - \hat{V}(Y_t, z_t; \hat{\Theta}_t)\big)\end{aligned} \tag{2.4}$$

Having derived the value iteration update rule for $\mathcal{D}$, we proceed to prove the convergence of $\hat{V}$ to the optimal policy V. We first state a known lemma, then proceed to the convergence result.

LEMMA 2.2: *For the state-action vector $X \in \mathcal{R}^n$ and the random process:*

$$\Delta_{t+1}(X) := \big(1 - g_t(X)\big)\Delta_t(X) + g_t(X)F_t(X)$$

converges to zero with probability one under the following conditions:

- $\sum_t g_t(X) = \infty \;, \;\; \sum_t g_t^2(X) < \infty \quad \forall X$
- $||\mathrm{E}[F_t(X)]||_\infty \leq ||\Delta_t(X)||_\infty$
- $\mathrm{Var}[F_t(X)] < C(1 + ||\Delta_t(X)||_\infty)^2$ *for some* $C > 0$

Proof. Refer to Jaakkola, Jordan & Singh (1993), Appendix A. □

THEOREM 2.2 (Convergence): *Assume the markov decision process defined by problem $\mathcal{D}$, the value function approximator $\hat{V}$ parameterized by the sequence $\{\hat{\Theta}_t\}$, and the reinforcement learning value iterates constructed as in equation 2.4. Further suppose that the state spaces $\mathcal{X}$ is bounded on $\mathcal{R}^n$ and that the gradient of $\hat{V}$ is well-conditioned in the following sense:*

- $\sum_t \nabla_\Theta \hat{V}(Y_t, z_t; \hat{\Theta}_t)^2 = \infty$
- $\sum_t \nabla_\Theta \hat{V}(Y_t, z_t; \hat{\Theta}_t)^4 < \infty$

element by element of the collection $\hat{\Theta}_t$. Then the approximator $\hat{V}$ converges to the true value function V with probability one.

Proof. We define the objects g_t and Δ_t in terms of $\hat{V}$:

$$\begin{aligned} g_t(X_t) &= g_t(Y_t, z_t, \hat{\Theta}_t) = 2\lambda \nabla_\Theta \hat{V}(Y_t, z_t; \hat{\Theta}_t)^2 \\ \Delta_t(X) &= \Delta_t(\hat{\Theta}_t; Y_t, z_t) = \hat{V}(Y_t, z_t; \hat{\Theta}_t) - V(Y_t, z_t)\end{aligned}$$

Rearranging the terms of equation 2.4 yields:

$$\hat{V}(Y_t, z_t; \hat{\Theta}_{t+1}) = \big(1 - g_t(Y_t, z_t\hat{\Theta}_t))\big)\hat{V}(Y_t, z_t; \hat{\Theta}_t) + g_t(Y_t, z_t, \hat{\Theta}_t)\big(\max_{q_t}\{u(q_t) + \mathrm{E}_t[\hat{V}(Y_{t+1}, z_{t+1}; \hat{\Theta}_t)]\}\big)$$

and subtracting the true value function V from both sides gives:

$$\begin{aligned}\Delta_{t+1}(\hat{\Theta}_{t+1}; Y_t, z_t) =& (1 - g_t(Y_t, z_t\hat{\Theta}_t)))\Delta_t(\hat{\Theta}_t; Y_t, z_t) \\ &+ g_t(Y_t, z_t, \hat{\Theta}_t)(\max_{q_t}\{u(q_t) + \mathrm{E}_t[\hat{V}(Y_{t+1}, z_{t+1}; \hat{\Theta}_t)]\} - V(Y_t, z_t)) \\ =& (1 - g_t(Y_t, z_t\hat{\Theta}_t)))\Delta_t(\hat{\Theta}_t; Y_t, z_t) + g_t(Y_t, z_t, \hat{\Theta}_t)F_t(Y_t, z_t, \hat{\Theta}_t)\end{aligned}$$

It is now left to verify the technical conditions of Lemma 2.2. First:

$$\begin{aligned}\mathrm{E}[F_t(Y, z, \hat{\Theta})] &= \max_{q_t}\{u(q_t) + \mathrm{E}_t[\hat{V}(Y_{t+1}, z_{t+1}; \hat{\Theta}_t)]\} - V(Y_t, z_t) \\ &= H\hat{V}(Y, z, \hat{\Theta}) - HV(Y, z) \\ &\leq ||\hat{V}(Y, z, \hat{\Theta}) - V(Y, z)||_\infty = ||\Delta_t(Y, z, \hat{\Theta})||_\infty\end{aligned}$$

where the first equality uses the fact that at the accumulation point $HV = V$, and the second equality applies Lemma 2.1. Lastly:

$$\begin{aligned}\mathrm{Var}[F_t(Y, z, \hat{\Theta})] &= \mathrm{E}\Big[\big(H\hat{V}(Y, z; \hat{\Theta}) - V(Y, z) - \mathrm{E}[H\hat{V}(Y, z; \hat{\Theta}) - V(Y, z)]\big)^2\Big] \\ &= \mathrm{E}\Big[V(Y, z) - \mathrm{E}[V(Y, z)]\Big] = \mathrm{Var}[V(Y, z)]\end{aligned}$$

which is trivially finite in the case that the state space $\mathcal{X}$ is bounded. It follows by Lemma 2.2 that $\hat{V}$ converges to V via the reinforcement learning value iterates with probability one. □

We have now demonstrated that Q-learning does in fact attain the accumulation point of the contraction H of $\hat{V}$. We now verify that this discrete-time algorithm also provides a solution that is optimal for the original continuous-time relaxation $\mathcal{D}$. Doing so will also address a significant computational issue that arises in the implementation of the reinforcement learning formulation. Returning to the value iterates of equation 2.4, the loss term:

$$l_t = \max_{q_t}\{u(q_t) + \mathrm{E}_t[\hat{V}(Y_{t+1}, z_{t+1}; \hat{\Theta}_t)]\} - \hat{V}(Y_t, z_t; \hat{\Theta}_t)$$

poses a potential problem due to the appearance of the expectation within the max operator. In a discrete-time setting, the presence of this term necessitates the computation of an integral, which is usually obtained via highly optimized linear algebra routines maintaining matrices of transition probabilities. However, in our setting we are armed with the parametric structure of the state space in continuous time, enabling us the apply Ito's Lemma to the expectation and thereby recover:

$$\begin{aligned}l_t &= \max_{q_t}\{u(q_t) + \hat{V}_t + \mathrm{E}[d\hat{V}]\} \\ &= \max_{q}\Big\{u(q_t) + V_Y\mu_t(Y_t, q_t) - V_z q_t + \frac{1}{2}\mathrm{Tr}\left(\sigma_t(Y_t, q_t)'V_{YY}\sigma_t(Y_t, q_t)\right) + V_t\Big\}\end{aligned}$$

the Bellman residual. As a consequence of the convergence result in Theorem 2.2, the value iterates

$\hat{V}(Y_t, z_t; \hat{\Theta})$ can be made arbitrarily close together given sufficient updates; likewise, the Bellman residual can be made arbitrarily small. At the fixed point V itself of the contraction H, the Bellman residual l_t is identically zero. By Theorem 2.1, the fixed point V then also represents an optimal solution to the continuous time variant $\mathcal{D}$, as desired.

C. Application: analytic solutions in continuous time

We now study the optimal solutions to problem $\mathcal{D}$ under a several increasingly complex (and increasing realistic) canonical parameterizations, focusing on the existence or non-existence of analytic solutions to the dynamic programming PDE and the computational properties of the resulting Q-learning iterates. All of these examples will assume a single traded price $y_t \in \mathcal{R}$. We first consider the case of random walk price dynamics, with a linear price impact term governed by a constant coefficient ϕ. The market impact term is deemed "absolute" in the sense that it does not depend on the level of prices.

$$\begin{aligned} \min_{\{q_t\}} \quad & \mathrm{E}_0\left[\int_0^T s_t q_t dt\right] \\ \text{subj. to} \quad & ds_t = -\phi q_t dt + \sigma dW_t \\ & dz_t = -q_t dt \\ & z_0 = Q \ , \quad z_T = 0 \end{aligned}$$

In the general formulation, this optimization problem statement corresponds to setting $u(q) = sq$, $\mu(s,q) = -\phi q$, and $\sigma(s,q) = \sigma$ constant. By Theorem 2.1, the optimal value function satisfies the PDE:

$$s - V_s - Vz = 0$$

Substituting $\alpha = s$ and $\beta = -s + \phi z$ yields the ODE in α only:

$$\alpha - \phi V_\alpha = 0 \implies V(\alpha, \beta) = \frac{\alpha}{2\phi} + f(\beta) \implies V(s,z) = \frac{s^2}{2\phi} + f(-s+\phi z)$$

Introducing the terminal condition $V(*, 0) = 0$ yields:

$$0 = \frac{s^2}{2\phi} + f(-s) \implies f(s) = \frac{s^2}{2\phi} \implies V(s,z) = \frac{s^2}{2\phi} + \frac{(\phi z - s)^2}{2\phi} = sz - \frac{\phi}{2}z^2$$

We thereby arrive at a closed form analytic solution for the optimal value function of the execution problem. Note that solving the PDE of the continuous time relaxation yields a continuous value function. When applied to any discrete, finite trading period $t \in [0, T]$, the value function gives rise to the optimal policy: $q_t^* = \frac{Q}{T} \ \forall \ t$. That is, the optimal behavior is to discretize the block trade into T equal components, and execute one component per period, regardless of the realized path of prices. This "naive" strategy arises due to the particular parameterization of the problem, in which market impact has no predictable interaction with prices such that the trader never derives any advantage from "timing" her trades based on price levels. This result is equivalent to the one derived by Bertsimas & Lo (1998) for the discrete time case.

The SGD iterates corresponding to the reinforcement learning formulation are:

$$\begin{aligned}\hat{\Theta}_{t+1} &= \hat{\Theta}_t - 2\lambda\ \big(s_t q_t^* - \phi q_t^* V_s - q_t^* V_z + \frac{1}{2}\sigma^2 V_{ss}\big)\ \nabla_\Theta \hat{V}(s_t, z_t; \hat{\Theta}_t)\\ &= \hat{\Theta}_t - 2\lambda\ \big(\frac{1}{T}s_t - \frac{1}{T}\phi Q V_s - \frac{1}{T} Q V_z + \frac{1}{2}\sigma^2 V_{ss}\big)\ \nabla_\Theta \hat{V}(s_t, z_t; \hat{\Theta}_t)\end{aligned}$$

Where the optimal policy is a known function of the parameters of $\mathcal{D}$, rather than a function of the partial derivatives of the value function as is typically the case (for example, see below).

D. *Application: numerical solutions in continuous time*

We now introduce a second example formulation, this one with geometric brownian motion (GBM) price dynamics and a *relative* linear market impact, again governed by a constant coefficient ϕ, but this time also a function of the current level of prices y_t.

$$\begin{aligned}&\min_{\{q_t\}} && \mathrm{E}_0\Big[\int_0^T s_t q_t dt\Big]\\ &\text{subj. to} && ds_t = -\phi q_t s_t dt + \sigma s_t dW_t\\ & && dz_t = -q_t dt\\ & && z_0 = Q\ ,\quad z_T = 0\end{aligned}$$

This simple modification to the structural dynamics of prices already renders the problem without a known analytic solution to the PDE. The corresponding SGD iterates for the reinforcement learning formulation are:

$$\hat{\Theta}_{t+1} = \hat{\Theta}_t - 2\lambda\ \big(s_t q_t^* - \phi q_t^* s_t V_s - q_t^* V_z + \frac{1}{2}\sigma^2 s_t^2 V_{ss}\big)\ \nabla_\Theta \hat{V}(s_t, z_t; \hat{\Theta}_t)$$

Finally, we consider a formulation containing one price governed by GBM dynamics with linear relative price impact, but this time introducing a signal that is a function of a second, correlated source of risk, and that impacts prices through the drift term of the SDE.

$$\begin{aligned}&\min_{\{q_t\}} && \mathrm{E}_0\Big[\int_0^T s_t q_t dt\Big]\\ &\text{subj. to} && d\begin{bmatrix} s_t \\ x_t \end{bmatrix} = \begin{bmatrix} (x_t - \phi q_t)s_t \\ 0 \end{bmatrix} dt + \begin{bmatrix} \sigma_s & \rho \\ \rho & \sigma_x \end{bmatrix} dW_t\\ & && dz_t = -q_t dt\\ & && z_0 = Q\ ,\quad z_T = 0\end{aligned}$$

The SGD iterates for the signal variant are:

$$\hat{\Theta}_{t+1} = \hat{\Theta}_t - 2\lambda \left(s_t q_t^* - V_s \begin{bmatrix} (x_t - \phi q_t) s_t \\ 0 \end{bmatrix} - q_t^* V_z + \frac{1}{2} \begin{bmatrix} \sigma_s & \rho \\ \rho & \sigma_x \end{bmatrix} V_{ss} \begin{bmatrix} \sigma_s & \rho \\ \rho & \sigma_x \end{bmatrix} \right) \nabla_\Theta \hat{V}(s_t, z_t; \hat{\Theta}_t)$$

III. Simulation Results

The section to follow contains an empirical calibration exercised based on historical microstructure data, as well as the results of performance simulations of the reinforcement learning algorithm under the increasingly comply model settings described in Section II.

A. Empirical estimation of realistic model parameters

We first present findings from an empirical analysis performed on hourly aggregations of historical quotes and trades data from the TAQ exchange trading database for the ticker SPY from January 2019. The analysis serves as a calibration of the parameter values used in the performance simulations of the RL algorithm to follow, as well as offering several observations about the nature of microstructure quantities that might be relevant for electronic trading applications most suitable for the method.

Let $K(t_1, t_2) = \{p_t : t \in [t_1, t_2]\}$ be the set of quoted prices list in the time interval $t \in [t_1, t_2]$ and let $M(t_1, t_2) = \{q_t : t \in [t_1, t_2]\}$ be the set of posted quantities of shares for all orders placed in $[t_1, t_2]$. Note that the elements of $M(t_1, t_2)$ include all orders, even those that may not ever be executed - as such it reflects the state of the order book during the time interval, not only the set of executed trades. Furthermore, the quantites may be positive or negative, reflecting "buy" or "sell" orders respectively.

We define and compute the following microstructure quantities for a single asset (SPY), where the increment $[t_1, t_2]$ are typically one hour long.

$$
\begin{aligned}
r(t_1,t_2) &= \frac{p_{\sup\{t:p_t \in K(t_1,t_2)\}} - p_{\inf\{t:p_t \in K(t_1,t_2)\}}}{p_{\inf\{t:p_t \in K(t_1,t_2)\}}} \\
s(t_1,t_2) &= \frac{1}{60}\sum_{i=1}^{60}\left[r\Big(t_1 + (i-1)\frac{t_2-t_1}{60}, t_1 + i\frac{t_2-t_1}{60}\Big) - \frac{r(t_1,t_2)}{60}\right]^2 \\
b(t_1,t_2) &= \frac{\sum_{q_t \in M(t_1,t_2), q_t>0} q_t - \sum_{q_t \in M(t_1,t_2), q_t<0} q_t}{\sum_{q_t \in M(t_1,t_2), q_t>0} q_t + \sum_{q_t \in M(t_1,t_2), q_t<0} q_t}
\end{aligned}
$$

Summary statistics for these quantities computed over the month long period 01-Jan-2019 to 31-Jan-2019 are reported in the table below. Both volatility and order imbalance were positive on average over the month, and both volatility and imbalance exhibit positive correlation with hourly returns

Empirical Sample Moments: 01-Jan-19 to 31-Jan-19 (N=504)							
	Mean	StDev	Skew	Kurt	P05	P95	Corr(Ret)
Hourly Returns	0.54%	0.72%	-0.24	4.56	-3.8%	2.7%	
Hourly Volatility	0.72%	0.38%	0.83	4.15	0.2%	3.2%	14.5%
Hourly Imbalance	0.083	0.185	-0.15	3.79	-0.856	0.763	8.3%

B. Replication of known analytic solution in random walk case

Now that we have established reasonable values for the parameters of the dynamic execution model, our first task will be to recover from a policy function trained via the reinforcement learning algorithm the known closed-form analytic solution to the simple random walk, linear absolute price impact problem derived in Section II.C. In particular, we follow our machinery to formulate the continuous time relaxation of the discrete-time problem, solve that problem numerically over the course of trading during a set of $N = 1,000$ stochastic simulations of the stock price with random walk dynamics and linear impact, discretize the resulting solution in the form of a discrete optimal policy function $q(s_t, z_t)^*$, and inspect the optimal policy for similarities to the known solution.

In the discussion that follows, we will at times focus on the properties of the "normalized" policy function:

$$\hat{q}(s_t, z_t)^* = \frac{q(s_t, z_t)^*}{z_t}$$

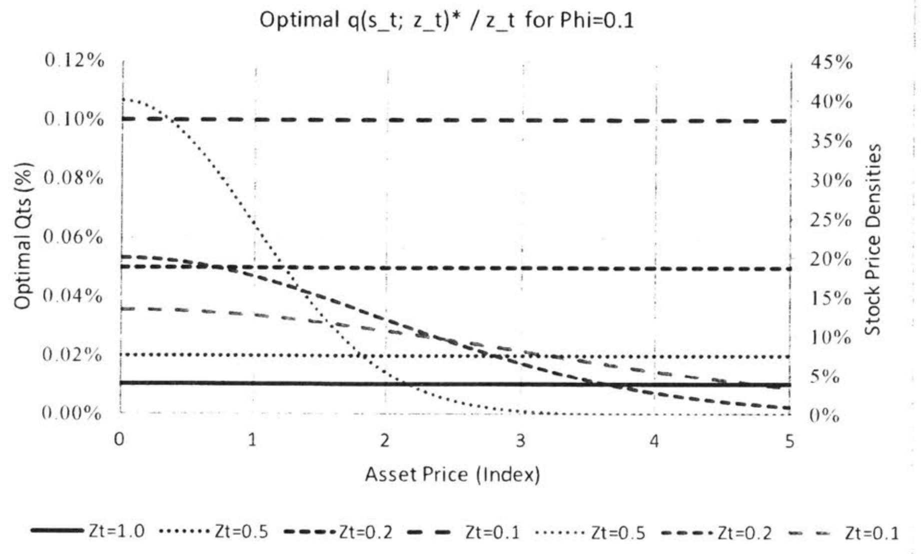
which represents the fraction of the residual shares that should optimally be executed in period t, rather than the non-normalized counterpart $q(s_t, z_t)^*$. This normalization aids in visualizing relationships between the policy function evaluated at different values of the state variables since during the course of trading, non-normalized share counts may become very small making comparisons between the value function or policy gradients across different values of z_t difficult.

The figure below plots the normalized policy for the random walk, linear impact problem for different residual share counts $z \in \{1.0, 0.5, 0.2, 0.1\}$, along with the simulated stock price densities observed across all simulated paths at the point at which $z_t = z$. We assume that the initial quantity of shares to sell is $S_0 = 100$, such that the non-normalized policy function can be interpreted as the percentage of the initial allocation of shares to be executed in period t. We assume a price impact parameter of $\phi = 0.1$. We see that the policy function is a constant function in the stock price and that:

$$\hat{q}(s_t, z_t)^* = \frac{1}{S_0 z_t}$$

ror all z_t. This numerical solution is consistent with the known analytic solution derived in Section ??, with the surprising interpretation that the full trade should be divided into equally-sized partitions and executed one partition per trading period regardless of the path of prices. This naive strategy is practically relevant only in the most stylized of settings, but serves as a logical first sanity check to verify that the RL algorithm produces a sensible solution for a known problem.

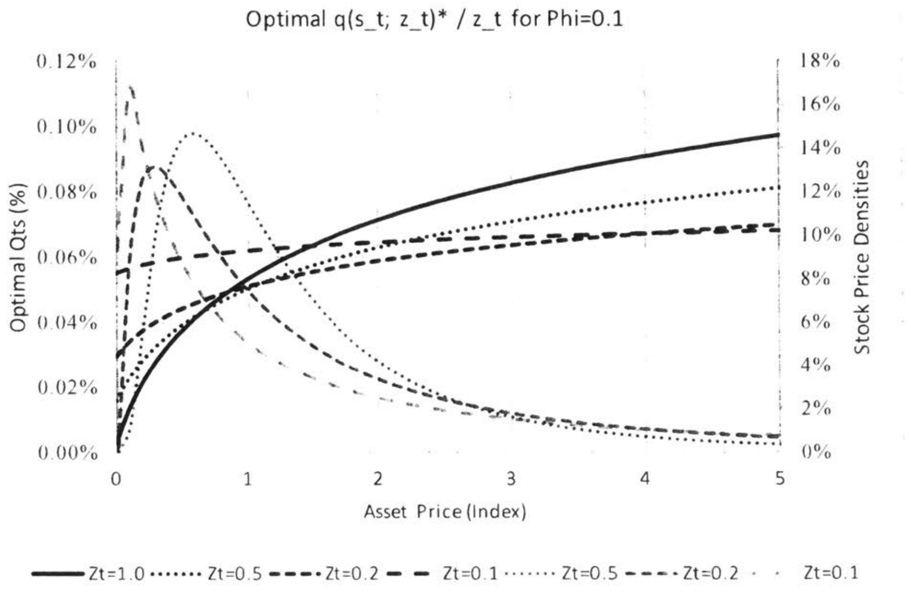
The figure also contains empirical densities of the asset price across the simulated scenarios at different stages of the simulation (i.e. when different remaining fractions of shares left to trade). Predictably for random walk dynamics, prices become more dispersed later in the simulation, as there are no upper or lower boundaries on the trajectories (in particular, prices can go negative - a particularly unrealistic characteristic of pure random walk prices). For example for $z_t = 0.5$, implying the point on each trajectory at which point half of the block remained to be traded, approximately 80% of the probability mass lies between index values of zero and two, and the optimal normalized policy is 2% of the remaining block. For $z_t = 0.1$, only 55% of the mass lies between zero and two, and $q_t^* = 0.10$.



C. Replication of numerical solution from Bellman iteration method in GBM case

We now proceed to apply the RL execution algorithm to a slightly more sophisticated problem: the geometric brownian motion price dynamics, linear price impact variant discussed in Section II.D. In this setting, since no simple analytic solution exists, the traditional approach to deriving an optimal policy function is to solve the formulated continuous time relaxation in the form of a dynamic program, either via a PDE solution method or, more typically, bellman iteration on a discretized grid of potential values of the state variable. By this method, one would arrive at a discretized version of the continuous time optimal policy, usable for live trading. The discussion to follow illustrates the solution to the GBM, linear impact problem derived by the RL approach and compares that solution to the one obtained via a bellman iteration DP solver.

The figure below plots the normalized optimal policy derived by the RL algorithm along with the stock prices densities across the simulated price paths. The normalized policy differs significantly from the naive solution to the random walk problem in several key respects. The policy gradients are no longer constant in either the stock price or the residual share count: it is now a convex three-dimensional surface. The surface is monotonically increasing in the stock price, reflecting the fact that when stock prices are low, the asymmetric price increments interact with the price impact term in the SDE and the zero lower bound (ZLB) in prices to drive the optimal number of shares to execute in the present period lower, since the risk of prices falling in the future is not so great. The surface is most concave in prices when the number of residual shares is high, such that the impact of the former effect is greatest on total trading profits; it flattens as the number of residual shares approaches zero at which point the trader is forced to execute the remaining shares regardless of price due to the impending terminal condition.

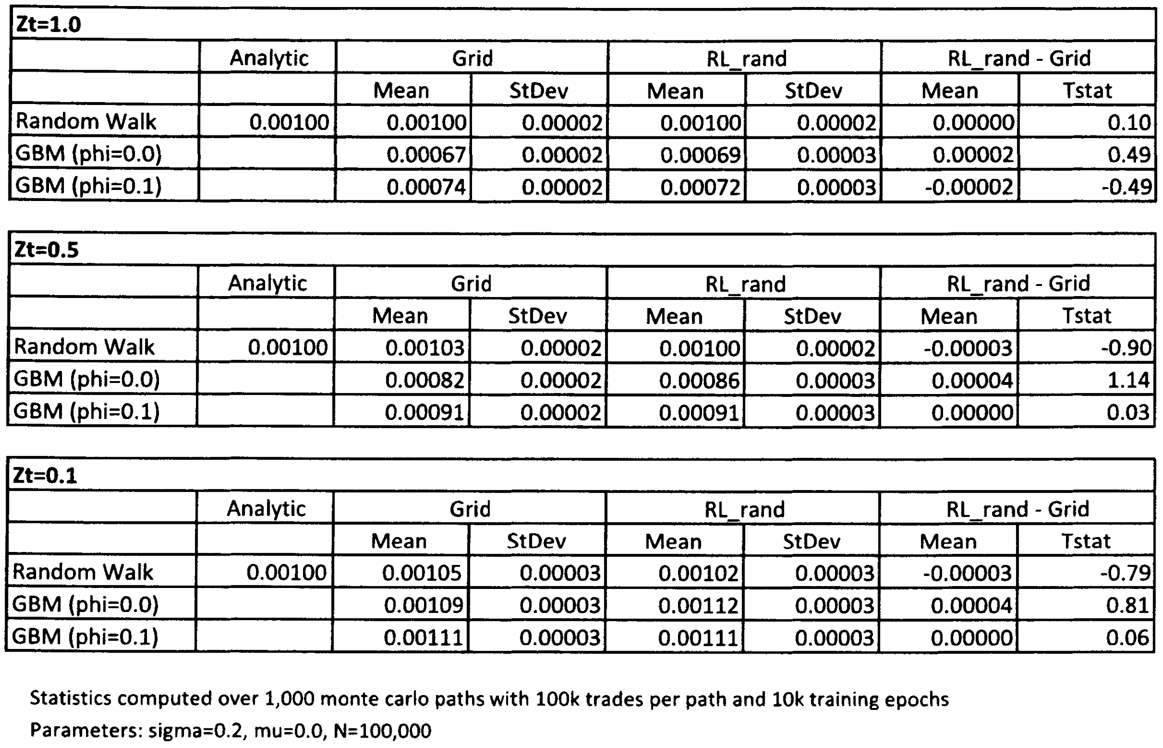


Given a general understanding of the shape and dynamics of the optimal policy derived numerically for the GBM, linear impact problem, we now assess whether there is a statisical difference between the RL algorithm's policy and the more traditional bellman iteration approach solved statically and executed on the same set of simulated price paths. We conduct hypothesis test by estimating the simulated sample moments of the non-normalized share counts evaluated at different points in the simulation (that is, for different $z \in \{1.0, 0.5, 0.1\}$):

$$\hat{\mathrm{E}}[q(s_t, z_t)^* | z_t = z] = \frac{1}{N} \sum_{i=1}^{N} q_i(s_t, z)^*$$

$$\hat{\mathrm{Var}}[q(s_t, z_t)^* | z_t = z] = \frac{1}{N} \sum_{i=1}^{N} \left(q_i(s_t, z)^* - \hat{\mathrm{E}}[q(s_t, z_t)^* | z_t = z] \right)^2$$

where $q_i(s_t, z)^*$ is the optimal quantity to be executed along simulated path indexed i in the period in which $z_t = z$. The table below reports the sample moments and resulting CLT test statistics comparing the solutions produced by the Bellman grid and the RL algorithm at three different points in the simulation. The optimal quantities are very close in economic magnitude, typically differing by less that one percent along most paths, and are statistically indistinguishable as expected.



Zt=1.0							
	Analytic	Grid		RL_rand		RL_rand - Grid	
		Mean	StDev	Mean	StDev	Mean	Tstat
Random Walk	0.00100	0.00100	0.00002	0.00100	0.00002	0.00000	0.10
GBM (phi=0.0)		0.00067	0.00002	0.00069	0.00003	0.00002	0.49
GBM (phi=0.1)		0.00074	0.00002	0.00072	0.00003	-0.00002	-0.49

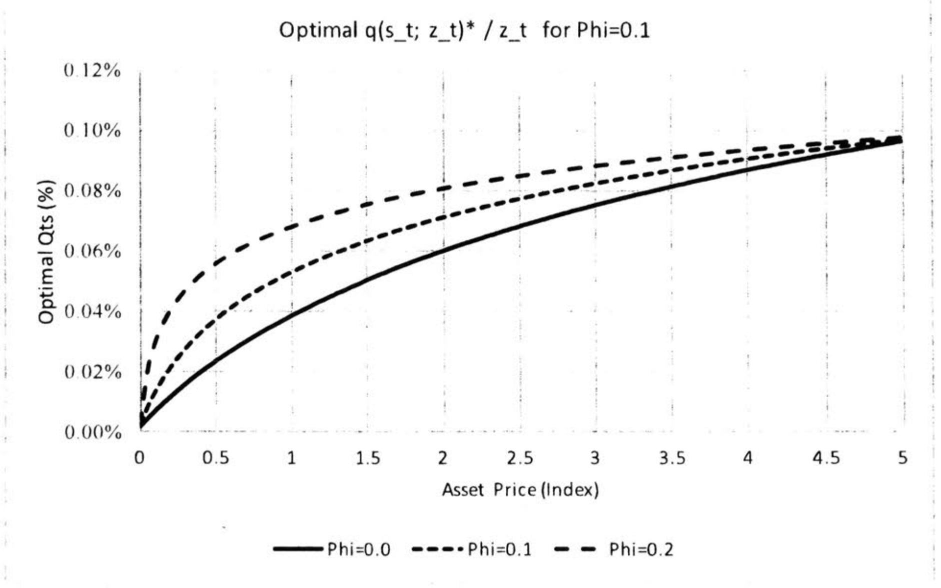
Zt=0.5							
	Analytic	Grid		RL_rand		RL_rand - Grid	
		Mean	StDev	Mean	StDev	Mean	Tstat
Random Walk	0.00100	0.00103	0.00002	0.00100	0.00002	-0.00003	-0.90
GBM (phi=0.0)		0.00082	0.00002	0.00086	0.00003	0.00004	1.14
GBM (phi=0.1)		0.00091	0.00002	0.00091	0.00003	0.00000	0.03

Zt=0.1							
	Analytic	Grid		RL_rand		RL_rand - Grid	
		Mean	StDev	Mean	StDev	Mean	Tstat
Random Walk	0.00100	0.00105	0.00003	0.00102	0.00003	-0.00003	-0.79
GBM (phi=0.0)		0.00109	0.00003	0.00112	0.00003	0.00004	0.81
GBM (phi=0.1)		0.00111	0.00003	0.00111	0.00003	0.00000	0.06

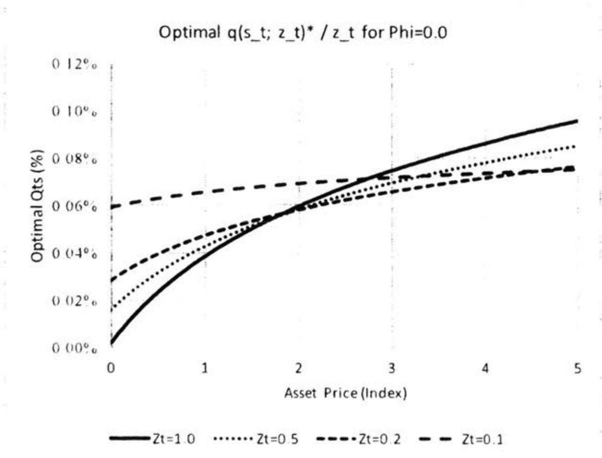
Statistics computed over 1,000 monte carlo paths with 100k trades per path and 10k training epochs
Parameters: sigma=0.2, mu=0.0, N=100,000

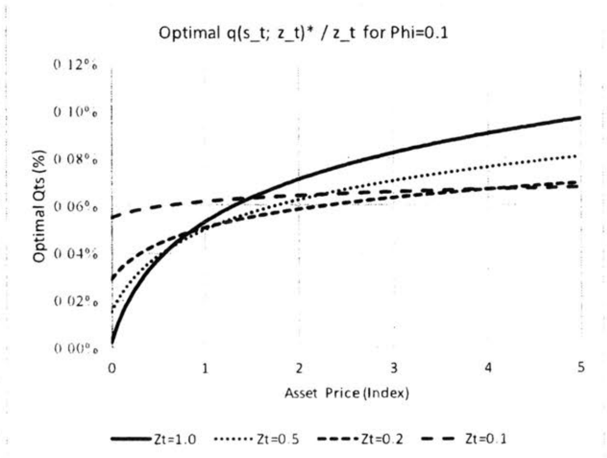
D. *Sensitivity of quantities & optimal cost to parameter assumptions*

Having assessed the general features of the solution to the GBM, linear impact produced by the RL execution algorithm and having verified that the results conform to its traditional DP counterpart, we now consider the sensitivity of the solution to changes in the key parameters of the model, in particular the price impact parameter ϕ and the asset volatility σ. The figure below plots the normalized optimal policy in the initial period (i.e. where $z_t = 1.0$), for different choices of ϕ. As ϕ increases, we see that the surface become more concave in the stock price without translating vertically. These statics reflect the manner in which the price impact term in the SDE interacts with the ZLB on prices to produce concavity in the optimal policy: as the degree to which future trades impact future prices increases, that change is asymmetrically reflected in the policy when prices are low, driving greater concavity in the policy gradient. We note also that the policy is everywhere monotonically increasing in ϕ, but that at the upper limit of prices it approaches an asymptote in ϕ as the state becomes so distant from the ZLB that the latter no longer meaningfully influences the policy.



Inspecting the entire s_t-z_t surface as ϕ changes from 0.0 ro 0.1, we observe an accentuation of several of the key attributes of the policy function. The surface is sharply decreasing in z_t for low price states, reflecting the fact that when prices are low early in execution, the optimal policy is highly incentivized to postpone large trades since it is protected from downside by the ZLB. As ϕ increases, the magnitude of this low price-large z_t gradient relationship also increases. In high price states on the other hand, the optimal policy is positively correlated with z_t, first decreasing as residual shares get executed while prices are high, but then flattening as the terminal period approaches forcing execution of the remaining position. The slope of this relationship also increases when price impact is greater. The further from the ZLB that prices move, the less influential these effects become, and the slope and levels of the surfaces for different parameterizations converge to each other in the limit.





We now turn our attention away from the optimal policy function, toward the economic consequences of executing that policy in the form of average execution cost experienced over the duration of the trading period. In particular, we define:

$$\hat{c}_i(K) = \left(\frac{1}{|K|}\right)\frac{\sum_{t\in K} p_t \; \hat{q}_i(s_t, z_t)^*}{\sum_{t\in K} \hat{q}_i(s_t, z_t)^*}$$

representing the share-weighted average price of all trades executed for time periods $t \in K$, the set of period indices. In the event that $K = [1, T]$, $\hat{c}_i(K)$ is simply the average share price experienced by the trader over the entire trading period. These costs are reported along with their simulated sample moments and test statistics for the $N = 1,000$ path simulation, and for the Bellman grid methodolgy as well as for two versions of the RL algorithm - one initialized with random neural network weights and trained from scratch, the other warm-started via pre training from the solution to the grid method, which is assumed to be computed ex ante (further discussion in Section 3-D).

The table below illustrates that both RL algorithms are able to recover statistically indistinguishable performance from the bellman grid methodology in terms of average trading cost paid across the simulated scenarios. In this instance, the warm started RL algorithm performs marginally better than the randomly initialized variant due to it's slightly improved learning rate based on the pre training procedure, but this benefit is not statistically significant. The table also illustrates how the total trading cost reflects the parameters of the problem: as the impact parameter ϕ increases, the average price achieved by selling the stock over the trading period decreases sharply, reflecting the effect that past sales have on the ability to liquidate future shares at high prices. Similarly, as asset volatility increases (while holding impact fixed), the average selling price achieved also decreases, though not nearly as sharply.

Average Execution Cost*								
		Grid	RL_rand			RL_warm		
				Versus Grid			Versus Grid	
				Diff (%)	Tstat**		Diff (%)	Tstat**
Phi (Fix Sigma=2%)	0.00	1.000	1.000	0.00%	0	1.000	0.00%	0
	0.05	0.957	0.956	-0.10%	-0.17	0.957	-0.01%	-0.03
	0.10	0.922	0.920	-0.20%	-0.31	0.922	-0.01%	-0.02
	0.15	0.886	0.887	0.10%	0.14	0.888	0.14%	0.31
	0.20	0.830	0.828	-0.20%	-0.25	0.833	0.30%	0.59
Sigma (Fix Phi=0.1)	1.0%	0.925	0.926	0.04%	0.36	0.925	0.00%	0.05
	1.5%	0.924	0.924	-0.05%	-0.33	0.924	0.00%	0.04
	2.0%	0.922	0.922	-0.08%	-0.43	0.922	-0.01%	-0.37
	2.5%	0.921	0.919	-0.11%	-0.52	0.920	-0.01%	-0.39
	3.0%	0.915	0.914	-0.10%	-0.45	0.915	0.02%	0.69

* Average execution cost computed over discrete monte carlo paths of 100k trades per path with 10k train epochs

** Simulated standard errors for tstats computed over N=1,000 monte carlo trials

E. RL solution in presence of signals

We now proceed to explore the implications of introducing an informative signal about future price movements, x_t, to the execution model, as formulated in Section II.D. In particular, we assume that the trader observes a realization of a signal in each trading period which is known to be non-trivially correlated with the price process SDE according to a constant correlation coefficient ρ. Consequently, the optimal policy $\hat{q}(s_t, z_t, x_t)^*$ is now a function of the stock price, the residual share count, *and* the realization of the signal today, which contains information about price movements tomorrow (and thereafter, through its "permanent" impact on the price process).

Below, we inspect the normalized optimal policy surface as a function of the signal realization and the stock price in period t, with different residual quantities during the trading period represented in the different panes. We see that the surfaces are again everywhere concave in prices and that they flatten as z_t approaches zero. Furthermore, the optimal policy is everywhere decreasing in the value of the signal. To understand this relationship, we consider the fact that the signal is positively correlated with the next period's return. Therefore, if the signal realization is high indicating the prices will increase in the next period, then the trading algorithm is incentivized to postpone trades today in favor of tomorrow (and the more distant future) when expected prices are higher than they are presently. Conversely, if the signal realization is low indicating a price decrease, the optimal policy will increase shares executed in the period t prior to the disadvantageous price decrease.

With these key features of the relationship between optimal policy and informative signal, we now assess the consequences of introducing a signal to the simulated average execution costs $\hat{c}_i(K)$. The table below reports, for different levels of ρ (i.e. greater or lesser degrees of informational content) the average costs for the bellman grid, which is executed ignoring the informative signal since it is not computationally feasible to solve this problem via the grid methodology for higher dimensional sets of states (further discussion in Section 3-C). Also reported are the average costs for the two RL algorithm variants, both of which do employ information from the signal. We see that for non-zero ρ, the RL algorithms outperform the signal-less grid methodology, and that the degree of outperformance increases as the quality of the signal improves. In terms of economic magnitudes, for a market impact $\phi = 0.1$, and informative signal with correlation $\rho = 0.20$ amounts to a 3.25% improvement in average execution price, whereas for a signal with $\rho = 0.40$, the improvement is over 8%. These performance improvements are strongly statistically significant in the simulated sample.

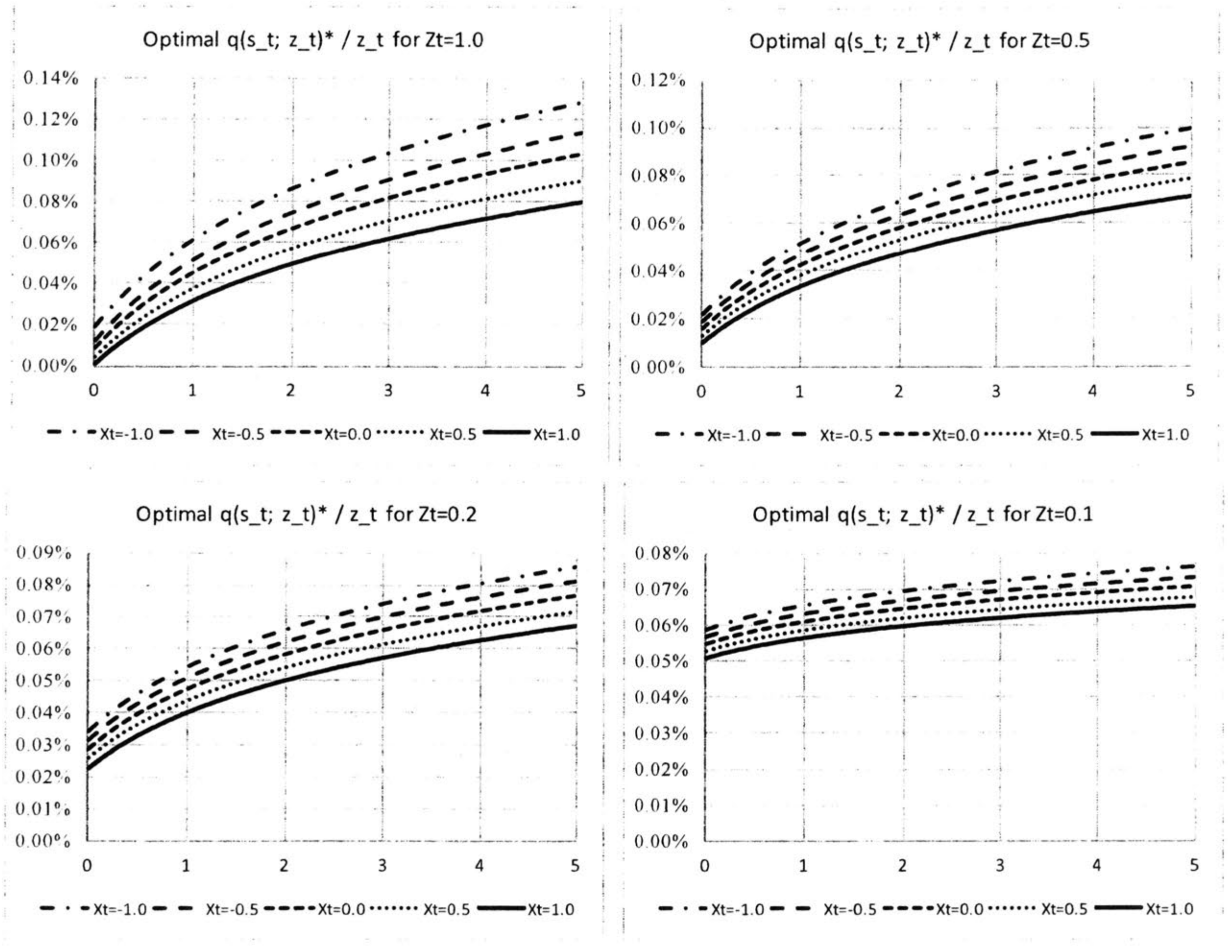


Figure 1. Optimal policies with respect to Signal value X_t for different residual share counts Z_t

A few other observations appear from the comparison of average execution cost. As market impact ϕ increases, the benefit of incorporating information increases. This is also true for increases in asset volatility (see appendix). The intuition underpinning this result is that when market impact is greater, the benefits of information about future price movements becomes more valuable due to the high costs associated with trading "incorrectly" leading to further losses later in the trading period due to the permanent impact of the incorrect trade. Secondarily, for low levels of signal correlation ρ, the warm started version of the RL algorithm outperforms the randomly initialized version because the pre training procedure, which does not take into account the presence of the signal in the model (as it is based on the output of the grid method) is sufficiently "close" to the optimal solution such that it's initialization is preferable to random initialization that then learns the full policy function from scratch. However, as the correlation increases, the warm started variant deviates from the optimal solution sufficiently that it becomes optimal to learn the full policy function for scratch rather than to pre train based on the approximate solution.

Average Execution Cost*								
	Rho (Signal Corr.)	Grid**	RL_rand			RL_warm**		
				Versus Grid			Versus Grid	
				Diff (%)	Tstat***		Diff (%)	Tstat***
Phi=0.10	0.00	0.922	0.921	-0.09%	-0.18	0.923	0.09%	0.31
	0.10	0.922	0.938	1.57%	2.61	0.948	2.58%	7.38
	0.20	0.922	0.955	3.25%	5.08	0.964	4.20%	9.99
	0.30	0.922	0.979	5.74%	8.19	0.980	5.83%	12.96
	0.40	0.922	1.005	8.28%	10.22	0.997	7.50%	14.71
Phi=0.20	0.00	0.833	0.834	0.08%	0.17	0.834	0.08%	0.28
	0.10	0.833	0.852	1.92%	3.20	0.859	2.58%	7.15
	0.20	0.833	0.871	3.79%	5.97	0.879	4.65%	11.07
	0.30	0.833	0.900	6.67%	9.69	0.901	6.76%	15.36
	0.40	0.833	0.931	9.82%	11.96	0.922	8.92%	18.08

* Avg. execution cost computed over discrete monte carlo paths of 100k trades per path with 10k train epochs

** During RL_warm warm start signals are ignored (random signals fed into training iterates); for grid solver, signals are ignored entirely

*** Simulated standard errors for tstats computed over N=1,000 monte carlo trials

Given the structure of the simulation and the ability to track the value of the signal and the subsequent price movements over the duration of each trading epoch, it is interesting to study the behavior or the algorithms conditional on high versus low realizations of the signal. In the table below, each simulated scenario is partitioned into periods in which the signal achieved realizes in each of the five quintiles of it's distribution. Thus, we obtain the collection of index sets $K_\alpha = \{t : x^l \leq x_t < x^u, P[x^l \leq x < x^u] = \alpha\}$ and record the total quantities executed and average execution costs over all $t \in K_\alpha$.

We see that when x_t falls in its lower quintile, the RL algorithm executes a disproportionately large quantity of shares, and receives a statistically significant improvement in execution price relative to the grid method which operates without the benefit of the informative signal. Conversely, in the highest quintile, the RL algorithm defers the quantity of shares to later periods, and receives a lower execution price than it does on average in low-signal states, though still higher than average execution price achieved by the grid methodology.

	Quantiles of Realized Signal (across all paths/periods)*				
	20%	40%	60%	80%	100%
Total Quantity Executed					
Grid**	0.193	0.208	0.200	0.199	0.199
RL_rand	0.214	0.219	0.201	0.189	0.178
Diff	0.021	0.010	0.001	-0.010	-0.022
Tstat	10.94	5.60	0.57	-5.82	-12.32
Avg. Execution Cost					
Grid**	0.921	0.922	0.923	0.923	0.921
RL_rand	0.958	0.957	0.955	0.953	0.949
Diff (%)	3.72%	3.51%	3.25%	3.09%	2.88%
Tstat	37.47	34.56	32.54	30.09	28.81

* Represents the subset of periods in which the signal took a value in a given quantile of it's distribution

** For grid solver, signals are ignored

*** Parameter values: sigma=0.2, mu=0.0, phi=0.1, rho=0.2

IV. Conclusion

This paper offers contributions to several literatures pertaining to the dynamic trading execution problem, incorporation of alternative risk premia or other potentially high-dimensional signals into optimal trading strategies, and the use of reinforcement learning methodologies for applications in financial markets. I propose a reinforcement learning algorithm to deliver high quality and computationally efficient solutions to a class of dynamic trading problems where potential knowledge about the underlying dynamics of asset prices and other informative signals can be exploited in an "online" fashion to generate performance gains with minimum computational burden. The results reported in Section II demonstrate that the algorithm converges with probability one for the relevant class of problems, and points out a useful duality between the fixed point of the Q-learning contraction and the optimal solution to the exact dynamic programming PDE in continuous time.

Section III reports simulated experimental results related to the performance of the algorithm under increasingly complex model settings, first replicating the known analytic solution for the case of random walk prices with linear absolute price impact function, then matching traditional tabular DP solvers for the case of geometric brownian motion prices with linear relative price impact; and finally outperforming the traditional solution methods once informative signals are introduced to the environment, potentially rendering other methods intractable. In this latter context, the strength of correlation between the signal and the target price has a major impact on the quality of the RL algorithm solution relative to the tabular solver, and also impacts the computational characteristics of the algorithm via the types of training procedures that are optimal in different information environments, for example at what point warm-started training or signal compression may become advantageous.

Future work will focus on several key areas of extension including further use of historical microstructure data for model calibration, training, and performance evaluation; experiments related to the computational cost of the algorithm relative to other methods for dynamic trading; incorporation of additional alternative risk premia and signals from the microstructure literature into assessments of the algorithm; and the potential for contemporaneous learning of model parameters "within" the Q-learning training procedure.

REFERENCES

Abadi, Martin et al. TensorFlow: Large Scale Machine Learning on Heterogeneous Distributed Systems (Tensorflow Standard). (2015).

Almgren, Robert. Optimal Trading with Stochastic Liquidity and Volatility, SIAM Journal of Financial Mathematics, 163-188 (2012). Bellman, R. Dynamic Programming, Princeton University. Press, Princeton, 1957.

Almgren, Robert and Neil Chriss. Optimal Execution of Portfolio Transactions, Journal of Risk 5-39 (2001).

Bechler, Kyle and Michael Ludkovski. Optimal Execution with Dynamic Order Flow Imbalance, SIAM Journal of Financial Mathematics (2015).

Baird, Leemon and Andrew Moore. Gradient Descent for General Rinforcement Learning, NIPS (1998).

Benveniste A., M. Metivier, and P. Priouret, Adaptive Algorithms and Stochastic Approximations, Springer-Verlag, 1990.

Bertsimas, Dimitris and Andrew W. Lo. Optimal Control of Execution Costs, Journal of Financial Markets, 1-50 (1998).

Betts, J.T. Practical Methods for Optimal Control and Estimation Using Nonlinear Programming, 2nd edn. SIAM, Philadelphia (2009).

Bhat, Nikhil. Tractable Approximation Algorithms for HIgh Dimensional Sequential Optimization Problems, WP (2016).

Borkar, V.S. and S.P. Meyn. The ODE Method for Convergence of Stochastic Approximation and Reinforcement Learning (1999).

Bouzaiene-Ayari, B, George, Q., Powell, W.B. and Simao, H.P. Approimate Dynamic Programming for High Dimensional Resource Allocation Problems, IEEE Conference Paper (2005).

Brunnermeier, M. and Sannikov, Y. Macro, Money, and Finance: A Continuous-Time Approach, Handbook of Macroeconomics (2016).

Capriotti, Luca. Fast Greeks by Algorithmic Differentiation, Journal of Computational Finance (2011).

Cartea, Alvaro and Sebastian Jaimungai. Incorporating Order Flow into Optimal Execution, Mathematics and Financial Economics (2016).

Drechsler, Itamar, Alan moreira and Alexi Savov. Liquidity Creation as Volatility Risk, WP (2018).

Drechsler, Itamar. Uncertainty, Time-Varying Fear, and Asset Prices, Journal of Finance (2013).

Feng, Yi, Michael Kearns and Nevmyvaka Yuriy. Reinforcement Learning for Optimized Trade Execution, ICML Conference Paper (2006).

Fleming W.H. and T. Pang, An application of stochastic control theory to financial economics, SIAM J. Control Optim. 43 (2004) 502531

Forsyth, P.A., J.S. Kennedy, S.T. Tse, and H. Windcliff. Optimal Trade Eecution: A Mean-Quadratic Variation Approach, Journal of Economic Dynamics and Control (2009).

Garg, Divya, Michael Patterson, Willian W. Hager, Anil Rao, David A. Benson, Geoffrey T. Huntingon. A Unified Framework for the Numerical Solution of Optimal Control Problems Using Pseudospectral Methods, Atomatica (2010).

Gatheral, Jim and Alexander Schied. Optimal Trade Execution under Geometric Brownian Motion in the Almgren and Chriss Framework, Journal of Theoretical and Applied Finance (2011).

Hendricks, Dieter and Diane Wilcox. A Reinforcement Learning Etension to the Almgren-Chriss Framework for Optimal Trade Execution, IEEE Conference Paper (2014).

Jaakkola, Tommi, Michael I. Jordan, and Satinder P. Singh. On the convergence of stochastic iterative dynamic programming algorithms. Neural Computation, 6(6):11851201, 1994.

Kharroubi, Idris and Huyen Pham. Optimal Portfolio Liquidation with Execution Cost and Risk, SIAM Journal of Financial Mathematics (2010).

Lehalle, Charles-Albert and Eyal Neuman. Incorporating Signals into Optimal Trading, Finance and Stochastics 23-2 (2019).

Merton, R.C. Optimal consumption and portfolio rules in continuous time, J. Economic Theory, 3 (1971) 373413.

Meyer, Renate, David Fournier and Andreas Berg. Stochastic Volatility: Bayesian Computation Using Automatic Differentiation and the Extended Kalman Filter, Econometrics Journal (2003).

Obizhaeva, Anna and Jiang Wang. Optimal Trading Strategy and Supply/Demand Dynamics, Journal of Financial Markets 16-1 (2013).

Park, Beomsoo and Benjamin Van Roy. Adaptive Execution: Exploration and Learning of Price Impact, Operations Research (2012).

Powell, Warren B. A Unified Framework for Stochastic Optimization, European Journal of Operational Research (2019).

Wang, Yazhen. Asymptotic Analysis via Stochastic Differential Equations of Gradient Descent Algorithms in Statistical and Computational Paradigms, WP (2018).