DOI:10.1145/1735223.1735247

# Censored Exploration and the Dark Pool Problem

By Kuzman Ganchev, Yuriy Nevmyvaka, Michael Kearns, and Jennifer Wortman Vaughan

**Abstract**

**Dark pools are a recent type of stock exchange in which information about outstanding orders is deliberately hidden in order to minimize the market impact of large-volume trades. The success and proliferation of dark pools have created challenging and interesting problems in algorithmic trading—in particular, the problem of optimizing the allocation of a large trade over multiple competing dark pools. In this work, we formalize this optimization as a problem of multi-venue exploration from censored data, and provide a provably efficient and near-optimal algorithm for its solution. Our algorithm and its analysis have much in common with well-studied algorithms for managing the exploration–exploitation trade-off in reinforcement learning. We also provide an extensive experimental evaluation of our algorithm using dark pool execution data from a large brokerage.**

## 1. INTRODUCTION

*Dark pools* are a relatively new type of exchange designed to address the problems that arise from the transparent (or "light") nature of a typical stock exchange—namely, the difficulty of minimizing the impact of large-volume trades.[3, 5, 7] In a typical exchange, the revelation that there is a large-volume buyer (seller) in the market can cause prices to rise (fall) at the buyer's (seller's) expense. If the volume is sufficiently large, and the trading period sufficiently short, such market impacts remain even if one attempts to fragment the trade over time into smaller transactions. As a result, there has been increasing interest in recent years in execution mechanisms that allow full or partial concealment of large trades.

In a typical dark pool, buyers and sellers submit orders that simply specify the total volume of shares they wish to buy or sell, with the price of the transaction determined exogenously by "the market".[a] Upon submitting an order to buy (or sell) $v$ shares, a trader is put in a queue of buyers (or sellers) awaiting transaction. Matching between buyers and sellers occurs in sequential arrival of orders, similar to a light exchange. However, unlike a light exchange, no information is provided to traders about how many parties or shares might be available in the pool at any given moment. Thus in a given time period, a submission of $v$ shares results only in a report of how many shares up to $v$ were executed.

While presenting their own trading challenges, dark pools have become tremendously popular exchanges, responsible for executing 10–20% of the overall US equity volume. In fact, they have been so successful that there are now approximately 40+ dark pools for the US Equity market alone. The popularity of these exchanges has left large-volume traders and brokerages facing a novel problem: How should one optimally distribute a large trade over the many independent dark pools?

To answer this question, we analyze a framework and algorithm for a more general multi-venue exploration problem. We consider a setting in which at each time period, we have some exogenously determined *volume* of $V$ units of an abstract good (for example, shares of a stock that a client would like to sell). Our goal is to "sell" or "consume" as many of these units as possible at each step, and there are $K$ abstract "venues" (for example, various dark pools) in which this selling or consumption may occur. We can divide our $V$ units into any way we like across the venues in service of this goal. What differentiates this problem from most standard learning settings is that if $v_i$ units are allocated to venue $i$, and all of them are consumed, we learn only that the total demand at venue $i$ was *at least* $v_i$, not the precise number of units that *could* have been consumed there. This important aspect of our framework is known as *censoring* in the statistics literature.

In this work, we make the natural and common assumption that the maximum amount of consumption available in venue $i$ at each time step (or the total liquidity available, in the dark pool problem) is drawn according to a fixed but unknown distribution $P_i$. Formally speaking, this means that when $v_i$ units are submitted to venue $i$, a value $s_i$ is drawn randomly from $P_i$ and the observed (and possibly censored) amount of consumption is $\min\{s_i, v_i\}$.

A learning algorithm in our framework receives a sequence of volumes $V^1, V^2, \ldots$ and must decide how to distribute the $V^t$ units across the venues at each time step $t$. Our goal is to *efficiently* (in time polynomial in the parameters of the model) learn a near-optimal allocation policy. There is a distinct *between-venue exploration* component to this problem, since the best number of shares to submit to venue $i$ may depend on both $V^t$ and the distributions for the *other* venues, and the only mechanism by which we can discover the distributions is by submitting allocations. If we routinely submit too-small volumes to a venue, we receive censored observations and are underutilizing the venue; if we submit

The original version of this paper was published in the *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence,* 2009.

[a] For our purposes, we can think of the price as the midpoint between the bids and ask in the light exchanges, though this is a slight oversimplification.

too-large volumes, we receive uncensored observations but have excess inventory.

Our main theoretical contribution is a provably polynomial-time algorithm for learning a near-optimal policy for any unknown venue distributions $P_i$. This algorithm takes a particularly natural and appealing form, in which *allocation* and distribution *reestimation* are repeatedly alternated. More precisely, at each time step we maintain estimates of the distributions $P_i$; pretending that these estimates are in fact exactly correct, we allocate the current volume $V$ accordingly. These allocations generate observed consumptions in each venue, which in turn are used to update the estimates. We show that when the estimates are "optimistic tail modifications" of the classical *Kaplan–Meier* maximum likelihood estimator for censored data, this estimate–allocate loop has *provably efficient between-venue exploration* behavior that yields the desired result. Venues with smaller available volumes are gradually given smaller allocations in the estimate–allocate loop, whereas venues with repeated censored observations are gradually given larger allocations, eventually settling on a near-optimal overall allocation distribution.

Finally, we present an extensive experimental evaluation of our model and algorithm on the dark pool problem, using trading data from a large brokerage.

The closest problem to our setting is the widely studied *newsvendor problem* from the operations research literature. In this problem, at each time period a player (representing a newsstand owner) chooses a quantity $V$ of newspapers to purchase at a fixed per-unit price, and tries to optimize profit in the face of demand uncertainty at a single venue (their newsstand).[b] Huh et al.[10] were the first to consider the use of the Kaplan–Meier estimator in this class of problems. They use an estimate–allocate loop similar to ours, and show *asymptotic* convergence to near-optimal behavior in a single venue. Managing the distribution of an *exogenously specified* volume $V$ across *multiple* venues (which are the important aspects of the dark pool problem, where the volume to be traded is specified by a client, and there are many dark pools) and the attendant exploration–exploitation trade-off *between venues* are key aspects and differentiators of our algorithm and analysis. We also obtain stronger (polynomial time rather than asymptotic) bounds, which require a modification of the classical Kaplan–Meier estimator.

## 2. THE FORMAL MODEL

Formally, we consider the following problem. At each time $t$, a learner is presented with a quantity or *volume* $V^t \in \{1, ..., V\}$ of units, where $V^t$ is sampled from an unknown distribution $Q$. The learner must decide on an *allocation* $\vec{v}^t$ of these shares to a set of $K$ known *venues*, with $v_i^t \in \{0, ..., V^t\}$ for each $i \in \{1, ..., K\}$, and $\sum_{i=1}^{k} v_i^t = V^t$. The learner is then told the number of units $r_i^t$ *consumed* at each venue $i$. Here $r_i^t = \min\{s_i^t, v_i^t\}$, where $s_i^t$ is the maximum consumption level of venue $i$ at time $t$, which is sampled independently from a fixed but unknown distribution $P_i$. If $r_i^t = v_i^t$, we say that the algorithm receives a *censored observation* because it is possible to infer only that $r_i^t \le s_i^t$. If $r_i^t < v_i^t$, we say that the algorithm receives a *direct observation* because it must be the case that $r_i^t = s_i^t$.

The goal of the learner is to discover a near-optimal one-step allocation policy, that is, an allocation policy that approximately optimizes the expected number of units out of $V^t$ consumed at each time step $t$. (We briefly discuss other objectives at the end of Section 4.4.)

Throughout the remainder of the paper, we use the shorthand $T_i$ for the *tail probabilities* associated with $P_i$. That is, $T_i(s) = \sum_{s' \ge s} P_i(s')$.[c] Clearly $T_i(0) = 1$ for all $i$. We use $\hat{T}_i^t(s)$ for an empirical estimate of $T_i(s)$ at time $t$.

## 3. A GREEDY ALLOCATION SCHEME

Before tackling the full exploration–exploitation problem, we must examine a more basic question: Given estimates $\hat{T}_i$ of the tail probabilities $T_i$ for each venue $i$, how can we maximize the (estimated) expected number of units consumed on a single time step? It turns out that this can be accomplished using a simple greedy allocation scheme. The greedy algorithm allocates one unit at a time. The venue to which the next unit is allocated is chosen to maximize the estimated probability that the unit will be consumed. It is easy to see that if $v_i$ units have already been allocated to venue $i$, then the estimated probability that the next allocated unit will be consumed is simply $\hat{T}_i(v_i + 1)$. A formal description of the Greedy algorithm is given in Figure 1.

THEOREM 1. *The allocation returned by* Greedy *maximizes the expected number of units consumed in a single time step, where the expectation is taken with respect to the estimated tail probabilities* $\{\hat{T}_i\}_{i=1}^{K}$.

The proof of this theorem is fairly simple. Using the fact that tail probabilities must satisfy $\hat{T}_i(s) \ge \hat{T}_i(s')$ for all $s \le s'$, it is easy to verify that by greedily adding units to the venues in decreasing order of $\hat{T}_i(s)$, the algorithm returns

$$\arg\max_{\bar{v}} \sum_{i=1}^{K} \sum_{s=1}^{v_i} \hat{T}_i(s) \qquad \text{s.t.} \sum_{i=1}^{N} v_i = V.$$

The remainder of the proof involves showing that the expression being maximized here equivalent to the expected number of units consumed. This can be done algebraically.[d]

## 4. THE CENSORED EXPLORATION–EXPLOITATION ALGORITHM

We now present our main theoretical result, which is a

[b] In our setting, it is important that we view $V$ as given exogenously by the client and not under the trader's control, which distinguishes our setting somewhat from the prior works.

[c] In the early literature on censored estimation, these tail probabilities were referred to as *survival probabilities*, as $T(s)$ usually represented the probability that a patient in a particular medical study survived for at least $s$ years past the start of the study. In this setting, observations were frequently censored when researchers lost track of a patient midway through the study and knew only that the patient lived *at least* until the point at which contact was broken.[1]

[d] The curious reader can find more details of this and other omitted proofs in the original version of this paper.[9]

Figure 1. Optimal allocation algorithm *Greedy*.

```
Input: Volume $V$, tail probability estimates $[\hat{T}_i]_{i=1}^K$
Output: An allocation $\vec{v}$
$\vec{v} \leftarrow \vec{0}$;
for $\ell \leftarrow 1$ to $V$ do
    $i \leftarrow \text{argmax}_i\ \hat{T}_i(v_i + 1)$;
    $v_i \leftarrow v_i + 1$;
end
return $\vec{v}$
```

polynomial-time, near-optimal algorithm for multi-venue exploration from censored data. The analysis of our algorithm bears strong resemblance to the exploration–exploitation arguments common in the $E^3$ and RMAX family of algorithms for reinforcement learning.[4, 12] In particular, there is an analogy to the notion of a *known state* inherent in those earlier algorithms, along with an *exploitation lemma* (proving that expected payoffs from known states are high) and an *exploration lemma* (proving that extended periods of low payoffs must result in more states becoming known). In our setting, however, the number of states is exponential and thus the special structure of our problem is required to obtain a polynomial time algorithm. We first provide an overview of the algorithm and its analysis before examining it in more detail.

At the highest level, the algorithm is quite simple and natural. It maintains estimates $\hat{T}_i^t$ for the true unknown tail probabilities $T_i$ for each venue $i$. These estimates improve with time in a particular quantifiable sense which drives between-venue exploration. At any given time $t$, the current volume $V^t$ is allocated across the venues by simply calling the optimal greedy allocation scheme from Figure 1 on the current set of estimated tail probabilities $\hat{T}_i^t$. This results in new censored observations from each venue, which in turn are used to update the estimates $\hat{T}_i^{t+1}$ used at the next time step. Thus the algorithm, which is formally stated in Figure 2, implements a continuous allocate–reestimate loop.

Note that we have not yet described the algorithm's subroutine *OptimisticKM*, which specifies how we estimate $\hat{T}_i^t$ from the observed data. The most natural choice would be the maximum likelihood estimator on the data. This estimator is well-known in the statistics literature as the *Kaplan–Meier* estimator. In the following section, we describe Kaplan–Meier and derive a new convergence result that suits our particular needs. This result in turn lets us define an optimistic tail modification of Kaplan–Meier that becomes our choice for *OptimisticKM*. Figure 3 shows the full subroutine.

The analysis of our algorithm, which is developed in more detail over the next few sections, proceeds as follows:

**Step 1:** We first review the Kaplan–Meier maximum likelihood estimator for censored data and provide a new finite sample convergence bound for this estimator. This bound allows us to define a *cut-off* for each venue $i$ such that the Kaplan–Meier estimate of the tail probability $T_i(s)$ for every value of $s$ up to the cut-off is guaranteed to be close to the true tail probability. We then define a lightly modified version of the Kaplan–Meier estimates in which the tail probability of the next unit above the cut-off is modified in an optimistic manner. We show that in conjunction with the greedy allocation scheme, this minor modification leads to increased exploration, since the next unit beyond the cut-off always looks at least as good as the cut-off itself.

**Step 2:** We next prove our main *Exploitation Lemma* (Lemma 3). This lemma shows that at any time step, if it is the case that the number of units allocated to each venue by the greedy algorithm is strictly below the cut-off for that venue (which can be thought of as being in a *known state* in the parlance of reinforcement learning) then the allocation is provably $\varepsilon$-optimal.

**Step 3:** We then prove our main *Exploration Lemma* (Lemma 4), which shows that on any time step at which the allocation made by the greedy algorithm is *not* $\varepsilon$-optimal, it is possible to lower bound the probability that the algorithm explores. Thus, any time we cannot ensure a near-optimal allocation, we are instead assured of exploring.

**Step 4:** Finally, we show that on any sufficiently long sequence of time steps (where *sufficiently long* is polynomial in the parameters of the model), it must be the case that either the algorithm has already implemented a near-

Figure 2. Main algorithm.

```
Input: Volume sequence $V^1, V^2, V^3, \ldots$
Arbitrarily initialize $\hat{T}_i^1$ for each i;
for $t \leftarrow 1, 2, 3, \ldots$ do
    % Allocation Step:
    $\vec{v}^t \leftarrow Greedy\ (V^t, \hat{T}_1^t, \ldots, \hat{T}_K^t)$;
    for $i \leftarrow \{1, \ldots, K\}$ do
        Submit $V_i^t$ units to venue $i$;
        Let $r_i^t$ be the number of shares sold;
        % Reestimation Step:
        $\hat{T}_i^{t+1} \leftarrow OptimisticKM\ (\{(v_i^\tau, r_i^\tau)\}_{\tau=1}^t)$;
    end
end
```

Figure 3. Subroutine *OptimisticKM*. Let $M_{i,s'}^t$ and $N_{i,s'}^t$ be defined in Section 4.1, and assume that $\varepsilon$, $\delta > 0$ are fixed parameters.

```
Input: Observed data $(\{(v_i^\tau, r_i^\tau)\}_{\tau=1}^t)$ for venue $i$
Output: Modified Kaplan–Meier estimators for $i$
% Calculate the cut-off:
$c_i^t \leftarrow \max\{s : s = 0 \text{ or } N_{i,s-1}^t \geq 128\ (sV/\varepsilon)^2 \ln(2V/\delta)\}$;
% Compute Kaplan–Meier tail probabilities:
$\hat{T}_i^t(0) = 1$;
for $s = 1$ to $V$ do
    $\hat{T}_i^t(s) \leftarrow \prod_{s'=0}^{s-1} (1-(M_{i,s'}^t / N_{i,s'}^t))$;
end
% Make the optimistic modification:
if $c_i^t < V$ then
    $\hat{T}_i^t(c_i^t + 1) \leftarrow \hat{T}_i^t(c_i^t)$;
return $\hat{T}_i^t$;
```

optimal solution at almost every time step (and thus will continue to perform well in the future), or the algorithm has explored sufficiently often to learn accurate estimates of the tail distributions out to $V$ units on every venue. In either case, we can show that with high probability, at the end of the sequence, the current algorithm achieves an $\varepsilon$-optimal solution at each time step with probability at least $1 - \varepsilon$.

### 4.1. Convergence of Kaplan–Meier estimators

We begin by describing the standard Kaplan–Meier maximum likelihood estimator for censored data,[11, 13] restricting our attention to a single venue $i$. Let $z_{i,s}$ be the true probability that the demand in this venue is *exactly* $s$ units given that the demand is *at least* $s$ units. Formally,

$$z_{i,s} = \frac{T_i(s) - T_i(s+1)}{T_i(s)} = 1 - \frac{T_i(s+1)}{T_i(s)}.$$

It is easy to verify that for any $s > 0$,

$$T_i(s) = \prod_{s'=0}^{s-1} \frac{T_i(s'+1)}{T_i(s')} = \prod_{s'=0}^{s-1} (1 - z_{i,s'}).$$

At a high level, we can think of Kaplan–Meier as first computing a separate estimate of $z_{i,s}$ for each $s$ and then using these estimates to compute an estimate of $T_i(s)$.

More specifically, let $M^t_{i,s}$ be the number of *direct* observations of $s$ units up to time $t$, that is, the number of time steps at which strictly more than $s$ units were allocated to venue $i$ and exactly $s$ were consumed. Let $N^t_{i,s}$ be the number of either direct or censored observations of *at least* $s$ units on time steps at which strictly more than $s$ units were allocated to venue $i$. We can then naturally define our estimate $\hat{z}^t_{i,s} = M^t_{i,s}/N^t_{i,s}$, with $\hat{z}^t_{i,s} = 0$ if $N^t_{i,s} = 0$. The Kaplan–Meier estimator of the tail probability for any $s > 0$ after $t$ time steps can then be expressed as

$$\hat{T}^t_i(s) = \prod_{s'=0}^{s-1} (1 - \hat{z}^t_{i,s'}), \tag{1}$$

with $\hat{T}^t_i(0) = T_i(0) = 1$ for all $t$.

Previous work has established convergence rates for the Kaplan–Meier estimator to the true underlying distribution in the case that each submission in the sequence $v^1_i, \ldots, v^t_i$ is independently and identically distributed (i.i.d.),[8] and asymptotic convergence for non-i.i.d. settings.[10] We are not in the i.i.d. case, since the submitted volumes at one venue are a function of the entire history of allocations and executions across all venues. In the following theorem, we give a new finite sample convergence bound applicable to our setting.

Theorem 2. *Let $\hat{T}^t_i$ be the Kaplan–Meier estimate of $T_i$ as given in Equation 1. For any $\delta > 0$, with probability at least $1 - \delta$, for every $s \in \{1, \ldots, V\}$,*

$$\left| T_i(s) - \hat{T}^t_i(s) \right| \le s\sqrt{2 \ln(2V/\delta)/N^t_{i,s-1}}.$$

This result shows that as we make more and more direct or censored observations of at least $s - 1$ units on time steps at which at least $s$ units are allocated to venue $i$, our estimate of the tail probability for $s$ shares rapidly improves.

To prove this theorem, we must first show that the estimates $\hat{z}^t_{i,s}$ converge to the true probabilities $z_{i,s}$. In an i.i.d. setting, this could be accomplished easily using standard concentration results such as Hoeffding's inequality. In our setting, we instead appeal to Azuma's inequality (see, for example, Alon and Spencer[2]), a tool for bounding *martingales*, or sequences $X_1, X_2, \ldots$ such that for each $n$, $|X_n - X_{n+1}| \le 1$ and $E\,[X_{n+1}|X_n] = X_n$. In particular, we show that the value $N^t_{i,s}(z_{i,s} - \hat{z}^t_{i,s})$ can be expressed as the final term of a martingale sequence, allowing us to bound its absolute value. This in turn implies that bound on $|z_{i,s} - \hat{z}^t_{i,s}|$ that we need, and all that remains is to show that these bounds imply a bound on the discrepancy between $T_i(s)$ and the estimator $\hat{T}_i(s)$.

### 4.2. Modifying Kaplan–Meier

In Figure 3, we describe the minor modification of Kaplan–Meier necessary for our analysis. As described above (Step 1), the value $c^t_i$ in this algorithm can intuitively be viewed as a cut-off up to which we are guaranteed to have sufficient data to accurately estimate the tail probabilities using Kaplan–Meier; this is formalized in Lemma 1. Thus for every quantity $s < c^t_i$, we simply let $\hat{T}^t_i(s)$ be precisely the Kaplan–Meier estimate as in Equation 1.

However, to promote exploration, we set the value of $\hat{T}^t_i(c^t_i + 1)$ optimistically to the Kaplan–Meier estimate of the tail probability at $c^t_i$ (*not* at $c^t_i + 1$). This optimistic modification is necessary to ensure that the greedy algorithm explores (i.e., has a chance of making progress towards increasing at least one cut-off value) on every time step for which it is not already producing an $\varepsilon$-optimal allocation. In particular, suppose that the current greedy solution allocated no more than $c^t_i$ units to any venue $i$ and exactly $c^t_j$ units to some venue $j$. Using the standard Kaplan–Meier tail probability estimates, it could be the case that this allocation is suboptimal (there is no way to know if it would have been better to include unit $c^t_i + 1$ from venue $j$ in place of a unit from another venue since we do not have an accurate estimate of the tail probability for this unit), and yet no exploration is taking place. By optimistically modifying the tail probability $\hat{T}^t_i(c^t_j + 1)$ for each venue, we ensure that no venue remains unexplored simply because the algorithm unluckily observes a low demand a small number of times.

We now formalize the idea of $c^t_i$ as a cut-off up to which the Kaplan–Meier estimates are accurate. In the results that follow, we think of $\varepsilon > 0$ and $\delta > 0$ as fixed parameters of the algorithm.[e]

Lemma 1. *For any $s \le V$, let $\hat{T}^t_i(s)$ be the Kaplan–Meier estimator for $T_i(s)$ returned by* OptimisticKM. *With probability at least $1 - \delta$, for all $s \le c^t_i$, $|T_i(s) - \hat{T}^t_i(s)| \le \varepsilon/(8V)$.*

Proof. It is always the case that $T_i(0) = \hat{T}^t_i(0) = 1$, so the result

[e] In particular, ε corresponds to the value ε specified in Theorem 3, and $\delta$ corresponds roughly to that $\delta$ divided by the polynomial upper bound on time steps.

holds trivially unless $c_i^t > 0$. Suppose this is the case. Recall that $N_{i,s}^t$ is the number of direct or censored observations of at least $s$ units on time steps at which strictly more than $s$ units were allocated to venue $i$. By definition, it must be the case that $N_{i,s}^t \geq N_{i,s'}^t$, whenever $s \leq s'$. Thus by definition of the cut-off $c_i^t$ in Figure 3, for all $s < c_i^t$, $N_{i,s}^t \geq 128(sV/\varepsilon)^2 \ln(2V/\varepsilon)$. The lemma then follows immediately from an application of Theorem 2. □

Lemma 2 shows that it is also possible to achieve additive bounds on the error of tail probability estimates for quantities $s$ much *bigger* than $c_i^t$ as long as the estimated tail probability at $c_i^t$ is sufficiently small. Intuitively, this is because the tail probability at these large values of $s$ must be smaller than the true tail probability at $c_i^t$, which, in this case, is known to be very small already.

LEMMA 2. *If* $\hat{T}_i^t(c_i^t) \leq \varepsilon/(4V)$ *and the high probability event in Lemma 1 holds, then for all s such that* $c_i^t < s \leq V$, $|T_i(s) - \hat{T}_i^t(s)| \leq \varepsilon/(2V)$.

### 4.3. Exploitation and exploration lemmas

We are now ready to state our main *Exploitation Lemma* (Step 2), which formalizes the idea that once a sufficient amount of exploration has occurred, the allocation output by the greedy algorithm is $\varepsilon$-optimal. The proof of this lemma is where the optimistic tail modification to the Kaplan–Meier estimator becomes important. In particular, because of the optimistic setting of $\hat{T}_i^t(c_i^t + 1)$, we know that if the greedy policy allocates exactly $c_i^t$ units to a venue $i$, it could not gain too much by reallocating additional units from another venue to venue $i$ instead. In this sense, we create a buffer above each cut-off, guaranteeing that it is not necessary to continue exploring as long as one of the two conditions in the lemma statement is met for each venue.

The second condition in the lemma may appear mysterious at first. To see why it is necessary, notice that the rate at which the estimate $\hat{T}_i^t(c_i^t + 1)$ converges to the true tail probability $T_i(c_i^t + 1)$ implied by Theorem 2 depends on the number of times that we observe a consumption of $c_i^t$ or more units. If $T_i(c_i^t)$ is very small, then the consumption of this many units does not frequently occur. Luckily, if this is the case, then we know that $T_i(c_i^t + 1)$ must be very small as well, and more exploration of this venue is not needed.

LEMMA 3 (EXPLOITATION LEMMA). *Assume that at time t, the high probability event in Lemma 1 holds. If for each venue i, either (1),* $v_i^t \leq c_i^t$ *or (2),* $\hat{T}_i^t(c_i^t) \leq \varepsilon/(4V)$, *the difference between the expected number of units consumed under allocation* $\vec{v}^t$ *and the expected number of units consumed under the optimal allocation is at most* $\varepsilon$.

PROOF SKETCH. The proof begins by creating an arbitrary one-to-one mapping between the units allocated to different venues by the algorithm and an optimal allocation. Consider any such pair in this mapping.

If the first condition in the lemma holds for the venue $i$ to which the unit was allocated by the algorithm, we can use Lemma 1 to show that the algorithm's estimate of the probability of this unit being consumed is close to the true probability; in particular, the algorithm is not overestimating this probability too much. If the second condition holds, then the algorithm's estimate of the probability of the share being consumed is so small that, again, the algorithm cannot possibly be overestimating it too much (because the lowest the probability could be is zero). This follows from Lemma 2.

Now consider the venue $j$ to which unit was allocated by the optimal allocation. If the number of units $v_j^t$ allocated to this venue by the algorithm is strictly less than the cut-off $c_j^t$, then by Lemma 1, the algorithm could not have underestimated the probability of additional units being consumed by too much. Furthermore, because of the optimistic tail modification of the Kaplan–Meier estimator, this also holds if $v_j^t = c_j^t$. Finally, if it is instead the case that the second condition in the lemma statement holds for venue $j$, then the algorithm again could not possibly have underestimated the probability of the unit being consumed too much because the true probability is so low.

Putting these pieces together, we can argue that for each pair in the matching (of which there are no more than $V$), since the algorithm did not overestimate the probability of unit it chose being consumed by too much (in this case, *too much* means more than $\varepsilon/(2V)$) and did not underestimate the probability of the corresponding unit in the optimal allocation by too much (again, by $\varepsilon/(2V)$), the difference in expected units consumed between the optimal allocation and the algorithm's is at most $\varepsilon$. □

Finally, Lemma 4 presents the main exploration lemma (Step 3), which states that on any time step at which the allocation is *not* $\varepsilon$-optimal, the probability of obtaining a useful observation is at least $\varepsilon/(8V)$.

LEMMA 4 (EXPLORATION LEMMA). *Assume that at time t, the high probability event in Lemma 1 holds. If the allocation is not* $\varepsilon$*-optimal, then for some venue i, with probability at least* $\varepsilon/(8V)$, $N_{i,c_i^t}^{t+1} = N_{i,c_i^t}^t + 1$.

PROOF. Suppose the allocation is not $\varepsilon$-optimal at time $t$. By Lemma 3, it must be the case that there exists some venue $i$ for which $v_i^t > c_i^t$ and $\hat{T}_i^t(c_i^t) > \varepsilon/(4V)$, i.e., a venue in which the algorithm has allocated units past the cut-off but for which the tail probability at the cut-off is not too close to zero. Let  be a venue for which this is true. Since $v_\ell^t > c_\ell^t$, it will be the case that the algorithm obtains a useful observation for exploration of this venue (i.e., an observation causing $N_{\ell,c_\ell^t}^t$ to be incremented) if the number of units consumed at this venue is sufficiently high (specifically, if $r_\ell^t > c_\ell^t$). Since $\hat{T}_\ell^t(c_\ell^t) > \varepsilon/(4V)$, Lemma 1 implies that $T_\ell(c_\ell^t) > \varepsilon/(8V)$, which in turn implies that the number of units consumed is high enough to constitute a useful observation with probability at least $\varepsilon/(8V)$. □

### 4.4. Putting it all together

With the exploitation and exploration lemmas in place, we are finally ready to state our main theorem.

THEOREM 3 (MAIN THEOREM). *For any* $\varepsilon > 0$ *and* $\delta > 0$, *with probability* $1 - \delta$ *(over the randomness of draws from Q and* $\{P_i\}$*), after running for a time polynomial in K, V,* $1/\varepsilon$, *and* $\ln(1/\delta)$, *the algorithm in Figure 2 makes an* $\varepsilon$*-optimal allocation* *on each subsequent time step with probability at least* $1 - \varepsilon$.

Proof Sketch. Suppose that the algorithm runs for $R$ time steps, where $R$ is a (specific, but unspecified for now) polynomial in the model parameters $K$, $V$, $1/\varepsilon$, and $\ln(1/\delta)$. If it is the case that the algorithm was already $\varepsilon$-optimal on a fraction $(1 - \varepsilon)$ of the $R$ time steps, then we can argue that the algorithm will continue to be $\varepsilon$-optimal on at least a fraction $(1 - \varepsilon)$ of future time steps since the algorithm's performance should improve on average over time as estimates become more accurate.

On the other hand, if the algorithm chose sub-optimal allocations on at least a fraction $\varepsilon$ of the $R$ time steps, then by Lemma 4, the algorithm must have incremented $N^t_{i,c^t_i}$ for some venue $i$ and cut-off $c^t_i$ approximately $\varepsilon^2R/(8V)$ times. By definition of the $c^t_i$, it can never be the case that $N^t_{i,c^t_i}$was incremented too many times for any *fixed* values of $i$ and $c^t_i$ (where *too many* is a polynomial in $V$, $1/\varepsilon$, and $\ln(1/\delta)$); otherwise the cut-off would have increased. Since there are only $K$ venues and $V$ possible cut-off values to consider in each venue, the total number of increments can be no more than $KV$ times this polynomial, another polynomial in $V$, $1/\varepsilon$, $\ln(1/\delta)$, and now $K$. If $R$ is sufficiently large (but still polynomial in all of the desired quantities) and approximately $\varepsilon^2$ $R/(8V)$ increments were made, we can argue that *every* venue must have been fully explored, in which case, again, future allocations will be $\varepsilon$-optimal. □

We remark that our optimistic tail modifications of the Kaplan–Meier estimators are relatively mild. This leads us to believe that using the same estimate–allocate loop with an *unmodified* Kaplan–Meier estimator would frequently work well in practice. We investigate a parametric version of this learning algorithm in the experiments described below.

## 5. THE DARK POOL PROBLEM

The remainder of this article is devoted to the application of our techniques to the dark pool problem. We begin with a description of the trading data we used, and go on to describe a variety of experiments we performed.

### 5.1. Summary of the dark pool data

Our data set is from the internal dark pool order flow for a major US broker–dealer. Each (possibly censored) observation is of the form discussed throughout the paper—a triple consisting of the dark pool name, the number of shares sent to that pool, and the number of shares subsequently executed within a short time interval. It is important to highlight some limitations of the data. First, note that the data set conflates the policy the brokerage used for allocation across the dark pools with the liquidity available in the pools themselves. For our data set, the policy in force was very similar to the bandit-style approach we discuss below. Second, the "parent" orders determining the overall volumes to be allocated across the pools were determined by the brokerage's trading needs, and are similarly out of our control.

The data set contains submissions and executions for four active dark pools: BIDS Trading, Automated Trading Desk, D.E. Shaw, and NYFIX, each for a dozen of relatively actively-traded stocks,[f] thus yielding 48 distinct stock–pool data sets. The average daily trading volume of these stocks across all exchanges (light and dark) ranges from 1 to 60 million shares, with a median volume of 15 million shares. Energy, Financials, Consumer, Industrials, and Utilities industries are represented. Our data set spans 30 trading days. For every stock–pool pair we have on average 1,200 orders (from 600 to 2,000), which corresponds to 1.3 million shares (from 0.5 to 3 million). Individual order sizes range from 100 to 50,000 shares, with 1,000 shares being the median. Sixteen percent of orders are filled at least partially (meaning that fully 84% result in no shares executed), 9% of the total submitted volume was executed, and 11% of all observations were censored.

### 5.2. Parametric models for dark pools

The theory and algorithm we have developed for censored exploration permit a very general form for the venue distributions $P_i$. The downside of this generality is that we are left with the problem of learning a very large number of parameters. More parameters generally mean that more data is necessary to guarantee that the model will generalize well, which means more rounds of exploration are needed before the algorithm's future performance is near-optimal. In some applications, it is therefore advantageous to employ a less general but more simple parametric form for these distributions.

We experimented with a variety of common parametric forms for the distributions. For each such form, the basic methodology was the same. For each of the $4 \times 12 = 48$ venue–stock pairs, the data for that pair was split evenly into a training set and a test set. The training data was used to select the maximum likelihood model from the parametric class. Note that we can no longer directly apply the nonparametric Kaplan–Meier estimator—within each model class, we must directly maximize the likelihood on the censored training data. This is a relatively straightforward and efficient computation for each of the model classes we investigated. The test set was then used to measure the generalization performance of each maximum likelihood model.

Our investigations revealed that the best models maintained a separate parameter for the probability of zero shares being available (that is, $P_i(0)$ is explicitly estimated)—a *zero bin* or ZB parameter. This is due to the fact that the vast majority of submissions (84%) to dark pools result in no shares being executed. We then examined various parametric forms for the nonzero portions of the venue distributions, including uniform (which of course requires no additional parameters), and Poisson, exponential and power law forms (each of which requires a single additional parameter); each of these forms were applied up to the largest volume submitted in the data sets, then normalized.

The generalization results strongly favor the power law form, in which the probability of $s$ shares being available is proportional to $1/s^{\beta}$ for real $\beta$—a so-called heavy-tailed

[f] Tickers represented are AIG, ALO, CMI, CVX, FRE, HAL, JPM, MER, MIR, NOV, XOM, and NRG.

**Table 1. Average per-sample log-loss (negative log likelihood) for each venue distribution models. The "Wins" column shows the number of stock-venue pairs where a given model beats the other four on the test data.**

| Model | Train Loss | Test Loss | Wins |
|---|---|---|---|
| Nonparametric | 0.454 | 0.872 | 3 |
| ZB + Uniform | 0.499 | 0.508 | 12 |
| ZB + Power Law | 0.467 | 0.484 | 28 |
| ZB + Poisson | 0.576 | 0.661 | 0 |
| ZB + Exponential | 0.883 | 0.953 | 5 |

distribution when $\beta > 0$. Nonparametric models trained with Kaplan–Meier are best on the training data but overfit badly due to their complexity relative to the sparse data, while the other parametric forms cannot accommodate the heavy tails of the data. This is summarized in Table 1. Based on this comparison, for our dark pool study we investigate a variant of our main algorithm, in which the estimate–allocate loop has an estimation step using maximum likelihood estimation within the ZB + Power Law model, and allocations are done greedily on these same models.

In terms of the estimated ZB + Power Law parameters themselves, we note that for all 48 stock–pool pairs the Zero Bin parameter accounted for most of the distribution (between a fraction 0.67 and 0.96), which is not surprising considering the aforementioned preponderance of entirely unfilled orders in the data. The vast majority of the 48 exponents $\beta$ fell between $\beta = 0.25$ and $\beta = 1.3$—so rather long tails indeed—but it is noteworthy that for one of the four dark pools, 7 of the 12 estimated exponents were actually *negative*, yielding a model that predicts *higher* probabilities for larger volumes. This is likely an artifact of our size- and time-limited data set, but is not entirely unrealistic and results in some interesting behavior in the simulations.

## 5.3. Data-based simulation results

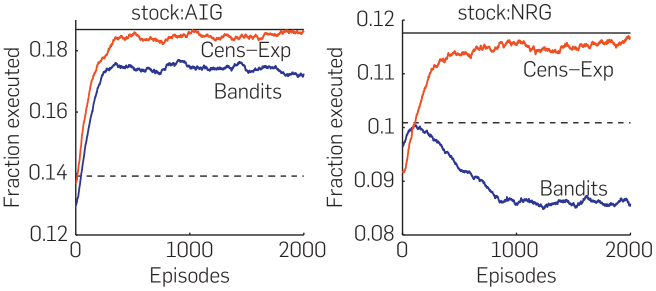
As in any control problem, the dark pool data in our possession is unfortunately insufficient to evaluate and compare different allocation algorithms. This is because of the aforementioned fact that the volumes submitted to each venue were fixed by the specific policy that generated the data, and we cannot explore alternative choices—if our algorithm chooses to submit 1000 shares to some venue, but in the data only 500 shares were submitted, we simply cannot infer the outcome of our desired submission.

We thus instead use the raw data to derive a *simulator* with which we can evaluate different approaches. In light of the modeling results of Section 5.2, the simulator for stock $S$ was constructed as follows. For each dark pool $i$, we used *all* of the data for $i$ and stock $S$ to estimate the maximum likelihood Zero Bin + Power Law distribution. (Note that there is no need for a training-test split here, as we have already separately validated the choice of distributional model.) This results in a set of four venue distribution models $P_i$ that form the simulator for stock $S$. This simulator accepts allocation vectors $(v_1, v_2, v_3, v_4)$ indicating how many shares some algorithm wishes to submit to each venue, draws a "true liquidity" value $s_i$ from $P_i$ for each $i$, and returns the vector $(r_1, r_2, r_3, r_4)$, where $r_i = \min(v_i, s_i)$ is the possibly censored number of shares filled in venue $i$.

Across all 12 stocks, we compared the performance of four different allocation algorithms. The (obviously unrealistic) *ideal allocation* is given the *true parameters* of the ZB + Power Law distributions used by the simulator and allocates shares optimally (greedily) with respect to these distributions. The *uniform allocation* divides any order equally among all four venues. Our *learning algorithm* implements the repeated allocate–reestimate loop as in Figure 2, using the maximum likelihood ZB + Power Law model for the reestimation step. Finally, the simple (and fairly naive) *bandit-style algorithm* maintains a weighting over the venues and chooses allocations proportional to the weights. It begins with equal weights assigned to all venues, and each allocation to a venue which results in any nonzero number of shares being executed causes that venue's weight to be multiplied by a constant factor $\alpha$. (Optimizing $\alpha$ over all stock–pool pairs resulted in a value of $\alpha = 1.05$.)

Some remarks on these algorithms are in order. First, note that the ideal and uniform allocation methods are nonadaptive and are meant to serve as baselines—one of them the best performance we could hope for (ideal), and the other the most naive allocation possible (uniform). Second, note that our algorithm has a distinct advantage in the sense that it is using the correct parametric form, the same being used by the simulator itself. Thus our evaluation of this algorithm is certainly optimistic compared to what should be expected in practice. Finally, note that the bandit algorithm is the crudest type of weight-based allocation scheme of the type that abounds in the no-regret literature[6]; we are effectively forcing our problem into a 0/1 loss setting corresponding to "no shares" and "some shares" being executed. Certainly more sophisticated bandit-style approaches can and should be examined.

**Figure 4. Sample learning curves. For the stock AIG (left panel), the naive bandits algorithm (labeled blue curve) beats uniform allocation (dashed horizontal line) but appears to asymptote short of ideal (solid horizontal line). For the stock NRG (right panel), the bandits algorithm actually deteriorates with more episodes, underperforming both the uniform and ideal allocations. For both stocks (and the other 10 in our data set), our algorithm (labeled red curve) performs nearly optimally.**



Each algorithm was run in simulation for some number of *episodes*. Each episode consisted of the allocation of a fixed number $V$ of shares—thus the same number of shares is repeatedly allocated by the algorithm, though of course this allocation will change over time for the two adaptive algorithms as they learn. Each episode of simulation results in some fraction of the $V$ shares being executed. Two values of $V$ were investigated—a smaller value $V = 1000$, and the larger and potentially more difficult $V = 8000$.

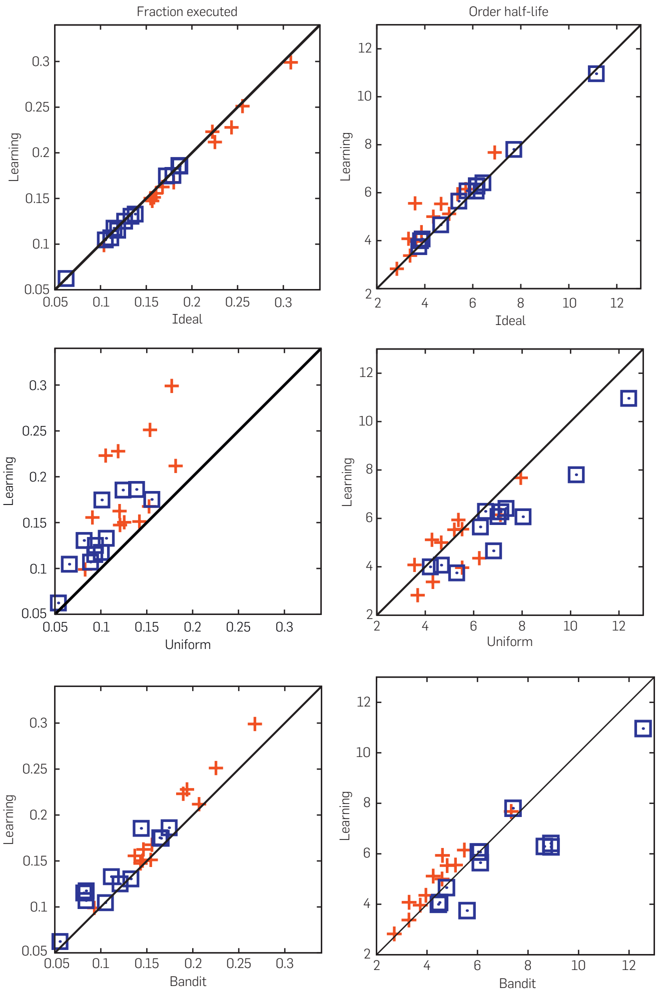
We begin by showing full learning curves over 2000 episodes with $V = 8000$ for a couple of representative stocks in Figure 4. Here the average performance of the two non-adaptive allocation schemes (ideal and uniform) are represented as horizontal lines, while learning curves are given for the adaptive schemes. Due to high variance of the heavy-tailed venue distributions used by the simulator, a single trial of 2000 episodes is extremely noisy, so we both average over 400 trials for each algorithm, and smooth the resulting averaged learning curve with a standard exponential decay temporal moving average.

We see that our learning algorithm converges towards the ideal allocation (as suggested by the theory), often relatively quickly. Furthermore, in each case this ideal asymptote is significantly better than the uniform allocation strawman, meaning that optimal allocations are highly nonuniform. Learning curves for the bandit approach exhibit one of the three general behaviors over the set of 12 stocks. In some cases, the bandit approach is quite competitive with our algorithm, though converging to ideal perhaps slightly slower (not shown in Figure 4). In other cases, the bandit approach learns to outperform uniform allocation but appears to asymptote short of the ideal allocation. Finally, in some cases the bandit approach appears to actually “learn the wrong thing”, with performance decaying significantly with more episodes. This happens when one venue has a very heavy tail, but also a relatively high probability of executing zero shares, and occurs because the very naive bandit approach that we use does not have an explicit representation of the tails of the distribution.

The left column of Figure 5 shows more systematic head-to-head comparisons of our algorithm’s performance versus the other allocation techniques after 2000 episodes for both small and large $V$. The values plotted are averages of the last 50 points on learning curves similar to Figure 4. These scatterplots show that across all 12 stocks and both settings of $V$, our algorithm competes well with the optimal allocation, dramatically outperforms uniform, and significantly outperforms the naive bandit allocations (especially with $V = 8000$). The average completion rate across all stocks for the large (small) order sequences is 10.0% (13.1%) for uniform and 13.6% (19.4%) for optimal allocations. Our algorithm performs almost as well as optimal—13.5% (18.7%)—and much better than bandits at 11.9% (17.2%).

In the right column, we measure performance not by the fraction of $V$ shares filled in one step, but by the natural alternative of *order half-life*—the number of steps of *repeated* resubmission of any remaining shares to get the total number executed above $V/2$. Despite the fact that our algorithm is not designed to optimize this criterion and that our theory does not directly apply to it, we see the same broad story on this metric as well—our algorithm competes with ideal, dominates uniform allocation and beats the bandit approach on large orders. The average order half-life for large (small) orders is 7.2 (5.3) for uniform allocation and 5.9 (4.4) for the greedy algorithm on the true distributions. Our algorithm requires on average 6.0 (4.9) steps, while bandits uses 7.0 (4.4) to trade the large (small) orders.

**Figure 5. Comparison of our learning algorithm to the three baselines. In each plot, the performance of the learning algorithm is plotted on the *y*-axis, and the performance of one of the baselines on the *x*-axis. Left column: Evaluated by the fraction of submitted shares executed in a single time step; higher values are better, and points above the diagonal are wins for our algorithm. Right: Evaluated by order half-life; lower values are better, and points below the diagonal are wins for our algorithm. Each point corresponds to a single stock and order size; small orders (red plus signs) are 1000 shares, large orders (blue squares) are 8000 shares.**



## 6. CONCLUSION

While there has been longstanding interest in quantitative finance in the use of models from machine learning and related fields, they are often applied towards the attempt to predict directional price movements, or in the parlance of the field, to "generate alpha" (outperform the market). Here we have instead focused on a problem in what is often called *algorithmic trading*—where one seeks to optimize properties of a specified trade, rather than decide what to trade in the first place—in the recently introduced dark pool mechanism. In part because of the constraints imposed by the mechanism and the structure of the problem, we have been able to adapt and blend methods from statistics and reinforcement learning in the development of a simple, efficient, and provably effective algorithm. We expect there will be many more applications of machine learning methods in algorithmic trading in the future.

### Acknowledgments

We are grateful to Curtis Pfeiffer and Andrew Westhead for valuable conversations and to Bobby Kleinberg for introducing us to the literature on the newsvendor problem. C

### References

1. Akritas, M.G. Nonparametric survival analysis. *Stat. Sci. 19*, 4 (2004), 615–623.
2. Alon, N., Spencer, J. *The Probabilistic Method, 2nd Edition*. Wiley, New York, 2000.
3. Bogoslaw, D. Big traders dive into dark pools. Business Week article, available at: http://www.businessweek.com/investor/content/oct2007/pi2007102_394204.htm, 2007.
4. Brafman, R., Tennenholtz, M. R-MAX—a general polynomial time algorithm for near-optimal reinforcement learning. *J. Mach. Learn. Res. 3* (2003), 213–231.
5. Carrie, C. Illuminating the new dark influence on trading and U.S. market structure. *J. Trading 3*, 1 (2008), 40–55.
6. Cesa-Bianchi, N., Lugosi, G. *Prediction, Learning, and Games*. Cambridge University Press, 2006.
7. Domowitz, I., Finkelshteyn, I., Yegerman, H. Cul de sacs and highways: an optical tour of dark pool trading performance. *J. Trading 4*, 1 (2009), 16–22.
8. Foldes, A., Rejto, L. Strong uniform consistency for nonparametric survival curve estimators from randomly censored data. *Ann. Stat. 9*, 1 (1981), 122–129.
9. Ganchev, K., Kearns, M. Nevmyvaka, Y., Vaughan, J.W. Censored exploration and the dark pool problem. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, 2009.
10. Huh, W.T., Levi, R., Rusmevichientong, P., Orlin, J. Adaptive data-driven inventory control policies based on Kaplan–Meier estimator. Preprint available at http://legacy.orie.cornell.edu/~paatrus/psfiles/km-myopic.pdf, 2009.
11. Kaplan, E.L., Meier, P. Nonparametric estimation from incomplete observations. *J. Am. Stat. Assoc.* 53 (1958), 457–481.
12. Kearns, M., Singh, S. Near-optimal reinforcement learning in polynomial time. *Mach. Learn.* 49 (2002), 209–232.
13. Peterson, A.V. Kaplan-Meier estimator. In *Encyclopedia of Statistical Sciences*. Wiley, 1983.

**Kuzman Ganchev** (kuzman@cis.upenn.edu), University of Pennsylvania.

**Yuriy Nevmyvaka** (yuriy@cs.cmu.edu), University of Pennsylvania.

**Michael Kearns** (mkearns@cis.upenn.edu), University of Pennsylvania.

**Jennifer Wortman Vaughan** (jenn@seas.harvard.edu), Harvard University.