Flow Trading*

Eric Budish† Peter Cramton‡ Albert S. Kyle§ Jeongmin Lee¶
David Malec‖
March 13, 2022

Abstract

We propose a new market design for trading financial assets. The design combines three elements: (1) orders are downward-sloping linear demand curves with quantities expressed as flows; (2) markets clear in discrete time using uniform-price batch auctions; (3) traders may submit orders for portfolios of assets, expressed as arbitrary linear combinations with positive and negative weights. Thus, relative to the status quo design: time is discrete instead of continuous, prices and quantities are continuous instead of discrete, and traders can directly trade arbitrary portfolios. Clearing prices and quantities are shown to exist, with the latter unique, despite the wide variety of preferences that can be expressed via portfolio orders. Calculating prices and quantities is shown to be computationally feasible. Microfoundations for portfolio orders are provided. The proposal has six advantages over the current market design. Flow trading (1) eliminates sniping and the speed race, (2) avoids the complexities and inefficiencies caused by tick-size constraints, (3) reduces the cost and complexity of trading large quantities over time, (4) reduces the cost and complexity of trading portfolios, (5) reduces the cost and complexity of

---

*We thank Elizabeth Baldwin, Paul Klemperer, Paul Milgrom, David Parkes, and Marzena Rostek for helpful conversations. Disclosures: Peter Cramton consults on market design and is an academic advisor to Carta on the design of a private equity exchange. Albert S. Kyle has worked as a consultant for various U.S. government agencies on issues related to competition and efficiency in financial markets. He is a non-executive director of a U.S.-based asset management company. The other authors have no relevant or material financial interests that relate to this research. Research support: Budish thanks the University of Chicago Booth School of Business. Cramton thanks the German Research Foundation (DFG, EXC 2126/1-390838866) and the European Research Council under the European Union's research and innovation program (grant 741409).

†Professor of Economics, University of Chicago, Booth School of Business.
‡Professor of Economics, University of Cologne and University of Maryland.
§Professor of Finance, University of Maryland.
¶Assistant Professor of Finance, Olin Business School, Washington University in St. Louis.
‖Research Scholar, University of Cologne and University of Maryland.
providing liquidity in correlated assets, and (6) improves fairness and transparency of optimal execution.
1 Introduction

Description of the status-quo design  Current exchanges for trading equities and many other financial assets, such as futures, options, and treasury bonds, implement a market design with the following features. Most orders are variations on a standard limit order, such as “Buy 1000 shares of AAPL at $150.00 or better,” which has one maximum quantity and one limit price. The orders are processed continuously, one at a time in order of arrival, with incoming “executable” orders matched in whole or in part with “non-executable” orders resting in the limit order book. Orders are for single securities rather than for portfolios of securities. Displayed bids and offers respect a minimum “tick size,” which is typically $0.01 per share for U.S. stocks, and a minimum “lot size,” which has historically been 100 shares for most U.S. stocks. Traded quantities must also respect a minimum tick size and minimum lot size.

This market design is the natural electronic version of the limit order books used in the era of human trading—tracing not only to the era of specialists and trading pits but back to trading under the buttonwood tree. A human can run a limit order market with pen-and-paper or simple electronic recordkeeping if the orders arrive slowly enough, and computers can run limit order markets at modern speeds and order volumes.\(^1\)

However, there are multiple ways that this market design creates unnecessary costs, complexity, and perceptions of unfairness for investors and other financial market participants, especially given the market design possibilities afforded by modern computers.

First, since any order resting on the limit order book is subject to immediate execution against the next incoming order, any time there is new public information that affects an asset’s market price, resting limit orders risk being “picked off” or “sniped” by high-frequency traders acting on this information. Such orders trade at a stale price. This sniping raises the cost of providing liquidity using limit orders and is perceived by many market participants to be unfair. Recent evidence suggests that such sniping races constitute over 20% of all trading volume and constitute from 17% to 33% of the market’s cost of liquidity depending on the measure used (Aquilina, Budish and O’Neill, 2022).

Second, the discrete minimum tick size, which is necessary to prevent an explosion of message traffic under the current market design, artificially constrains the market’s

---

\(^1\)See MacKenzie (2021) for a history of this evolution from human-based trading to computer-based trading. See Aquilina, Budish and O’Neill (2022) Section 2 for a detailed overview of the market design and associated computer systems architecture for handling modern levels of speed and order activity.
cost of liquidity. This constraint has been shown to lead to (i) high-frequency trading races for queue position, (ii) a proliferation of complex order types to navigate this race for queue position, and (iii) the proliferation of exchanges with creative fee schedules designed to circumvent this constraint. Both sniping and tick-size constraints also likely play a role in the proliferation of off-exchange trading, now nearly 50% of equity volume in the United States.\(^2\)

Third, institutional investors trading large quantities of stock typically now do so by placing and canceling thousands of small orders spread out over time to reduce price impact and disguise trading intentions. If an institutional investor’s trading leaves a detectable statistical trace, algorithmic trading firms who detect the trading demand can profitably trade in front of the institutional investor.\(^3\) Institutional investors therefore must have access to complex, expensive trading platforms to manage their orders, or risk being algorithmically front run. Such trading tools are unavailable to many smaller investors.

Fourth, these costs and complexities of optimal trading are even more severe for investors trading portfolios or engaging in long-short arbitrage strategies. Not only must investors manage price impact and smoothing their trading over time for each individual security in the trading strategy, they must handle the additional complexity that comes from managing the relative rates of trade across the different securities in the trading strategy. Some indirect evidence on the value of efficiently trading portfolios comes from the rise of exchange traded funds (ETFs). ETFs are redundant assets that enable investors to trade portfolios efficiently, in exchange for a management fee on holdings that averages about 20 basis points. ETFs now constitute a remarkable 40% of all U.S. stock market volume.\(^4\)

\(^2\)See the series of papers Chao, Yao, and Ye (2017, 2019), Yao and Ye (2018) and Li, Wang and Ye (2021) on the various complexities created by tick-size constraints in U.S. equity markets, with additional references contained therein. Data on the share of off-exchange trading is available from SIFMA and was cited in Sept 2021 Senate testimony by SEC Chair Gary Gensler.

\(^3\)As one simple example, Hasbrouck and Saar (2013) point out that execution algorithms that trade every second, on the second, leave an obvious statistical trace in a continuous-time market. If trading can take place at any nanosecond, it would be an astonishing coincidence for a sequence of trades to occur at exactly 1.00000000, 2.00000000, 3.00000000, etc. Note that the same trading would not leave as obvious a statistical trace in a discrete-time batch process market, in which all trade occurs at exactly 1, 2, 3, etc.

\(^4\)ETF volume is computed from CRSP (ETFs are share code 73). The 40% figure is ETFs’ proportion of on-exchange trading volume in dollars. Vanguard reports that the asset-weighted average ETF expense ratio for non-Vanguard ETFs is 0.24% in 2020.
Flow trading This paper proposes a new market design for financial exchanges, “flow trading.” The design is motivated by the costs and complexities described above and the design possibilities enabled by modern computational power.

<table>
<thead>
<tr>
<th>Flow Trading</th>
<th>Traditional Exchange</th>
</tr>
</thead>
<tbody>
<tr>
<td>Downward-sloping piecewise-linear supply and demand curves for flows</td>
<td>Discontinuous step functions for discrete quantities</td>
</tr>
<tr>
<td>Batch auctions once per second</td>
<td>Sequential matching one at a time</td>
</tr>
<tr>
<td>Orders for portfolios (linear combinations)</td>
<td>Orders for one asset</td>
</tr>
</tbody>
</table>

Table 1: Comparison of Flow Trading with the Status Quo Design

Flow trading is a combination of three key elements (Table 1). First, instead of limit orders that define demands as step functions of price, traders place flow orders that specify demands as piecewise-linear downward-sloping functions of price, with quantities expressed as flows rather than as discrete quantity changes (Kyle and Lee 2017). For example, “Buy a maximum of one share per second at $150.00 or better, declining to zero shares per second at $150.10 or worse, until 1000 shares are bought.”

Second, instead of the market processing orders one at a time in sequence, orders are processed in discrete time using uniform-price batch auctions (“frequent batch auctions,” Budish, Cramton and Shim 2015). Suppose the discrete-time interval is one second. A flow order to buy at a maximum rate of one share per second will buy one share per batch if fully executable at the clearing price, a fraction of a share per batch if partially executable (i.e., the clearing price is in the range where the order’s demand is strictly downward sloping), or no shares per batch if non-executable. Orders persist over many auctions. An order remains outstanding until the trader cancels it or a user-defined termination criterion is met, such as the cumulative purchase of 1000 shares.

The combination of flow orders and batch auctions allows prices and quantities to be approximately continuous—tiny fractions of shares can trade each second within a nearly continuous price grid. For example, quantities could be expressed in nano-shares (billionths of shares) and prices in micro-dollars (millionths of dollars). In the status quo market design, making prices and quantities approximately continuous would cause an explosion of message traffic, with traders constantly canceling and replacing orders.
to improve their queue position. In our proposed design, prices and quantities can be approximately continuous without issue.

Third, instead of restricting to orders for single assets, our design allows traders to directly trade arbitrary portfolios of assets. In our design, a “portfolio” is any user-defined linear combination of assets, in which asset weights can be arbitrary positive or negative real numbers. Portfolio orders allow assets to be either complements or substitutes. If two assets in a portfolio have weights with the same sign, the assets are complements in the usual sense that an increase in the price of one asset decreases the quantity demanded of the other. If two assets have weights with opposite signs, the assets are substitutes because an increase in the price of one asset increases the quantity demanded of the other. For example, a pairs trade has a positive weight on the stock being bought and a negative weight on the stock being sold. An order to buy the S&P 500 has positive weights on each of the 500 stocks in the index. An order to sell a portfolio of assets, which represents an upward-sloping supply curve for the portfolio, is implemented as an equivalent downward sloping demand curve for a portfolio with negative weights on the assets being sold and zero weight on all other assets.

In sum, relative to the status quo market design, the proposed market design makes time discrete instead of continuous, prices and quantities continuous instead of discrete, and allows participants to directly trade arbitrary user-defined portfolios.

Benefits

Flow trading directly addresses the four concerns raised above about the status quo market design. First, sniping is directly addressed by discrete-time batch processing (Budish, Cramton, and Shim 2015). Moreover, flow trading makes the executed quantity proportional to the length of time, which means that even if new public information arrives just before the next batch auction, so that regular traders are vulnerable to sniping, the actual quantity executed at unfavorable prices will remain small. Second, the complexities and inefficiencies caused by tick-size and lot-size constraints are directly addressed by making prices and quantities approximately continuous. There no longer would be a reason to use non-standard exchange fee schedules or off-exchange trading venues to “hack the penny.” Additionally, there no longer would be an incentive to race for advantageous queue position, further reducing the rents from speed. Third, investors who wish to trade large quantities over some time can do so directly, with a single order. They can easily tune the urgency of trade by choosing the maximum flow rate—trading more slowly if their information is not time-sensitive and vice
versa. In effect, the ability to trade at the time-weighted average price (TWAP) is built directly into the market design. Moreover, since trading is batched, it is easier for a large trader to blend in with other traders (as in models such as Kyle 1989, Vayanos 1999, Kyle, Obizhaeva, Wang 2017, Du and Zhu 2017) without complex infrastructure. Fourth, investors who wish to trade portfolios can do so directly. Investors can define and directly trade their own custom ETFs, or long-short arbitrage portfolios, etc. Again, this reduces the need for costly trading infrastructure—expensive for large investors and unavailable to many smaller investors.

Another benefit of flow trading, related both to this last point about trading portfolios and to arguments in Budish, Cramton and Shim (2015), is that market participants can more easily provide liquidity across correlated assets and link price discovery across correlated assets. Suppose A and B are highly correlated assets. In the continuous market, a change in the price of one asset can lead to a sniping race in the other asset—this adds to the expense of providing liquidity. With flow trading, a market participant can directly provide liquidity in the pairs trades “Buy A, Sell B” and “Sell A, Buy B” (indeed, the latter is just an offer to sell the former). Even if an investor arrives wanting to buy just A, the order is automatically incorporated into A and B clearing prices. There need not be a sniping race in asset B, nor is there any “correlation breakdown” of prices between A and B (Budish, Cramton, and Shim (2015)). The pairs trade order is like a string that ties the correlated assets’ prices together, maintaining their underlying economic pricing relationships.

Last, the new market design significantly improves transparency and fairness. The key feature is that all orders that are executable at the clearing prices are executed, either at their full rate or a partial rate depending on the order’s pricing parameters, and all orders that execute for a given asset receive the same pricing for that asset. This feature allows every trader, whether trading 100 shares or 100,000, to infer exactly the execution rate on their order from publicly announced clearing prices. An institutional investor trading a sophisticated portfolio can confirm they received the correct execution. This ability perhaps does not sound radical, but it is a significant transparency improvement over the current market design, where checking whether one’s order received appropriate execution is difficult (see Tyc (2014)).

Having mentioned these potential benefits, we add an important caveat, which is that flow trading is not designed to mitigate market failures related to market power or private information (see Rostek and Yoon 2020 for a recent survey of these issues). Mar-
ket participants still must think strategically about how to trade on private information and manage their price impact, just as in the status quo market design. Flow trading removes some of the unnecessary technological costs and complexities surrounding this game, but the fact remains that large trades will move prices.

**Technical Foundations** We provide three sets of technical results: on existence and uniqueness of market-clearing prices and quantities; on computability of these prices and quantities; and results that provide microfoundations for the bidding language.

To prove existence of equilibrium prices and quantities, we transform the problem into a well-understood quadratic optimization problem with linear constraints. To do so, we first impute a quasi-linear quadratic utility function to each order by interpreting the order as an expression of preferences defining a linear marginal utility curve over the range where it is partially executable. The sum of these utility functions creates a concave objective function. The restrictions that each order must execute at a rate between zero and its maximum rate (e.g., one share per second) are linear inequality constraints. Market clearing defines linear equality constraints for each asset. Zero trade is feasible since it satisfies both sets of constraints. This setup allows us to use known results from convex optimization to prove existence of unique equilibrium quantities.

Equilibrium prices are Lagrange multipliers of the primal problem. Regardless of whether assets are complements or substitutes, market-clearing prices exist because our language imposes downward-sloping demand curves on all user-defined portfolios. (We discuss the connection to other existence and non-existence results in the next subsection.) Prices, however, may be non-unique when there are no partially executable orders from which unique prices can be inferred. For example, when there is only one order to buy or sell some asset, the market clearing quantity must be zero, but any price at which the order is non-executable clears the market. Prices can easily be made unique by introducing a tie-breaking rule.

To show computational feasibility of the market design, we start by showing our problem has a structure such that the gradient method (equivalent to Walrasian tatonnement) is guaranteed to converge. This proves that our problem is computationally simpler than some cases of finding competitive equilibrium prices (Scarf and Hansen (1973)), as the reader will anticipate from the quadratic-programming setup described just above. It is well known, however, that the gradient method may be slow and inaccurate for problems with this structure. We therefore add to the market design that the
exchange itself can serve as a “market maker of last resort.” Formally, the exchange is willing to buy or sell an epsilon amount of any portfolio at the clearing prices. This allows us to use interior point methods, which are much faster and more accurate than the gradient method. Without the exchange as market maker, we know that zero trade is feasible but it is not strictly on the interior of the constraint set; with the exchange as market maker, we can easily find a feasible point strictly on the interior, from which the algorithm can be initialized.

We provide computational proof-of-concept by calculating clearing prices for a simulated order book using our own implementation of a public-domain interior-point method on an ordinary workstation. In a market with 100,000 orders and 500 assets, with parameters chosen to make the problem difficult, our algorithm calculates prices in about 0.30 seconds. If the number of assets exceeds 2000, the computation time approaches one second with the same number of orders. If the number of orders increases to 1,000,000, computation time approaches 10 seconds with 500 assets. Conceptually, our goalpost for the computational exercise is to suggest that serious computing power can solve a practical problem of realistic size in less than one second, not just to illustrate the solution to the problem is in P and not NP.

We provide a stylized microfoundation for portfolio orders. Portfolio orders cannot express arbitrary preferences. Indeed, with wealth effects, demand schedules may slope upward. Such demands cannot be expressed in our language because we require demand schedules to be downward sloping. For a “CARA–normal” investor (with exponential utility or constant absolute risk aversion and subjective beliefs that liquidation values are normally distributed), the demands for assets are linear functions of the asset’s own price and the prices of other assets. Such demands cannot be implemented with standard limit orders due to the dependence of demand on prices for other assets. We show that, by rotating the assets in portfolios in a specific manner, such demands can be implemented with downward-sloping portfolio orders consistent with our proposal. In general, implementing $N$ asset demands requires $N$ portfolio orders. If traders believe that assets have a factor structure of rank $K < N$, they can implement the optimum with only $K$ portfolio orders, which may be practically appealing. Moreover, we then find that a trader who wishes to use $K' < K$ orders optimally does so by sorting on the portfolio Sharpe ratios, which may be practically appealing as well.
Structure of the paper  The rest of the paper is structured as follows. Section 2 discusses related literature. Section 3 describes flow orders. Section 4 discusses the existence and uniqueness of market clearing prices and quantities. Section 5 shows computational feasibility of our proposal. Section 6 provides a microfoundation for portfolio orders. Section 7 discusses implementation and policy issues. Section 8 concludes.

2 Related Literature

We divide our discussion of related literature into two parts. Section 2.1 discusses the prior work related to the flow trading market design. Section 2.2 discusses prior work that is related to the results that equilibrium prices and quantities exist.

2.1 Literature Related to the Proposed Market Design

The conceptual ideas behind this paper’s market design proposal—piecewise-linear downward-sloping demand schedules, portfolios as linear combinations of assets, general equilibrium theory, quadratic programming, batch auctions, reducing temporary price impact by trading slowly—are well-understood by researchers in economics and finance. Our contribution is to combine these ideas into a coherent and practical market design for trading financial assets such as stocks, bonds, and futures contracts.

The two prior works most closely related to our paper are Kyle and Lee (2017) and Budish, Cramton, and Shim (2015). Kyle and Lee (2017) propose downward sloping, piecewise-linear flow orders for individual assets (“continuously scaled limit orders”). Budish, Cramton, and Shim (2015) propose frequent batch auctions as a market design for financial exchanges. Combining these two market design ideas yields a market design for financial assets in which time is discrete instead of continuous, and prices and quantities are continuous instead of discrete. This is appealing for many reasons described above. Put another way, the present paper shows that these two prior market design ideas are complements, not substitutes.

The third ingredient of the market design proposal, portfolio orders, is novel to this paper. To be more precise, the broad idea of bidding for financial portfolios instead of individual assets is obvious from the combinatorial auctions literature, but our specific language for portfolio bidding is novel. We suggest via a long list of example use cases that our proposed language is practically useful for real-world financial markets. Diff-
ferent ways of representing preferences for portfolios also might not yield the existence and computability results we obtain here.

Sophisticated expression of preferences over multiple objects is a theme in the market design literature more broadly. Research on this topic has straddled computer science, economics, and operations research (Lahaie and Parkes (2004); Sandholm and Boutilier (2006); Milgrom (2009); Klemperer (2010); Vohra (2011); Bichler (2017); Cramton (2017); Budish, Cachon, Kessler, and Othman (2017); Parkes and Seuken (2018); Budish and Kessler (forthcoming)). This literature has focused on indivisible-goods combinatorial allocation problems, such as spectrum auctions. Relative to this burgeoning literature, our contribution is our proposed language for portfolio orders, which treats all goods as perfectly divisible, and allows complementarities and substitutabilities only to the extent that they can be expressed with linear portfolio weights. This language is simple enough to obtain strong existence and computational results, while being expressive enough to capture many important use cases in financial markets.

Another closely-related body of work is Li, Wang, and Ye (2021), Chao, Yao, and Ye (2019), Chao, Yao, and Ye (2017) and Yao and Ye (2018). This research highlights the complexities created by tick-size constraints in modern markets and associates tick-size constraints with an important aspect of high-frequency trading, the race for queue position. As emphasized, our market design makes time discrete and prices continuous, thus eliminating the inefficiencies caused by tick-size constraints.

The idea that optimal trading strategies involve flow trading to reduce temporary price impact costs, even when prices and quantities are continuous, emerges as an equilibrium result in game-theoretic models of rationally-optimizing strategic traders. Black (1971) conjectures that more urgent execution of large orders incurs greater price impact costs. In the context of a continuous-time model of information-based trading among overconfident and privately informed traders, Kyle, Obizhaeva, and Wang (2018) describe an equilibrium in which exponential utility and normal distribution imply all traders optimally submit linear flow strategies. In discrete-time models with trading motivated by private values or endowment shocks, Vayanos (1999) and Du and Zhu (2017) derive optimal trading strategies in which quantities are linear functions of price and inventories become differentiable functions of time in the limit as the time interval between auctions becomes zero.

A growing literature studies the implications of allowing orders to trade one asset to be contingent not only on the asset’s price but on the price of other assets (Cespa 2004,
Rostek and Yoon (2021) study strategic behavior in a market with multiple assets and imperfectly competitive traders, under market designs with both contingent and non-contingent orders. In their framework, contingent orders allow a trader's demand for one asset to be a function of the price of all other assets, whereas non-contingent orders require that a trader's demand for each asset is a function only of the price of that asset. Portfolio orders as defined here are an intermediate case between contingent and non-contingent orders. A trader's demand for one asset can depend contingently on the prices of other assets, but only through these other asset prices' effects on the price of the trader-defined portfolio the assets belong to. In Section 6 we show that traders can use a collection of portfolio orders to implement their optimal fully-contingent demand.

Surprisingly, in these models, which study imperfectly competitive trading in a one-shot Walrasian auctioneer framework, the efficiency and welfare consequences of allowing for contingent demands are ambiguous. Each trader's individually optimal demand is indeed contingent on all asset prices, but allowing for contingent orders affects incentives to strategically shade demand and supply, and the net effects of this incremental strategic behavior can effect efficiency or welfare in either direction. A related insight is in Chen and Duffie (2021), who study fragmentation of trade of the same asset across multiple trading venues.\footnote{Though see also Antill and Duffie (forth.), who find that fragmentation of trade of the same asset across multiple trading venues is unambiguously negative for efficiency in the case where one trading venue is the center of price discovery and other trading venues engage in size discovery (i.e., trade at the price discovered by the price-discovery venue).} We note that these analyses do not study the efficiency issues which motivate this paper's market design proposal, such as reducing the technology and intermediation costs of trading portfolios (in these models, trading, including complex trading, is free) or reducing latency arbitrage opportunities across venues (in these models there is mostly just a single trading period). We also emphasize in the conclusion that extending this style of analysis to the case of portfolio orders as introduced here is an interesting open question for research.

\subsection{2.2 Literature Related to Existence Results}

We obtain existence of market-clearing prices and quantities despite the wide range of preferences, including both substitutes and complements, that can be expressed using portfolio orders. In this subsection we describe the relationship of our existence results
to the textbook general equilibrium theory approach and to the literature on indivisible goods.

**Relationship to General Equilibrium Theory** Readers familiar with the standard treatment of general equilibrium theory will notice differences in our approach to existence and uniqueness. Mas-Colell, Whinston, and Green (1995, Chapter 17) ("MWG") is a reference for the standard treatment, descending from Arrow and Debreu (1954) and McKenzie (1959). This standard approach uses fixed-point theorems to derive existence results for general convex preferences which include income effects. Finding the fixed point is known to often be computationally intractable (Scarf and Hansen (1973); Daskalakis, Goldberg, and Papadimitriou (2009); Budish, Cachon, Kessler, and Othman (2017)). By contrast, our market design approach focuses on a language for preferences that yields existence and uniqueness within a computationally tractable framework.

There are three main differences with the standard treatment, as explicated in MWG. First, the setting and assumptions are different.

1. While MWG define preferences for the entire positive orthant, our model defines preferences for a given portfolio on the line segment \((0, q)\), representing partial execution of an order to buy the portfolio. The portfolio can be a short position. By defining utility to be minus-infinity off the line segment, we preserve convexity over a larger space, but we lose continuity.

2. While MWG allow general preferences that allow income effects, we assume quasi-linear utility functions of the form \(u(x) - \pi^T x\), which do not have income effects.

3. While MWG require strongly monotone preferences and strictly positive prices, our preferences are not strongly monotone and prices can be negative. Moreover, it may be difficult to make preferences monotone, even over the restricted domain of agents’ demands, because there is no natural “up” direction for the legs of a pairs trade.

Second, the technique to prove the existence of equilibrium is distinct. While MWG relies on Kakutani’s fixed-point theorem, we use quadratic programming.

Third, while equilibrium may not be unique in MWG, we have uniqueness up to a convex set. This results from using quasi-linear utility, making the second derivative of
the planner’s objective function negative (semi) definite. This guarantees that all equilibria must lie in a convex set. In our framework, substitutes and complements do not matter for existence or uniqueness, since the matrix is negative semi-definite anyway.

**Relationship to the Indivisible Goods Literature** Our assumptions are in some respects more similar to assumptions made in the literature on indivisible goods, which typically uses quasi-linear utility.

A classic reference is Kelso Jr and Crawford (1982), who show that competitive equilibrium is guaranteed to exist in an indivisible goods setting under a substitutes condition. There have been many different variations of the Kelso-Crawford substitutes condition defined in the literature; see Gul and Stacchetti (1999); Milgrom (2000); Hatfield and Milgrom (2005); Ostrovsky (2008); Hatfield et al. (2013). Hatfield, Kominers, and Westkamp (2021) discuss the relationship among many of these criteria and provides a maximum domain result for existence.

Baldwin and Klemperer (2019), on the other hand, use tropical geometry to show that existence can be obtained not only when indivisible goods are substitutes but also in some cases when they are complements. For example, left-shoes and right-shoes are clearly complements, but prices for shoes may nevertheless be guaranteed to exist if all agents’ preferences regard them as complements in ways that enable the application of the Baldwin and Klemperer (2019) existence theorems. For example, if all agents purchase shoes as pairs, and no agents regard left shoes and right shoes as substitutes for each other, prices are guaranteed to exist.

Unlike in Baldwin and Klemperer (2019), or in most of the indivisible-goods substitutes literature, we obtain existence for any preferences expressible in our language. This stronger existence result relies on our treatment of assets as perfectly divisible (avoiding the potential difficulties of exact market-clearing when there are indivisibilities) and—as noted above in the discussion of the relationship to general equilibrium theory—the restriction that preferences are only defined for each portfolio on a line segment exactly corresponding to those portfolio weights, as opposed to preferences being well defined on a richer consumption space.

Two other papers in the indivisible goods literature that deserve special mention are Klemperer (2010), which proposes the product-mix auction, and Milgrom (2009), which proposes the assignment auction. (These papers in turn descend from Shapley and Shubik 1971 and Demange, Gale, and Sotomayor (1986)). Both papers describe multi-object
auction designs that use linear preference languages and are motivated in part by financial applications—Klemperer’s auction, in particular, was designed for the Bank of England to purchase toxic financial assets during the financial crisis. Technically, the key difference versus our proposal is the preference language. In our design, participants have piecewise-linear demands for portfolios of assets, which can have arbitrary user-defined positive and negative asset weights. In Klemperer’s and Milgrom’s designs participants have piecewise-constant demands, expressing preferences over mutually-exclusive substitutable assets, including Shapley-Shubik unit-demand for substitutes preferences as a special case. For example, our design would allow a user to buy a portfolio consisting of assets A and B in some user-specified ratio, whereas Klemperer’s and Milgrom’s auction designs would allow the user to buy whichever of A or B gives them more surplus at realized prices. This difference in language then drives differences in the statements and methods of proof for existence and uniqueness results. Practically, the papers have different intended use cases. We have in mind near-continuous trading of financial assets, in which users trade portfolios in flows. Klemperer’s and Milgrom’s designs are intended for one-shot, high-value allocation problems—e.g., a high-value auction for toxic assets during the financial crisis, or a spectrum auction.

3 Flow Orders

3.1 Formal Definition of Flow Orders

Traditional limit orders consist of a price, quantity, and direction of trade for a single symbol. For example, buy 1000 shares of AAPL at $150.00 per share. The order implicitly defines a step-wise demand curve, with full demand (1000 shares) at any price weakly better than the limit and zero demand at any price strictly worse than the limit.

Flow orders depart from traditional limit orders in 3 ways:

1. **Orders are for portfolios of assets instead of individual assets.** A portfolio is defined by a vector of weights, \( w_i := (w_{i1}, \ldots, w_{iN})^\top \), where \( i \) identifies the order, \( N \) denotes the number of assets in the market, and \( w_{in} \in \mathbb{R} \) denotes the portfolio weight of asset \( n \) in order \( i \). A strictly positive weight denotes buying the asset, a strictly negative weight denotes selling the asset, and a zero weight denotes that the asset is not a part of that portfolio.
2. Flow orders specify piecewise-linear downward-sloping demands. Each order specifies two prices: a lower limit \( p^L_i \) and an upper limit \( p^H_i \), with \( p^L_i < p^H_i \). The flow order interprets \( p^L_i \) as a demand to buy the portfolio in full quantity at prices weakly lower than \( p^L_i \). It interprets \( p^H_i \) as indicating zero demand for the portfolio at prices weakly higher than \( p^H_i \). Then, in the interval \([p^L_i, p^H_i]\), the flow order linearly interpolates the quantity demanded from full quantity at \( p^L_i \) to zero quantity at \( p^H_i \).\(^6\) Note that we use the phrase “buy the portfolio” to include the case of selling assets—in our language, selling an asset is buying a portfolio with a negative weight on the asset at a negative price (i.e., receiving a transfer). We will clarify this point in detail below.

3. Quantities are expressed as flows per batch interval, up to a total quantity limit. For each order \( i \), the user specifies two quantity parameters, \( q_i > 0 \) and \( Q^\text{max}_i > 0 \), expressing their demand to buy up to quantity \( q_i \) of the portfolio per batch interval, up to a cumulative total purchased quantity of \( Q^\text{max}_i \). Instead of requiring that quantities express a demand to trade immediately (1000 shares now), users can tune their urgency to trade.

Thus, a flow order is described by the tuple \((w_i, p^L_i, p^H_i, q_i, Q^\text{max}_i)\). (Throughout this paper, we use a lower-case bold font to denote vectors, an upper-case bold font to denote matrices, a subscript \( i \) to denote orders, and a subscript \( n \) to denote assets.)

Next, we define a flow order’s demand within a batch auction. Assume that the order’s cumulative purchased quantity is not within \( q_i \) of \( Q^\text{max}_i \), so that the order can purchase its full quantity \( q_i \) in the next batch without exceeding \( Q^\text{max}_i \).\(^7\) Let \( \pi = (\pi_1, \ldots, \pi_N)^T \) denote the column vector of market prices of all assets \( n = 1, \ldots, N \). The market price for the portfolio defined by the weight vector \( w_i \) is the inner product

\[
p_i = w_i^\top \pi := \sum_{n=1}^{N} w_{in} \pi_n. \quad (1)
\]

Order \( i \)’s demand per batch auction, which we call its “flow demand,” is the downward-sloping linear function of the portfolio price \( p_i = w_i^\top \pi \) defined by:

\(^6\)In a traditional limit order at price \( p \), the implied demand is the full quantity at prices weakly better than \( p \) and zero quantity at prices strictly worse than \( p \). In our language, these two implications of the traditional limit price are split into two separate parameters: demand in full at prices weakly better than the lower limit \( p^L_i \), and demand zero at prices weakly worse than the upper limit \( p^H_i \).

\(^7\)In the case where the order’s cumulative purchased quantity, say \( Q^\text{t}_i \), is within \( q_i \) of the limit \( Q^\text{max}_i \), replace \( q_i \) with the remaining quantity demanded \( Q^\text{max}_i - Q^\text{t}_i \).
\[ D_i(p_i|w_i, q_i, p_i^L, p_i^H) = q_i \text{trunc} \left( \frac{p_i^H - p_i}{p_i^H - p_i^L} \right), \quad \text{where} \quad \text{trunc}(z) := \begin{cases} 1, & \text{for } z \geq 1 \\ z, & \text{for } 0 < z < 1 \\ 0, & \text{for } z \leq 0 \end{cases}. \]  

(2)

Notice how the rate at which order \( i \) buys the portfolio depends on the order’s quantity limit \( q_i \) and where the price for the portfolio is relative to the order’s price parameters \( p_i^L \) and \( p_i^H \). If the portfolio price \( p_i \) is less than or equal to \( p_i^L \), the order is “fully executable,” and the portfolio is bought at the maximum rate \( q_i \). If the portfolio price \( p_i \) is higher than \( p_i^H \), then the order is “nonexecutable” and does not buy at all. If the portfolio price is somewhere between \( p_i^H \) and \( p_i^L \), the order is “partially executable” and buys at the rate determined by linear interpolation between the two price parameters.

**Buying vs. Selling**  
This formulation treats “selling” an asset as buying a portfolio with a negative weight on that asset at a negative price. This not only generates compact notation for representing both buying and selling but also emphasizes a symmetry between buying and selling, which will be important for understanding how market clearing works. General equilibrium theory often uses this idea that an upward sloping supply curve for positive quantities is equivalent to a downward sloping demand curve for negative quantities.

Whether buying or selling, we have \( p_i^L < p_i^H \) and demand defined according to equation (2). However, when selling, both \( p_i^L \) and \( p_i^H \) are negative. For example, an order to sell XYZ in full at price \( $42.00 \) or higher, with the sell rate declining linearly to zero at price \( $41.00 \), would be encoded with \( p_i^L = -$42.00 \) and \( p_i^H = -$41.00 \). There are two equivalent ways to remember this. First, think of \( p_i^L \) as analogous to the price limit in a limit order (willing to trade in full at this price or better), with demand then declining linearly to zero in the interval \([p_i^L, p_i^H]\). Alternatively, think of \( p_i^H \) as the price at which the trader is exactly indifferent between trading and not. Then, as the price improves from \( p_i^H \), the trader’s quantity demanded increases linearly, up to a maximum quantity of \( q_i \) when the price reaches \( p_i^L \) or better.

See Figure 1 for an illustration of buying and selling.

Last, note that if a portfolio has both positive and negative weights, there may not be a natural buying versus selling direction to the order. The trader is always “buying
the portfolio" under our approach, but whether their pricing parameters $p_i^L$ and $p_i^H$ are positive or negative will depend on the weighted valuations of the assets in the portfolio.

Additional Technical Remarks on the Formulation  We make two additional remarks on this formulation.

First, observe that while the above demand function in equation (2) has a single downward-sloping segment, the user can define an arbitrary piecewise-linear downward-sloping demand function for a given portfolio with multiple flow orders.

Second, order specification using the tuple of parameters $(w_i, p_i^L, p_i^H, q_i, Q_i^\text{max})$ contains an intentional redundancy of notation. Buying a portfolio containing one share of each stock at a rate of ten portfolio units per batch auction is equivalent to buying a portfolio containing half a share of each stock at a rate of twenty portfolio units per batch auction. More generally, for some parameter $\alpha > 0$, changing the order parameters from $(w_i, p_i^L, p_i^H, q_i, Q_i^\text{max})$ to $(\alpha w_i, \alpha p_i^L, \alpha p_i^H, q_i / \alpha, Q_i^\text{max} / \alpha)$ has no effect on the trade rates for each asset as a function of asset prices. We do this because in some circumstances it will be natural to normalize some stocks' individual weights to one or minus one, while in others it may be more natural to normalize the sum of weights.
**Proxy Instructions For Orders Over Time**  As in the traditional market design, users may modify or cancel their flow orders at any moment in time throughout the trading day. Additionally, users may want to specify what we will refer to as “proxy instructions” that modify or cancel their orders under specified contingencies.

The parameter \(Q_{i}^{\text{max}}\) is an example of such a proxy instruction: cancel the order from the market once the cumulative total quantity \(Q_{i}^{\text{max}}\) has been reached. Another example is time-in-force instructions, such as “good for day” or good for some other specified period. In principle, the exchange could provide more complex examples, such as allowing an order’s pricing parameters to vary dynamically over time as a function of recent prices (“Ensure that my order’s price impact is never more than ten basis points”), or allowing an order’s quantity parameter to vary over time (“Reduce this order’s flow quantity if I am averaging above ten percent of trading volume”). We will not discuss such complex order contingencies in this paper.

### 3.2 Examples

We give several examples to illustrate the flexibility of portfolio orders.

1. **Standard limit order.**

   A standard limit order expresses preferences to buy or sell a specified quantity of one asset at one limit price. A flow order can be specified to approximate a limit order. First, when only one weight \(w_i\) is nonzero, the order is an order to buy one asset if the weight is positive or to sell one asset if the weight is negative. Second, the maximum rate \(q_i\) can be set to equal the quantity the trader wants to buy or sell, \(Q_{i}^{\text{max}}\). Third, the price parameters can be set so that \(p_{i}^{L}\) corresponds to the intended limit price and \(p_{i}^{H}\) is as close as possible to \(p_{i}^{L}\). Theoretically, we obtain a limit order in the limit as \(p_{i}^{H} \rightarrow p_{i}^{L} +\).

2. **Time-weighted average price (TWAP) order.**

   In the traditional market design, a market order executes immediately at the clearing price. The analog here is a time-weighted average price (TWAP) order. The user specifies a price parameter \(p_{i}^{L}\) that is sufficiently aggressive relative to recent prices that it is essentially guaranteed to execute.\(^8\) Then, the user will trade quan-

---

\(^8\)In the traditional formulation of a market order, one thinks of the limit price as \(\infty\) if buying and as 0 if selling. The 0 for selling implicitly encodes that assets are “goods” that can always be sold at a weakly
tity $q_i$ of the portfolio every batch auction until their quantity limit is achieved, i.e., they will trade at the TWAP over this period.

3. Pairs trades.

A pairs trade is executed by specifying a portfolio weight vector $\mathbf{w}_i$ with one strictly positive entry, one strictly negative entry, and the rest zeros.

4. Portfolio trades.

A portfolio trade is executed by specifying a portfolio weight vector $\mathbf{w}_i$ with either all entries weakly positive (if buying the portfolio) or all entries weakly negative (if selling the portfolio). The assets whose weights are strictly positive or strictly negative comprise the portfolio.

Traders can construct and trade their own index portfolios. For example, an order to buy the S&P 500 has positive weights on each stock in the S&P 500 index, with weights proportional to S&P 500 weights and zero weight on stocks not in the S&P 500 index. An order to sell an index has negative weights on all stocks in the index. Traders can easily customize index portfolios by adjusting portfolio weights—e.g., adjusting weights based on valuation models or setting to zero weights for assets that fail a screening criterion such as environmental, social, and governance criteria.

5. General long-short strategies.

A general long-short strategy combines the previous two cases: multiple positive and negative entries.


A trader can engage in market making, whether for a single asset, a pairs trade, a portfolio trade, or a general long-short strategy, using two orders with opposite-signed weights and price parameters. For example, a market maker who is willing to buy portfolio $\mathbf{w}_i$ in full at 41.00 and sell it in full at 42.00 could use orders like

- Buy leg: weights $\mathbf{w}_i$, price parameters $p_i^L = 41.00$, $p_i^H = 41.25$

\footnote{positive price. Here, if the order is for a portfolio with both positive and negative weights, it is not automatic from the order itself whether the portfolio is a “good” that should always trade at a positive price or a “bad” that should trade at a negative price. Either way, the trader can guarantee execution by specifying $p_i^L$ sufficiently large.}
• Sell leg: weights $-w_i$, price parameters $p^L_i = -$42.00, $p^H_i = -$41.75

3.3 Limitations of the Language

There are limitations of the language for representing trading demands.

First, trading demands are only defined at exactly the ratio of portfolio weights specified in the order. If an order specifies it wants to buy assets A and B at a ratio of 2:1, the order contains no information about the trader’s willingness to trade at, say, a ratio of 2.2:1 or 1.8:1. This restriction relative to traditional consumer theory, where preferences are typically defined on the whole positive orthant, is key to our method of existence proof (below in Section 4.2).

Second, trading demands are linear within each order. In principle, we could replace the linear trunc function with the flexibility to specify an arbitrary downward-sloping function on the interval of prices $[p^L_i, p^H_i]$. However, our existence proof and computational results take advantage of this linearity. We view the linearity restriction as a less important limitation because arbitrary downward-sloping functions can be approximated, if needed, with a set of linear orders.

Third, the language does not allow for indivisibilities. Most importantly, a user cannot specify a minimum transaction quantity per batch, only a maximum. So, for example, an order cannot be “fill or kill”, or “at least 100 shares per batch, otherwise stay out”. That said, a user may approximate such preferences with marketable orders if prices are sufficiently continuous.

Last, the language does not allow for in-order contingencies. This includes cases like “buy A if the price of B is high enough” or “buy whichever of A or B gives me more surplus given my valuations”. This latter kind of preference expression is analyzed in Demange, Gale, and Sotomayor (1986) and is present in market design proposals of Klemperer (2010) and Milgrom (2009). As with indivisibilities, a user may approximate such preferences with marketable orders if prices are continuous enough.

4 Market Clearing Prices and Quantities

Now we turn our attention to the exchange’s problem of finding clearing prices and quantities.
4.1 Definition of Market Clearing

To define market clearing, we need to convert individual traders’ demand curves for portfolios as a function of portfolio prices into a market demand curve for assets as a function of asset prices. For each portfolio $i$, first replace the portfolio price $p_i$ by the weighted vector of asset prices, using $p_i = \mathbf{p} \cdot \mathbf{w}_i$. Then, convert the demand for portfolio units $D_i(\mathbf{p} \cdot \mathbf{w}_i)$ into the demand for individual assets by multiplying by the portfolio weights $\mathbf{w}_i$. Last, sum up the demand for assets across all orders $i$ to obtain the market net excess demand curve for assets as a function of asset prices:

$$D(\mathbf{p}) := \sum_{i=1}^I D_i(\mathbf{p} \cdot \mathbf{w}_i | w_i, q_i, p^L_i, p^H_i) \mathbf{w}_i. \quad (3)$$

The function $D(\cdot)$ maps asset price vectors $\mathbf{p} \in \mathbb{R}^N$ to net asset quantity vectors $\mathbf{q} \in \mathbb{R}^N$. A price vector is market clearing if each asset’s net excess demand is zero:

$$D(\mathbf{p}) = \mathbf{0}. \quad (4)$$

This market clearing condition defines $N$ equations in $N$ unknowns. At clearing prices $\mathbf{p}$, order $i$’s trading rate for the individual assets is given by $D_i(\mathbf{p} \cdot \mathbf{w}_i \mathbf{w}_i)$, i.e., by its demand for portfolio units at the clearing prices times the portfolio weights.

For arbitrary, non-clearing price vectors, the quantity vector $\mathbf{q} = D(\mathbf{p})$ may have both positive and negative components. Note that we do not enforce a constraint that prices be nonnegative. Negative prices arise naturally in some commodity markets, such as electricity, with limited storage and costly curtailment.

4.2 Existence of Market Clearing Prices and Quantities

To show the existence of clearing prices, which then determine market-clearing quantities, we formulate an optimization problem by imputing to each order “as-bid” preferences which define the dollar utility value of the number of portfolio units bought, then sum the utility functions across orders to obtain the objective function to be maximized.

An order’s demand is a linear function of prices in the range of prices where the order is partially executable. Therefore, a quadratic quasilinear utility function defines preferences. The constraints preventing overfilling or underfilling the order are linear inequality constraints. Market clearing consists of linear equality constraints. Putting
this together results in a quadratic program—maximizing a quadratic objective function subject to linear constraints.

Quadratic programs have been thoroughly studied and are well-understood. Given the structure of our problem, we can use well-known results to show that unique utility-maximizing quantities exist, and the solution implies Lagrange multipliers which correspond to clearing prices. A solution to the dual problem of calculating optimal (market-clearing) prices also exists and implies the same solution as the original (primal) problem.

Imputing utility functions to orders is a convenient mathematical modeling device. We proceed as though orders directly represent traders’ preferences, even though, in practice, traders submit orders strategically. Thus, our methodology does not measure actual economic welfare and does not generate welfare results on market efficiency. Rather, the method provides a practical approach to finding clearing prices and quantities consistent with bids.

**Pseudo-Utility** Let $V_i(x)$ denote the dollar utility of order $i$ from a trade rate of $x$ in portfolio units per second. To find $V_i(x)$, we first define the marginal utility function $M_i(x)$ as the inverse demand curve, $p_i = M_i(x_i)$, where recall the order $i$ demand curve is denoted by $D_i(p_i) = x_i$. In words, the inverse demand curve maps order $i$’s trade rate $x \in [0, q_i]$ into prices $p \in [p_i^L, p_i^H]$.\(^9\) Rearranging equation (2) we have:

$$M_i(x) := p_i^H - \frac{p_i^H - p_i^L}{q_i} x \quad \text{for } x \in [0, q_i]. \tag{5}$$

The value of $M_i(x)$ measures marginal as-bid flow value in dollars per portfolio unit. Utility $V_i(x)$, as a function of the trade rate $x$, is defined as the integral of the marginal utility function for trade rate over the interval $[0, x]$: 

$$V_i(x) := \int_0^x M_i(u) \, du \tag{6}$$

Since the marginal value is linear in $x$, the total value is quadratic and therefore strictly

---

\(^9\)For trade rates in the interval $(0, q_i)$, the fact that the order chooses an interior quantity tells us that the order’s as-bid marginal utility is equal to the corresponding price in the interval $(p_i^L, p_i^H)$. The same logic extends to the boundary points 0 and $q_i$, corresponding respectively to prices $p_i^H$ and $p_i^L$, by assuming as-bid utility is continuous.
concave in \( x \):

\[
V_i(x) = p_i^H x - \frac{p_i^H - p_i^L}{2q_i} x^2
\]  

(7)

We will think of \( V_i(x) \) as defined for all \( x \in \mathbb{R} \), with order specifications imposing the constraint \( x \in [0, q_i] \).\textsuperscript{10}

**Value Maximization**  

Our problem of finding clearing prices is formulated as two optimization problems, a primal problem of finding quantities that maximize as-bid dollar value and a dual problem of finding prices that minimize the cost of non-clearing prices. The first-order conditions for optimality of these two problems imply market-clearing quantities and prices.

The exchange, acting analogously to a social planner in general equilibrium theory, chooses a vector of execution rates for all orders \( x = (x_1, \ldots, x_I) \) to maximize aggregate value, defined as the sum of pseudo-utility functions across orders,

\[
V(x) := \sum_{i=1}^I V_i(x_i) \quad \text{for } x \in \mathbb{R}^I,
\]  

(8)

subject to choosing quantities consistent with market clearing constraints and order execution rate constraints:

\[
\begin{align*}
\max_x V(x) \quad \text{subject to} \quad & \sum_{i=0}^I x_i w_i = 0 \quad \text{(market clearing)} \\
& x_i \in [0, q_i] \text{ for all } i \quad \text{(order execution rate)}
\end{align*}
\]  

(9)

The objective function \( V(x) \) is concave because it is a sum of concave functions.

Indeed, since the objective function is quadratic and the constraints are linear, this is a quadratic program. To make this quadratic structure apparent using matrix and vector notation, let \( W \) denote the \( N \times I \) matrix whose \( i \)th column is \( w_i \). Let \( p^H \) denote the column vector whose \( i \)th element is \( p_i^H \). Let \( D \) denote the \( I \times I \) positive definite diagonal matrix whose \( i \)th diagonal element is \((p_i^H - p_i^L)/q_i\). Then problem in equation (9) may be written compactly as

\[
\max_x \left[ x^T p^H - \frac{1}{2} x^T D x \right] \quad \text{subject to} \quad W x = 0 \quad \text{and} \quad 0 \leq x \leq q.
\]  

(10)

We first show that quantities that maximize aggregate utility exist. Then we show that

\textsuperscript{10}We could equivalently think of the domain of \( V_i(x) \) as \( x \in [0, q_i] \) or define \( V_i(x) = -\infty \) for \( x \notin [0, q_i] \).
clearing prices exist by examining the dual problem to the utility maximization problem.

**Theorem 1** (Existence and Uniqueness of Optimal Quantities). There exists a unique quantity vector \( \mathbf{x}^* \) which solves the maximization problem in equation (10).

**Proof.** The problem has the following properties:

1. Compactness and convexity: The inequality constraints on trade rates define the Cartesian product of \( I \) intervals, \([0, q_1] \times \cdots \times [0, q_I]\), which is compact and convex. The market-clearing conditions are linear constraints, which define the intersection of hyperplanes. The intersection of a compact, convex set with hyperplanes is compact and convex. Thus, the set of vectors of trade rates \( \mathbf{x} \) that satisfies all constraints is compact and convex.

2. Feasibility: No trade \( (\mathbf{x} = \mathbf{0}) \) generates well-defined utility for each order \( (V_i(0) = 0) \), clears markets and is allowed on each order. No-trade is feasible.

3. Strict concavity: Each function \( V_i(x_i) \) is quadratic and therefore strictly concave for all \( x_i \in \mathbb{R} \). Since \( V \) is the sum of \( V_i \) across \( i \), the function \( V \) is concave on the domain \( \mathbb{R}^I \) and thus also on the compact and convex subset defined by the constraints.

It is a well-known principle of convex analysis (Boyd and Vandenberghe (2004); Bertsekas (2009); Nocedal and Wright (2006)) that a strictly concave objective function on a non-empty compact and convex set has a unique maximizing vector \( \mathbf{x}^* \).

\( \square \)

Our approach makes the problem compact by assuming that traders are not interested in trading additional quantities beyond some very favorable prices. This is like putting upper and lower bounds on quantities and linear combinations of quantities.

To prove that clearing prices exist, we exploit the duality between the problems of finding optimal quantities and prices. For this, we define a Lagrangian function of the vector of quantities \( \mathbf{x} \) with three constraints: (1) the market clears \((\sum_{i=1}^{I} x_i \cdot w_i = 0)\); (2) the order execution rate is greater than or equal to zero \((\mathbf{x} \geq \mathbf{0})\); (3) the order execution rate is less than or equal to the maximum \((\mathbf{x} \leq \mathbf{q})\). In vector notation, the Lagrangian is defined by

\[
L(\mathbf{x}, \mathbf{\pi}, \mathbf{\lambda}, \mathbf{\mu}) := \mathbf{x}^\top \mathbf{p}^H - \frac{1}{2} \mathbf{x}^\top \mathbf{D} \mathbf{x} - \mathbf{\pi}^\top \mathbf{W} \mathbf{x} + \mathbf{\mu}^\top \mathbf{x} + \mathbf{\lambda}^\top (\mathbf{q} - \mathbf{x}).
\]  

(11)

Since the multipliers associated with the market-clearing equality constraint have the economic interpretation of market prices for assets, we use the notation \( \mathbf{\pi} = (\pi_1, \ldots, \pi_N) \) for these multipliers. Two vectors of order-execution-rate multipliers, \( \mathbf{\mu} = (\mu_1, \ldots, \mu_I) \)
and $\lambda = (\lambda_1, \ldots, \lambda_I)$, are associated with inequality constraints on order execution rates, with two constraints for each order.

The dual problem associated with the primal problem of maximizing aggregate utility in equation (10), is then defined by

$$\hat{G}(\pi, \lambda, \mu) := \max_{x} L(x, \pi, \lambda, \mu) \quad \text{for} \quad \pi \in \mathbb{R}^N, \quad \mu \geq 0, \quad \lambda \geq 0. \quad (12)$$

The dual problem is a minimization problem with infimum $g^*$ defined by

$$g^* := \inf_{\pi, \lambda, \mu} \hat{G}(\pi, \lambda, \mu) \quad \text{subject to} \quad \pi \in \mathbb{R}^N, \quad \mu \geq 0, \quad \lambda \geq 0. \quad (13)$$

The dual problem in equation (13) is formulated as an infimum rather than a minimum because we have not yet shown that there exists a solution $(\pi^*, \lambda^*, \mu^*)$ which attains the infimum.

**Theorem 2** (Existence of clearing prices). *There exists at least one optimal solution $(\pi^*, \lambda^*, \mu^*)$ to the dual problem in equation (13). The solutions $x^*$ and $(\pi^*, \lambda^*, \mu^*)$ are a primal-dual pair which satisfies the strict duality relationship

$$g^* = V(x^*). \quad (14)$$

**Proof of Theorem 2.** The primal problem has the following properties:

1. Concavity: The objective function $V(x)$ is strictly concave.
2. Finite solution: The primal objective is the sum of a finite number of concave quadratic functions. Since each quadratic function is bounded above, the solution to the primal problem is bounded above.
3. Linear constraints: The minimum execution rate constraint $x \geq 0$, the maximum execution rate constraint $x \leq q$, and the market clearing constraint $Wx = 0$ are all linear.
4. Feasibility: No trade ($x = 0$) is feasible because it clears the market and is allowed on each order.\footnote{Feasibility does not require a strict interior point (Slater’s condition) because the constraints are linear in this problem (linear constraint qualification).}

It is a standard result from convex programming that a concave primal problem, a finite supremum on the primal problem, feasibility, and linear constraints guarantee that a solution to the dual problem exists and has the same optimal value as the supremum to the primal problem even if a solution to the primal problem does not exist as it does
in our problem; see Boyd and Vandenberghe (2004), Bertsekas (2009, Proposition 5.3.4, p. 173), Nocedal and Wright (2006, Theorem 16.4, p. 464). Since Theorem 1 guarantees that a solution to the primal problem does exist, the solution to the primal problem has the same value as the solution to the dual problem.

There are three Lagrange multipliers in this problem: $\pi, \lambda$, and $\mu$. The multiplier on the market clearing condition $\pi$ is the vector of prices for all assets. The other multipliers $\lambda$ and $\mu$ ensure that orders are not underfilled ($x < 0$) or overfilled ($x > q$).

Theorem 2 does not guarantee that clearing prices are unique. The set of clearing prices is convex and may be unbounded. A trivial example occurs when all orders are buy orders for individual assets, and there are no sell orders. Then any sufficiently high price clears the market with zero trade. There may also be cases where the clearing price is not unique even when trade occurs. A trivial example occurs when there is one buy order and one sell order for the same asset (or portfolio) with the same quantities $q$, and the buyer's lower-limit price exceeds the absolute value of the seller's lower limit price. In this case, there is an interval of prices where both orders are fully executable. We discuss a tie-breaking rule to pick a unique price in the next section.

5 Computation

In this section, we study the computational feasibility of flow trading. The objective is to provide a proof of concept, finding market-clearing solutions in less than a second for a reasonably difficult problem with 500 assets and 100,000 orders, using an ordinary workstation. We also study how computation time varies with the number of assets and orders and parameters that affect how orders are generated. Section 5.1 proposes a computational methodology. Section 5.2 explores computational performance in a simulation environment.

5.1 Methodology

Gradient Method For economists, Walrasian tatonnement is an intuitive approach for calculating market-clearing prices. An auctioneer announces tentative prices, and traders respond with their quantities. The auctioneer then adjusts prices in the direction proportional to net excess demand. The process continues until the market clears.
Tatonnement is equivalent to applying the gradient optimization method to an objective function of prices whose first-order conditions correspond to market clearing. In our setting, such a function can be obtained as

\[
G(\pi) := \inf_{\lambda, \mu} \hat{G}(\pi, \lambda, \mu) \quad \text{subject to} \quad \mu \geq 0, \quad \lambda \geq 0,
\]

where \( \hat{G}(\pi, \lambda, \mu) \) is given by equation (12). Theorem 2 implies that this function’s first order conditions, \( G'(\pi) = 0 \), correspond to market clearing.

Since the gains function has a piecewise-linear derivative, it is continuously differentiable, and the derivative satisfies a Lipschitz condition. These conditions assure that the gradient method converges (Nesterov (2004, Corollary 2.1.2, p. 70)). While the guaranteed convergence rate is much faster than for the traditional general-equilibrium theory problems discussed by Scarf and Hansen (1973), it is too slow for our purpose. Reducing the error by a factor of one million may require approximately one million iterations, a prohibitively large number in our setting, where we need to solve for prices frequently throughout the trading day (as opposed to a single high-stakes allocation problem in a combinatorial auction).

5.1.1 Interior Point Method

We propose to solve the minimization problem by using an interior point method for quadratic programming. The literature shows that interior point methods are computationally more efficient than the intuitive gradient method, both theoretically (see Nesterov (2004, Chapter 4); Bertsekas (2009), Boyd and Vandenberghe (2004)) and in practice (Gondzio (2012)).

Exchange as a Small Market Maker

Theoretically, the interior point method requires the existence of an interior point, a feasible allocation on the interior of the constraint

\[12\]There is a Lipschitz constant \( L \) such that \( |\nabla G(\pi + \Delta \pi) - \nabla G(\pi)| < L|\Delta \pi| \) for all \( \pi \) and all \( \Delta \pi \).

\[13\]More modern work in computer science has focused on the complexity of computing Brouwer and Kakutani fixed points (Daskalakis, Goldberg, and Papadimitriou (2009); Budish, Cachon, Kessler, and Othman (2017)) and supports the claim that computing competitive equilibrium prices can be computationally difficult.

\[14\]For interior point methods, the maximum number of iterations has an upper bound proportional to \( O(\log(1/\epsilon)) \), where \( \epsilon \) is the proportion by which the error is reduced (Nesterov (2004) Theorem 3.1). For example, reducing error by proportion 0.000001 (one-millionth) is \( O(\log(1,000,000)) \approx O(13.8) \). For gradient methods, the upper bound is proportional to \( O(1/\epsilon) \) or \( O(1/\epsilon^2) \) depending on the structure of the problem (Gondzio (2012)).
set. Such an allocation clears the market and strictly satisfies the inequality constraints $(0 < x < q)$. However, there is no natural candidate for such an interior point in our setting. For example, no-trade ($x = 0$), which satisfies market clearing, does not lie on the interior of the constraints.

To ensure an interior point, we let the exchange act as a small market maker for every asset. Specifically, the exchange submits a linear demand curve for each asset $n$,

$$\epsilon_n(\pi_{0n} - \pi_n),$$

where $\epsilon_n$ is the slope, and $\pi_{0n}$ is a base price below which the exchange buys and above which it sells. Here, $\epsilon_n$ can be a small positive number such that the exchange does little trading. The strategy can be implemented by placing two flow orders for each asset: one order to buy at prices below $\pi_{0n}$ and the other to sell at prices above $\pi_{0n}$.

With the exchange as a small market maker, existence of an interior point is assured. For example, pick any $x$ such that $0 < x < q$. Then the exchange can soak up any uncleared quantities to clear the market.

Allowing modest exchange trading has two other benefits. First, it can resolve the tiebreaker problem, which arises when the convex set of market-clearing prices contains more than one point (as we know is possible from Theorem 2). Since the exchange has an active order for every asset at every relevant price vector, market prices are chosen uniquely for all assets when multiple prices would otherwise be possible. For example, if $\pi_{0n}$ is set at the previous clearing price for asset $n$, then the exchange’s small trading demand will tend to break ties in favor of the prices closest to the previous prices. Second, the exchange can absorb uncleared quantities due to rounding error and inexact algorithm convergence to clearing prices, even when the algorithm has converged to a target tolerance.

**Solving the KKT Conditions** The interior-point method solves the Karush–Kuhn–Tucker (KKT) conditions to find market-clearing prices and quantities, utilizing information about both quantities from the primal problem and prices and multipliers from the dual.
problem.

From here on, we redefine $p^H$, $p^L$, $D$, $W$, $q$, and $x$ to include the exchange’s orders. Then the results from Section 4 hold, and it is straightforward to show that a solution to the KKT conditions clears the market. Further, since the exchange has an active order at every price, the solution is unique.

**Theorem 3** (Karush–Kuhn–Tucker (KKT) Conditions with Exchange Trading). *Any solution of the KKT conditions in equations* (17)–(20) *for quantities $x^* := (x^*_1, \ldots, x^*_I)$ and multipliers $(\pi^*, \lambda^*, \mu^*)$ is a solution to both the primal problem and dual problem:*

\begin{align*}
W x^* &= 0, \quad 0 \leq x^* \leq q, \quad (Primal Feasibility) \tag{17} \\
\pi^* &\in \mathbb{R}^N, \quad \lambda^* \geq 0, \quad \mu^* \geq 0, \quad (Dual Feasibility) \tag{18} \\
p^H - Dx^* - W^T \pi^* + \mu^* - \lambda^* &= 0, \quad (Primal Optimality) \tag{19} \\
\lambda^* \cdot (q - x^*) &= 0, \quad \mu^* \cdot x^* = 0, \quad (Complementary Slackness) \tag{20}
\end{align*}

*With exchange trading defined in equation* (16), *there exists a unique solution to the KKT conditions.*

**Proof of Theorem 3.** Existence is a straightforward consequence of Theorems 1 and 2, which imply that a unique optimal primal solution $x^*$ exists and some optimal dual solution $(\pi^*, \lambda^*, \mu^*)$ exists, and these solutions form a primal-dual pair with the same optimized value. Uniqueness follows from the exchange having a partially executable order for every asset. If market-clearing prices were not unique, then any change in the price of any asset would change the aggregate quantity demanded, which implies multiple market-clearing quantities. Since the quantities are unique from Theorem 1, prices must therefore also be unique. \hfill \square

Instead of solving these conditions directly, the interior point method first modifies the problem by replacing the complementary slackness conditions in equation (20) with a set of constraints parameterized by $\bar{\nu} > 0$:

\begin{align*}
\lambda^* \cdot (q - x^*) &= \bar{\nu} \cdot 1, \quad \mu^* \cdot x^* = \bar{\nu} \cdot 1. \tag{21}
\end{align*}

Then as we take a limit as $\bar{\nu} \to 0$, the sequence of solutions to the modified KKT conditions satisfies the original KKT conditions in Theorem 3.
The modified complementary slackness conditions in equation (21) imply that a solution to the modified KKT conditions satisfies the constraints with strict inequality: \( 0 < x < q \). Exchange trading plays a role in guaranteeing the existence of such a solution for any \( \bar{\nu} > 0 \).

**Implementation Details**  Our algorithmic strategy is to solve the modified KKT conditions in equations (17), (18), (19), and (21) iteratively by starting with an initial guess for \( x, \pi, \mu, \lambda \). To find search directions \( (\Delta x, \Delta \pi, \Delta \mu, \Delta \lambda) \), we first substitute \( x + \Delta x, \pi + \Delta \pi, \mu + \Delta \mu, \) and \( \lambda + \Delta \lambda \) for \( x, \pi, \mu, \) and \( \lambda \), respectively into the system of equations representing the modified KKT conditions with the value of \( \bar{\nu} \) set to 0. Then we linearize the system of equations by dropping the second order terms (\( \Delta x \Delta \mu \) and \( \Delta x \Delta \lambda \) in the modified complementary slackness conditions in equation (21)) and solve the resulting linear system for \( \Delta x, \Delta \pi, \Delta \mu, \) and \( \Delta \lambda \).\(^{16}\) The solution vectors are then multiplied by a scalar \( \alpha \) (with \( 0 < \alpha \leq 1 \)) to ensure that the best guess for the next iteration \( x + \alpha \Delta x, \pi + \alpha \Delta \pi, \mu + \alpha \Delta \mu, \lambda + \alpha \Delta \lambda \) is such that \( x \) remains an interior point and the multipliers remain strictly positive. Since the KKT conditions are essentially first-order conditions, the linearized approximation is a version of Newton's method.

On each iteration, the linear system is solved in the following way. The multipliers \( \Delta \mu \) and \( \Delta \lambda \) are expressed as functions of \( \Delta x \), easy invertibility of the diagonal matrix \( D \) allows \( x \) to be expressed as a simple function of \( \pi \), and substituting the solution for \( x \) into the market clearing condition reduces the problem to solving an \( N \times N \) positive definite system for a price update to \( \pi \), for which a Cholesky decomposition is used.\(^{17}\)

\(^{16}\)The KKT system is nonlinear in the unknowns \( \pi, x, \mu, \) and \( \lambda \) only because the complementary slackness condition in equation (21) involves element-by-element multiplication of \( x \) by \( \mu \) and \( \lambda \). For \( \mu \) (and analogously for \( \lambda \)), linearizing \( (x + \Delta x)(\mu + \Delta \mu) \) sets the second-order term term \( \Delta x \Delta \mu \) to zero.

\(^{17}\)As in the original KKT system, the revised KKT system is nonlinear because the revised complementary slackness condition involves element-by-element multiplication of \( x \) by \( \mu \) and \( \lambda \). To correct for the error created by dropping the second-order term, we solve the linear system a second time on each iteration (using the same Cholesky decomposition), including a correction term described by Mehrotra (1992) in the second solution.

Our solution uses a Python implementation of the interior point methodology in the CVXOPT package, as described by Vandenberghe (2010). The Python programming language and the CVXOPT package are free and publicly available. One version of the algorithm is implemented on the cpu using the Python packages numpy and scipy. Another equivalent version is implemented on both cpu and gpu using the Python package Pytorch. Results are reported for the Pytorch implementation on the gpu, which was three times faster than either cpu version. Our implementation is tailored to our specific quadratic program, which has an invertible diagonal matrix \( D \) and simple “Euclidean cone” constraints \( 0 \leq x \leq q \). Our implementation code will be posted publicly upon publication and is available immediately to interested readers upon request.
The positive definite matrix to be decomposed changes with each iteration because it is constructed by implicitly assigning weights to each order based on values of multipliers. The weights are close to zero when the multipliers push the order execution rate \( x_i \) close to the boundary of the interval \([0, q_i]\), and closer to one if the execution rate \( x_i \) implied by the multipliers is far away from the boundary of the interval \([0, q_i]\).

The order is expected to be relevant for price discovery at the margin when it is partially executable. A new Cholesky decomposition is needed on each iteration to incorporate updated weights from the most recent iteration into the calculation of the new search direction.

When there is little liquidity for some assets, the matrix to be decomposed is nearly singular. By supplying small amounts of liquidity to all assets, exchange trading improves the condition number of this matrix, which makes the Cholesky decomposition unlikely to fail. When there is insufficient exchange trading to prevent the Cholesky decomposition from failing, the algorithm regularizes the matrix by adding small quantities to the diagonal.

5.2 Results

5.2.1 Simulating the Order Book

Four sets of parametric assumptions are used to simulate an order book. A list of parameters is presented in Table 2, with base-case, high, and low values.

The first set of assumptions includes the number of assets, the number of orders, and the extent of exchange trading. We start with 500 assets and 100,000 orders per simulated order book as a base case. The number 500 is chosen based on the number of stocks in the S&P 500 index. The number 100,000 is chosen somewhat arbitrarily. In our base case, there will be an average of 100 orders per asset, as well as an additional 50,000 portfolio orders, which seems reasonable as an initial base case for the number of active orders in a given batch interval. After reporting outcomes for the base case, we study how the numbers of assets and orders affect computation.

Exchange trading is characterized by the slope of its demand schedule (\( \epsilon_n \) in equation (16)) and the base price. For simplicity, we set the slope constant across all assets, and choose a small number, \( 10^{-2} \), so that the exchange does minimal trading, buying one dollar’s worth of an asset when its price falls by one percent. The base price, when the exchange starts buying and selling assets, is set equal to the initial price. The ini-
Table 2: Parameters for simulating an order book.

<table>
<thead>
<tr>
<th>Description</th>
<th>Base</th>
<th>Low</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of assets</td>
<td>500</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>Number of orders</td>
<td>100000</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>Slope of exchange's demand schedule (shares traded per dollar price change at $100/share)</td>
<td>0.0100</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>Fraction of orders for individual asset</td>
<td>0.5000</td>
<td>0.0500</td>
<td>0.9500</td>
</tr>
<tr>
<td>Fraction of orders for indexes among orders for portfolios</td>
<td>0.5000</td>
<td>0.0500</td>
<td>0.9500</td>
</tr>
<tr>
<td>Number of size indexes</td>
<td>5</td>
<td>2</td>
<td>50</td>
</tr>
<tr>
<td>Number of industry indexes</td>
<td>10</td>
<td>2</td>
<td>50</td>
</tr>
<tr>
<td>Probability an index order is a market index order</td>
<td>0.8000</td>
<td>0.0500</td>
<td>0.9500</td>
</tr>
<tr>
<td>Probability a size or industry index ord is a size index order</td>
<td>0.5000</td>
<td>0.0500</td>
<td>0.9500</td>
</tr>
<tr>
<td>Probability a mkt index order is an EW mkt index order</td>
<td>0.0625</td>
<td>0.0500</td>
<td>0.9500</td>
</tr>
<tr>
<td>Probability a size index order is an EW size index order</td>
<td>0.2500</td>
<td>0.0500</td>
<td>0.9500</td>
</tr>
<tr>
<td>Probability an industry index order is an EW industry index order</td>
<td>0.2500</td>
<td>0.0500</td>
<td>0.9500</td>
</tr>
<tr>
<td>Standard deviation of expected number of orders across assets</td>
<td>1.7000</td>
<td>0.1000</td>
<td>3.0000</td>
</tr>
<tr>
<td>Standard deviation of order size given asset</td>
<td>1.5000</td>
<td>0.1000</td>
<td>3.0000</td>
</tr>
<tr>
<td>Standard deviation of upper limit price as fraction of initial price</td>
<td>0.1000</td>
<td>0.0100</td>
<td>1.0000</td>
</tr>
<tr>
<td>Mean deviation of upper limit price as fraction of initial price standard deviation</td>
<td>0.3000</td>
<td>0.0100</td>
<td>1.0000</td>
</tr>
<tr>
<td>Mean difference between upper and lower limit prices (basis points)</td>
<td>1.0000</td>
<td>0.0100</td>
<td>100.0000</td>
</tr>
<tr>
<td>Standard deviation of difference between upper and lower limit prices</td>
<td>2.0000</td>
<td>0.1000</td>
<td>3.0000</td>
</tr>
<tr>
<td>Fraction buy orders for indexes and assets</td>
<td>0.5000</td>
<td>0.1000</td>
<td>0.9500</td>
</tr>
</tbody>
</table>

The potential price of all assets and index portfolios is normalized to $100. These prices are pure scaling factors, which do not affect the economics of market clearing in any way.

The second set of assumptions defines how orders are distributed across individual assets and various types of portfolios and how orders are distributed between buy and sell. Theoretically, investors can choose from infinitely many different portfolios by combining any of the 500 assets with arbitrary weights for each asset. In our simulations, however, we restrict portfolios to six different types of index portfolios and to randomly generated pairs trades. This is a limitation of our simulation environment.

For index portfolios, we construct value-weighted and equal-weighted portfolios of the market index, “size” indices, and “industry” indices. Assets are sorted by expected dollar volumes, assigned to size quantiles, and the groups created form size indices. Assets are evenly allocated to industries by first sorting assets by size, then assigning assets to the same industry if they have the same rank modulo the number of industries. Pairs trades randomly buy either an asset or an index portfolio and sell an equal expected dollar value of another random asset or index.

In the base case, we divide the 100,000 orders evenly between orders for individual assets and orders for portfolios. The orders for portfolios are then divided evenly between index portfolios and pairs trades. Orders for index portfolios are randomly assigned to the six categories with corresponding probabilities in parentheses: the valued-weighted market index (75%), the equal-weighted market index (5%), five value-weighted
size indices (7.5%), five equally-weighted size indices (2.5%), ten value-weighted industry indices (7.5%), and ten equal-weighted industry indices (2.5%). The numbers here are chosen somewhat arbitrarily, except to reflect that the value-weighted market indices, such as those which track the S&P 500 (including even the CME’s S&P 500 E-mini contract), have high trading volume. We later vary the probabilities to study how they may affect computation times.\textsuperscript{18} Finally, each order for individual assets and indexes has an equal probability of being buy or sell.

The third set of assumptions concerns the distribution of the number of orders across assets and the size of orders. To represent the skewness of trading volume in the real world, we allow the number and size of orders to be drawn from heterogeneous distributions.

For each asset, we draw a random number from a lognormal distribution with mean of 1 and standard deviation of 1.7. (Here, we adopt the convention that the mean of a lognormal random variable refers to the mean of the random variable itself, and the standard deviation refers to the standard deviation of the logarithm of the random variable.\textsuperscript{19}) Dividing these numbers by the sum of all realizations across 500 assets, we generate the probability that a given order is allocated to that asset. Then for each order for individual assets, we pick an asset from a multinomial distribution with the chosen probabilities. The probability multiplied by the total number of orders for assets (50,000) is the expected number of orders for that asset.

The size of orders for a given asset is lognormally distributed with standard deviation of 1.5. Following the market microstructure invariance hypothesis of Kyle and Obizhaeva (2016), the mean order size is set proportionally to the square root of the expected number of orders for that asset. The proportionality constant is chosen to make the aggregate expected order volume from individual stocks equal to the arbitrary scaling constant of $10 million per batch using arbitrary expected ex-ante prices of $100 per

\textsuperscript{18}To allow conveniently varying these probabilities, we generate them from five parameters: the probability that an index order is for either the equal-weighted or the value-weighted market index; the probability that a non-market index order is for a size index portfolio; the probability that a market index order is for the equal-weighted market index portfolio; and the probability that a size (industry) index order is for an equal-weighted size (industry) index portfolio. The five parameters and the restriction that the probabilities sum to one determine all six probabilities. We let each of the five parameters vary from 5\% to 95\%.

\textsuperscript{19}Specifically, the random variable can be written $\mu \exp(\sigma Z - \sigma^2/2)$, where $\mu$ is the mean of the random variable, $\sigma$ is the standard deviation of the logarithm, and $Z$ is a standard normal random variable. The standard deviation of the random variable itself is $\mu(\exp(\sigma^2) - 1)^{1/2}$. This implies that the mean $\mu$ has the same units as the random variable, and the standard deviation $\sigma$ is dimensionless.
share. Then the standard deviation of the order size equals \( \sqrt{\exp(1.5^2) - 1} \) multiplied by the mean, approximately two times the mean.

For index portfolios, the expected number of orders for each size index is the same, and the expected number of orders for each industry index is the same. The size of the index orders is determined by multiplying the square root of the expected number of orders by the same proportionality factor used for individual orders. Since orders for the value-weighted market index are much larger and more numerous than orders for individual stocks, the overall value of the market index is primarily determined by these index orders. For pairs trades, each individual asset leg is generated randomly in the same manner as orders for the asset or portfolio. The dollar size of the larger leg is then truncated to match the dollar size of the smaller leg, again using expected ex-ante prices.

The last set of assumptions concerns limit prices. First, the absolute value of the upper limit price \( p_{Hi} \) is lognormally distributed with a mean of $97 if buying and $103 if selling and a standard deviation of 0.1, which implies that the standard deviation of the upper limit price is approximately $10. Second, the difference between the upper and lower limit prices, \( p_{Hi} - p_{Li} \), is lognormally distributed with a mean of one basis point relative to the "midpoint" \( (p_{Li} + p_{Hi})/2 \) and a standard deviation of 2. If the midpoint is set at the mean value of $100 and \( p_{Hi} - p_{Li} \) is set equal to its mean of one basis point, we have \( p_{Hi} = $100.005 \) and \( p_{Li} = $99.995 \). A one standard deviation change makes \( p_{Hi} - p_{Li} \) vary by a factor of \( e^2 \approx 7.4 \). The problem is better behaved when orders have a larger region in which their demands are strictly downward sloping in the portfolio price. The small mean of one basis point is expected to stress the algorithm by making demand highly nonlinear in prices. The random variables are independently distributed.

### 5.2.2 Computation Outcomes

When performed on an ordinary office workstation (an AMD Ryzen Threadripper 3960X processor, 24 cores running at 3.8GHz, and 128GB of memory running at 3600MHz; RTZ 2070 gpu at 1710 MHZ with 8 GB of RAM), computation of market-clearing prices and quantities takes about 0.1451 seconds (median) in the baseline scenario with 500 assets and 100,000 orders. Our results are obtained using the gpu and two cores.\(^20\) Uncleared computation times do not change much when more cores are used. This is probably because easily parallelized computations are done on the gpu while not easily parallelized computations, like Cholesky decomposition and some sparse matrix operations, use the cpu but do not benefit much from using multiple cores. The computation times are stable across 401 repetitions, with a maximum of 0.1603 seconds, a minimum of 0.1365 seconds, and a standard deviation of 0.0058 seconds.
quantities are near zero, equal to 8.7 dollars per trillion dollars of total volume.\textsuperscript{21}

The amount of exchange trading is small. On average, the exchange trades 3.2 dollars per million dollars of trading volume. Across 51 repetitions, the maximum, minimum, and standard deviation of exchange trading are 5.99, 2.32, and 0.73 dollars per million dollars. In a dynamic market, the exchange can avoid accumulating significant inventories by adjusting its base prices over time to liquidate existing inventories.

Exchange trading, while small, has a significant effect on the cross-sectional standard deviation of market-clearing prices for assets with thin order books. Without exchange trading, assets with thin order books have essentially indeterminate prices. The algorithm chooses arbitrarily among multiple clearing prices. This drives the standard deviation of price changes from the baseline price to an immense value of 14.4 million percent. The small amount of exchange trading used in the simulations brings the standard deviation of prices down to a much more reasonable 14.19%. In practice, we would expect market-making firms to provide liquidity in thinly traded stocks and stabilize prices, possibly with a wide spread.

Next, we study how computation times vary with the number of assets and the number of orders; see Figure 2. In the first panel, as the number of orders increases from 100,000 to 1,000,000 and 10,000,000, while keeping the number of assets constant, computation times increase from 0.1451 to 0.5639 and 4.67 seconds. The computation time crosses one second with approximately 1,930,000 orders. When the number of orders is large, computation time is approximately proportional to the number of orders. In the second panel, as the number of assets increases from 500 to 2,000 and 10,000, again keeping the number of orders constant, computation times increase to 1.1021 and 56.3 seconds. The computation time crosses one second with 1,800 assets and ten seconds with 5,200 assets. The increased computation times when the number of assets increase are mainly due to the computation costs of constructing the matrix input to the Cholesky decomposition and the Cholesky decomposition itself (which is an $O(N^3)$ algorithm in the number of assets). Both figures are almost flat initially. With a small number of orders or assets, the overhead associated with the Python interpreter becomes a significant fraction of computation times. When there are only 10 assets and 20 orders, the computation time is about 0.0552 seconds, which we believe is likely a good estimate

\textsuperscript{21}All calculations are done with 64-bit floating-point numbers. If calculations are done with 32-bit floating-point numbers, computation time is more than twice as fast, but uncleared quantities are typically an unacceptably large $100 per million.
Figure 2: Computation Time As a Function of the Number of Orders and the Number of Assets
Panel A varies the number of orders. Panel B varies the number of assets. In both panels, all other parameters are set to their baseline values. Each dot represents one simulated order book, and there are approximately 500 simulations in each panel. The small discontinuity in Panel B around 600 assets likely is a gpu artifact, such as the need to use RAM rather than cache for sufficiently large problems.
of the overhead associated with the Python interpreter.

For robustness, we alter each parameter's value to the minimum and the maximum of a wide range, as described in Table 2, while keeping the number of orders, the number of assets, and the slope of exchange trading constant. Computation times remain of the same order of magnitude (0.1159 to 0.2655 seconds compared to 0.1459 seconds in the baseline setting). Most parameters have a modest effect on computation times except for the standard deviation of order size and the fraction of buy orders. These two parameters affect the balance of the supply and demand of the order book. Changing the standard deviation of order size from 1.5 to 3 increases computation time to 0.2078 seconds. Changing the fraction of buy orders from 0.5 to 0.1 increases computation time to 0.2344 seconds. By making the order book more asymmetric, extreme values for these parameters make the problem more difficult to solve.

While having little effect on computation times, the difference between upper and lower limit prices significantly affects the quality of market clearing. The low mean difference scenario (0.01 basis points) and the high standard deviation scenario (3.00) increase the amount of uncleared quantities from 8.7 dollars per trillion to 120 and 160 dollars per trillion. Further making the mean difference ten times smaller makes uncleared quantities ten times larger. This occurs because as the upper and lower limit prices become closer \( p_H - p_L \rightarrow p \), the upper limit price also becomes closer to the market-clearing price \( p_H - p_L \rightarrow p \) for all executable orders. The ratio between the two differences \( \frac{(p_H - p_L)}{(p_H - p_L)} \), each of which approaches zero, becomes numerically inaccurate. This makes the flow demand in equation (2), which depends on the ratio, and thus market-clearing quantities inaccurate and increases uncleared quantities.

Finally, we consider an extreme scenario by setting all parameters simultaneously to those that increase computation times (either the minimum or the maximum of the range depending on the parameters). In this case, the computation time increases to 0.4291 seconds, approximately a factor of three relative to the base case and still below half a second. The uncleared quantities increase to 1.5 dollars per million of total volume, mainly from varying the mean and the standard deviation of the difference between the upper and lower limit prices. The analysis suggests that computation times are not sensitive to the parameter values used for the order book construction, while market clearing accuracy is sensitive to order book parameters, especially the difference between the two limit prices. While 1.5 dollars per million is still arguably small as an amount of inaccuracy, it may be prudent to adopt a lower bound for the difference be-
between the upper and lower limit prices, such as one basis point in the baseline case.

**Discussion**  Overall, market-clearing allocations are computed quickly and accurately with minimal trading by the exchange. Computational times grow with the number of orders and assets, as expected. Reassuringly, the growth in the number of orders appears to be linear, and problems with several thousand assets can be solved in about one to ten seconds on an ordinary workstation. We interpret these results as an initial computational proof of concept for the flow trading market design.

In a production environment, we expect more powerful computers and more refined algorithms will make it easier to calculate market-clearing allocations with even greater speed. Future algorithms may be better able to take advantage of parallel processing across more cores and take advantage of advances in quadratic programming and sparse matrix multiplications, which play key roles in our computation and are active areas of research in computer science.

6 Microfoundation for Portfolio Orders

Flow orders specify demand for a user-specified portfolio as a function of the price of that portfolio. Although such portfolio orders are more general than limit orders, this language is still restrictive. In general, a market participant's demands depend on the complete vector of asset prices, not just on the price of the portfolio. This section provides a microfoundation for our approach to expressing trading demands.

6.1 The Static CARA-Normal Framework

The CARA-normal model (Grossman (1976), Grossman and Stiglitz (1980), Admati (1985)), in which agents have constant absolute risk aversion (CARA) and asset returns are jointly normally distributed, is widely used in economics and finance. We use the CARA-normal model to study trading portfolio orders. The model is static, so there is no distinction between trading in quantities and flows. Models that study dynamic strategic trading in the CARA-normal environment have found that trading gradually over time is optimal to manage price impact (Vayanos (1999); Du and Zhu (2017); Kyle, Obizhaeva, and Wang (2018), Sannikov and Skrzypacz (2016)). While these models focus on the case of a single risky asset, we conjecture that the insights would carry over to the trade of portfolios.
Assume there are $N$ risky assets and one safe asset, whose return is normalized to one. Assume there is a single trader who subjectively believes that the risky assets’ payoffs, denoted by vector $\mathbf{v}$, are joint-normally distributed with mean $\mathbf{m}$ and variance-covariance matrix $\Sigma$. The trader has CARA preferences with risk aversion parameter $A$. There are no wealth effects with CARA preferences, so the trader’s wealth is set to zero for simplicity.

Initially, consider the trader’s optimization problem given a fixed, known set of prices—let $\mathbf{\pi}$ denote the vector of prices for the $N$ risky assets. Assume that the trader is a perfect competitor who cannot affect these prices with their trading; we will discuss the case where the trader has price impact shortly. The trader’s portfolio optimization problem, given her beliefs, risk preferences, and prices, is given by:

$$\max_{\omega} \mathbb{E}[-\exp^{-A(\mathbf{v} - \mathbf{\pi})^\top \omega}], \quad (22)$$

Joint normality allows us to transform the above into the quadratic optimization problem:

$$\max_{\omega} \left( (\mathbf{m} - \mathbf{\pi})^\top \omega - \frac{1}{2} A \omega^\top \Sigma \omega \right). \quad (23)$$

The first order condition implies that the optimal portfolio is given by:

$$\omega^* = (A \Sigma)^{-1}(\mathbf{m} - \mathbf{\pi}). \quad (24)$$

Observe that the optimal demand for each asset depends on its covariance with the other assets, via the associated row of the inverse covariance matrix, and the entire vector $(\mathbf{m} - \mathbf{\pi})$. Thus, as is well known, demand for each asset generally depends on the prices of all assets.

**Implementing the Optimum with Portfolio Orders** If the prices $\mathbf{\pi}$ are known and fixed, the trader can implement their optimum as defined in equation (24) with a single portfolio order with portfolio weights $\mathbf{w}_i$ and quantity parameter $Q_i^{\max}$ such that $\mathbf{w}_i Q_i^{\max} = \omega^*$. This single portfolio order would specify pricing parameters such that it is fully executable at the known prices.

What if the trader does not know the asset prices? This might reflect the environment in which prices are rapidly fluctuating over time. Next, we show that traders can implement their optimum according to equation (24) with portfolio orders, without any
knowledge of prices. To do this, we need to rotate the asset space such that independent portfolios span it.

Since the variance-covariance matrix $\Sigma$ is positive semidefinite, its singular value decomposition has a form

$$\Sigma = U \Delta U^T,$$

where $U$ is an orthonormal matrix, and $\Delta$ is a diagonal matrix with nonnegative elements. Let $K \leq N$ denote the rank of $\Sigma$, let $\delta_i$ denote the $i$th nonzero diagonal entry of $\Delta$, and let $u_i$ denote the corresponding column of $U$. \(^{22}\) Then we have

$$\Sigma^{-1} = \sum_{i=1}^{K} \frac{1}{\delta_i} u_i u_i^T. \quad (26)$$

Using this, we can express the optimal portfolio in equation (24) as

$$\omega^* = \sum_{i=1}^{K} \left( \frac{u_i^T m - u_i^T \pi}{A \delta_i} \right) u_i, \quad (27)$$

which is a combination of demand schedules for portfolios. Here, $u_1, \ldots, u_K$ are portfolios of assets, whereas in equation (24) demand was expressed in terms of individual assets. Since the portfolios are independent of one another and there is no wealth effect, the optimal portfolio chooses the demand for each of them separately as if in a single-asset model. \(^{23}\) That is, the optimal demand for the $i$th portfolio is given by

$$\frac{1}{A \delta_i} (u_i^T m - u_i^T \pi), \quad (28)$$

where $\delta_i, u_i^T m,$ and $u_i^T \pi$ correspond to the variance, the expected payoff, and the price of the portfolio $u_i$. Since the demand for each portfolio only depends on the portfolio’s price, traders can achieve the optimal trade in equation (24) by utilizing $K$ orders for portfolios where each order is a function of that portfolio’s price.

Recall, we require orders’ demands for portfolios to be downward sloping. Since the optimal demand for each portfolio in equation (27) is decreasing in the portfolio’s price, the demand is indeed downward sloping.

\(^{22}\)When $K$ is strictly less than $N$, i.e., the matrix $\Sigma$ is positive semidefinite but not positive definite, we can use the pseudo-inverse instead of the inverse to define the demand function.

\(^{23}\)Observe that in equation (24), if the covariance matrix $\Sigma$ is diagonal, then the demand coefficients on each of the individual assets are scalars, so the optimal portfolio can choose the demand for each asset separately as if in a single-asset model too.
The theorem below summarizes the results.

**Theorem 4.** Consider a static CARA-normal framework in which a trader believes that the variance-covariance matrix of the asset payoffs has rank \( K \). Then the trader’s optimal portfolio (equation (24)) can be represented as the sum of \( K \) downward-sloping demand schedules for portfolios, each of which depends only on that portfolio’s price (equation (27)).

**Practical Implementation** We can decompose the expected utility from the optimal portfolio into the contribution of each rotated asset. Substituting the optimal portfolio in equation (27) into equation (23), and some algebraic manipulations (see details in Appendix A), allows us to express the expected utility from trading at prices \( \pi \) as

\[
\sum_{i=1}^{K} \frac{1}{2A} \left( \frac{u_i^T m - u_i^T \pi}{\sqrt{\delta_i}} \right)^2.
\]  

(29)

This formula shows that the benefit of each portfolio is determined by its squared Sharpe ratio as perceived by the trader.\(^{24}\) In practice, traders may select a few portfolios, which they perceive to have a sufficiently high Sharpe ratio (more precisely, its absolute value), and choose to trade only those portfolios rather than all \( K \) portfolios.

**Price Impact and Strategic Trading** Thus far, we have assumed that traders are perfect competitors, behaving as if they have no price impact. In practice, trades can move prices. Many institutional traders dedicate considerable time and resources to managing their price impact. Now we show that flow orders can still be used to implement the optimal portfolio when traders behave strategically, considering their price impact.

Following the literature (for example, Kyle (1989); Malamud and Rostek (2017)), we assume that traders believe that their price impact is linear in the quantity they trade. We further assume that the matrix of price impact is positive semidefinite.\(^{25}\) That is, for

---

\(^{24}\)Recall, the Sharpe ratio refers to the risk premium (i.e., the expected return minus risk-free rate) divided by the standard deviation. Here, the risk-free rate is zero since the safe asset’s return is normalized to one.

\(^{25}\)Malamud and Rostek (2017) show that when the variance-covariance matrix is the same for all traders, each trader’s equilibrium price impact matrix is proportional to the variance-covariance matrix, which implies that all price impact matrices are positive semidefinite. It is left for future study to determine under what conditions the price impact matrix is positive semidefinite in a more general setting.
each trader, there is an $N \times N$ positive semidefinite matrix $\Lambda$, such that

$$\pi = \pi_0 + \Lambda \omega,$$  \hspace{1cm} (30)

where $\pi_0$ is the vector of hypothetical prices that would prevail if the trader were not to trade, and the $n$th row of $\Lambda$ corresponds to the marginal impact of trading assets 1 to $N$ on the price of asset $n$. With a slight abuse of notation, we use the demand schedule $\omega$ to also refer to the actual quantities that a trader trades at given prices.

With price impact, the trader’s optimal strategy is a slight modification of the competitive solution in equation (24), given by

$$\omega^* = (A \Sigma + \Lambda)^{-1}(m - \pi).$$  \hspace{1cm} (31)

Since the sum of two positive semidefinite matrices is also positive semidefinite, $\Lambda \Sigma + \Lambda$ is positive semidefinite. Thus, we can use singular value decomposition to rotate the asset space such that independent portfolios span it. Then the same logic as above implies that the optimal portfolio can be implemented by combining portfolio orders that only depend on the portfolio’s price. The number of required portfolio orders corresponds to the rank of $A \Sigma + \Lambda$.

**Theorem 5.** Consider a static CARA-normal framework in which a trader believes that her price impact is linear and positive semidefinite (equation (30)). Then the strategic trader’s optimal portfolio (equation (31)) can be represented as the sum of downward-sloping demand schedules for portfolios, each of which depends only on that portfolio’s price.

Recall, when proving the existence and uniqueness of market-clearing quantities in Section 4, we treat orders as if they represent traders’ true valuations. This simplification does not imply that we can infer traders’ valuations from their orders. Strategic trading is an important reason that there often is a gap between true and as-bid valuations.

6.2 Approximations for General Preferences and Limitations

The logic above extends to any strictly concave, twice continuously differentiable, quasi-linear utility function over assets provided asset payoffs are joint normally distributed. To see why this is the case, recall the two CARA preference properties we use: 1) no wealth effects and 2) strict concavity.
First, with no wealth effects, a trader’s optimal demand for each asset does not depend on the prices of other assets in the case where assets’ payoffs are independent of one another. If the assets have correlated, joint normal payoffs, we can, as shown above, rotate the asset space such that it is spanned by a set of portfolios whose payoffs are independent of one another. Thus, any quasilinear preference implies no wealth effects, and we can use the independent portfolios such that the trader’s optimal demand for each portfolio only depends on that portfolio’s price.

Second, strict concavity implies that the optimal demand for any portfolio must be downward sloping. This is crucial since we require demands for portfolios to be downward sloping in the portfolio’s price. Although the optimal strategy may not be linear for any strictly concave, twice continuously differentiable, quasilinear preference, we can approximate the optimal strategy by combining multiple linear downward sloping schedules.

However, portfolio orders will be unable to approximate the optimal portfolio of every concave utility function. First, with wealth effects, the demand for an independent portfolio may still depend on the prices of other portfolios and may also increase in that portfolio’s price. Second, with asymmetric information, the prices of other portfolios may be useful to learn about the payoff of a given portfolio, even if the payoffs of the two portfolios are independently distributed. In this case, the optimal demand for an independent portfolio may again depend on the prices of other portfolios.

7 Implementation and Policy Issues

This section discusses several practical implementation and policy issues related to the flow trading market design.

Batch Interval  What is the optimal batch interval? The best choice likely depends on the asset class. We discuss three considerations: computation limits, factors that favor a shorter interval subject to computation constraints, and an open question about whether a longer interval is desirable for thinly traded assets.

The batch interval must be long enough to compute prices and trades. Our simulations suggest that a batch interval on the order of one second may be sufficient for many asset classes. That said, the computational simulations serve as a proof-of-concept rather than as a final word. There are many reasons why a real-world implementation
could be meaningfully faster than our implementation. There is also the possibility that real-world markets may take longer to compute for reasons not anticipated by our simulation environment.

Next, three factors favor a batch interval as short as computationally feasible. First, a fast batch interval makes trading smoother—that is, smaller quantities are traded per batch. Smoother trading can be helpful to traders with complex dynamic trading strategies, who may wish to adjust their orders over time as information evolves. At the same time, traders with simpler strategies can leave their orders be, without much adjustment over time, whether the batch interval is long or short. Second, if the assets in question are traded in fragmented markets, and some of those markets are continuous, then the interaction of a discrete-time market with a continuous-time market is likely simplest if the discrete-time interval is short (Budish, Lee, and Shim (2018)). Third, information policy is more robust with a shorter batch interval, as there will be less pressure for within-batch information dissemination.

Last, we acknowledge the open question of whether a longer batch interval is desirable for more thinly traded assets. This is a common intuition among regulators and practitioners (see, e.g., U.S. Securities and Exchange Commission (2019a,b), Schwartz (2012), and references therein). Du and Zhu (2017) provide some theoretical grounding for this intuition. A hard conceptual question is how to think about the batch interval for markets consisting of heavily traded and thinly traded assets. For instance, the US equities market consists of about 7000 assets, some of which trade many times per second while others trade only a few times per hour. Similarly, in many sovereign debt markets, on-the-run assets are heavily traded while off-the-run assets are thinly traded.

**Information Policy** Information policy is typically discussed in terms of pre-trade and post-trade transparency. Concerning post-trade transparency, we propose that the exchange publish the trading volume and clearing price of each asset promptly after the quantities and price have been calculated. In addition, the exchange may also publish information about the aggregate net demand curve for each asset, holding the prices of all other assets fixed. This policy is analogous to publishing the outstanding limit order book in continuous markets. Then traders can make inferences about the price impact costs of their orders. The exchange would not publish information about the identity of traders, nor would it publish individual orders, since portfolio weights may reveal trading strategies.
For pre-trade transparency, we envision that the post-trade information from the auction at time $t$ is the complete pre-trade information for the auction at time $t + 1$. As discussed by Budish, Cramton, and Shim (2015), this is the appropriate discrete-time analog of information policy in the continuous market. In both cases, the exchange (i) receives an order, (ii) economically processes the order, and then (iii) disseminates information about what happened (e.g., a trade or an order book update). The difference with discrete time is that the economic processing in (ii) occurs in discrete time, and hence the information dissemination in (iii) occurs in discrete time as well.\footnote{As pointed out by Budish, Cramton, and Shim (2015), in the continuous-time market it may look like traders can see information about the state of an asset's order book “right now,” but that is an illusion — a trader's information is always as of a latency ago because it takes non-zero time for the exchange's matching engine to economically process new messages in step (ii) and to disseminate updates in step (iii). Discrete time makes more transparent that a trader's information as of time $t$ is the state of the order book as of time $t - \Delta$, whether $\Delta$ is the latency of information travel in a continuous market or the duration of the batch interval in a discrete market. Discrete time also eliminates the arms race for speed to reduce $\Delta$.}

The reason not to disseminate additional information between batch auctions, e.g., about the arrival of new orders or the cancelations of outstanding orders, is that such information could lead to gaming. For example, suppose the batch interval is one second. A trader could submit a new order to buy a large quantity 100 milliseconds into the batch interval with the intention to cancel that order and instead send a new order to sell 999 milliseconds into the batch interval. In this scenario, the order to buy was never economically binding nor economically processed by the exchange, so sending this purposefully misleading message was costless.

We acknowledge that the pressure to disseminate information between auctions grows with the batch interval. This is one reason why a batch interval that is as short as computationally feasible may be appropriate for many asset classes. Additionally, one could extend flow trading to include order types that are economically binding for the duration of the current auction (i.e., cannot be canceled until after the next auction), with within-auction information disseminated about updates to this binding subset of the order book. Professional market-making firms might deploy such orders to attract trading volume, but we view this discussion as speculative and in need of future research.

With arbitrary portfolio orders, information about the depth of the order book is inherently complex — there are infinitely many possible portfolios. The exchange might publish limited depth information about a fixed list of reference portfolios, alongside the depth information for individual assets.
**Trust and Transparency**  Flow trading has the desirable property that all orders that are executable at published clearing prices do in fact execute. This property allows investors to confirm that their orders received correct execution from published prices.

By contrast, potentially executable orders in current markets do not consistently execute when other orders execute at the same price at nearly the same time. Uncertain execution erodes trust and market confidence, particularly among traders without state-of-the-art speed technology, whose orders are more apt to get poorer execution.

This difference between flow trading and the current market design arises from combining discrete time and continuous prices and quantities. Continuous prices and quantities make it possible to execute all executable orders at a market-clearing price without any need for rationing. Discrete time makes it possible to process multiple executable orders simultaneously in a batch process.

**Fairness**  In traditional markets, the concept of “bid-ask spread” captures many of the features participants complain about as unfair. When there is a minimum tick size and the bid-ask spread is one-tick wide, buyers and sellers cannot offer price improvement by quoting better prices between the best bid price and best offer price. Instead, buyers and sellers queue up at the best bid and offer, where the fastest traders have the highest priority in the queue. Slower traders perceive this as unfair. In dealer markets, dealers do not allow customers to post limit orders to trade directly with other customers. Instead, customers must trade with dealers in transactions where the dealer buys at the bid price and sells at the offer price. Customers perceive that dealer markets are unfair because dealers have privileges that customers do not have.

With flow trading, the concept of bid-ask spread is irrelevant when trade occurs because the market demand schedule for each asset is continuous and strictly downward sloping. All trades clear at the same price. All executable orders execute. There are of course still trading costs. Trading a larger quantity, or trading a given quantity faster, requires offering a better price — that is, walking up the market’s supply curve if buying or down the market’s demand curve if selling — which creates price impact. The essential difference is that a trader can trade an epsilon quantity at the market-clearing price without any bid-ask spread. A practical interpretation of this point is that institutional investors will have to manage their price impact, but small retail investors can trade small quantities at the market-clearing price with negligible trading costs.
Regulatory Objectives  The US Securities and Exchange Commission, which regulates US securities markets, pursues various policy objectives, including economic efficiency, competition, maintaining trust and confidence, and investor protection.

Flow trading is consistent with these objectives. It improves economic efficiency by reducing wasteful expenditure on fast data feeds, communication technologies, and trading algorithms. It does this by decreasing the arms race among traders to pick off orders and reducing the messages needed to implement dynamic trading strategies. It increases competition by enabling institutional investors to trade arbitrary long-short portfolios without the need for sophisticated trading platforms. Flow trading is also consistent with the current demand of small investors to trade fractions of shares and construct diversified portfolios consisting of tiny positions in many stocks. It promotes trust and confidence in markets by trading all executable orders at the same transparent price. It protects investors from poor order execution by making quality of order execution easy for customers to measure.

8 Conclusion

This paper has introduced a new market design for trading financial assets, such as stocks, bonds, futures, and currencies. It combines three elements: flow orders from Kyle and Lee (2017), frequent batch auctions from Budish, Cramton, and Shim (2015), and a novel language for trading portfolios of assets. Technical foundations for the proposed market design include existence and uniqueness results, computational results, and microfoundations for portfolio orders.

The combination of flow orders and frequent batch auctions yields a market design in which time is discrete, and prices and quantities are continuous. The status quo market design has these reversed. As has been widely documented, treating time as a continuous variable and imposing discreteness on prices and quantities causes significant complexity, inefficiency, and rent-seeking in modern financial markets. Policy debates on the arms race for trading speed, the proliferation of complex order types, the importance of proprietary market data and exchange access, the cat-and-mouse game between institutional investors and high-frequency traders, and the internalization of retail investors’ order flow, all relate to continuous time and discrete prices and quantities.

The novel language for portfolio orders is, on the one hand, rich enough to allow
traders to directly express many important kinds of trading demands — customized ETFs, pairs trades, general long-short strategies, general market-making strategies, all with tunable urgency — while also allowing for guaranteed existence and uniqueness of equilibrium prices and quantities and their fast computation. This seems a useful new point on the frontier of language design, that is, an attractive tradeoff between expressiveness and computability.

An open topic left for future research is the efficiency and welfare consequences of allowing market participants to directly trade portfolios, rather than trading separately for all individual assets. We conjecture there are two main efficiency benefits. First, allowing market participants to directly trade portfolios should reduce trading costs, such as the costs of trade execution and intermediation. This includes costs paid indirectly for portfolio trade execution through ETF fees. Second, portfolio orders make it more efficient for sophisticated financial market participants to endogenously link prices and liquidity provision for correlated assets. Portfolio orders enable, for example, Bertrand competition on the cost of executing a Buy A, Sell B pairs trade, which is impossible under the status quo market design. This should improve the efficiency of price discovery and reduce arbitrage rents. Another interesting dimension of the problem is how allowing participants to trade portfolios affects strategic issues around demand- and supply-reduction to manage price impact. In models with fully-contingent trading demands, these effects can go in either direction (Rostek and Yoon 2021, Wittwer 2021).
References


Budish, Eric, and Judd B Kessler. forthcoming. “Bringing real market participants’ real preferences into the lab: An experiment that changed the course allocation mechanism at Wharton.” *Management Science*.


Tyc, Stephane. 2014. “A technological solution to best execution and excessive market complexity.” *Quincy Data, LLC*.


Appendix

A Proofs

**Derivation of Equation** (29) Recall, from equation (23), the expected utility from the optimal portfolio is

\[(m - \pi)^\top \omega^* - \frac{1}{2} A \omega^{*\top} \Sigma \omega^*.\]  

(32)

Equalizing the marginal benefit (the expected return) and the marginal cost (risk), the optimal portfolio in equation (27) is essentially the ratio of the expected return to risk.

Substituting the optimal portfolio in equation (27) into the first term above, we have

\[(m - \pi)^\top \omega^* = (m - \pi)^\top \sum_{i=1}^{K} \mathbf{u}_i \left( \frac{\mathbf{u}_i^\top (m - \pi)}{A \delta_i} \right)\]

\[= \sum_{i=1}^{K} (m^\top \mathbf{u}_i - \pi^\top \mathbf{u}_i) \left( \frac{\mathbf{u}_i^\top (m - \pi)}{A \delta_i} \right)\]

\[= \sum_{i=1}^{K} \left( \frac{\mathbf{u}_i^\top (m - \pi)}{A \delta_i} \right)^2 = \frac{1}{A} \sum_{i=1}^{K} \left( \frac{\mathbf{u}_i^\top (m - \pi)}{\sqrt{\delta_i}} \right)^2.\]  

(33)

Notice, \(\mathbf{u}_i^\top (m - \pi)\) is a scalar and thus symmetric. Thus, the total expected return from the optimal portfolio is represented by the sum of squared Sharpe ratios of rotated portfolios, divided by risk aversion.

Now, we want to do the same thing to the second term in the expected utility.

\[\frac{1}{2} A \omega^{*\top} \Sigma \omega^*.\]  

(34)

Here, since \(\Sigma = \mathbf{U} \Delta \mathbf{U}^\top\), and \(\Delta\) is a diagonal matrix, we can express it as

\[\Sigma = \mathbf{U} \Delta \mathbf{U}^\top = \sum_{i=1}^{K} \delta_i \mathbf{u}_i \mathbf{u}_i^\top.\]  

(35)

Also, \(\mathbf{U}\) is an orthonormal matrix, which implies that \(\mathbf{U} \mathbf{U}^\top = \mathbf{I}\), an identity matrix. That is, \(\mathbf{u}_i^\top \mathbf{u}_i = 1, \forall i\) and \(\mathbf{u}_j^\top \mathbf{u}_i = 0, \forall j \neq i\). Then substituting the optimal portfolio, we have
Thus, similar to the total expected return, the total risk from the optimal portfolio is represented as the sum of squared Sharpe ratios of rotated portfolios, except that it is divided by 2 times the risk aversion. Thus, the total risk is exactly half of the total expected return, where half comes from the fact that the risk is a quadratic function of the portfolio, while the return is linear.

Finally, combining equations (33) and (36) yields equation (29).