Proving You Can Pick Stocks Without Revealing How

Alex Chinco[*]

April 6, 2024
[Click here for the latest version]

Abstract

You are raising capital to trade on a profitable new stock-picking strategy. You have approached a wealthy investor, Alice, for seed funding. To convince Alice, you must offer proof that you can pick winning stocks while revealing nothing about how you do it. The standard way to do this right now is through trading, but this approach is expensive and does not speak to the novelty of your strategy. In this paper, I show how to give Alice a zero-cost zero-knowledge proof that you can pick stocks in a way that is different from what others are doing.

Keywords: Zero-Knowledge Proofs, Homomorphic Encryption, Active Management, Emerging Managers, Cross-sectional Predictability

[*]Baruch College, Zicklin School of Business. alexchinco@gmail.com

I would like to thank Snehal Banerjee, Zahi Ben-David, Will Cong, Michele Fabi, Tyler Muir, Dave Solomon, Sunil Wahal, Michael Weber, and Jeff Wurgler for extremely helpful comments and excellent suggestions. This paper has also benefited from the generous feedback provided by seminar audiences and conference participants.

Introduction

Imagine that you are an emerging manager who knows about a brand-new way to pick stocks. When your stock-picking rule says “buy”, a stock tends to have above-average returns. When your rule says “sell”, the stock usually has below-average returns. No one else has thought to trade like this before.

Return predictability and trading profits are two sides of the same coin (Fama, 1976). So you could make money by going long the stocks with buy recommendations and selling short the ones marked as sell. You are currently trying to raise capital to do just that. This is why you are sitting in a pitch meeting with Alice, a wealthy potential investor.

Of course, anyone can claim to be skilled after the fact. Once returns have been realized on the 30th of the month, it is easy to come up with long and short positions that would have been profitable to trade on had you known about them on the 1st. So before giving you any money, Alice wants to see proof that you knew about your stock-picking rule ahead of time.

One way to provide such a proof would be to trade on your rule with the small amount of money you have right now. As an emerging manager, you do not currently have much personal wealth to draw on. But you do have some. Suppose that if you had used this money to go long the stocks where your new rule said “buy” and short the ones where it said “sell”, then you would have earned a 10% profit. If you were able to show Alice this 10% value at the top of a profit-and-loss statement on the 30th, she could be certain that you knew about a profitable stock-picking rule on the 1st.

The main benefit of giving Alice this sort of proof-by-trading is that it reveals no additional information about how you pick stocks, only that you can do it. In other words, trading produces a “zero-knowledge proof” (Goldwasser, Micali, and Rackoff, 1989) of your stock-picking skill.

After returns have been realized on the 30th of the month, Alice knows which stocks she should have bought and which she should have sold on the 1st. With the benefit of hindsight, Alice can think up a vast number of different combinations of buy and sell recommendations that would also have produced a 10% profit last month. She just cannot trade on any of these combinations because she did not know how to compute them on the 1st of the month. When Alice sees 10% at the top of your profit-and-loss statement on the 30th, she learns that you did know how to compute your positions on the 1st. However, since she can think up so many other equally profitable positions in hindsight, she learns nothing else about your particular stock-picking rule.

This logic is computational not Bayesian. It follows the contours of the "simulation paradigm" (Goldwasser and Micali, 1982), the standard approach to modeling secrecy in computer science. It does not make sense to apply Bayes' rule. Alice cannot have priors about the likelihood of encountering a manager with knowledge of your specific stock-picking rule. If she knew enough to form such priors, she would already be trading on your stock-picking rule herself.

Unfortunately, there are two important drawbacks to proof-by-trading. First, trading strategies need to be funded. And, as an emerging manager, you do not currently have much cash. Even if the long and short positions implied by your stock-picking rule exactly cancel each other out, you still have to pay trading costs, cover margin, etc. You must pay a large startup cost to begin trading.[1]

Second, proof-by-trading says nothing about the novelty of your stock-picking rule. Alice has already invested money with a number of other managers. She is not interested in whether it would be profitable to invest money with you in isolation. She does not care about whether your stock-picking rule predicts raw returns. She cares about whether it predicts the residual component of returns that she herself cannot predict. Alice wants to know whether it would be profitable to add your return profile to her existing portfolio. Even if you have stock-picking skill, Alice may not want to invest money with you if you are just duplicating the positions of an existing manager.

In this paper, I develop an alternative way for you to prove your stock-picking skill to Alice that solves both these problems. In other words, I introduce a zero-cost zero-knowledge protocol for proving novel stock-picking skill. Under a homomorphic encryption scheme (Rivest, Adleman, and Dertouzos, 1978; Gentry, 2009), it is possible to add and multiply pairs of encrypted numbers and

[1]Sonali Basak. "Veteran Hedge Fund Manager on 'Insane' Startup Costs." *Bloomberg*. Jun 1 2023.

then decrypt the result to reveal the mathematically correct answer. Thus, if Alice sends you encrypted numbers, you can add and multiply them together without needing to decrypt them first. This fact allows you to compute the accuracy of your stock-picking rule without revealing any additional information about how it works to Alice. At the end of the day, building portfolio positions is expensive. Sending encrypted messages is free.

Before anything else happens on the 1st of the month, you start by computing a buy/sell recommendation for each stock. Then, after returns have been realized on the 30th, Alice encrypts the component of each stock's return that she failed to predict, and she sends you this encrypted ciphertext. Because she uses a homomorphic encryption scheme, this ciphertext will allow you to calculate whether it would have been profitable to trade on your stock-picking rule given Alice's residual returns. The result of your calculation will be encrypted, so it will look like gibberish to you. But Alice will be able to decrypt it when you send it to her. If she sees a large value, Alice can conclude that you knew about a profitable new way to pick stocks at the start of the month.

This novel protocol solves both of the problems associated with proof-by-trading. First, it requires no upfront costs because you are exchanging encrypted messages with Alice rather than building costly portfolio positions. Second, because it verifies that your buy/sell recommendations predict Alice's residual returns rather than raw returns, the protocol demonstrates that you can pick stocks in a way that is different from what Alice is already doing. It does not just prove you have skill. It proves you have novel skill.

When using the proof-by-encryption protocol, Alice never sees your specific recommendation for any particular stock on the 1st of the month. She only ever sees the profit from trading on these recommendations given her residual returns. And after these returns have been realized on the 30th, she can think of a vast number of different combinations of buy and sell recommendations that would have been equally profitable to trade on. When following my protocol, Alice would have received an equivalent encrypted ciphertext if you had been trading on any of these other combinations. So Alice can learn nothing else about the particular combination you are trading on besides its profitability.

My new protocol offers the same security guarantee as proof-by-trading. Proof-by-encryption confirms that you knew about profitable buy/sell recommendations on the 1st of the month. But it reveals no additional information about how you generated them because Alice can think of lots of ways to produce equally accurate predictions on the 30th.

Proof-by-encryption can be implemented using a number of existing homomorphic encryption schemes. I provide an example of what an encrypted buy order would look like using the approach in Brakerski and Vaikuntanathan (2011). I use simulations to verify the numerical precision of encrypted estimates of stock-picking skill under this approach. While my baseline setup involves binary buy/sell recommendations, it is also possible to extend proof-by-encryption to cover continuous stock-picking rules.

Moreover, managers typically trade on only their strongest predictions. A stock-picking rule might involve sorting stocks by some characteristic and then going long/short the highest/lowest decile. In this setting, I show that proof-by-encryption actually produces a more informative signal than proof-by-trading. There is information in the predictions you made but chose not to trade on. This information does not show up in your profit-and-loss statement.

It is possible to characterize when a zero-knowledge proof of stock-picking skill must exist more generally. The key insight is that, under standard assumptions, all problems in "NP have zero-knowledge proof systems (Goldreich, Micali, and Wigderson, 1991)". NP problems have solutions that are easy to verify once you see one.[2] So to rule out the possibility of a zero-knowledge proof of skill, it would have to be computationally hard for Alice to verify your skill even if she knew how you were picking stocks. This is unrealistic. There is a reason why traders are forced to sign non-compete agreements.[3] Current employers worry about them leaving and describing their most profitable ideas to someone else. This fear only makes sense if these ideas are easy to recognize.

[2]A crossword puzzle is a good example of an NP problem (Garey and Johnson, 2002). Solving this Sunday's grid might take you an hour, but it will only take a second to verify your guess for 31 down once you see the answer key in next week's paper.

[3]Lindsay Fortado. "DE Shaw Imposes Non-Compete Demands On Staff." *The Financial Times.* September 19th, 2019.

Right now, emerging managers like yourself must rely on private wealth or personal connections to raise seed capital. Then, after trading for a year or two, if you want to grow your fund, you would then have to convince investment consultants who serve as gatekeepers for their large institutional clients. The process is expensive and time-consuming. Investment consultants represented roughly $47t in institutional assets as of 2022.[4] And at the moment an emerging manager must "have [at least] a three-year track record to be in a consultant database."[5] There is skill involved. Some people are better at it than others, and these people are not necessarily the ones with the most profitable trading ideas.

Proof-by-encryption offers a way to bypass this arrangement and create a spot market for profitable trading ideas. This is a market where you can sell the rights to trade on a profitable stock-picking rule at a price that depends only on the rule's performance. Before purchasing the rights to trade on an idea, a buyer can see how profitable it would be to add the stock-picking rule to their existing portfolio…but only via a zero-cost zero-knowledge proof. The buyer would not get to see your underlying approach to picking winners and losers.

The existence of such a spot market would allow emerging managers who lack personal capital and wealthy friends to raise seed capital. It is likely that, given how the odds are stacked against them, many would-by emerging managers never bother trying to raise capital in the first place. To address this problem, many large institutional investors now have funds earmarked for under-represented groups.[6] A spot market is a more direct solution to this problem, akin to running a "blind" orchestra audition (Goldin and Rouse, 2000).

A spot market for profitable trading ideas would also have numerous downstream consequences for the active management industry more broadly. Right now, due to privacy concerns, idea generation is done almost entirely in-house. The results in this paper suggest that there is no fundamental reason why this has to be the case. I show how a trader who is good at coming up with new

[4] https://www.pionline.com/specialreports/investment-consultants

[5] Amy Whyte. "Allocators Need Them. Asset Managers Resent Them. And Everyone Is Afraid of Them." *Institutional Investor.* Mar 4 2019.

[6] J.P. Morgan Asset Management. (Mar 8 2021) "J.P. Morgan launches Project Spark to support diverse emerging alternative investment managers."

stock-picking ideas can sell them to the highest bidder without fear of disclosing their secret sauce. This fact has clear implications for how profits are shared between managers (who have ideas) and allocators (who fund them).

Large institutional investors face a difficult portfolio problem (Basak and Pavlova, 2013). Given their size, these investors must give money to multiple active managers, and it is hard to tell whether two active managers are taking similar positions by studying their past track records. They have to worry about unknowingly doubling down on a particular trading strategy. This worry likely has played a role in the shift towards passive investing over the past two decades (Chinco and Sammon, 2023). Proof-by-encryption directly solves this problem by measuring whether a manager can produce uncorrelated alpha.

This is just the tip of the iceberg. Much of the active management industry is organized around preserving the secrecy of good trading ideas. It is hard to imagine all the knock-on effects of being able to sidestep this concern.

Paper Outline. Section 1 describes how to use homomorphic encryption to give a zero-cost zero-knowledge proof that you have novel stock-picking skill. Section 2 shows how to implement this protocol. Section 3 characterizes the robustness of this construction. Section 4 discusses how this proof system makes it possible to have a spot market for profitable trading ideas.

Notation. Uppercase Latin letters are counts indexed by the corresponding lowercase Latin letter—e.g., there are $n = 1, \ldots, N$ stocks. Greek letters are continuous parameters. Words in italics are *Variables*, words in sans serif are Sets, and words in teletype are `Protocols`. $\{0,1\}^\ell$ denotes all binary strings of length ℓ, and $\{0,1\}^* \overset{\text{def}}{=} \cup_{0 \leq \ell} \{0,1\}^\ell$ denotes the set of all binary strings.

Related Work. Previous researchers have studied the computational complexity of finding equilibrium allocations (Nisan, Roughgarden, Tardos, and Vazirani, 2007; Roughgarden, 2010; Papadimitriou, 2015). In particular, Nisan and Segal (2006) shows that, to arrive at a Pareto efficient allocation, you must convey at least as much information as is contained in prices. While proof-by-encryption might at first seem to contradict this finding, on closer inspection it does not. Stock-picking skill is not an equilibrium phenomenon.

It is now common to see computational considerations play a role in economics research. The applications range from finance (Hatfield, Kominers, Nichifor, Ostrovsky, and Westkamp, 2013; Chinco and Fos, 2021; Dworczak, Kominers, and Akbarpour, 2021; Akbarpour, Kominers, Li, Li, and Milgrom, 2023; Veldkamp, 2023) to game/auction theory (Mu'Alem and Nisan, 2008; Milgrom, 2021; Gonczarowski, Heffetz, and Thomas, 2023) to mechanism design (Roth, 2002; Milgrom, 2009; Budish, 2011; Budish, Che, Kojima, and Milgrom, 2013; Kominers, Teytelboym, and Crawford, 2017; Azevedo and Budish, 2019; Budish and Kessler, 2022).

However, at the moment, economists primarily think about computation as a constraint. "Computational constraints can be just as real as the constraints imposed by information or incentives. (Golowich and Li, 2022)" This paper pushes in a different direction. I constructively apply results from the computer science literature to open up new ways of organizing financial markets.

There is a strand of finance literature studying cryptocurrencies (Biais, Bisiere, Bouvard, Casamatta, and Menkveld, 2022; Makarov and Schoar, 2020; Cong, Li, and Wang, 2021, 2022; Sockin and Xiong, 2022; Griffin and Shams, 2020; Liu, Tsyvinski, and Wu, 2022; Duchin, Solomon, Tu, and Wang, 2022; Pagnotta, 2022). However, this work largely treats cryptocurrencies as yet another asset class. Cryptography plays little to no role in that analysis. By contrast, it is a central building block in this paper.

I know of only a handful of other economics papers that directly incorporate cryptographic protocols. Abbe et al. (2012) proposes that banks use them to disclose risk exposures, and Hastings et al. (2022) suggests using similar tools to perform stress tests. In an accounting context, Cao et al. (2019) and Cao et al. (2020) deploy zero-knowledge proofs to verify accounting statements rather than portfolio positions.

Recently, a number of computationally-minded papers have shed new light on how trading is currently organized via exchanges (Rostek and Yoon, 2021; Wittwer, 2021; Budish, Cramton, Kyle, Lee, and Malec, 2023; Rostek and Yoon, 2023). In this paper, I show how active managers can construct off-exchange signals as a way to prove their stock-picking skill.

There is a large literature looking at the problem of performance evaluation (Wermers, 2000; Berk and Green, 2004; Kosowski et al., 2006; Barras et al., 2010; Berk and van Binsbergen, 2015; Pástor et al., 2015; Giglio et al., 2021). These papers assume that past trading success is the only credible way that an active manager can signal their ability to pick stocks without revealing how they do it. Getmansky (2012), Fung, Hsieh, Naik, and Ramadorai (2008), Aggarwal and Jorion (2010), and Cao, Farnsworth, and Zhang (2021) document how difficult it is for emerging managers to raise startup capital. However, these papers do not propose better ways of accomplishing the goal.

Finally, this paper departs from classic work on information-based asset pricing in an important way (Grossman and Stiglitz, 1980; Hellwig, 1980; Admati, 1985; Kyle, 1985; Veldkamp, 2023). In those papers, investors already know which variables to focus on. For example, in Grossman and Stiglitz (1980), traders know to interpret the equilibrium price as a noisy signal about the fundamental value of a single asset. By contrast, in this paper, an emerging manager wants to keep the source of his predictive success a secret.

1 Main Result

Subsection 1.1 outlines the return-predictability problem that Alice is trying to solve. Subsection 1.2 defines what it means to "prove" your skill to Alice and gives a full-information benchmark (proof-by-revelation). Subsection 1.3 describes a subset of proof systems that generate zero-knowledge proofs and gives a notable example of one such system (proof-by-trading). Proof-by-trading is the current industry standard. However, it suffers from two key drawbacks: proof-by-trading requires upfront capital to trade with and does not demonstrate the originality of a stock-picking rule. Subsection 1.4 shows how to produce a zero-cost zero-knowledge proof of novel stock-picking skill (proof-by-encryption). This new proof system is the main result.

	Benefits	Drawbacks
Proof-By-Revelation, p15 **Protocol 1 (`RevealIt`)**	• Simple. Full-information benchmark • Does not require upfront capital. • Does not require trusted 3rd party. • Allows Alice to learn whether your stock-picking rule is different from the one she is already trading on.	• Discloses too much information. Allows Alice to trade on your profitable new stock-picking rule without paying you a dime. As far as possible from being a zero knowledge proof.
Proof-By-Trading, p17 **Protocol 2 (`TradeOnIt`)**	• Familiar. This is the industry standard that is currently used by asset managers at the moment. • Produces a zero-knowledge proof of your stock-picking skill.	• Requires upfront capital. • Requires trusted 3rd party to issue profit-and-loss statement. • Alice cannot be certain that your stock-picking rule is different from the one she is already trading on.
Proof-By-Encryption, p21 **Protocol 3 (`EncryptIt`)**	• Does not require upfront capital. • Produces zero-knowledge proof of your stock-picking skill. • Does not require trusted 3rd party. • Allows Alice to learn whether your stock-picking rule is different from the one she is already trading on.	

Table 1. Main benefits and drawbacks of the three different proof systems that I study in this paper.

1.1 General Framework

You are an emerging manager who is raising money to trade on a profitable new stock-picking strategy. Alice is an asset allocator whom you have approached for seed funding. Alice is in charge of the active portfolio for a large institutional investor. Here is the core problem she is trying to solve.

Market Environment. There are $N \gg 1$ stocks in your investable universe. Each of these stocks has K characteristics. Think about the kth characteristic, $\mathit{Characteristic}_{n,k}$, as a stock's past returns, its market cap, its sales growth, etc. $\mathit{Characteristics}_n \overset{\text{def}}{=} \{\mathit{Characteristic}_{n,k}\}_{k=1}^{K}$ is the K-dimensional vector of characteristics for the nth stock, and CharacteristicSpace is the set of all possible K-dimensional characteristic vectors that a stock could have.

In the model, you and Alice both have your own stock-picking rules. As an asset allocator, Alice has invested money with other managers in the past. Think about her stock-picking rule as the consensus recommendation for each stock coming from her existing stable of managers.

Let Return_n denote the nth stock's realized return at the end of the current period, and let Resid_n denote the component of this stock's return that Alice cannot explain using her own stock-selection rule. Assume that $\mathit{Resid}_n = \pm 1\%$ with $\Pr[\,\mathit{Resid}_n = +1\%\,] = 1/2$. Alice knows that half the stocks in her investable universe will have positive residuals at the end of the period. So do you. But neither of you knows which $N/2$ stocks it will be.

Let $\mathtt{f}(\cdot)$ denote a deterministic polynomial-time rule for making buy/sell recommendations at the start of the period:

$$\mathtt{f} : \text{CharacteristicSpace} \mapsto \{\text{“buy”}, \text{“sell”}\} \tag{1}$$

When $\mathtt{f}(\mathit{Characteristics}_n) = \text{“buy”}$, the nth stock tends to realize above-average returns, $\mathit{Return}_n = +1\%$. Whereas, if $\mathtt{f}(\mathit{Characteristics}_n) = \text{“sell”}$, the nth stock would have been more likely to have below-average returns, $\mathit{Return}_n = -1\%$.

Let $\alpha \in [0, 1/2)$ denote your rule's accuracy over Alice's residuals:

$$\begin{aligned} 1/2 + \alpha &\overset{\text{def}}{=} \Pr\big[\, \mathit{Resid}_n = +1\% \mid \mathtt{f}(\mathit{Characteristics}_n) = \text{“buy”} \,\big] && \text{(2a)} \\ &= \Pr\big[\, \mathit{Resid}_n = -1\% \mid \mathtt{f}(\mathit{Characteristics}_n) = \text{“sell”} \,\big] && \text{(2b)} \end{aligned}$$

Alice does not care whether you can predict raw returns directly. She cares about whether you can predict them in a way that is different from what her existing managers are already doing. Can you generate uncorrelated alpha? Does your stock-picking rule explain her residual returns?

Since your stock-picking rule accurately predicts Alice's residuals, $\alpha > 0$, it would be profitable for her to implement a strategy with positions

$$\mathtt{Position}_n \overset{\text{def}}{=} \$0.50 \times \begin{cases} +1 & \text{if } \mathtt{f}(\mathit{Characteristics}_n) = \text{“buy”} \\ -1 & \text{if } \mathtt{f}(\mathit{Characteristics}_n) = \text{“sell”} \end{cases} \tag{3}$$

This strategy is long \$0.50 in each buy and short \$0.50 of each sell.

In other words, the parameter α lives a double life. In addition to capturing the accuracy of your stock-picking rule, α also corresponds to the profit π from a long/short strategy based on your recommendations:

$$\pi \overset{\text{def}}{=} \left(\frac{1}{N \cdot |\pm \$0.50|} \right) \times \sum_{n=1}^{N} \underbrace{\mathit{Position}_n}_{\substack{\text{your position} \\ \text{in the } n\text{th stock}}} \times \underbrace{\mathit{Resid}_n}_{\substack{\text{residual return} \\ \text{on this position}}} \tag{4}$$

For example, if $\alpha = 0.10$ meaning that your stock-picking rule correctly predicts 60% of Alice's residuals, then she would take home $\pi = \$0.10$ in profit for each \$1.00 spent on offsetting $\pm\$0.50$ long/short positions—i.e., $\pi = \$1.00 \times \alpha$.

Decision Problem. “A decision problem consists of a specification of a subset of possible instances. Given an instance, one is required to determine whether the instance is in the specified set… [The goal is to decide] whether a given object has some predetermined property. (Goldreich, 2010)”

Let Problem denote an arbitrary decision problem. Instances represents all instances that conform to this decision problem. We write that $i \in$ Problem if the ith instance has the predetermined property associated with that specific decision problem. For example, the Boolean satisfiability decision problem (SAT; Cook, 1971) asks you to determine whether there is a collection of "true"/"false" assignments that will satisfy the ith Boolean formula. If yes, then $i \in \mathsf{SAT}$.

Let's now look at Alice's specific decision problem. She wants to know whether, at the start of the period, you knew about a profitable stock-picking rule different from her own. We say that you have novel stock-picking skill if $\alpha \geq \alpha_{\min}$ for some $\alpha_{\min} > 0$. That is the "predetermined property". The "given object" consists of your stock-picking rule $\mathtt{f}(\cdot)$ with $\alpha \in [0, 1/2)$; the characteristics displayed at the start of the current trading period, $\{Characteristics_n\}_{n=1}^{N}$; and, Alice's residual returns at the end of the period, $\{Resid_n\}_{n=1}^{N}$.

<u>Definition 1</u> ($\mathsf{Alice'sProblem}$).

(Instance) *A stock-picking rule* $\mathtt{f}(\cdot)$ *with* $\alpha \in [0, 1/2)$; $\{Characteristics_n\}_{n=1}^{N}$ *at the start of the period; and,* $\{Resid_n\}_{n=1}^{N}$ *at the end of the period.*

(Question) *Is* $\alpha \geq \alpha_{\min}$ *for some prespecified* $\alpha_{\min} > 0$*?*

If the correct answer to this question is "yes" for the ith instance, then we write that $i \in \mathsf{Alice'sProblem}$. You have novel stock-picking skill in that sort of instance. Conversely, if the correct answer is "no", then $i \notin \mathsf{Alice'sProblem}$.

The timing is important. Note that Alice solves her decision problem at the end of the current period. At that point in time, she can observe each stock's characteristics at the start of the period, $\{Characteristics_n\}_{n=1}^{N}$, as well as her residual returns, $\{Resid_n\}_{n=1}^{N}$. She also has access to any additional information she can glean from interacting with you over the course of the trading period.

Let C_α denote the number of stocks for which your stock-picking rule correctly predicts Alice's residual:

$$C_\alpha \overset{\text{def}}{=} N \cdot (1/2 + \alpha) \tag{5}$$

After returns have been realized at the end of the period, Alice can enumerate all $\binom{N}{C_\alpha}$ possible ways to make C_α correct predictions. Her question is whether you knew how to calculate one of these possible ways at the start of the period. And she can only figure this out by talking to you.

Lemma 1 (20/20 Hindsight). *After returns have been realized, there is a probabilistic polynomial-time algorithm for creating recommendations that correctly predict Alice's residuals with probability* $(1/2+\alpha)$ *for any choice of* $\alpha \in [0, 1/2)$.

The intuition behind this lemma is straightforward. Suppose you want a collection of N buy/sell recommendations that are correct with probability $(1/2+\alpha)$ for some $\alpha \in [0, 1/2)$. Draw N IID samples from $\text{Unif}([0,1])$. Then, rank these samples. For stocks $1 : C_\alpha$, set the recommendation equal to the sign of Alice's residual. For the remaining stocks ranked $(C_\alpha + 1) : N$, do the opposite.

1.2 Interactive Proof Systems

I now describe what it means to prove your stock-picking skill to Alice and give a natural full-information benchmark example of such a proof system.

The Definition of "Proof". For each instance of her decision problem, you either know about a stock-picking rule that predicts the component of returns that she cannot: $\alpha > 0 \Leftrightarrow i \in \text{Alice'sProblem}$. Or you do not: $\alpha = 0 \Leftrightarrow i \notin \text{Alice'sProblem}$. A proof that you have novel stock-picking skill should allow her to discern which situation applies to the current instance. In other words, a proof is a message that you send to Alice, $\textit{Proof} \in \{0,1\}^*$, that allows her to use a polynomial-time machine, `Verifier`, to verify your claim:

$$\texttt{Verifier}(\, i \,|\, \textit{Proof}\,) = 1 \qquad \text{for all } i \in \text{Alice'sProblem} \tag{6}$$

The complexity class NP contains all decision problems that have short easy-to-verify solutions.

Definition 2 (Complexity Class NP). *We say that* $\text{Problem} \in \text{NP}$ *if there exists a polynomial-time machine,* `Verifier`*, such that for each conforming instance* $i \in \text{Instances}$ *there exists some* $\textit{Proof} \in \{0,1\}^{\text{Poly}(|i|)}$ *whereby*

$$i \in \text{Problem} \quad \Leftrightarrow \quad \texttt{Verifier}(\, i \,|\, \textit{Proof}\,) = \text{“yes”} \tag{7}$$

An NP problem may be hard to solve, but any solution to one of these problems must be obviously correct the moment you see it. Determining whether there are true/false assignments that satisfy a Boolean formula is a classic example of an NP problem, called Boolean satisfiability (SAT; Cook, 1971).

Clearly we have that $\mathsf{Alice'sProblem} \in \mathsf{NP}$. If you described your stock-picking rule to Alice at the start of the period, she would be able to easily verify whether it was any good. This description would constitute a short proof that you have stock-picking skill. It would also allow Alice to trade on your stock-picking rule. This is why traders go on garden leave when moving from one employer to the next. A trader's previous employer does not want them to give a description of his best trading ideas to the new employer.

Interactive Protocol. How do you generate the *Proof* $\in \{0,1\}^{\mathrm{Poly}(|i|)}$ for the ith instance? You interact with Alice (Babai, 1985; Goldwasser, Micali, and Rackoff, 1989). The transcript of your conversation is the proof.

Let $\langle \mathtt{Verifier} \leftrightarrow \mathtt{Prover} \rangle_i$ denote the proof produced by the interaction between a $\mathtt{Prover}$ (this is you) and a $\mathtt{Verifier}$ (this is Alice) for the ith instance. I will use $\langle \mathtt{Verifier} \leftrightarrow \mathtt{Prover} \rangle(i) \in \{\text{“yes”}, \text{“no”}\}$ as shorthand for the verifier's decision about the ith instance after interacting with the prover:

$$\langle \mathtt{Verifier} \leftrightarrow \mathtt{Prover} \rangle(i) \overset{\text{def}}{=} \mathtt{Verifier}(\, i \,|\, \langle \mathtt{Verifier} \leftrightarrow \mathtt{Prover} \rangle_i \,) \tag{8}$$

Definition 3 (Interactive Proof System). *An interactive proof system is a protocol between a computationally unbounded* $\mathtt{Prover}$ *and a probabilistic polynomial-time* $\mathtt{Verifier}$ *that satisfies the following two conditions:*

(Completeness) *No false negatives. For all instances* $i \in \mathsf{Problem}$

$$\Pr\Big[\, \langle \mathtt{Verifier} \leftrightarrow \mathtt{Prover} \rangle(i) \;=\; \text{“yes”} \,\Big] = 1 \tag{9}$$

(Soundness) *Few false positives. For all instances* $i \notin \mathsf{Problem}$

$$\Pr\Big[\, \langle \mathtt{Verifier} \leftrightarrow \widetilde{\mathtt{Prover}} \rangle(i) \;=\; \text{“yes”} \,\Big] < 1/2 \tag{10}$$

The protocol $\widetilde{\texttt{Prover}}$ represents any choice of prover, including but not limited to the one specified by the proof system.

The completeness condition captures the idea that a good proof system should convince Alice that you have novel stock-picking skill on instances where that is actually the case. The soundness condition captures the idea that, when following this proof system, a trader who does not have novel stock-picking skill should not be able to fool Alice into thinking otherwise.

The specific choice of $1/2$ in Equation (10) is not important. By iterating on an interactive proof system, Alice can make the protocol arbitrarily sound (see subsection 2.3 below). However, it is important that the soundness condition holds for any choice of $\widetilde{\texttt{Prover}}$, not just the one specified in the proof system. The prover should not be able to fool the verifier using a different protocol.

Proof-By-Revelation. What is the simplest possible way that you could prove to Alice that you have novel stock-picking skill? You could give Alice your buy/sell recommendation for each stock at the start of the period. Then, after returns are realized, Alice could calculate whether your initial recommendations predicted her residuals sufficiently well—i.e., whether $\alpha > \alpha_{\text{min}}$ for some $\alpha_{\text{min}} > 0$ value chosen by Alice ahead of time.

<u>Protocol 1</u> (`RevealIt`).

(Step #1) *At the start of the period, you compute recommendations for each stock, $\{\texttt{f}(Characteristics_n)\}_{n=1}^{N}$, and send the positions implied by these recommendations to Alice.*

(Step #2) *At the end of the period, Alice uses Equation (4) to calculate α given her realized residual returns, $\{Resid_n\}_{n=1}^{N}$.*

If she finds $\alpha > \alpha_{\text{min}}$ for some $\alpha_{\text{min}} > 0$, she decides that $i \in \mathsf{Alice'sProblem}$—i.e., that you have stock-picking skill. Otherwise, she decides $i \notin \mathsf{Alice'sProblem}$.

If you know about a novel stock-picking rule, proof by revelation will clearly demonstrate this fact to Alice (completeness). And, if you do not know about such a rule, then you cannot fool Alice more than half the time by randomly making buy/sell recommendations at the start of the period (soundness). And calculating the accuracy of your rule via Equation (4) is as easy as computing an inner product, meaning it can be done in polynomial time.

Proposition 1. `RevealIt` *is an interactive proof system for* Alice'sProblem*.*

Of course, there is a good reason why you should not reveal your pick for each stock to Alice at the start of the period as highlighted in the first row of Table 1. If you did that, you would make it easy for her to reverse engineer how you were picking stocks. She could ask herself: "What kind of rule would have lead you to say "buy" for this stock but "sell" for that one?"

By guessing the correct answer, Alice would be able to profit from your novel stock-picking skill in future periods without paying you a dime. This is why only new and/or inexperienced traders use proof by revelation to prove that they have novel stock-picking skill. Instead, they trade on their stock-picking rule using whatever money they have.

1.3 Zero-Knowledge Proofs

Next, I show that, by trading on your buy and sell recommendations, you can prove to Alice that you can pick stocks without revealing anything else about how you do it. In other words, proof-by-trading will generate a zero-knowledge proof of your stock-picking skill.

Proof-By-Trading. For simplicity, assume your short positions must be fully funded. At the start of the period, suppose you send your broker a briefcase with $\$N/2$ in cash and you tell him to put on the long and short positions implied by Equation (3). At the end of the period, you will have profits

$$\pi \;=\; (N \cdot | \pm \$0.50|) \times \alpha \tag{11}$$

And your broker sends this value to Alice. If $\pi > \pi_{\text{min}}$ for some $\pi_{\text{min}} > \$0$ chosen ahead of time, Alice can be sure your initial positions were profitable.

Protocol 2 (`TradeOnIt`).

(Step #1) *At the start of the period, you compute recommendations for each stock, $\{\mathtt{f}(Characteristics_n)\}_{n=1}^N$. You send your broker $\$N/2$ in cash and instructions to build the positions dictated by Equation (3).*

(Step #2) *At the end of the period, your broker tells Alice what the combined profit was from these long and short positions, π.*

If $\pi > \pi_{\min}$ for some $\pi_{\min} > 0$, then Alice decides that $i \in \mathsf{Alice'sProblem}$—i.e., that you have stock-picking skill. Otherwise, she decides $i \notin \mathsf{Alice'sProblem}$.

If Alice cared about raw returns rather than her own residual returns, then Equation (4) tells us that seeing your trading profits, π, at the end of the period would have the same information content as directly computing the accuracy of your stock-picking rule, α, making `TradeOnIt` complete and sound.

Proposition 2. *If Alice has no stock-picking skill of her own, $\{Resid_n\}_{n=1}^N = \{Return_n\}_{n=1}^N$, then* `TradeOnIt` *is an interactive proof system for $\mathsf{Alice'sProblem}$.*

What's more, notice that Alice learns nothing about which stocks you predicted correctly or how you produced these predictions by interacting with you according to the `TradeOnIt` protocol. She only learns how profitable your predictions would have been to trade on.

Here is the logic. After returns have been realized, Alice can think up lots of signals that would have correctly predicted returns for a fraction $(1/2 + \alpha)$ of all stocks. It does not matter which $\alpha \in [0, 1/2)$ you choose. Any collection of buy/sell recommendations that makes C_α correct returns predictions will do. If you had been trading on any one of these equally accurate signals, then your broker would have sent Alice the exact same message, π. And that insight is the essence of why Alice learns so little about your stock-picking rule from proof-by-trading. If every interaction with a skilled trader looks the same and Alice can simulate all these outcomes herself, then she cannot learn anything new from interacting with you (a skilled trader).

Simulation Paradigm. This line of reasoning is known as the "simulation paradigm" (Lindell, 2017). The `Simulator` protocol for `TradeOnIt` samples random collections of buy/sell recommendations that are correct with probability $(1/2 + \alpha)$ where α corresponds to the accuracy of your stock-picking rule. The protocol then computes the profitability of these simulated recommendations according to Equation (4). The key point is that Alice would be unable to distinguish this simulated π from the actual signal sent to her by your broker. This is true for any accuracy level $\alpha \in [0, 1/2)$. Thus, if Alice only cared about whether you could predict raw returns, then you could use `TradeOnIt` to prove your ability to pick stocks in zero knowledge.

The Bayesian-updating paradigm says that, in order to prove Alice learns nothing besides your skill level from `TradeOnIt`, you need a detailed model of Alice's information set. And this would be very problematic. How can you be sure what Alice does and does not know?

The simulation paradigm sidesteps this quagmire. If Alice cannot distinguish her conversation with you from a simulated conversation she had with herself, then she cannot be learning anything else from interacting with you other than the accuracy of your stock-picking rule, α. It does not matter what her information set looks like.

Zero-Knowledge Proof. With this idea in hand, we can now define the notion of a zero-knowledge proof (Goldwasser, Micali, and Rackoff, 1989).

Definition 4 (Zero-Knowledge Proof)**.** *An interactive proof system gives zero-knowledge proofs for* Problem *if for every probabilistic polynomial-time* $\widetilde{\texttt{Verifier}}$ *there is a probabilistic polynomial-time* `Simulator` *such that for all* $i \in$ Instances

$$\langle \widetilde{\texttt{Verifier}} \leftrightarrow \texttt{Prover} \rangle(i) = \texttt{Simulator}(\, i \,|\, \widetilde{\texttt{Verifier}} \,) \tag{12}$$

The `Simulator` protocol is conditioned on Alice's choice of verification scheme because the simulation paradigm requires that Alice be able to simulate her interaction with you all by herself based on things she already knows. And one of the things is the list of questions she will ask you.

Proposition 3. *If Alice has no stock-picking skill of her own,* $\{Resid_n\}_{n=1}^N = \{Return_n\}_{n=1}^N$*, then* `TradeOnIt` *gives a zero-knowledge proof of* Alice'sProblem*.*

But again, there are reasons why you might not be able to give Alice a proof-by-trading as highlighted in Table 1. Here is one reason: real-world strategies need to be funded, and you might not have the upfront capital to fund the long and short positions implied by your stock-picking rule. Here is another: your potential investor will typically be invested with other managers, who have stock-picking rules of their own. As a result, Alice cares about both the profitability and novelty of your stock-picking rule.

1.4 Proof-By-Encryption

By using a homomorphic encryption scheme, it is possible to construct an alternative interactive proof system that fixes both problems associated with proof-by-trading. This proof-by-encryption protocol allows you to prove to Alice that you know about a novel way to pick stocks at no upfront cost and without revealing any additional information to her about how you do it.

Encryption Scheme. A public-key encryption scheme (Diffie and Hellman, 1976) is composed of two protocols, $(\texttt{Enc}, \texttt{Dec})$. Each of these protocols takes two arguments as inputs. The first argument to the `Enc` protocol is the message you want to encrypt. The second argument is a public key, *PublicKey*. The first argument to the `Dec` protocol is the encrypted ciphertext you want to decrypt, and the second argument to this protocol is the secret key, *SecretKey*, which is needed to do the decrypting.

Here is the fundamental identity behind any encryption scheme:

$$\mathit{Dec}\big[\,\mathit{Enc}(\mathit{Message}, \mathit{PublicKey}), \mathit{SecretKey}\,\big] \;=\; \mathit{Message} \tag{13}$$

Anyone can encrypt a message because everyone can see the *PublicKey*. But the *SecretKey* is only known by the person who set up the encryption scheme. As a result, they are the only ones who can decrypt an encrypted ciphertext:

$$\mathit{Dec}\big[\,\mathit{Enc}(\mathit{Message}, \mathit{PublicKey}), \mathit{AnyOtherValue}\,\big] \;=\; \mathit{Gibberish} \tag{14}$$

Going forward, I will suppress the dependence of both $\texttt{Enc}$ and $\texttt{Dec}$ on the public and secret key values. For example, I will write $\texttt{Enc}(\mathit{Message})$ rather than $\texttt{Enc}(\mathit{Message}, \mathit{PublicKey})$. However, it is important to point out this wrinkle about the asymmetry about who can encrypt vs decrypt a message. Both you and Alice will be encrypting messages in the proof protocol I propose, but only Alice will be able to decrypt a message.

Homomorphic Property. Alice will be using a homomorphic encryption scheme in my protocol. In a homomorphic encryption scheme, it is possible to perform operations analogous to addition and multiplication, $\oplus$ and $\otimes$, on an encrypted ciphertext. The results of the computation can then be decrypted to reveal the mathematically correct answer. It is as if $+$ and $\times$ were directly applied to the original message.

Definition 5 (Homomorphic Property)**.** *Let* $(\texttt{Enc}, \texttt{Dec})$ *be a public-key encryption scheme. This scheme has the homomorphic property if there are two operations,* $\oplus$ *and* $\otimes$*, such that for messages, Message and Message′, we have*

$$\texttt{Enc}(\mathit{Message}) \oplus \texttt{Enc}(\mathit{Message}') = \texttt{Enc}(\mathit{Message} + \mathit{Message}') \tag{15a}$$

$$\texttt{Enc}(\mathit{Message}) \otimes \texttt{Enc}(\mathit{Message}') = \texttt{Enc}(\mathit{Message} \times \mathit{Message}') \tag{15b}$$

Again, note that anyone will be able to calculate $\texttt{Enc}(\mathit{Message}) \oplus \texttt{Enc}(\mathit{Message}')$ and $\texttt{Enc}(\mathit{Message}) \otimes \texttt{Enc}(\mathit{Message}')$, but not everyone will be able to decrypt the output of these calculations. To do that, you would need to know the *SecretKey*.

Proof-By-Encryption. Here is how my new protocol works. At the beginning of the period, Alice sends you the *PublicKey* for a homomorphic encryption scheme and keeps the *SecretKey* to herself. You then use this *PublicKey* to encrypt the positions implied by your buy/sell recommendations, $\{\texttt{Enc}(\mathit{Position}_n)\}_{n=1}^{N}$, and send them to an independent third party, whom I will call a "notary". This step commits you to specific long and short positions at the start of the period.

Once you send the notary your encrypted positions, you cannot change them. However, unlike the broker in proof-by-trading, you do not need to trust the notary. He only ever sees the encrypted ciphertext associated with your portfolio positions, not the plaintext values. From the notary's point of view, each $\texttt{Enc}(\mathit{Position}_n)$ is as likely to represent a long position as a short position.

Once returns have been realized at the end of the period, Alice sends the notary her encrypted residuals, $\{\texttt{Enc}(\mathit{Resid}_n)\}_{n=1}^N$. At this point, the notary computes the inner product of these two vectors of encrypted ciphertexts

$$\begin{aligned} &\{\,\texttt{Enc}(\mathit{Position}_1)\ \otimes\ \texttt{Enc}(\mathit{Resid}_1)\,\} \\ &\quad \oplus\ \{\,\texttt{Enc}(\mathit{Position}_2)\ \otimes\ \texttt{Enc}(\mathit{Resid}_2)\,\} \\ &\qquad\qquad\qquad \ddots \\ &\qquad\qquad \oplus\ \{\,\texttt{Enc}(\mathit{Position}_N)\ \otimes\ \texttt{Enc}(\mathit{Resid}_N)\,\}\ =\ \texttt{Enc}(\pi) \end{aligned} \tag{16}$$

The output of this computation is still encrypted. It looks like gibberish to the notary. But, when he sends it to Alice, she is able to decrypt it to reveal π, which allows her to verify whether you have novel stock-picking skill.

Protocol 3 ($\texttt{EncryptIt}$)**.**

(Step #1) *At the start of the period, Alice creates a* $(\mathit{PublicKey}, \mathit{SecretKey})$ *pair and sends you the PublicKey. You compute a recommendation for each stock,* $\{\texttt{f}(\mathit{Characteristics}_n)\}_{n=1}^N$*, and the long/short positions that they imply. Then, you commit yourself to these specific values by encrypting them,* $\{\texttt{Enc}(\mathit{Position}_n)\}_{n=1}^N$*, and sending the resulting vector of ciphertexts to the notary.*

(Step #2) *At the end of the period, Alice sends the notary a ciphertext of her encrypted residual returns,* $\{\texttt{Enc}(\mathit{Resid}_n)\}_{n=1}^N$*. The notary calculates* $\texttt{Enc}(\pi)$ *as the inner product of your encrypted positions and Alice's encrypted residuals. Finally, he sends the encrypted output from this calculation back to Alice, who decrypts it to reveal* π*.*

If $\pi > \pi_{\min}$ *for some* $\pi_{\min} > 0$*, then Alice decides that* $i \in \mathsf{Alice'sProblem}$*—i.e., that you have stock-picking skill. Otherwise, she decides* $i \notin \mathsf{Alice'sProblem}$*.*

It is important to emphasize that the notary is nothing more than a pedagogical device—i.e., a helpful construction for introducing the key ideas. Unlike the broker in proof-by-trading, the notary never gets to see whether you took a long or short position in any stock, $\textit{Position}_n$. He only ever sees encrypted values of your positions, $\texttt{Enc}(\textit{Position}_n)$. The notary also never sees the accuracy of your buy/sell recommendations, π. He only ever has access to the associated ciphertext, $\texttt{Enc}(\pi)$.

The notary has two jobs. He ensures that your profits are computed (a) correctly and (b) based on the positions you calculated at the start of the period. Both these jobs can be performed by protocols that are standard in the computer science literature. "Zero-knowledge proofs are a major tool for forcing participants in cryptographic protocols to behave correctly, that is, without compromising anyone's privacy. (Wigderson, 2019)" Likewise, as long as public-key encryption schemes have existed, researchers have been using them to make private commitments (e.g., see Goldwasser and Micali, 1982).

The economically interesting thing about `EncryptIt` is the shift in perspective it requires. You are trying to prove something to Alice. The onus is on you to back up your claim about having a new way to pick stocks. So we naturally think about Alice as a passive participant in your conversation. She just listens to your pitch with her arms folded and says nothing. But what if Alice played a more active role? She could just sit there and listen. But you can both benefit if she takes the lead and creates a public-key encryption scheme for you both to use. That Gestalt shift makes it possible to construct a better proof system. It is what opens up new possibilities.

Main Result. Proposition 4 below shows that the `EncryptIt` protocol solves both of the problems associated with proof-by-trading. `EncryptIt` is just as secure as `TradeOnIt`. However, it requires no upfront capital to implement and demonstrates that you have novel stock-picking skill. It solves Alice's problem in the general case where she has already invested money with other active managers in the past. Alice can use `EncryptIt` to determine whether you can generate uncorrelated alpha.

Proposition 4 (Main Result)**.** `EncryptIt` *is an interactive proof system for* Alice'sProblem *that generates zero-knowledge proofs and has no upfront cost.*

`EncryptIt` is clearly an interactive proof system for Alice'sProblem even when Alice has a stock-selection rule of her own, $\{Resid_n\}_{n=1}^N \neq \{Return_n\}_{n=1}^N$. If you have novel stock-picking skill, Alice will uncover this fact after interacting with you according to `EncryptIt` (completeness). Otherwise, you have a 50-50 chance of guessing your way to above-average profits (soundness).

It is also clear that `EncryptIt` generates a zero-knowledge proof for deciding each problem instance. The logic is the same as in proof-by-trading. The only difference is that, in proof-by-encryption, security is guaranteed by a cryptographic assumption; whereas, in proof-by-trading, it is guaranteed by an assumption about how trustworthy your broker is as highlighted in Table 1.

In both cases, once returns have been realized at the end of the period, Alice can simulate for herself all the possible ways of making C_α correct predictions from among N possible stocks. Every one of these ways is observationally equivalent to Alice. So all she can learn is that you knew how to calculate one of them at the start of the period.

2 Implementation

The previous section shows how you can prove to Alice that you know about a profitable new way to pick stocks by making use of a homomorphic encryption scheme. Subsection 2.1 gives an example of a homomorphically encrypted buy order. Then, in subsection 2.2, I verify the numerical precision of encrypted estimates for your stock-picking skill. I do this for both the baseline setup where you make binary buy/sell recommendations and for a continuous extension.

What's more, in the baseline model, your stock-picking rule says "buy" for half the stocks in your investable universe and "sell" for the other half. But, in practice, you typically would not trade on every prediction. You would only act on the strongest predictions. In subsection 2.3, I show that proof by encryption produces a more informative signal than proof-by-trading in this scenario.

2.1 Example Ciphertext

It is possible to implement the `EncryptIt` protocol using the homomorphic encryption scheme in Brakerski and Vaikuntanathan (2011). Under this approach, each message gets converted into a pair of large polynomials with integer coefficients. The size of the polynomials and the coefficient range are free parameters. For example, an encrypted ciphertext might take the form of a pair of degree-15 polynomials with integer coefficients drawn from $\{0, 1, 2, \ldots, 99999\}$, turning a single $\mathtt{f}(\mathit{Characteristics}_n)$ = "buy" recommendation into

$$\begin{aligned} \mathit{Ciphertext}_n = \Big\{ & 16828 \cdot x^0 + 7073 \cdot x^1 + 17328 \cdot x^2 + 23267 \cdot x^3 \\ & + 18258 \cdot x^4 + 91451 \cdot x^5 + 2214 \cdot x^6 + 15985 \cdot x^7 \\ & + 33554 \cdot x^8 + 2110 \cdot x^9 + 88677 \cdot x^{10} + 28392 \cdot x^{11} \\ & + 67165 \cdot x^{12} + 625 \cdot x^{13} + 1172 \cdot x^{14} + 69731 \cdot x^{15}, \\ & 90576 \cdot x^0 + 56869 \cdot x^1 + 64823 \cdot x^2 + 76276 \cdot x^3 \\ & + 71635 \cdot x^4 + 12277 \cdot x^5 + 40128 \cdot x^6 + 36492 \cdot x^7 \\ & + 76006 \cdot x^8 + 93634 \cdot x^9 + 5486 \cdot x^{10} + 61917 \cdot x^{11} \\ & + 41216 \cdot x^{12} + 6846 \cdot x^{13} + 5987 \cdot x^{14} + 25979 \cdot x^{15} \Big\} \end{aligned} \tag{17}$$

It would be more secure to use even larger polynomials with more possible coefficient values. For example, I could have used degree-150 polynomials with integer coefficients drawn from $\{0, 1, 2, \ldots, 999999\}$. But this choice would have entailed a much larger computational burden. There is a trade off.[7]

2.2 Numerical Precision

How precise are calculations performed on encrypted ciphertexts in Brakerski and Vaikuntanathan's scheme?

[7]The variable x in these polynomials is an arbitrary placeholder analogous to the 10 in base-10 numbers or the 2 in binary. For instance, I could write the number 73 as $7 \cdot 10^1 + 3 \cdot 10^0$ in base 10 and the number 12 as $1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 0 \cdot 2^0$ (i.e., 1100) in binary. If it seems weird to add and multiply polynomials, remember that you are perfectly content with $73 + 12$ and 73×12.

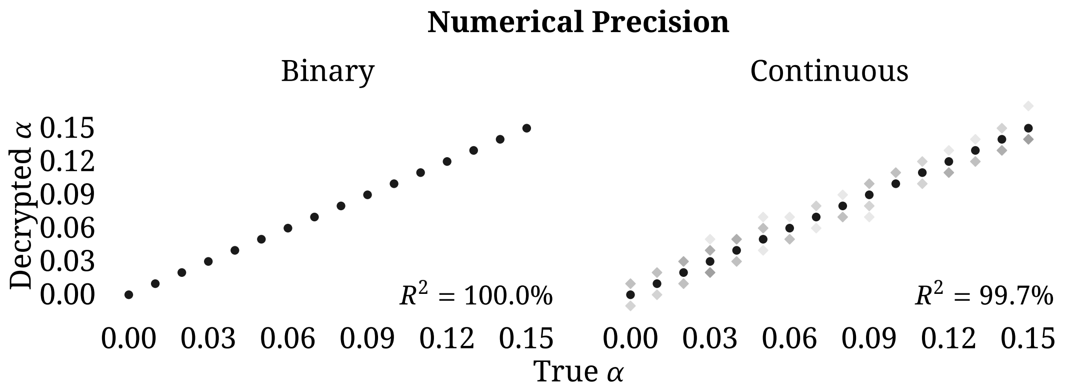

Figure 1. Each dot corresponds to a simulated market environment with $N = 500$ stocks. x-axis: true accuracy of your stock-picking rule in a particular simulation, $\alpha \in \{0.00, 0.01, \ldots, 0.15\}$. y-axis: implied accuracy based on the π value that Alice decrypts at the end of the `EncryptIt` protocol. I run 10,000 simulations at each accuracy level. Diamonds depict outcomes where there is a gap between the true α and the value that Alice estimates based on $\mathtt{Enc}(\pi)$.

Very.

Each dot in Figure 1 represents the results of a simulated market environment with $N = 500$ stocks. The x coordinate corresponds to the accuracy of your stock-picking rule in a particular simulation, $\alpha \in \{0.00, 0.01, \ldots, 0.15\}$. The y coordinate corresponds to the implied accuracy based on what Alice decrypts at the end of `EncryptIt`. I run 10,000 simulations at each accuracy level.

The left panel reports results where $\mathit{Position}_n \in \{\pm\$0.50\}$ and $\mathit{Resid}_n \in \{\pm 1\%\}$. The right panel reports results for an analogous exercise where both variables are normally distributed: $\mathit{Position}_n \sim \text{Normal}(0, 1/\sqrt{N})$ and $\mathit{Resid}_n \sim \text{Normal}(0, 10\%)$. When working with continuous random variables, the accuracy parameter is defined as $\alpha \overset{\text{def}}{=} \text{Corr}(\mathit{Position}, \mathit{Resid})$ and I discretize the continuous values using the approach in Hall, Fienberg, and Nardi (2011).

If the homomorphic encryption scheme worked perfectly, then every dot would sit right on top of the $y = x$ diagonal. This is exactly what I find in the left panel using binary data, $R^2 = 100.0\%$. When using continuous variables, the fit is hardly any worse, $R^2 = 99.7\%$. `EncryptIt` works as advertised using an off-the-shelf homomorphic encryption scheme.

2.3 A Better Signal

While the baseline model assumes that you trade on all buy/sell recommendations, managers typically trade on their strongest predictions. For example, a stock-picking rule might involve sorting stocks according to some characteristic and then going long/short the highest/lowest decile. In this sort of setting, proof-by-encryption generates a more informative signal about your stock-picking skill than proof-by-trading.

I demonstrate this fact using a simulation exercise. Each simulation looks at the monthly returns to 500 stocks over a two-year period, and I run 10,000 different simulations. The data-generating process for returns each month is

$$Return_{n,t} \leftarrow 1\% + 3\% \cdot \left\{ \underbrace{\sqrt{2\%} \cdot \textstyle\sum_{k=1}^{25} Characteristic_{n,k}}_{\text{Can predict 2\% using all 25 predictors}} + \underbrace{\sqrt{98\%} \cdot Noise_{n,t}}_{\text{Remaining 98\% of variation is random}} \right\} \tag{18}$$

The average monthly return for each stock is 1%. The standard deviation of returns across stocks is 3%. There are 25 characteristics, $Characteristic_{n,k} \overset{\text{IID}}{\sim} \text{Normal}(0, 1/\sqrt{25})$, which explain 2% of the cross-sectional variation. The remaining 98% of the variation is due to random noise, $Noise_{n,t} \overset{\text{IID}}{\sim} \text{Normal}(0, 1)$.

Alice is choosing between two emerging managers: you and Jim. I equate your stock-picking skill with knowledge of the first characteristic, $k = 1$. Jim does not have stock-picking skill. Instead of knowing about one of the 25 characteristics in Equation (18), Jim makes his return forecasts based on an arbitrary 26th characteristic $Characteristic_{n,26} \overset{\text{IID}}{\sim} \text{Normal}(0, 1/\sqrt{25})$ that plays no direct role in determining monthly returns. You and Jim both implement market-neutral strategies that go long/short the highest/lowest decile and use 65% leverage.

Figure 2 shows that, on average your strategy has an annualized $\pi = 10\%$ regardless of whether you trade for a month (left panel) or two years (right panel). Likewise, Jim's strategy has $\pi = 0\%$ at every horizon. However, the longer you trade, the more informative your realized profits are. There is much more separation between the light- and dark-gray distributions in the upper-right panel than in the upper-left panel. Based on this logic, asset allocators often refuse to meet with managers who have not been trading for 2+ years.

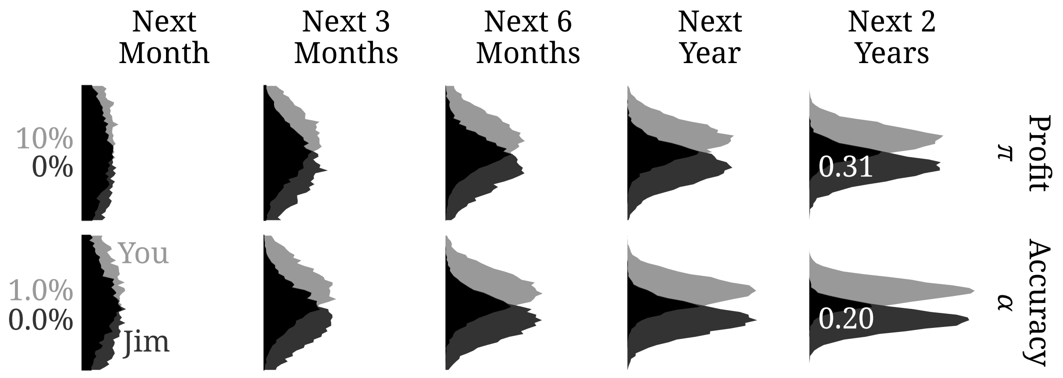

Figure 2. Outcome distributions for you (light gray) and Jim (dark gray) across 10,000 simulations over the next month, three months, six months, one year, and two years. Top panels show realized profits (π; %/year). Bottom panels show forecast accuracy (α; %/1σ change). Numbers on y-axis denote the average across simulations. Numbers in white denote the distributional overlap. A smaller overlap implies a more diagnostic signal.

My proof-by-encryption protocol can shorten that timeline. Rather than simply waiting to see your profit-and-loss statement each month, Alice could arrange to check the accuracy of your forecasts, α. The average accuracy of your stock-picking rule is $\alpha = 1.0\%$ while Jim has $\alpha = 0.0\%$. But again, the key thing is the variance. The bottom panels of Figure 2 show that your forecast accuracy is even more diagnostic of your skill level than your trading profits. After two years, Jim's forecasts are $(0.31 - 0.20)/0.31 \approx 35\%$ less likely to be as accurate as yours by pure chance.

The economic intuition behind this result is simple. There is information about your stock-picking skill in the picks that you chose not to trade on. Your long/short strategy does not trade 80% of the stocks in your investable universe. But you still made return forecasts for these stocks. The information in these untraded predictions can help Alice distinguish you from Jim. But, since you did not trade on this information, it has no effect on your annualized π. By contrast, α reflects the accuracy of your return forecasts across all stocks in your investable universe. As a result, Alice will be able to identify you as the skilled manager faster when you use `EncryptIt` to demonstrate the accuracy of your forecasts.

3 Robustness

Is it always possible for you to give Alice a zero-cost zero-knowledge proof of novel stock-picking skill? In subsection 3.1, I show that such a proof system will exist in nearly all quantitative trading applications. Subsection 3.2 addresses several edge-case concerns. And, in subsection 3.3, I give an example where no zero-knowledge proof of skill can exist because skill is hard to recognize when you see it. This is what it would take to rule out a proof-by-encryption.

3.1 Existence Result

`EncryptIt` produces zero-knowledge proofs to Alice′sProblem in a simple highly stylized model of cross-sectional predictability. Suppose a researcher were to complicate Alice's problem to make it more realistic. Let's call the resulting problem RealisticAlice′sProblem. Because of these additional features, the `EncryptIt` protocol might no longer give valid zero-knowledge proofs for RealisticAlice′sProblem. However, I claim that, if RealisticAlice′sProblem really is a more realistic model, then there will still exist a zero-knowledge proof system for RealisticAlice′sProblem.

This claim stems from a classic result in the computer science literature. A "one-way function" (Diffie and Hellman, 1976) is a function that is easy to compute but hard to invert. And Goldreich, Micali, and Wigderson (1991) proved that, if one-way functions exist, then all problems in NP have zero-knowledge proof systems.

Definition 6 (One-Way Function). *The polynomial-time function,* $\texttt{OWF} : \{0,1\}^\star \mapsto \{0,1\}^\star$*, is a one-way function if there is a negligibly small probability that any polynomial-time adversary* $\widetilde{\texttt{Adversary}}$ *can succeed at inverting it:*

$$\Pr\big[\, \texttt{OWF}(\widetilde{\texttt{Adversary}}[\texttt{OWF}(x)]) = \texttt{OWF}(x) \,\big] \leq 1/\text{Poly}(|x|) \tag{19}$$

Theorem (Goldreich, Micali, and Wigderson, 1991). *If one-way functions exist, then there exists a zero-knowledge proof system for every problem in* NP.

I am obviously oversimplifying the problem that Alice faces when deciding whether you have novel stock-picking skill. But, to overturn my main result, Goldreich, Micali, and Wigderson (1991)'s theorem tells us that a researcher would have to assume that $\mathsf{RealisticAlice'sProblem} \notin \mathsf{NP}$. This would make $\mathsf{RealisticAlice'sProblem}$ unrealistically complicated, which is exactly the opposite of what the researcher was trying to do in the first place.

For there to be no zero-cost zero-knowledge proofs of skill, it would have to be computationally hard for Alice to verify your skill level even if you told her how you picked stocks. We do not live in that sort of world. Non-compete clauses are common for quantitative equity traders.[8] Garden leave is a thing.[9]

3.2 Specific Concerns

The `EncryptIt` protocol can be generalized in various ways. For one thing, the protocol only requires multiplying pairs of encrypted numbers—i.e., you compute $\mathtt{Enc}(a) \otimes \mathtt{Enc}(b)$ but never $\mathtt{Enc}(a) \otimes \mathtt{Enc}(b) \otimes \mathtt{Enc}(c)$. There is no need to use a fully homomorphic scheme. `EncryptIt` can accommodate a partially homomorphic scheme, like Paillier (1999).

In the last section, we also saw how it is possible to apply `EncryptIt` to continuous random variables. One can only homomorphically encrypt integer-valued messages. So you must first discretize any continuous values before encrypting them using some scheme like the one in Hall, Fienberg, and Nardi (2011). But after doing this, the right panel of Figure 1 shows that mathematical results computed using homomorphically encrypted ciphertexts are still numerically precise.

I have been working under the assumption that Alice does not deviate from the proof protocol you both agree on at the start of a trading period. In this scenario, Alice is typically called an "honest-but-curious verifier". This assumption is without loss of generality. Goldreich, Micali, and Wigderson

[8]Amy Whyte. "Noncompete Agreements Have Become 'Ubiquitous' in Financial Services and Elsewhere." *Institutional Investor.* December 10th, 2019.

[9]Alex Morrell. "The Wall Street Quant Battle to Lock Up Traders With Non-Competes." *Business Insider.* July 31st, 2022.

(1991) showed how to make any zero-knowledge proof system for an honest-but-curious verifier into a proof system that works against even a verifier might strategically deviate from the proof protocol when it is in their own interest. In that case, Alice would be called a “malicious verifier”.

The `EncryptIt` protocol does not reveal anything to you about how Alice is picking stocks. Alice only shares the encrypted values of her residual returns at the end of the period. The `EncryptIt` protocol also works in situations where Alice does not have a stock-picking rule of her own—i.e., where $\{Resid_n\}_{n=1}^N = \{Return_n\}_{n=1}^N$. When Alice sends the notary her encrypted ciphertext on the 30th of the month, he cannot decrypt this vector of ciphertexts. Hence, he has no way of knowing whether it represents raw or residualized returns. The logic behind `EncryptIt` is the same either way.

3.3 Counterexample

Are there places where you cannot give a zero-cost zero-knowledge proof that you have a new way to pick good investments? Yes. But these scenarios are different from the one in this paper. Again, from Goldreich, Micali, and Wigderson (1991), we know that it must be a situation where it is hard to recognize good investment rules when you see them.

This sort of situation occurs in venture capital where it is often hard to recognize when you have hit a home run even after the fact. For example, the CEO of Excite famously turned down an offer to buy Google for $1 million in 1999 (Mallaby, 2022). This price tag seemed like an exorbitant sum, even to Google's existing investors. It was hard for them to recognize that Google would be worth orders of magnitude more money in just a few short years.

Hence, there is no way for a venture capitalist to give a zero-cost zero-knowledge proof that he/she has a new way to pick successful start-ups. This counterexample points to a previously unappreciated economics. There is a fundamental reason why it is harder to raise capital for a venture capital fund than for a quantitative equities fund.

4 Implications

This paper offers a new solution to a core problem in the field of asset pricing. Before giving a manager money, an asset allocator wants to see evidence that you have novel stock-picking skill. I show how to provide this evidence in zero-knowledge and at zero cost using homomorphic encryption techniques. In subsection 4.1, I describe how this proof system makes it possible to have a spot market for profitable trading ideas. In subsection 4.2, I highlight the fact that my proof-by-encryption protocol offers asset allocators a way to find uncorrelated alpha. This is something that cannot be done with proof-by-trading. Finally, subsection 4.3 discusses how it is possible to extend the machinery in this paper to other strategy characteristics, such as trading capacity.

4.1 Spot Market

There is already a market for profitable trading ideas. But this market is not a “spot market”. At the moment the price depends on your past track record. If you have a long history of producing strong predictors, you will be able to sell your next predictor at a fair price. Whereas, if this is your first transaction, you may have to sell at a steep discount.

The existence of zero-cost zero-knowledge proofs of skill makes it possible to have a spot market for profitable trading ideas. Under my proof-by-encryption protocol, all that matters is the quality of the stock-picking rule that you are currently selling. Before purchasing the rights to trade on your latest idea, any buyer will be able to see how profitable this stock-picking rule is. But, because you can give them a zero-cost zero-knowledge proof of this result, the buyer need not learn anything else about your stock-picking rule.

A spot market for profitable trading ideas would fundamentally alter how the active-management industry is structured. So much of the industry is organized around keeping good trading ideas secret, and zero-knowledge proofs change the requirements of secrecy. It is hard to imagine all the knock-on effects. That being said, I think there are two particular implications worth highlighting.

First, a spot market for profitable trading ideas would change how profits are shared between traders and the people who fund them. Under the current system, a trader will typically receive a tiny fraction of the profits from their first profitable stock-picking rule. Early successes merely allow them to bargain for a larger share of the profit from any future ideas.

With a spot market for profitable trading ideas, there is no reason why the profits need to be split this way. A skilled stock picker would command the competitive market price when selling his/her first good idea. For example, student trading competitions do not need to end with the winner revealing their approach. Likewise, with a spot market, social-investing platforms would become a suboptimal way of advertising your stock-picking skill (Cookson, Engelberg, and Mullins, 2023). There is no need to publicly broadcast your buy/sell recommendations as a way to commit to knowing about them ahead of time. Homomorphic encryption makes it possible to do this privately, allowing you to profit from the idea.

Second, a spot market for profitable trading idea would also alter the kind of information that gets into prices. For example, suppose Charles is a biochemistry PhD student who has noticed an overlooked detail about the drug development process. This is where his stock-picking skill comes from.

Because he is a poor PhD student, Charles needs to raise capital from Alice in order to trade on it. As things stand now, he would likely have to give Alice an in-depth description of this overlooked detail in order to get funding. And this description would give Alice all the information she needed to trade on his idea herself without paying Charles a dime. As a result, he would be unlikely to trade on the idea and incorporate this information into prices under the current regime.

This scenario would play out in a completely different way if there were a spot market for profitable trading ideas. Proof-by-encryption makes it possible to incorporate the information in Charles's first (and likely only) stock-picking rule into prices. And, by encouraging other amateur traders like Charles to do like he did, this spot market would allow for more diverse kinds of information to get impounded into prices.

4.2 Uncorrelated Alpha

Asset allocators like Alice will invest money with a large number of different managers. As such, they are not just interested in whether it would be profitable to invest money with you in isolation. Alice wants to know whether it would be profitable to add you to her existing portfolio. Even if you have stock-picking skill, Alice may not want to invest money with you in a world where you are merely duplicating the positions of one of her existing managers.

Asset allocators currently try to minimize the overlap between their managers' positions by looking for managers with different investable universes. For example, suppose your investable universe is US large-cap stocks. If Alice has already invested money with another manager who specializes in US large-cap stocks, then she would not give money to you regardless of your skill level. Whereas, if your investable universe was the FTSE 100 rather than the S&P 500, Alice would consider investing with you. However, this approach is far from perfect. Two managers with different investable universes may have highly correlated returns. A momentum strategy involving S&P 500 stocks will have similar returns to a momentum strategy involving FTSE 100 stocks.

Proof-by-encryption offers a better solution. Alice can use `EncryptIt` to verify whether your stock-picking rule can predict her residual returns, allowing her to learn whether you are generating profits in a way that is different from what she is already doing. Moreover, she can run this test even in the same investable universe. She can check whether you can forecast the component of S&P 500 returns left unexplained by her other US large-cap manager.

The rise of passive investing has been one of the most talked about financial-market trends over the past decade (Wurgler, 2011). This trend has important consequences for both corporate decisions (e.g., see Azar, Schmalz, and Tecu, 2018; Bebchuk and Hirst, 2019) and price informativeness (e.g., see Bai, Philippon, and Savov, 2016; Buffa, Vayanos, and Woolley, 2022). It is hard to form an active portfolio for a large institutional investor. As asset allocator must identify a group of active managers who are not only profitable on their own but also different from one another.

This is a difficult ask in a world where proof-by-trading is the norm. And the difficulty of this problem is a major reason why many institutional investors choose to simply track an established index, such as the S&P 500. By contrast, proof-by-encryption gives asset allocators a straightforward way to accomplish both goals. `EncryptIt` can be used to directly measure whether a manager can produce uncorrelated alpha.

4.3 Accuracy and...

Suppose that Alice is worried about whether you have a novel way to pick stocks AND whether this new stock-picking rule can be implemented at sufficient scale. How much money could you pump into the long/short strategy implied by your buy and sell recommendations? Jegadeesh and Titman (1993)'s momentum stock-picking rule looks good on paper and at small scale. But the turnover it requires can be very costly as the size of the portfolio increases.

There is no way to convey this sort of information via proof-by-trading. If you had enough money to implement your long/short strategy at scale, then you would not need outside funding from Alice. Goldreich, Micali, and Wigderson (1991)'s result implies that it is possible for you to demonstrate the scale of your novel stock-picking rule by modifying proof-by-encryption slightly.

Suppose you just told Alice about how your new stock-picking rule worked. Before implementing a long/short strategy based on this rule, she would want to predict its maximum scale. e.g., she might look at things like the turnover of its portfolio positions. As long as Alice has a way to predict whether a strategy has some property, then it is possible to prove that your new stock-picking rule satisfies these criteria in zero knowledge and at zero cost.

Conclusion

This paper studies a classic problem in asset pricing. You are an emerging manager with novel stock-picking skill. The trouble is that, in addition to being skilled, you are also poor. So to implement a reasonably sized long/short strategy based on your new stock-picking rule, you need to raise money from an outside investor, Alice. Before giving you seed capital, Alice wants to see proof that you can pick stocks as well as you say you can. What is the best way to prove to Alice that you know about a profitable new way to pick stocks?

Previously, researchers only knew of two options. On one hand, you could implement a small-scale version of your long/short strategy and show Alice the resulting profits. This approach is nice because it reveals nothing about how you pick stocks, only that you can do it. Unfortunately, it is expensive and says nothing about the novelty of your recommendations. If Alice is already trading on a cross-sectional predictor of her own, then she cannot tell whether your stock-picking rule is different from hers by studying your excess returns.

On the other hand, if you really have no money at all, then the conventional wisdom is that the only way to prove to Alice that you can pick stocks is to describe to her how you pick them. This would be unwise. Even if successful, proof-by-revelation would put you in a bad bargaining position when deciding how to split the profits from trading on your new stock-picking rule.

This paper offers a third option based on homomorphic encryption. Proof-by-encryption does not require you to have any upfront capital to trade with. It also proves that your stock-picking rule is different from anything that Alice is already trading on. The existence of a proof-by-encryption protocol suggests that one could create a spot market for profitable trading ideas. Such a spot market would fundamentally alter the active-management industry.

Technical Appendix

Proof (Lemma 1)**.** The proof follows from the existence of the protocol below, which uses realized returns to create a random collection of N buy/sell recommendations, which are correct with probability $(1/2 + \alpha)$ for any choice of $\alpha \in [0, 1/2)$.

Protocol A (`20/20Hindsight`).

(Input) *List of stocks, N; an accuracy level,* $\alpha \in [0, 1/2)$*; and, a collection of realized returns,* $\{Resid_n\}_{n=1}^N$.

(Output) *N recommendations,* $\{ExPostRec_n\}_{n=1}^N \in \{\text{“buy”}, \text{“sell”}\}^N$.

...

(Step #1) *Draw N IID samples from a uniform distribution,*

$$Sample_n \overset{\text{IID}}{\sim} \text{Unif}([0, 1]), \tag{20}$$

(Step #2) *Rank* $\{Sample_n\}_{n=1}^N$ *in ascending order.*

(Step #3) *Compute required number of correct predictions,* C_α.

(Step #4) *For stocks ranked* $1 : C_\alpha$*, assign recommendations as follows:*

$$ExPostRec_n = \begin{cases} \text{“buy”} & \textit{if } Resid_n = +1\% \\ \text{“sell”} & \textit{if } Resid_n = -1\% \end{cases} \tag{21}$$

For the remaining stocks ranked $(C_\alpha + 1) : N$*, use the opposite rule:*

$$ExPostRec_n = \begin{cases} \text{“buy”} & \textit{if } Resid_n = -1\% \\ \text{“sell”} & \textit{if } Resid_n = +1\% \end{cases} \tag{22}$$

Every collection of recommendations produced by the `20/20Hindsight` protocol, $\{ExPostRec_n\}_{n=1}^N$, will be correct with probability $(1/2 + \alpha)$ for whatever initial accuracy level is provided, $\alpha \in [0, 1/2)$. And this protocol involves executing a fixed number of polynomial-time operations. □

Proof (Proposition 1). To prove that `RevealIt` is an interactive proof system for Alice'sProblem, I need to verify completeness and soundness. I also need to confirm that the `Verifier`'s role in `RevealIt` can be executed by a probabilistic polynomial-time machine—i.e., that a tractability condition is satisfied.

(Completeness) If you give Alice recommendations, $\{\mathtt{f}(Characteristics_n)\}_{n=1}^N$, she can compute the α of the recommendations via Equation

(4). Thus, for all instances where $\alpha > \alpha_{\text{min}}$, she will decide $i \in$ Alice'sProblem at the end of `RevealIt`.

(Soundness) Instances where $i \notin$ Alice'sProblem correspond to situations where you lack stock-picking skill—i.e., situations where your stock-picking rule makes buy/sell recommendations that are correct with probability $1/2$.

If you send Alice the buy/sell recommendations produced by this rule, then she will infer that $\alpha = 0$ and decide that $i \notin$ Alice'sProblem.

Alternatively, because you lack stock-picking skill, any other collection of buy/sell recommendations that you might send Alice at the start of the period will be correct no more than half the time. So, if you sent Alice a randomly guessed collection of buy/sell recommendations, you would fool her into thinking $i \in$ Alice'sProblem no more than half the time.

(Tractability) The `Verifier` protocol only requires Alice to calculate the accuracy of N buy/sell recommendations via Equation (4), which can be done in polynomial time.

Hence, we can conclude that `RevealIt` represents a valid interactive proof system for solving Alice'sProblem. □

Proof (Proposition 2)**.** To prove that `TradeOnIt` is an interactive proof system for Alice'sProblem when Alice has no stock-picking skill of her own, I need to demonstrate that the same three properties hold (i.e., completeness, soundness, and tractability) when we assume that $Resid_n = Return_n$ for all $n = 1, \ldots, N$.

(Completeness) Equation (4) implies that, in the final step of the `TradeOnIt` protocol, your broker will send Alice a message containing, $\pi = (N \cdot |\pm\$0.50|) \times \alpha$. So, if Alice sets $\pi_{\text{min}} \overset{\text{def}}{=} (N \cdot |\pm\$0.50|) \times \alpha_{\text{min}}$, she will conclude that $\alpha > \alpha_{\text{min}}$ on all instances $i \in$ Alice'sProblem.

(Soundness) Instances where $i \notin$ Alice'sProblem correspond to situations where you lack stock-picking skill—i.e., situations where you make recommendations that are correct with probability $1/2$.

If you implemented the long/short strategy associated with your stock-picking rule, then your broker would send Alice a message containing $\pi = 0$, and she would correctly decide that $i \notin$ Alice'sProblem.

Because you lack stock-picking skill, any other collection of buy/sell recommendations will be profitable no more than half the time. If you implemented a random long/short strategy, then the message sent by your broker would fool Alice into thinking that $i \in$ Alice'sProblem no more than half the time.

(Tractability) Alice only has to perform a single comparison, $\pi > \pi_{\text{min}}$, which can clearly be done in polynomial time.

Again, all three conditions are satisfied, so we can conclude that `TradeOnIt` represents a valid interactive proof system for solving the restricted version of Alice'sProblem where Alice has no stock-picking rule of her own. □

Proof (Proposition 3)**.** Assume that Alice has no stock-picking rule of her own: $Resid_n = Return_n$ for all $n = 1, \ldots, N$. For a given level of accuracy $\alpha \in [0, 1/2)$, I need to define an algorithm that can simulate the interaction Alice would have with your broker in every possible instance where you know about a stock-picking rule with that level of accuracy. And I need to show that all such simulations are computationally indistinguishable to Alice.

First, I define the simulator algorithm for `TradeOnIt`.

<u>Protocol B</u> (`TradeOnItSimulator`)**.**

(Input) *List of stocks, N; an accuracy level, $\alpha \in [0, 1/2)$; and, a collection of realized returns, $\{Return_n\}_{n=1}^{N}$.*

(Output) *A message containing profit level, π.*

...

(Step #1) *Use the* `20/20Hindsight` *protocol to generate a random recommendations,* $\{ExPostRec_n\}_{n=1}^N$*, that are correct with probability* $(1/2 + \alpha)$*.*

(Step #2) *Use Equation (3) to compute the portfolio positions implied by these recommendations.*

(Step #3) *Use Equation (4) to compute the profit,* π*, from trading on these implied positions in the current period.*

There are $\binom{N}{C_\alpha}$ collections of recommendations that lead to Alice receiving a message with profit $\pi = (N \cdot |\pm \$0.50|) \times \alpha$. And all messages containing the same value of π are observationally equivalent to her. So she is unable to distinguish a π signal that she herself simulated using the protocol above from the one sent to her by your flesh-and-blood broker so long as both are based on stock-picking rules with the same level of accuracy, α. □

Proof (Proposition 4)**.** This proof demonstrates that `EncryptIt` is an interactive proof system for Alice'sProblem which produces zero-knowledge proofs.

First, I prove `EncryptIt` is an interactive proof system for Alice'sProblem. To do this, I need to demonstrate that the proof system satisfies the same completeness, soundness, and tractability requirements as above.

(Completeness) Equation (4) implies that the notary will compute an encrypted profit value of $\texttt{Enc}(\pi) = \texttt{Enc}[(N \cdot |\pm \$0.50|) \times \alpha]$. Alice then decrypts this ciphertext to reveal the plaintext value of π. So, if Alice sets $\pi_{\text{min}} \overset{\text{def}}{=} (N \cdot |\pm \$0.50|) \times \alpha_{\text{min}}$, she will conclude that $\alpha > \alpha_{\text{min}}$ on all instances $i \in$ Alice'sProblem.

(Soundness) Instances where $i \notin$ Alice'sProblem correspond to situations where you lack stock-picking skill, $\alpha = 0$, meaning a long/short strategy based on your stock-picking rule has $\pi = \$0$.

If you sent the notary encrypted values of the long/short positions implied by your recommendations, he would calculate an encrypted ciphertext for $\pi = \$0$. When Alice saw this value, she would correctly conclude $i \notin$ Alice'sProblem.

Any other random collection of buy/sell recommendations at the start of the period will be profitable with probability $1/2$. So if you sent the notary some other randomly selected long/short strategy, it would fool Alice with probability $1/2$.

(Tractability) Each encryption and decoding task can be executed in polynomial time. Alice has to do $(N+1)$ of them. She has to encrypt her residual returns for N stocks, and she has to decrypt the profit value you send her. Thus, the overall computational burden of the `Verifier` algorithm scales polynomially.

All three requirements are satisfied, so we can conclude that `EncryptIt` is a valid interactive proof system for solving Alice'sProblem.

Now, I want to show that `EncryptIt` generates zero-knowledge proofs for deciding membership in Alice'sProblem. To do this, I need to define another algorithm that simulates the interaction Alice would have with you via the notary in every possible instance where you know about a stock-picking rule with a given level of accuracy, $\alpha \in [0, 1/2)$. Then I need to verify that all such simulations look the same to Alice.

Here is how the simulator algorithm works for `EncryptIt`.

<u>Protocol C</u> (`EncryptItSimulator`)**.**

(Input) *List of stocks, N; an accuracy level, $\alpha \in [0, 1/2)$; and, a collection of residual returns, $\{Resid_n\}_{n=1}^{N}$.*

(Output) *An encrypted ciphertext,* $\texttt{Enc}(\pi)$.

. . .

(Step #1) *Use the* `20/20Hindsight` *protocol to generate a random recommendations, $\{ExPostRec_n\}_{n=1}^{N}$, that are correct with probability $(1/2 + \alpha)$.*

(Step #2) *Use Equation (3) to compute the portfolio positions implied by these recommendations.*

(Step #3) *Use Equation (4) to compute the profit, π, generated by these implied positions in the current period.*

(Step #4) *Encrypt this profit value to create* $\texttt{Enc}(\pi)$.

Just as we saw when analyzing the `TradeOnIt` protocol, there are $\binom{N}{C_\alpha}$ possible collections of recommendations that might lead to Alice receiving an encrypted ciphertext that decrypts to reveal $\pi = (N \cdot | \pm \$0.50|) \times \alpha$. And all messages that decrypt to the same π value are observationally equivalent to her. So Alice is unable to distinguish an encrypted ciphertext that she herself simulated using the protocol above from the one sent to her by you if both are based on stock-picking rules with the same level of accuracy, α. Hence, `EncryptIt` produces zero-knowledge proofs for $\mathsf{Alice'sProblem}$.

Finally, `EncryptIt` does not require the prover to make any payments. It is also an interactive proof system for the full version of $\mathsf{Alice'sProblem}$, not just the restricted version where Alice has no stock-picking rule of her own. □

References

Abbe, E., A. Khandani, and A. Lo (2012). Privacy-preserving methods for sharing financial risk exposures. *American Economic Review*.

Admati, A. (1985). A noisy rational expectations equilibrium for multi-asset securities markets. *Econometrica*.

Aggarwal, R. and P. Jorion (2010). The performance of emerging hedge funds and managers. *Journal of Financial Economics*.

Akbarpour, M., S. Kominers, K. Li, S. Li, and P. Milgrom (2023). Algorithmic mechanism design with investment. *Econometrica*.

Azar, J., M. Schmalz, and I. Tecu (2018). Anticompetitive effects of common ownership. *Journal of Finance*.

Azevedo, E. and E. Budish (2019). Strategy-proofness in the large. *Review of Economic Studies*.

Babai, L. (1985). Trading group theory for randomness. In *Proceedings of the seventeenth annual ACM symposium on Theory of computing*.

Bai, J., T. Philippon, and A. Savov (2016). Have financial markets become more informative? *Journal of Financial Economics*.

Barras, L., O. Scaillet, and R. Wermers (2010). False discoveries in mutual fund performance: Measuring luck in estimated alphas. *Journal of Finance*.

Basak, S. and A. Pavlova (2013). Asset prices and institutional investors. *American Economic Review*.

Bebchuk, L. and S. Hirst (2019). Index funds and the future of corporate governance: Theory, evidence, and policy. *Columbia Law Review*.

Berk, J. and R. Green (2004). Mutual fund flows and performance in rational markets. *Journal of Political Economy*.

Berk, J. and J. van Binsbergen (2015). Measuring skill in the mutual fund industry. *Journal of Financial Economics*.

Biais, B., C. Bisiere, M. Bouvard, C. Casamatta, and A. Menkveld (2022). Equilibrium bitcoin pricing. *Journal of Finance*.

Brakerski, Z. and V. Vaikuntanathan (2011). Fully homomorphic encryption from ring-LWE and security for key dependent messages. In *Annual Cryptology Conference*.

Budish, E. (2011). The combinatorial assignment problem: Approximate competitive equilibrium from equal incomes. *Journal of Political Economy*.

Budish, E., Y.-K. Che, F. Kojima, and P. Milgrom (2013). Designing random allocation mechanisms: Theory and applications. *American Economic Review*.

Budish, E., P. Cramton, A. Kyle, J. Lee, and D. Malec (2023). Flow trading. *Working Paper*.

Budish, E. and J. Kessler (2022). Can market participants report their preferences accurately (enough)? *Management Science*.

Buffa, A., D. Vayanos, and P. Woolley (2022). Asset management contracts and equilibrium prices. *Journal of Political Economy*.

Cao, C., G. Farnsworth, and H. Zhang (2021). The economics of hedge fund startups: theory and empirical evidence. *Journal of Finance*.

Cao, S., L. W. Cong, M. Han, Q. Hou, and B. Yang (2020). Blockchain architecture for auditing automation and trust building in public markets. *IEEE Computer*.

Cao, S., L. W. Cong, and B. Yang (2019). Financial reporting and blockchains: Audit pricing, misstatements, and regulation. *Working Paper*.

Chinco, A. and V. Fos (2021). The sound of many funds rebalancing. *Review of Asset Pricing Studies*.

Chinco, A. and M. Sammon (2023). The passive-ownership share is double what you think it is. *Working Paper*.

Cong, L. W., Y. Li, and N. Wang (2021). Tokenomics: Dynamic adoption and valuation. *Review of Financial Studies*.

Cong, L. W., Y. Li, and N. Wang (2022). Token-based platform finance. *Journal of Financial Economics*.

Cook, S. (1971). The complexity of theorem-proving procedures. In *Proceedings of the third annual ACM symposium on Theory of computing*.

Cookson, A., J. Engelberg, and W. Mullins (2023). Echo chambers. *Review of Financial Studies*.

Diffie, W. and M. Hellman (1976). New directions in cryptography. *IEEE Transactions on Information Theory*.

Duchin, R., D. Solomon, J. Tu, and X. Wang (2022). The cryptocurrency participation puzzle. *Working Paper*.

Dworczak, P., S. Kominers, and M. Akbarpour (2021). Redistribution through markets. *Econometrica*.

Fama, E. (1976). *Foundations of finance: Portfolio Decisions and securities prices*. Basic Books.

Fung, W., D. Hsieh, N. Naik, and T. Ramadorai (2008). Hedge funds: Performance, risk, and capital formation. *Journal of Finance*.

Garey, M. and D. Johnson (2002). *Computers and intractability*. WH Freeman New York.

Gentry, C. (2009). Fully homomorphic encryption using ideal lattices. *ACM Symposium on Theory of Computing*.

Getmansky, M. (2012). The life cycle of hedge funds: Fund flows, size, competition, and performance. *Quarterly Journal of Finance*.

Giglio, S., Y. Liao, and D. Xiu (2021). Thousands of alpha tests. *Review of Financial Studies*.

Goldin, C. and C. Rouse (2000). Orchestrating impartiality: The impact of “blind” auditions on female musicians. *American Economic Review*.

Goldreich, O. (2010). *P, NP, and NP-completeness: the basics of computational complexity*. Cambridge University Press.

Goldreich, O., S. Micali, and A. Wigderson (1991). Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. *Journal of the ACM*.

Goldwasser, S. and S. Micali (1982). Probabilistic encryption & how to play mental poker keeping secret all partial information. In *Proceedings of the fourteenth annual ACM symposium on Theory of computing*.

Goldwasser, S., S. Micali, and C. Rackoff (1989). The knowledge complexity of interactive proof systems. *SIAM Journal on computing*.

Golowich, L. and S. Li (2022). On the computational properties of obviously strategy-proof mechanisms. *Working Paper*.

Gonczarowski, Y., O. Heffetz, and C. Thomas (2023). Strategyproofness-exposing mechanism descriptions. *Working Paper*.

Griffin, J. and A. Shams (2020). Is Bitcoin really untethered? *Journal of Finance*.

Grossman, S. and J. Stiglitz (1980). On the impossibility of informationally efficient markets. *American Economics Review*.

Hall, R., S. Fienberg, and Y. Nardi (2011). Secure multiple linear regression based on homomorphic encryption. *Journal of Official Statistics*.

Hastings, M., B. Falk, and G. Tsoukalas (2022). Privacy-preserving network analytics. *Management Science*.

Hatfield, J., S. Kominers, A. Nichifor, M. Ostrovsky, and A. Westkamp (2013). Stability and competitive equilibrium in trading networks. *Journal of Political Economy*.

Hellwig, M. (1980). On the aggregation of information in competitive markets. *Journal of Economic Theory*.

Jegadeesh, N. and S. Titman (1993). Returns to buying winners and selling losers: Implications for stock-market efficiency. *Journal of Finance*.

Kominers, S., A. Teytelboym, and V. Crawford (2017). An invitation to market design. *Oxford Review of Economic Policy*.

Kosowski, R., A. Timmermann, R. Wermers, and H. White (2006). Can mutual fund "stars" really pick stocks? new evidence from a bootstrap analysis. *Journal of Finance*.

Kyle, A. (1985). Continuous auctions and insider trading. *Econometrica*.

Lindell, Y. (2017). How to simulate it: A tutorial on the simulation proof technique. *Tutorials on the Foundations of Cryptography*.

Liu, Y., A. Tsyvinski, and X. Wu (2022). Common risk factors in cryptocurrency. *Journal of Finance*.

Makarov, I. and A. Schoar (2020). Trading and arbitrage in cryptocurrency markets. *Journal of Financial Economics*.

Mallaby, S. (2022). *The power law: Venture capital and the making of the new future*. Penguin.

Milgrom, P. (2009). Assignment messages and exchanges. *American Economic Journal: Microeconomics*.

Milgrom, P. (2021). Auction research evolving: Theorems and market designs. *American Economic Review*.

Mu'Alem, A. and N. Nisan (2008). Truthful approximation mechanisms for restricted combinatorial auctions. *Games and Economic Behavior*.

Nisan, N., T. Roughgarden, E. Tardos, and V. Vazirani (2007). *Algorithmic game theory*. Cambridge University Press.

Nisan, N. and I. Segal (2006). The communication requirements of efficient allocations and supporting prices. *Journal of Economic Theory*.

Pagnotta, E. (2022). Decentralizing money: Bitcoin prices and blockchain security. *Review of Financial Studies*.

Paillier, P. (1999). Public-key cryptosystems based on composite degree residuosity classes. In *International conference on the theory and applications of cryptographic techniques*.

Papadimitriou, C. (2015). The complexity of computing equilibria. In *Handbook of Game Theory with Economic Applications*.

Pástor, L., R. Stambaugh, and L. Taylor (2015). Scale and skill in active management. *Journal of Financial Economics*.

Rivest, R., L. Adleman, and M. Dertouzos (1978). On data banks and privacy homomorphisms. *Foundations of Secure Computation*.

Rostek, M. and J. H. Yoon (2021). Exchange design and efficiency. *Econometrica*.

Rostek, M. and J. H. Yoon (2023). Innovation in decentralized markets: Technology vs synthetic products. *American Economic Journal: Microeconomics*.

Roth, A. (2002). The economist as engineer: Game theory, experimentation, and computation as tools for design economics. *Econometrica*.

Roughgarden, T. (2010). Computing equilibria: A computational complexity perspective. *Economic Theory 42*(1), 193–236.

Sockin, M. and W. Xiong (2022). Decentralization through tokenization. *Journal of Finance*.

Veldkamp, L. (2023). *Information choice in macroeconomics and finance*. Princeton University Press.

Wermers, R. (2000). Mutual fund performance: An empirical decomposition into stock-picking talent, style, transactions costs, and expenses. *Journal of Finance*.

Wigderson, A. (2019). *Mathematics and computation*. Princeton University Press.

Wittwer, M. (2021). Connecting disconnected financial markets? *American Economic Journal: Microeconomics*.

Wurgler, J. (2011). On the economic consequences of index-linked investing. In *Challenges to Business in the Twenty-First Century*.