CoFiX

A Computable Trading System

Written by
Zaugust
NEST Community

Translated by
James Sangalli james.sangalli@alphawallet.com
Victor Zhang victor.zhang@alphawallet.com

July 2020

Abstract

DeFi projects have a problem; they cannot calculate and manage risk because they are not using an oracle with controllable and computable risk. The NEST oracle solves this problem by providing a reliable price feed with a computable risk factor that does not rely on centralized infrastructure. By using the NEST oracle, we can provide an entirely new DEX, CoFiX, that brings reliable prices and a computable risk factor for traders and market makers. This article will discuss the background of DEX's today, explain in detail the works of our new DEX, CoFiX, and how we expect it to become an essential product in the space.

1 Background

Decentralized exchanges have always been the focus of the blockchain industry and for a good reason: they provide anonymity, fairness, ease of use, and openness.

While decentralized exchanges today are a step in the right direction, they are limited by on-chain performance barriers such as computing power, storage and communication shortfalls. They are also limited by the blockchain's technical characteristics like the time per block (in the real world or another digital system, time is almost continuous, but in a blockchain system, the smallest unit of time is a block, e.g., the smallest unit of time in Ethereum is around 14s).

Exchanges based on an Order Book model using centralized computing platforms are all more performant than on-chain decentralized services and can generate more accurate pricing, better efficiency, and lower costs. Today, assets with high liquidity and transaction volume (such as ETH and BTC) are much better suited to centralized exchanges. DEX's need a new model other than the Order Book model. In this paper, we take ETH as an example to study and analyze DEX design and offer a new solution.

1.1 Automatic Market Makers (AMM)

In addition to the matching model order book, trading with a market maker is a popular form of DEX on the Ethereum blockchain today.

An example of a centralized market making model is Tokenlon, where specific market makers carry out transactions on-chain to fill orders from traders. The benefit of this model is that it is easy to get an equilibrium price[1] from centralized exchanges, and trades can still be executed on-chain without using an extra trusted third party. This model has many drawbacks, including reliance on the capital of the specific market makers, low transparency, and reliance on centralized servers. This model looks more like an OTC using a smart contract to settle a trade than a DEX.

In order to compensate for this, DEX's today have opted for an AMM model that allows anyone to be the market maker, and it generates the price on chain based on a formula. It is always on, fair, transparent, and anonymous. The most popular DEX using the AMM model is Uniswap. Uniswap derives price pairs based on a constant formula x*y=k. It has around 200 million USD worth of trading volume per day, but most of that has been made up of low liquidity asset pairs. These facts show us that despite being very easy to use, they have failed to capture the mainstream market of crypto traders.

In addition, other platforms such as Balancer also use the AMM mechanism, and they allow anyone to build a combined asset pool that is not bound to one trading pair. It is an improvement to Uniswap but does not solve the liquidity issues plaguing all DEX's today.

There are also other DEX's like Bancor that use a different pricing formula, but it is also based on partial equilibrium, which cannot reflect the real equilibrium price. They also have similar shortcomings with liquidity and pricing so we will not expand further on this.

1.2 AMM problems

The main problems with AMM's are not their automatic market making or asset pooling but instead the following:

1. Formula based partial equilibrium pricing causes asset pool losses: Uniswap's pricing is based on a constant formula that is indifferent to the outside market. It means that if a large fluctuation happens on the outside, prices on Uniswap will not be able to adjust dynamically and will instead be moved by arbitrageurs who know something that Uniswap does not. If Uniswap is used for trading of highly liquid asset pairs, market makers need to charge higher fees to account for the risk of outside price fluctuations. [1] [2]
2. Poor risk management: on Uniswap, there is no way to quantify risks that traders and market makers undertake accurately. Market makers are only compensated by a fixed trading fee, which may not be able to cover risks or may conversely cause traders to pay an unfair trading fee. Arbitrageurs move the price back to general equilibrium at the expense of market makers and traders. [1] [2] [3]
3. Trading sizes are limited by Partial Price Impact[2]: trades on Uniswap are subject to unfavorable Partial Price Impact depending on the liquidity of the pool. This problem is severe in highly illiquid pairs as a 1% change in the asset pool size caused by trade will result in a 1% Partial Price Impact in the price for the trader. Larger liquidity pools reduce Partial Price Impact but also result in less profit for market makers who have to share the profits with other market makers in the pool. [1] [2]

[1] The equilibrium price is the market price where the quantity of goods supplied is equal to the quantity of goods demanded.

[2] Partial Price Impact is the difference between the equilibrium price and the execution price of a trade, which is caused by the trading impact on partial equilibrium, not the general equilibrium.

1.3 EPM (Equilibrium Pricing Model) and NEST oracle

If an accurate source provides prices based on general equilibrium instead of a fixed algorithm and partial equilibrium, it is easy for market makers to control their risks and effectively hedge against market forces while also providing a fair price for traders. This is the idea behind the EPM DEX model.

A DEX based on this model has several advantages:

1. It can support highly liquid mainstream trading pairs, which represent 80% of the total trading volume in exchanges today.
2. It eliminates Partial Price Impact because prices are based on general equilibrium and computable risks rather than a fixed formula and Partial equilibrium pricing, where the execution prices are heavily affected by trading sizes.
3. A single trade can use up the full liquidity of a pool, however, this can be an additional risk if the trading sizes are too big as they themselves may impact the equilibrium price.
4. By using the NEST oracle, all risks in such a DEX can be calculated, so market makers and traders can do accurate risk management to maximize their profit.

We expect that this model will become mainstream, and we have already seen projects like DODO and Bancor v2 start using the equilibrium price.

Obtaining equilibrium prices will require a decentralized oracle because the discovery of equilibrium prices heavily relies on centralized trading systems. This is completely unsuitable for a DeFi product. At present, most oracles carry centralization risk, and even the famous Chainlink project cannot completely remove a central point of failure with its node model. This is because the collateral requirements to run an oracle node providing inputs on Chainlink are decoupled from the actual cost of being inaccurate, and the prices cannot be verified in a trustless manner.

The only oracle on the market today that provides accurate and trustless pricing information is the NEST oracle [4] [5] . Its structure is entirely trustless, decentralized, and has no central point of failure. We believe that NEST is the only option for DEX's using the EPM model.

1.4 Computability and Algorithm-based risk control

As mentioned above, most of DeFi today does not have a comprehensive risk quantification and control. Projects like MakerDAO and Compound use price oracles but do not quantify the price deviation risk, volatility risks, and other risks, and therefore cannot do algorithm-based risk control. Similarly, recent EPM based DEX's like DODO and Bancor v2 use the same rough trading sizes based management model that Uniswap uses.

Current DeFi models only remove counterparty risk; they do not have a way to quantify other risks through algorithms. This is mainly because of the oracle, as it is the key module to pass equilibrium prices to DeFi smart contracts. Equilibrium prices are the only states that cannot be generated solely on-chain. If we can quantify the risks of an oracle, we can do algorithm-based risk control to optimize DeFi.

The NEST oracle gives us the ability to develop a new kind of DeFi model, based on the computability of the NEST-price[3]. We can quantify the risks of using the equilibrium price and design DeFi applications

[3] Prices from Nest Oracle

accordingly. With such a model, we can create DeFi 2.0, a new service where the risks can be quantified and managed by accurate algorithms. We call DeFi 2.0 “CoFi” (Computable Finance).

When risks can be calculated and priced based on an algorithm, we will have a financial system that is truly non-custodial and automated. We believe that CoFi is the way to push the financial industry into a new era.

2 CoFiX: a computable DEX

CoFiX is the first innovative product in the CoFi field, and it is a DEX based on the NEST oracle. CoFiX consists of an oracle module, a market-making module, and a trading module. The idea of CoFiX is this: market makers and traders get their prices from the NEST oracle and can price in their computable risks with each respective trade. It ensures that market makers are motivated to make the market, and the traders trade with minimal price spread and without Partial Price Impact. We call it CoFiX because all the above parameters are computable.

Note: CoFiX only supports trading pairs that are included in the NEST oracle; if users want to trade a pair on CoFiX that is not included in the NEST oracle, they will need to add a new pair.

2.1 NEST price performance and computability

NEST is a decentralized price oracle based on financial arbitrage. Through Double Options, Price chains, and the beta coefficient, we can achieve the effective prices (almost equal to the current equilibrium prices) on-chain. The difference between the NEST-price and the real-time equilibrium price is within a computable and acceptable range [4] [5] .

2.1.1 Basic parameters of the NEST price

Parameters	The meaning of the parameters
P	Prices from the NEST oracle corresponding to a block number; if the current block does not contain a price, it will revert to the price in the previous block.
P_0	The equilibrium prices
g	The deviation in prices, it is calculated with $\|P - P_0\|/P$
σ	On-chain volatility rate
σ_0	Off-chain volatility rate
T_0	Verification time
T	Price delay, if an application needs to use the price in block N, but the latest price from NEST is in block N_0. $T=N-N_0$
q	The probability of standard quotation being arbitraged. When the price deviation is large, the standard quotation will also be arbitraged. This probability describes one of the essential characteristics of the NEST oracle.

According to NEST's research paper [5] , under normal circumstances, the theoretical deviation in prices from NEST oracle, g is about 0.3%. The actual data is very close to this, as it is currently at about 0.4%. See NEST's research paper [5] for more detail about how to quantify the two significant price risks of NEST: deviation and delay.

2.1.2 Price compensation coefficient

The risk of using a decentralized oracle like NEST is caused by the deviation and the delay (T). CoFiX needs to compensate for this risk when quoting prices from NEST to ensure that the market maker is sufficiently incentivized to continue making the market.

Compensation factor **K** [6] [7] is the coefficient related to the volatility rate δ and delay T. When a trader makes a transaction, he does not directly use price **P** but rather **P' =P*(1+K) (or P' =P*(1-K))**. Similarly, when a market maker enters and exits the market, they use the price variable **P'** instead of **P**. This price is known as the transaction price.

2.2 Market Maker Mechanism

The market maker is still using an AMM, but unlike Uniswap, they can inject any asset into the pool without it being a pair. It is here that we introduce the concept of the net worth of the asset pool, which is that the market maker treats the asset pool as a fund. Every time a market maker enters and exits the pool, the amount they receive will be calculated based on their share of the net worth of the fund. This is an essential improvement to Uniswap because CoFiX does not require bilateral asset pairs to form prices because of the NEST oracle.

2.2.1 Asset pool and trading pairs

Each trading pair corresponds to an asset pool where the main trading asset is ETH. That is each asset and ETH form one trading pair. As noted earlier, each pair needs to be supported by the NEST oracle.

2.2.2 Market makers receive XToken to represent their stake in a pool

If a market maker adds ETH or USDT to the ETH-USDT pool, the market maker will receive XToken that represents their stake in the pool (share of the fund).

2.2.3 Market maker net worth calculation

If a market maker adds ETH or USDT to the pool, the net worth of their stake will be adjusted every time a market maker enters or exits the pool [7] . The net worth calculation is priced in ETH based on price P'.

2.2.4 Perfectly hedge the market making risk

Market makers need to do hedging to lock down the profit from price spread. The amount to hedge can be easily calculated according to the market maker's stake in the pool.

2.2.5 Other risks of market makers

CoFiX quantifies and manages risks based on a set of fundamental finance principles, but there are still additional potential risks if market makers improperly manage their assets. These potential risks are as follows:

1. Since the coefficient K is designed based on statistical analysis, if the exchange pair is not frequently traded, the market maker may be at risk of loss. Besides, if the market maker only provides liquidity for a short period, they may be at risk of loss. These losses are probabilistic.

2. If the market maker does not hedge on other exchanges, they may bear price risks of an asset changing in price.
3. The volatility rate is estimated based on historical data that may deviate from the actual volatility.
4. If an asset has inferior liquidity, it could trigger Circuit breakers. This may affect the profitability of market makers.

2.3 Trader mechanism

Traders complete transactions on CoFiX at prices acceptable to them. According to the analysis above, the deviation between the trading price P' and the equilibrium price P is minimal, and there is no Partial Price Impact caused by trading volume. These are benefits unique to CoFiX.

2.3.1 Transaction mechanism

With ETH-USDT as an example, as long as there are assets in the pool, the trader can enter the market according to the transaction price P' based on the NEST oracle. This process is KYC free, and the trader does not need to worry about Partial Price Impact, exchange security, or trading with prices different from the equilibrium price. For each trade, the NEST oracle must be compensated for providing quotes, but this process is as simple as paying for gas.

A trade is settled with the latest price in the block of the transaction, not the price when they submit a transaction. There may be a time delay for their transaction to be included in a block, so certain protections should be set up for traders based on time delay or price differences.

2.3.2 Circuit breakers

According to the CoFiX trading compensation algorithm [6] , if volatility rate rises to an extreme level or the NEST system is attacked, CoFiX needs to activate an emergency procedure to protect both the trader and market maker. The system should be able to trigger Circuit breakers when the following conditions are met:

1. The K_0 value exceeds a range, e.g., K_0>5%
2. The volatility rate σ rises to a limit, e.g., σ>0.1% per second
3. Delay T exceeds a range, e.g., T>900s

2.3.3 Transaction risk

Traders have risks outside the model, such as transaction delays from low gas prices, which can cause differences between the expected price and actual settlement price[7] . This is an inherent risk of all AMM trading mechanisms.

2.4 Equilibrium Price Impact cost

When the size of the CoFiX asset pool is large enough, it will be difficult for a transaction to reach the upper limit of the asset pool's liquidity; however, a single large transaction could affect the market maker's hedging cost, as such a transaction may have a significant impact on the equilibrium price.

For example, trading 1 million ETH in a CoFiX pool without the price changing would result in a loss for market makers hedging on the exchange as this trade would change the equilibrium price of ETH.

Such large transactions will pay a premium, known as the impact cost **C**, to cover the risk. The impact cost is calculated by the price difference coefficient **K_1 =K+C**, causing the price in the system to change to **P' =P*(1+K_1)**. Trading Compensation of CoFiX [6] and CoFiX Product Documentation [7] explains in more detail the Equilibrium Price Impact cost.

3 System advantages and outlook

CoFiX can be said to be the first DEX to achieve a comprehensive solution that is not only decentralized but also accounts for risks to traders and market makers.

With quantifiable risk optimization for all parties involved, CoFiX is vastly superior to the DEX's on the market today. We believe that these characteristics make CoFiX suitable for mainstream users and mainstream crypto assets that represent more than 80% of the total crypto trading volume. With this in mind, it is easy to see how the trading volume on CoFiX will be 10x that of traditional DEX's.

References

[1] Uniswap v2 Core
https://uniswap.org/whitepaper.pdf

[2] Bancor Protocol
https://storage.googleapis.com/website-bancor/2018/04/01ba8253-bancor_protocol_whitepaper_en.pdf

[3] The Math Behind PMM
https://dodoex.github.io/docs/docs/math

[4] NEST Protocol
https://nestprotocol.org/doc/ennestwhitepaper.pdf

[5] How Accurate the NEST Price Is
https://nestprotocol.org/doc/NEST_Price_Performance.pdf

[6] Trading Compensation of CoFiX
https://cofix.io/doc/Trading_Compensation_CoFiX.pdf

[7] CoFiX Product Documentation
https://docs.cofix.io/

Appendix 1:

Trading Compensation of CoFiX

September 23, 2020

ABSTRACT

We estimate the difference between the trading price and a source price by which liquidity providers of CoFiX can hedge their risk. The compensation rate is determined by the upper bound of the difference.

Keywords: Blockchain, Ethereum, smart contract, Oracle, decentralized exchange

1. Transaction Fee and Price Inaccuracy

While the reference price is the decentralized price generated from the Oracle, we are interested in controlling the maximal loss of liquidity providers, for the purpose of designing a reliable decentralized trading system.

Let P_t be the decentralized price (effective price generated by the Oracle), S_t be the price outside of the blockchain (i.e. the price from an exchange), $\tilde{P}_t$ be the reference price for a trade occurred in the blockchain. All are at time t.

Let $\hat{P}_t$ be the execute price adopted by the trading system. It is linked to $\tilde{P}_t$ by the relation: if one sells the asset

$$\hat{P}_t = \tilde{P}_t(1 - K)$$

if one buys the asset

$$\hat{P}_t = \tilde{P}_t(1 + K).$$

That is, the liquidity providers require an extra percentage charge $K\tilde{P}_t$ for a transaction in the trading system. The amount is proportional to the reference price and is used to cover the potential loss due to the difference between the reference price and the outside price. We identify two major sources that contribute to the difference: delay of the blockchain system and inaccuracy of the Oracle prices.

There are many reasons[1] that make referred price $\tilde{P}_t$ (the true price generated by the Oracle) is a delayed version of optimal input price P_t (the theoretical price generated by the Oracle mechanism). Let T be the delay of time. The two prices are connected by

$$\tilde{P}_t = P_{t-T}.$$

Assuming that the normalized price P_t/P_{t-T} follows a Gaussian distribution

[1] The major reasons include malicious attacks to the Oracle, large price changes, transaction blocked/delayed in the blockchain system.

$N(0, \sqrt{T})$, we can find that

$$|P_t - P_{t-T}| < P_{t-T}b$$

for a positive constant b, with probability $\alpha := \phi(\frac{b}{\sigma\sqrt{T}}) - \phi(\frac{-b}{\sigma\sqrt{T}})$, where $\phi(\cdot)$ is the CDF of standard normal random variable.

On the other hand, the Oracle system generates P_t different from S_t, with a boundary

$$|P_t - S_t| < S_t a,$$

where $a = a(\sigma^2)$ is the difference of Oracle price P_t and S_t. (This quantity equals $2V + \epsilon$ as described in a separate paper of the NEST system.)

So the difference between the referred price and the outside price is

$$|\tilde{P}_t - S_t| = |P_{t-T} - S_t| \leq |P_{t-T} - P_t| + |P_t - S_t| = P_{t-T}b + S_t a = \tilde{P}_t b + S_t a$$

It follows that $(1 - a > 0)$

$$\frac{1+b}{1-a} > \frac{S_t}{\tilde{P}_t} > \frac{1-b}{1+a}.$$

Thus,

$$|\tilde{P}_t - S_t| < \tilde{P}_t(b + \frac{1+b}{1-a}a). \tag{1}$$

The trading system ensures liquidity providers that with probability α, their loss from trading on inaccuracy prices is covered if the extra charge (transaction fee) rate K satisfies

$$K\tilde{P}_t \geq |\tilde{P}_t - S_t|.$$

It is sufficient that we select K such that

$$K \geq \frac{1}{\tilde{P}_t}\tilde{P}_t(b + \frac{1+b}{1-a}a) = \frac{a+b}{1-a}. \tag{2}$$

We summarize the above analysis into a proposition as follows.

PROPOSITION 1. *If the trading transaction fee rate K satisfies (2), then the liquidity providers in the trading system will not lose with a probability $\alpha = \phi(\frac{b}{\sigma\sqrt{T}}) - \phi(\frac{-b}{\sigma\sqrt{T}})$.*

For a given set of (σ^2, T, α), we can find b, a and then select a value of K satisfying (2). The figure (1) depicts the quantitative relations of K and σ^2, T, where a linear approximation plane is given for efficient usage in smart contracts, $\alpha = 0.95$.

We illustrate the result by several numerical examples.

$\mu = 0$, $\sigma = 0.0002$, $T = 3600$ (one hour), $a = 0.005$, we will find the difference is less than 1.5% with probability 99%. In other words, the parameters k in the formula should be 1.5%. for this case.

The inequality (1) also provide an upper bound of loss encountered by liquidity providers. Let $b_0 = b(\alpha)$, which is a positive solution to $\alpha = \phi(\frac{b(\alpha)}{\sigma\sqrt{T}}) - \phi(\frac{-b(\alpha)}{\sigma\sqrt{T}})$ for a given α. Let $Z \sim normal(0, \sigma\sqrt{T})$ and

$$P_t - P_{t-T} = P_{t-T}Z.$$

Then the expected loss (relative to the trading price $\tilde{P}_t$) is (conditional on the region $|Z| > b_0$)

$$E[L_t/\tilde{P}_t||Z| > b_0] = E[\frac{a+|Z|}{1-a}1_{\{|Z|>b_0\}}] = \frac{a}{1-a}(1-\alpha) + \frac{1}{1-a}E[|Z|1_{\{|Z|>b_0\}}]. \tag{3}$$

The unconditional expected loss in percentage of the refereed price is

$$E[L_t/\tilde{P}_t] = E[\frac{a+|Z|}{1-a}] = \frac{a}{1-a} + \frac{1}{1-a}E[|Z|] = \frac{a}{1-a} + \frac{1}{1-a}\frac{\sigma\sqrt{T}\sqrt{2}}{\sqrt{\pi}}. \tag{4}$$

This expected loss can be used to determine K as well. The result is summarized as the following proposition.

PROPOSITION 2. *The expected loss of liquidity providers relative to reference price $\tilde{P}_t$ is bounded by*

$$K_0 = \frac{a}{1-a} + \frac{1}{1-a}\frac{\sigma\sqrt{2T}}{\sqrt{\pi}}, \tag{5}$$

where a is the upper bound of the difference between the Oracle price and the price in an exchange.

The related numerical result is shown in Figure 2.

If we use the upper bound directly as the rate K to adjust buy or sell prices, the liquidity providers may be over-compensated, because extra skills/efforts are required for the traders to make the liquidity providers lose in most of time. To balance the interests of traders and liquidity providers, we let

$$K = \gamma \cdot K_0 \tag{6}$$

in practice, where $\gamma \in (0, 1]$ represents the percentage of the expected loss compensated to the liquidity providers. For a choice of $\gamma < 1$, we suppose that liquidity providers and the traders shall share the risk of price inaccuracy. Considering traders have to pay transaction cost of the blockchain, it may be fair that liquidity providers bear part of the cost through this channel of γ.

In practice, value of K can not be updated for each trade due to gas expense. It is necessary to update K's value periodically. Denote value of K by $K(a, T)$, which is calculated from a and T by (5) and (6). Then the value of K after t seconds from the last update is adjusted as $K' = K(a, T + t)$. This adjustment assumes that no effective NEST price is available in the current block.

1.1 Geometric Brownian Motion

In this subsection we consider an alternative assumption that the price sequence P_t follows a geometric Brownian motion. Precisely, we assume that

$$dlog(P_t) = \sigma dB_t.$$

It follows that

$$P_t = P_{t-T} e^{\sigma Z_T},$$

where $Z_T \sim normal(0, \sqrt{T})$. By similar steps as above, the expected loss under the assumption of geometric Brownian motion is

$$E[L/P_{t-T}] = E[\frac{a + |e^{\sigma Z} - 1|}{1 - a}]$$

The numerical test shows that the difference of expected losses under the two assumptions is as small as 0.0006. Not significant. Actually, for small σ the Taylor expansion explains the result.

1.2 Conventional Computation Methods

It will be convenient for DApps if the NEST also provide volatility in addition to price. Considering expense of calculation in blockchain, we adopt the exponentially weighted moving average model to calculate volatility per block in a timely manner. In addition, we assume a geometric Brownian motion for the asset price, or equivalently assume a Brownian motion model for the log-return of the asset price. Denote

$$u_t = \frac{P_t}{P_{t-1}} - 1.$$

Let

$$\sigma_t^2 = \lambda \sigma_{t-1}^2 + (1-\lambda) u_{t-1}^2,\ t = 2, 3...$$

where $\lambda \in (0,1)$ and σ_1^2 can be calculated by a usual method, or just let it be u_1^2 (after some time, the result will be fine).

The weight λ causes a term u_{t-m} declined at a speed of λ^m. Therefore, more recent returns (u_{t-k}) impact the volatility (σ_t) greater.

For the application in the NEST, we have to consider the block gap between two successive effective prices. Let n_t be the number of blocks between the block with price P_t and that with price P_{t-1} and let *timespan* denote the average time (in seconds) between two successive blocks.

Then the formula is adjusted as follows.

$$\sigma_t^2 = \lambda \sigma_{t-1}^2 + (1-\lambda) \frac{u_{t-1}^2}{n_{t-1} \cdot timespan},\ t = 2, 3... \tag{7}$$

with the start point $\sigma_1 = \frac{u_1^2}{n_1 \cdot timespan}$. The weight λ can be set as 0.95.[2]

In addition, we also provide a linear approximation formula for economically calculating the NEST price deviation as follows.

$$a(\sigma) = -0.0014687 + 19.8898 \times \sigma + gascost/10. \tag{8}$$

where "gascost=gas price × gas consumed per transaction" is set to be 0.03 currently. It may be updated according to a real situation.

As a summary, the conventional way to compute K_0 is: compute σ by (7), compute a by (8), then compute K_0 by (5).

[2] With this weight, the latest 50 (25) effective prices take more than 90% (70%) weights of the total. The same idea can be applied to construct an exponentially weighted moving average (EWMA) of the asset price. That is, $\bar{P}_t = \lambda \bar{P}_{t-1} + (1-\lambda) P_t$, with $\bar{P}_1 = P_1, t = 2, 3, ...$ By a set of data, the probability of a price greater than the EWMA more than 2.5% is only 0.19%. **Therefore, an extra constraint $|P_t/\bar{P}_{t-1} - 1| < 2.5\%$ may be imposed in the system to avoid extraordinary prices from the NEST oracle.**

Furthermore, an even more cost-saving approach is to consider the average of K_0 over all pairs of (T, σ). That is, calculate

$$\bar{K}_0 = E[K_0(T, \sigma)],$$

where the expectation is with respect to the joint distribution of random (vector) variable (T, σ).

We estimate the expectation based on a data set spanning from July 13,2020- Sep 13,2020 and find that $\bar{K}_0 \approx 0.005$.

Adjusting the execution price by $\bar{K}_0$, the price deviation average with respect to randomness (risk) from asset price P_t, delay time T, and volatility σ, the liquidity providers are supposed to be compensated fully over a relative long time period.

2. The Risk of Impact Cost

An important issue is *impact cost*, that is usually referred to the extra cost due to significant price change when a large volume of assets are traded. For the trading system referring Oracle prices, it is unwise to fix a trading price for a relative large volume trade. This actually introduces a new risk to liquidity providers, we refer it as *the risk of impact cost.*

To handle this issue, we have to carefully study an order-book of an off-chain centralized exchange where liquidity providers hedge the risk from the decentralized trading system. The idea is to adjust trading prices according to trading volume. The mathematical relation between the price and volume shall be derived from the order book. By this design, the liquidity providers can hedge the impact cost risk effectively.

Through initial analysis of a set of order book, the impact of trading on price (impact cost) is depicted by Figure 3. It indicates that the impact is not significant, below 0.3% for trading volume less than 1000 units of ETH. The data is from huobi.com.

3. The Delay Risk to Traders

A trading order may be delayed due to blockchain's performance. One trader places an order when seeing a specific reference price, his order most likely will be delayed for several blocks (up to 50!) until it is packaged in a new block. At the moment, the order is executed by **the reference price at that moment ($\tilde{P}_t$ in the preceding analysis).** Suppose the delay time is ν whose value is relevant with the situation of the blockchain system. The risk encountered by the trader is controlled by a normalized random variable as follows.

$$\tilde{P}_t/\tilde{P}_{t-\tau} - 1 = Z \sim normal(0, \sigma\sqrt{\nu})$$

So the trader may buy/sell at a higher price than what he sees with probability 50% (half-to-half, that is fair). The price is higher than 1% is with probability

$$\Pr(Z > 0.01) = Pr(Z_0 > \frac{0.01}{\sigma\sqrt{\nu}}),$$

where $Z_0 \sim normal(0,1)$. For $\sigma = 0.0005, \nu = 60(seconds)$, the probability is only 0.5%. If we change 1% to 0.5%, the probability increases to 9.8%. That matters! Thus, **we suggest to show a reminder about this delay risk to traders.**

For market markers, this delay is not a risk because they hedge at the execute price (the reference price), not the price that the trader sees.

Figures

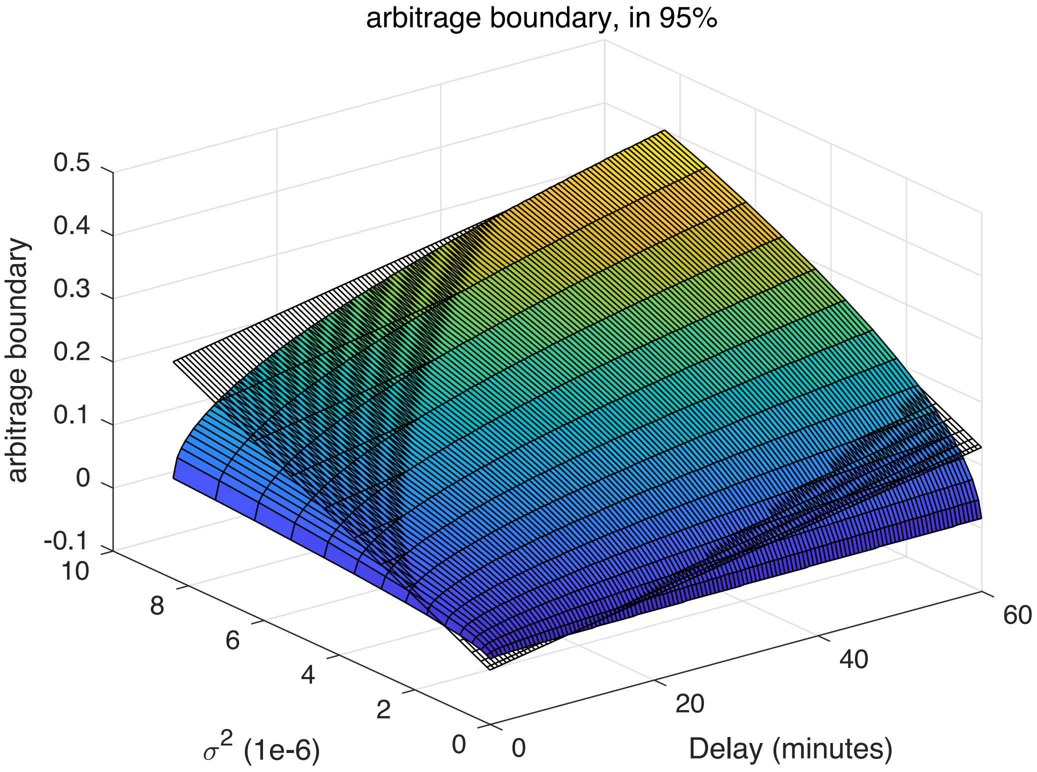

Figure 1. This figure depicts price inaccuracy when using the effective price as a reference for a trade. The gray flat plane is a linear approximation to the surface.

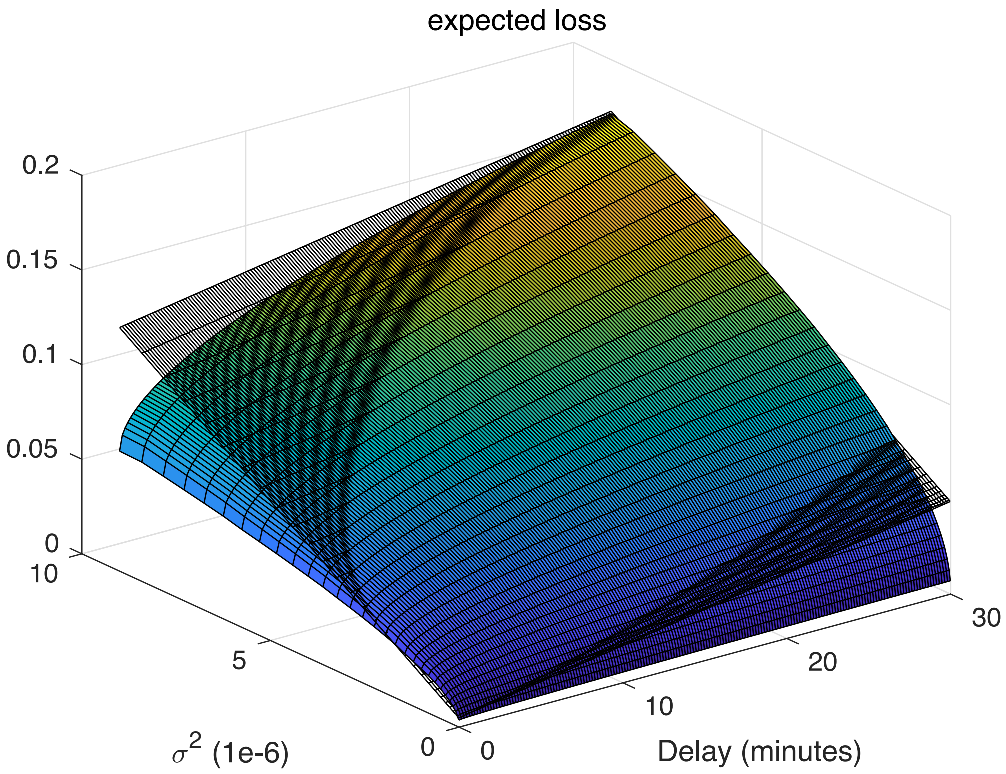

Figure 2. This figure depicts expected loss when using the effective price as a reference for a trade. The gray flat plane is a linear approximation to the surface.

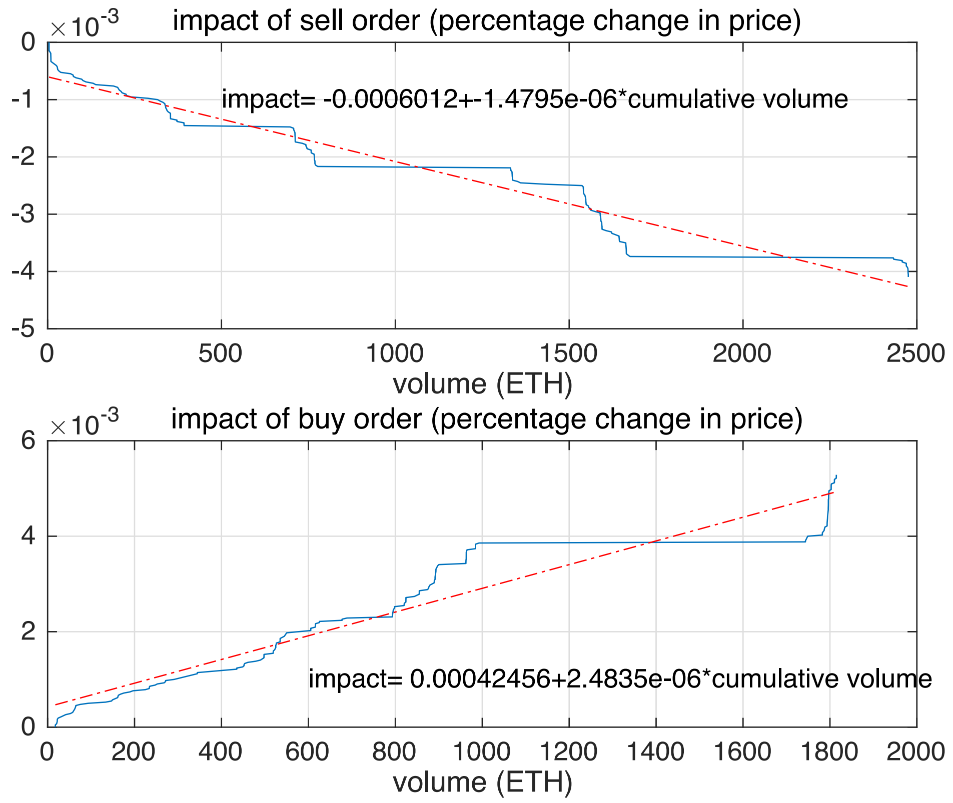

Figure 3. This figure depicts possible trading cost magnitude.

Appendix 2:

How Accurate the NEST Price Is

ABSTRACT

This short article develops a model to estimate the difference between the NEST price and a source price, e.g. price from an exchange. Under plausible assumptions, we show that the difference can be as small as 0.003 when volatility is small. It can even be lower if the transaction cost in the blockchain gets lower.

1. Model Setup

A *price-provider* is an individual who inputs a price into the NEST system and waits for a certain number of blocks passing to be verified by other individuals. The operation is equivalent to write an American type call and put option that anyone else can exercise it by using the input price as the exercise price. Thus, the price-provider shall minimize the value of this option by carefully choosing an input price. Precisely, the price-provider's objective problem is

$$P^* = \arg\min_{P} \left(\max_{\tau} E^Q[e^{-r\tau}|S_\tau - P|] \right), \tag{1}$$

where $\tau \leq T_0$ is a stopping time and T_0 is a fixed time horizon[1] , P is the input price decided by the price-provider. In other words, the price-provider has to minimize the value of one American type option by choosing an appropriate exercise price P . Here asset price S_t, $t \geq 0$ shall be referred to the price in an exchange at time t. Thus, the market is complete and we price the derivative in a risk-neutral framework by taking the expectation under the risk-neutral probability Q.

Denote the solution to the above problem by $P^* = P(S_0; \sigma)$, where σ is the volatility of the source price sequence S_t. Noting that the price-provider inputs a price optimally based on all of his information from a centralized market and/or from the decentralized world.

1.1 Arbitrageur

The price-provider writes an American option when he inputs a price K. It seems that anybody can exercise the option without any cost. However, the NEST requires that the one (arbitrageur) who exercises the derivative must input another price and lock in as much as β times the original asset requirement. In other words, to exercise one option, the arbitrageur

[1] For the NEST system, the time horizon actually is random because the time interval between two successive Ethereum blocks is. The framework in this note can be extended to study this case.

has to write β units of the same type of American options, where $\beta > 1$ is a specific multiplier.

One arbitrageur who wishes to make profit from the derivative can construct (sell) a portfolio in the outside market that replicates the derivative. Then the arbitrageur can make a risk-free profit the same as the value of the derivative. However, there is risk that the arbitrageur can not obtain the opportunity to exercise the derivative because it is competitive to take the arbitrage. Therefore, instead of making the risk-free profit, a realistic strategy is to make a *quick* profit in the sense of statistic arbitrage as follows.

The arbitrageur does nothing but waits until the difference between the outside asset price and the input price P is sufficiently large. Then he exercises the option and buys or sells in the exchange simultaneously to make money without any risk. Such an opportunity may not be available for all time, but in long time there are many chances. So statistically the arbitrageur can make money.

We calculate the following objective function for the arbitrageur:

$$\max_{\tau} E[(|S_\tau - P| \ - a) 1_{|S_\tau - P| > A, \tau < T_0} \tag{2}$$

where A represents all costs of the transaction, including Ethereum transaction fee and the value of the derivative multiplied by β. The stopping time τ in the above indicates that the arbitrageur will wait for the best time to take the arbitrage. However, considering the competitive environment, most likely, the profit is taken when the first time a target is reached. So the objective function turns to be

$$E[(|S_\eta - K| - A) 1_{\eta \le T_0}] \, , \tag{3}$$

where $\eta = \inf\{t : |S_t - P| - A > \epsilon\}$ and ϵ is the minimum target profit of the arbitrageur. Along with the arbitrage-taking method (3), the corresponding loss (or the cost of inputing a price) of the price-provider is

$$E(|S_\eta - P| \, 1_{\eta \le T_0}) \, .$$

The price-provider shall minimize the cost by choosing an appropriate K. That is, the objective function of the price-provider is

$$\min_P E[|S_\eta - P|\, 1_{\eta \le T_0}.$$

In fact, we should price it in a risk-neutral sense:

$$V^*(0) = \min_P E^Q[e^{-r\eta}|S_\eta - P|\, 1_{\eta \le T_0}]\ ,$$

where r is the risk-free interest rate. It yields that the price-provider can construct a portfolio in the outside market to hedge this derivative, so that his loss is a deterministic value same as V^*.

2. A Solution of the Model

Given the design of the NEST, we let

$$A = \beta V^*(\eta),$$

where $V^*(\eta)$ denotes value of the same derivative at time η. We let ϵ be the transaction fee in the blockchain (the gas fee).

Aware of the way the option is exercised, the price-provider actually considers the objective problem as follows.

$$V^*(0) = \min_P E^Q[e^{-r\eta}|S_\eta - P|\, 1_{\eta \le T_0}] = \min_P E^Q[e^{-r\eta}(A + \epsilon)1_{\eta \le T_0}] = \min_P E^Q[e^{-r\eta}(\beta V^*(\eta) + \epsilon)1_{\eta \le T_0}]. \tag{4}$$

We assume that the asset price follows a Brownian motion with drift:

$$S_t = S_0 + \mu t + \sigma Z_t,$$

where Z_t is a standard Brownian motion. Then $V^*(\cdot)$ is identical at any time. The recursive formula (4) is simplified (for a stationary solution under constant state variables μ and σ)

$$V^* = \min_P E^Q[e^{-r\eta} 1_{\eta \leq T_0}]\,(\beta V^* + \epsilon). \tag{5}$$

Exploiting the density function of η, the first hitting time of Brownian motion, we can evaluate the expecation in (5) and solve for V^* and P^* numerically.

Set $\mu = r = 0$, $\epsilon = 0.003$ (the gas fee of one transaction in the Ethereum/10ETH), $S_0 = 1$, we obtain the following results.

For $\sigma = 0.0001,\ 0.001,\ 0.003$ per second:

$\beta = 1.5$: $V^* = 0.0030, 0.0104, 0.0327$; probability of arbitrage: 0.0726, 0.3353, 0.3765

$\beta = 2$: $V^* = 0.0003$, 0.0092, 0.0291; probability of arbitrage= 0.0792, 0.4301, 0.4755,

$\beta = 3$: $V^* = 0.0002$, 0.0074, 0.0233; probability of arbitrage= 0.0894, 0.6064, 0.6696, where the probability of arbitrage is defined by $E^Q[1_{\eta \leq T_0}]$. For all of these cases, the optimal input-price $P^* = S_0 = 1$. Since S_t is assumed to be a Brownian motion without a drift, this answer is obvious.

The sensitivity analysis regarding verification during time T_0 , probability of arbitrage, β, volatility σ are shown in Figure 1 and 2.

2.1 Difference between NEST Price and Price of Exchange

By the preceding analysis, the difference between the NEST price and the price from an exchange is bounded by $a := \beta V^* + \epsilon$. Figure 3 indicates the upper bound can be as small as 0.003. The upper bound can be decreased if the transaction (arbitrage) cost in the blockchain becomes small. Alternatively, We may increase the asset requirement of inputing a price to decrease the relative weight of ϵ. For example, if we increase the asset requirement to 50 ETHs, the difference bound turns to be 0.002 only.

Figures

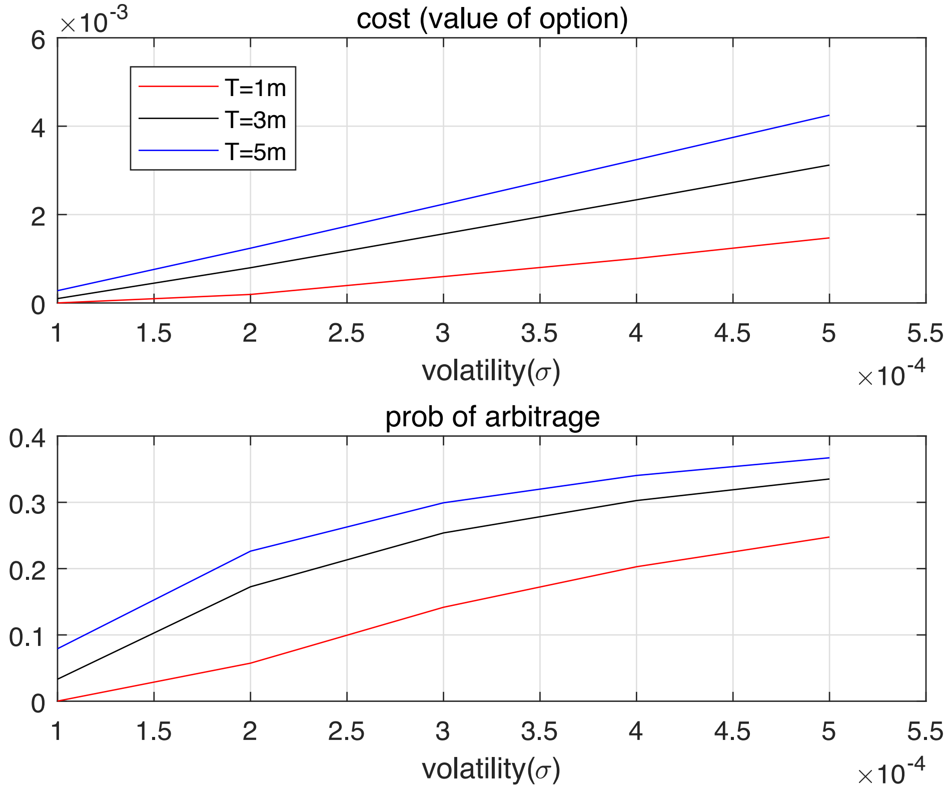

Figure 1. This figure depicts effects of volatility σ on cost of price-inputing and probability of arbitrage.

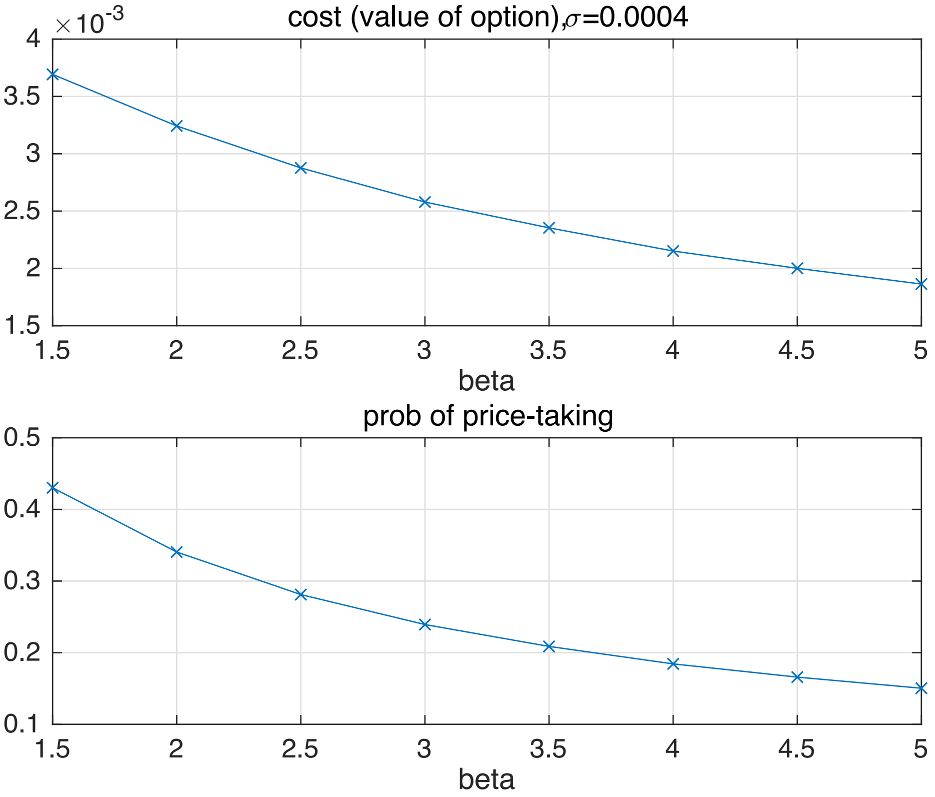

Figure 2. This figure depicts the effect of β on cost of price-inputing and probability of arbitrage.

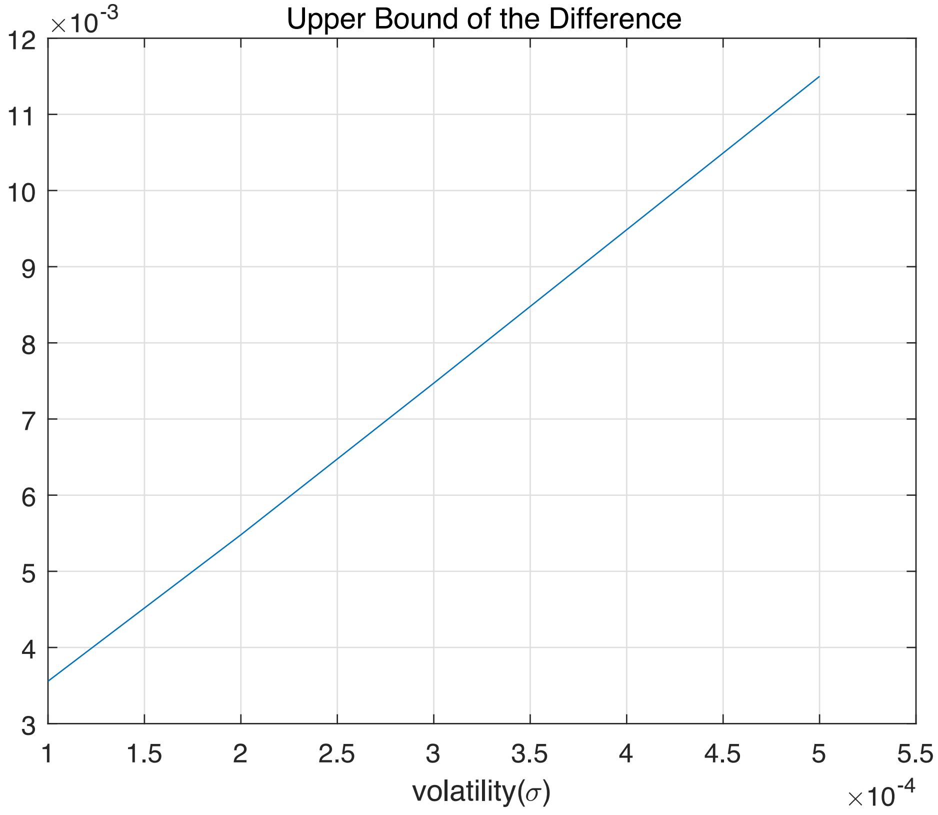

Figure 3. This figure shows the upper bound of difference between the NEST price and the price of an exchange at the same time.