CoFiX：一种可计算的金融交易模型

摘要：当前 DeFi 项目普遍没有对风险进行量化和使用算法进行风险管理，一个重要的原因是缺少风险可控的预言机。NEST 预言机提供了风险可计算的价格序列，这使得基于 NEST 价格的 DeFi 有了全新的特性：风险可计算且可使用算法管理，我们称之为 CoFi。在 CoFi 领域，第一个创新性产品是 **CoFiX**：一种可计算的 DEX。本文详细论述了 DEX 的背景和 CoFiX 的原理，并预计其将成为未来区块链交易领域最重要的产品。

一、背景

去中心化交易所 DEX 一直是区块链行业的焦点，因为其匿名性、公平性和开放性受到追捧。但去中心化交易所受限于区块链的性能劣势：无论是计算、存储还是通信，中心化的计算系统都全面优于区块链，这使得在链上展开基于 Order Book（即双边撮合）的交易模型成本高昂，效率低下。如果再考虑区块链系统是以区块为时间轴的世界，相比链下要显得粗糙和不确定。因此对于高流动性和高交易频次的资产诸如 ETH、BTC 等，其交易和定价在中心化交易所完成更为自然。在 DEX 交易需要特别的、适合链上的交易模型。本文以 ETH 为例，研究和分析对其交易的 DEX 设计，并给出一种新的方案。

1. AMM

在撮合模型之外，做市模型在链上得到了应用。一种中心化做市模型的例子是 Tokenlon：由指定的做市商与交易者进行链上交易。这种交易模型的好处是在获得市场价格（主要来自中心化交易所）的同时也能保证一定程度的安全性。但缺点也很明显：流动性完全取决于指定做市商的资本实力，同时开放性较差、依赖中心化服务器等。总体来说，这一模型是智能合约的应用，而不是 DEX。

为了实现匿名性、开放性和公平性，行业开发出了 AMM 模型（自动做市商模型）。AMM 与 Tokenlon 的差异在于，做市商是完全开放的，整个交易过程完全在链上，真正实现了去中心化，其代表是 Uniswap：基于 X*Y=K 这样的无差异曲线为一个交易对进行定价，任何人都可以成为做市商，并向市场提供流动性。Uniswap 非常简单开放，其日交易规模最高已经达到 2 亿美金，但大部分交易是由低流动性资产（大型中心化交易所没有或交易量较低的资产）贡献的，这说明 Uniswap 在解决高流动性资产交易方面的实践并不成功。

除此之外，像 Balancer 等合约，也采用了 AMM 机制，但可以构建组合性的资产池，即不限于一个交易对的资产池，以方便兑换。这是对 Uniswap 的一种改进，减少了 ETH 的重复消耗，然而并没有改变 Uniswap 对流动性资产交易不足的缺点。

还有一类交易合约，即 Bancor 协议等，也应用了不同类型的算法定价，本质上是基于供给需求关系的局部均衡设计，和 Uniswap 比较类似，其缺点也一致，在此不再展开。

2. 当前 AMM 的问题

当前 AMM 的主要问题不在于自动做市或资产池，这两点都非常适合区块链，主要问题在于以下几点：

1) 算法定价被套利：Uniswap 的算法定价是基于局部均衡的无差异曲线定价，而没有考虑一般均衡市场（也即外部更大更有效的交易场所），这导致在外部市场价格波动的情况下，Uniswap 的算法定价被持续套利。Uniswap 如果用来给高流动性资产提供交易，做市商就需要向交易者收取较高的费用以覆盖价格波动的风险，即使满足这样的条件，交易者和做市商之间达成的均衡也极为脆弱，系统很难持久。除 Uniswap 之外，Bancor 等算法定价模型存在同样的问题。

2) 风险管理很粗糙：基本上没有对交易双方的风险进行分类和有效定价，或者使用成交量管理所有风险。这一点在 Uniswap 上比较突出：做市商承担了什么风险在模型里没有描述，对该风险的合理补偿也没有计算，只是简单的基于套利机制使得价格回归——而且是只要价格回归即可，至于回归的成本是否合理则不在模型考虑范围之内。

3) 交易规模受制于各种滑点：此问题和 2）是一致的，因为用了成交量进行风险管理或者价格回归，必然导致各种不精确的滑点，而滑点和交易者承担的风险无法匹配，交易规模受到诸多限制。以高流动性资产为例，Uniswap 交易 1%的资产池规模会导致价格滑点为 1%，这相当于于向用户收取极为高昂的交易费用（或者通过多次交易减少滑点，但 GAS 费用又会大幅上升），或者要求资产池的规模巨大，但这又导致收益率可能不足以吸引做市商做市（这就需要向做市商进行补贴）。

3. EPM 与 NEST 预言机

如果不采用算法定价，而是直接采用交易所均衡价格，则做市商容易控制风险并完成有效对冲，这一思路即为均衡定价模型 EPM 的 DEX：交易双方基于交易所的均衡价格进行交易。基于 EPM 设计的 DEX 有几大好处：首先可以交易主流资产（交易所的流动性资产），而不是类似 Uniswap 的长尾资产，要知道主流资产的交易量占到 80%以上；其次是不再有滑点的问题，因为不是采用交易量定价，所以价差较小；再者单次交易规模可以接近整个资产池中某类资产的规模，只需要考虑全市场的冲击成本即可；最后在使用了合理的预言机的基础

上，DEX 上各类风险是可以计算并可以通过算法进行风险管理的。我们预计，基于 EPM 设计的 DEX 将成为行业的主流发展方向，近期类似的项目诸如 DODO 和 Bancor2.0 等都引入了市场均衡价格。

在链上得到均衡价格需要去中心化预言机（前面已经提到，不会在链上直接通过 Order Book 生成均衡价格）。目前市场上大部分预言机都带有中心化风险，即使像 Chainlink 这样知名的预言机项目，其节点模型的中心化风险也无法消除：要么抵押品需要和下游应用的风险匹配，要么因为引用的价格没有得到验证而产生信任风险。目前市场上唯一先验证再使用的去中心化预言机项目为 NEST，其结构符合区块链和去中心化精神，整个机制设计巧妙，比较符合事物的本质（参见文献 1,2）。目前其报价密度远超过各类预言机，我们认为 NEST 是 DEX 设计的唯一选择。

4. 可计算性与算法风控

当前大部分 DeFi 项目没有对自身的各种风险进行链上计算，并用算法有效管理，知名项目诸如 MakerDAO，Compound 等，其预言机的价格偏差风险、波动率风险等都没有在模型里体现，更不要说精细的算法管理了。同样，新近的基于 EPM 的 DEX 如 DODO、Bancor2.0 等，依然沿用 Uniswap 类的粗糙的成交量管理模型，这一切都是因为当前 DeFi 只是简单的剔除了信用风险，而没有做到对波动率风险进行有效计算。

NEST 预言机给了我们开发全新 DeFi 的可能：基于 NEST-price 的可计算性，对均衡价格的风险进行量化，再结合下游 DeFi 的业务特征进行设计，可以做出风险可算法化管理的 DeFi2.0（注意那种粗糙的交易量管理和对风险进行有效定价的算法管理不是一个概念），我们将此类 DeFi2.0 称之为 **CoFi**，也即计算金融 (Computational/Computable Finance)。

当风险可以被有效计算，基于算法可以对风险进行定价和管理，这样的金融系统将实现真正的无托管和自动化运行，这是几代金融从业者的理想。计算金融 CoFi 是区块链领域真正可落地的闭环应用，它将推动金融行业进入全新的时代。

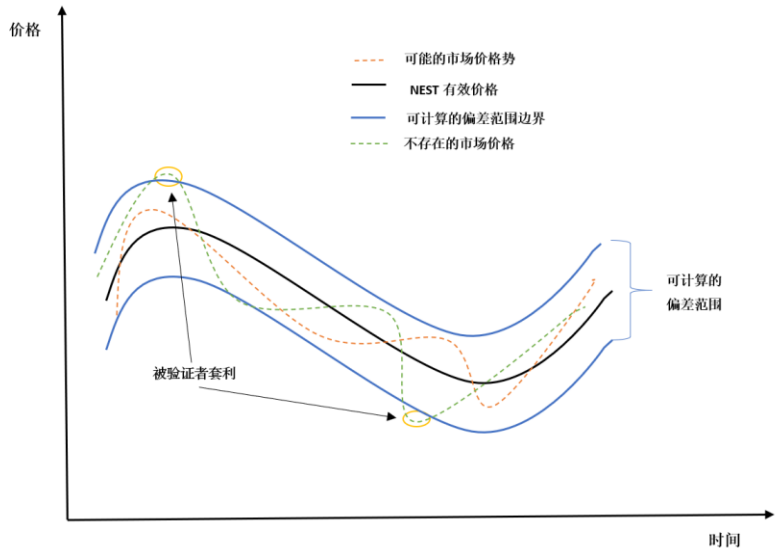
二、CoFiX：一种可计算的 DEX

CoFiX 是 **CoFi** 领域的第一个创新型产品：基于 NEST 预言机设计的 DEX，主要包含预言机模块、做市模块和交易模块。**CoFiX** 的思路是：做市商和交易者引用 NEST 预言机的价格进行交易，双方对 NEST 价格风险进行定价，确保做市商有动力做市，交易者交易买卖价差极小且不存在滑点问题。之所以称之为 **CoFiX** 是因为上述一切参数都是可计算的。

需要注意的是，目前 **CoFiX** 只支持有 NEST 预言机报价的交易对，没有开通 NEST 预言机的资产如想要在 **CoFiX** 交易，则需要先开通 NEST 预言机并报价。

1. NEST 价格性能及其计算

NEST 是基于金融套利设计的去中心化价格预言机，通过双向期权、价格链和 beta 系数等创新实现了价格的有效验证，使得 NEST 预言机的价格和即时的市场均衡价格的偏差在一个可计算的合理范围之内（详见参考文献 1）。



1.1 NEST 价格基本参数

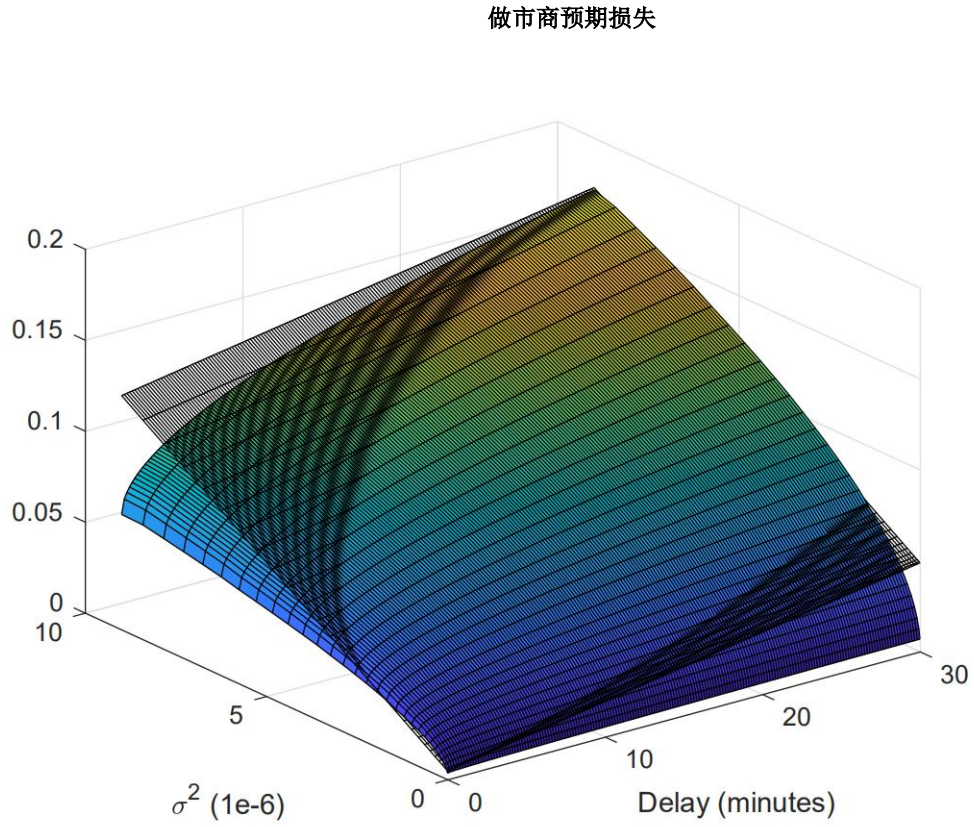
参数	具体意义
P	NEST 预言机提供的价格，每一区块对应一个价格，如果某个区块未更新，则沿用上一个区块的价格
P_0	交易所的均衡价格
g	价格的偏差，即 $\|P - P_0\|/P$
σ	链上波动率
σ_0	交易所波动率
T_0	验证周期
T	引用价格的延时，即使用价格在第 N 个区块，但最近的更新价格在第 N_0 个区块，则 $T=N-N_0$
q	被套利概率，价格偏差较大时，正常报价也会被套利者套利成交，该概率描述了 NEST 预言机的基本特征

根据 NEST 预言机的研究（参考文献 2），正常情况下 NEST 理论价差在 0.3%左右，实际数据也与此极为接近，约为 0.4%。参考文献 2 中详细论述了 NEST 的两大价格风险：价差风

险、延时风险的量化特征（被套利概率本质是延时风险的一种），给出了在不同波动率和不同激励程度下的风险函数。

1.2 价格补偿系数

由于 NEST 价格和均衡价格存在一定的偏差和延时（可以理解成去中心化的成本），因此 **CoFiX** 在应用该价格时需要对风险给出一些补偿，确保做市商有动力持续做市。补偿系数记为 **K**，该系数与波动率 δ 和延时变量 T 有关，当交易者进行交易时，并不是直接使用价格 **P**，而是使用$\mathbf{P' = P * (1 + k)}$或者$\mathbf{P' = P * (1 - k)}$，同样做市商在进入和退出做市时，计算净值用的价格变量也是$\mathbf{P'}$而不是 **P**，这一价格称之为交易价格。



2. 做市商机制

做市商依然是基于 AMM 的，即完全开放且去中心化，但 **CoFiX** 的做市商不需要像 Uniswap 一样打入双边资产到资产池中，而是打入任一单边资产即可。这里需要引入资产池净值的概念，即做市商将资产池当成一个基金，每次进出按照基金净值及份额进出，这是对 Uniswap 的一种改进。**CoFiX** 是不需要双边资产形成价格，NEST 均衡价格足以让资产池轻松结算。

2.1 资产池和交易对

每一个交易对对应一个资产池，在这里主交易资产是 ETH，即每个资产与 ETH 形成一个交易对，没有开通 NEST 预言机的资产，则不能在 CoFiX 里交易。比如 ETH/USDT 是一个交易对，也是一个资产池。同样，ETH/HBTC 也是一个交易对和资产池，没有开通预言机则需要先开通 NEST 预言机。

2.2 做市商份额 XToken

以 ETH/USDT 资产池为例，做市商将自己持有的的 USDT 或 ETH 打入资产池，将按照净值计算出一个份额，该份额用 XToken 来代表，即份额为 10，则得到 10 个 XToken，XToken 和资产池对应并有一个编号，比如 ETH/USDT 为 S001。

2.3 做市商净值计算

以 ETH/USDT 为例，初始净值为 1。做市商每次进出将更新净值，净值计算以 ETH 为本位，计算方法为将 USDT 按照当时的$\mathbf{P}'$换算成 ETH，然后除以总份额即可，具体计算公式（参考文献 4）。

2.4 按份额完美对冲

做市商需要在交易所进行对冲，从而获得价差收益。因为按照份额进行计算，因此可以进行完美的对冲。方法如下：交易者或做市商有操作，系统状态就会发生改变，程序每次轮询，记录和比较两次系统的状态，包括：ETH 总量 $\mathbf{A_e}$，ERC20 总量 $\mathbf{A_u}$, 做市商自己的份额$\boldsymbol{S_i}$，池子的总份额 $\boldsymbol{S}$。根据份额，做市商按比例拥有的 ETH 数量：

$$\boldsymbol{e = A_e * S_i / S}$$

根据份额，做市商按比例拥有的 ERC20 数量：

$$\boldsymbol{u = A_u * S_i / S}$$

通过上述公式可以算出本次状态做市商按比例拥有的 ETH 数量为e_1，按比例拥有的 ERC20 数量为 u1；上次状态做市商按比例拥有的 ETH 数量为e_0，按比例拥有的 ERC20 数量为u_0。则两次状态 ETH 的差额为：$d_{e=}e_1 - e_0$，ERC20 差额：$d_{u=}u_1 - u_0$， 分别设阀值 d_0, d_1, 当 abs(d_e)> d_0 或 abs(d_u)> d_1 开始对冲，对冲的方法理论上有多种，实践中最简单的就是通过卖出增量大于 0 的资产，补齐小于 0 的资产。这一部分对冲操作可以开发脚本，使做市商以较低成本在交易所完成对冲。

2.5 做市商的风险

CoFiX 基于基本的金融学原理对主要风险进行了量化和管理，但依然存在做市商自身操作不当或者在模型范围外的风险：

1）由于 K 系数是基于统计分析设计的，因此如果整个系统交易不够频繁，则做市商可能会出现亏损；或者做市商在资产池中做市的时间太短，也可能出现亏损，当然这种亏损是概率性的。

2）如果做市商不在交易所进行对冲（Uniswap 同样需要对冲），则可能承担某一资产价格下跌的风险。

3）波动率都是基于历史数据进行估计，和真实波动率可能有偏差。

4）用于某些流动性较差的资产时，如果频繁触发停机，则可能会影响做市商收益。

3. 交易者机制

交易者按照自己可接受的价格在 **CoFiX** 里完成交易。根据前面的分析，即使在提供了价格补偿的情况下，交易价格$\mathbf{P}'$与均衡价格偏差也极小，基本上代表了市场的公允价格，而且不存在成交量滑点，这是此前 DEX 所不具备的。

3.1 交易机制

以 ETH/USDT 为例，只要资产池有资产，交易者依据 NEST 预言机提供的交易价格$\mathbf{P}'$进行交易：买入 ETH（卖出 USDT）或者买入 USDT（卖出 ETH），在一般情况下将得到极好的交易体验。此过程不需要担心 KYC、交易所安全、滑点及非均衡价格问题。在交易过程中，调用价格需要向 NEST 预言机付费，其流程和支付 GAS 一样简单。

值得提醒的是，交易者并不是按照发起交易时的价格进行交易，而是按照打包时刻引用的最新价格进行成交，因此 GAS 费用的高低会影响交易者交易的确定性。一般来说，可以设计价差和时间的约束，对交易者进行保护。

3.2 停机机制

根据文献 3，当波动率上升到一个极端情况，或者 NEST 系统被攻击时，需要 **CoFiX** 启动应急方案，主要是为了保护交易双方，特别是做市商。此时系统可以设置停机机制，即触发一类条件时暂停交易。目前设置的停机条件有三类：

1）K_0 值超过一个范围，比如 5%

2）秒级波动率 σ 上升到一个上限，比如 0.1%

3）延时 T 超过一个范围，比如 200 区块

具体的停机条件详见文献 4

3.3 交易风险

交易者存在一些模型之外的风险，比如 ETH 打包延时的风险，导致交易者得到的价格与交易者预期的价格有偏差，这取决于交易的手续费。这是所有 AMM 交易机制的固有风险。

4. 冲击成本

当 **CoFiX** 资产池规模足够大的时候，单笔交易或者单位时间累积交易很难达到资产的上限，但单笔的大交易量可能影响做市商的对冲成本。比如在 **CoFiX** 资产池交易 100 万 ETH 而价格不变，做市商在交易所对冲的话，必然拉高 ETH 成本，这造成亏损。因此单位时间内数额较大的交易应当支付一定的价差，此价差称之为冲击成本 C（给定交易量对价格的影响）。因此在考虑了冲击成本的情况下，价差补偿系数 $\mathbf{K_1=K+C}$，当冲击成本触发，则系统内价格 $\mathbf{P' = P * (1 + K_1)}$。由于冲击成本是考虑全市场的交易总规律，因此在一般的交易行为下，不太会发生，只有巨额交易才需要，其规模取决于交易所的交易深度。详细的冲击成本见文本 3 和 4。

三、系统优势及展望

CoFiX 可以说是目前第一个在链上实现了逻辑闭环的 DEX：不仅仅去中心化，还对交易过程中的风险进行了量化和算法管理，无论在交易规模约束、价差管理和做市商预期收益计算等方面，都全面领先于当前的 DEX。考虑到 **CoFiX** 更适合主流资产的交易，而主流资产占到交易规模的 80%以上，因此可以预期 **CoFiX** 和 Uniswap 这样的上一代 DEX 竞争中，将拥有更高数量级的市场规模。而 **CoFiX** 的拓展性、可改良性以及开放性更是优于当前 DEX，它将代表未来 DEX 的设计潮流和方向。

参考文献

[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/

附录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 $\times$ 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

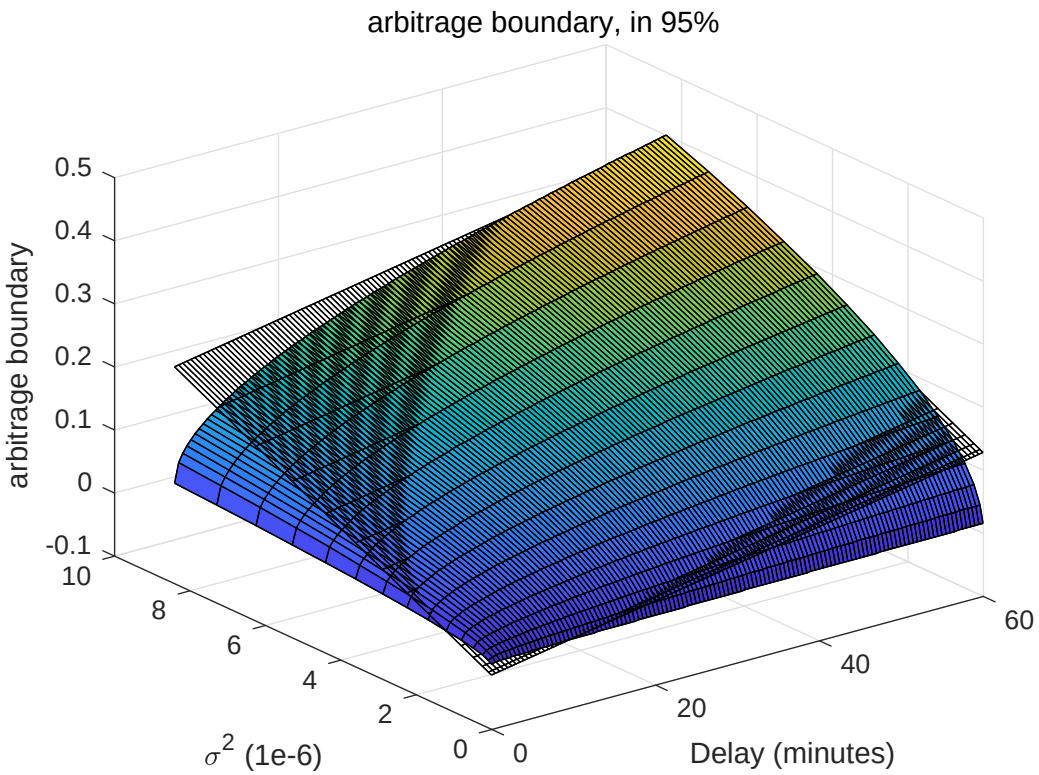


Figure 1. This figure depicts price inaccuracy when using the effective price as a reference for a trade. The gray flat plane is a linear approximation to the surface.

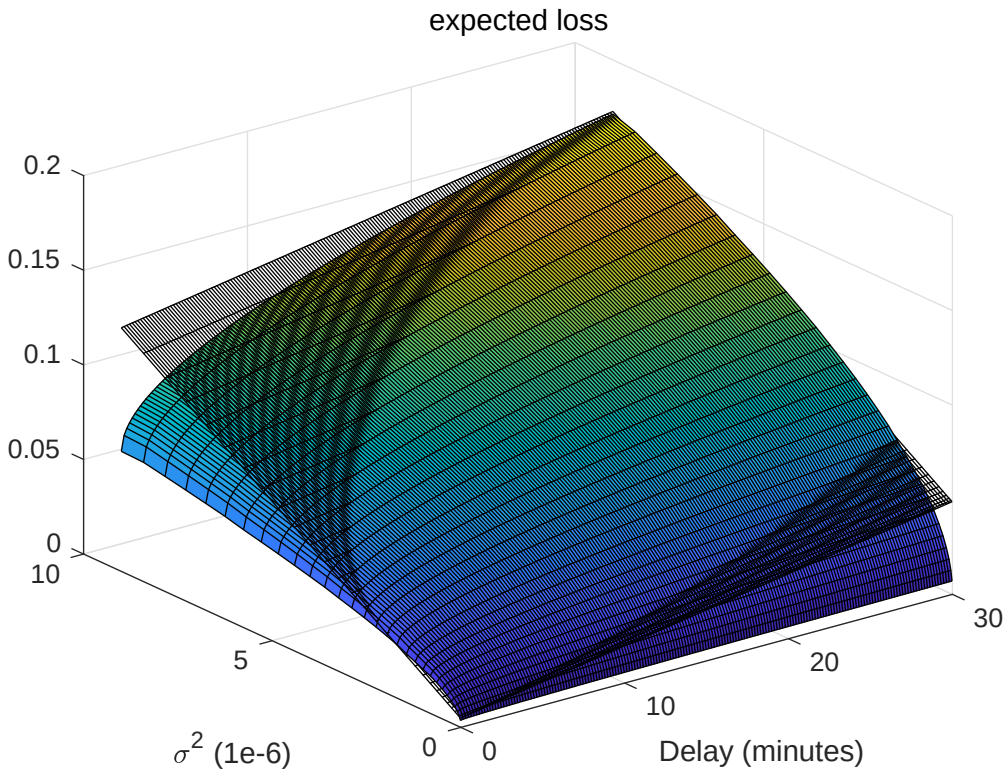


Figure 2. This figure depicts expected loss when using the effective price as a reference for a trade. The gray flat plane is a linear approximation to the surface.

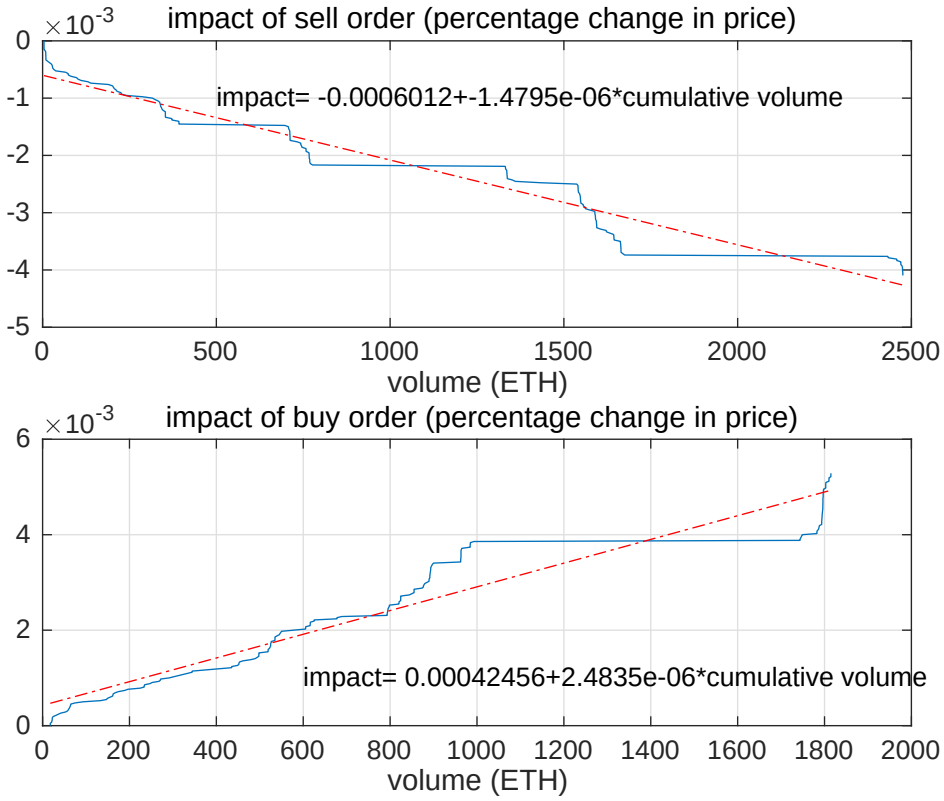


Figure 3. This figure depicts possible trading cost magnitude.

附录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 $\tau \leq \mathrm{T_0}$ is a stopping time and $\mathrm{T_0}$ 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)

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

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 $E^Q[1_{\eta \le T_0}]$. 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

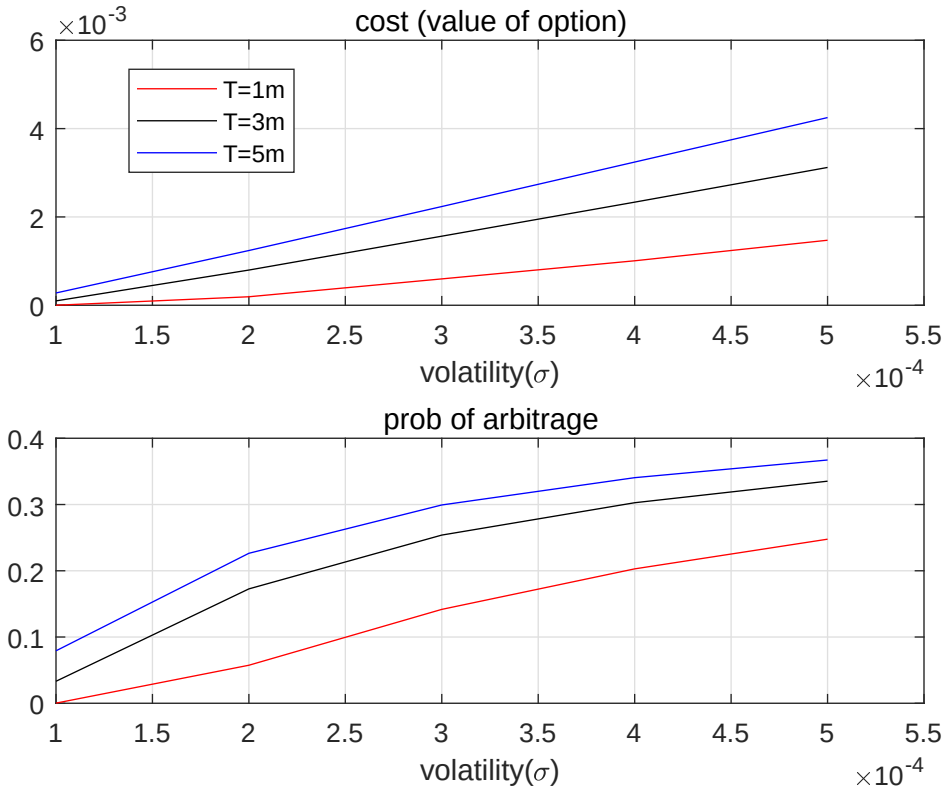


Figure 1. This figure depicts effects of volatility σ on cost of price-inputing and probability of arbitrage.

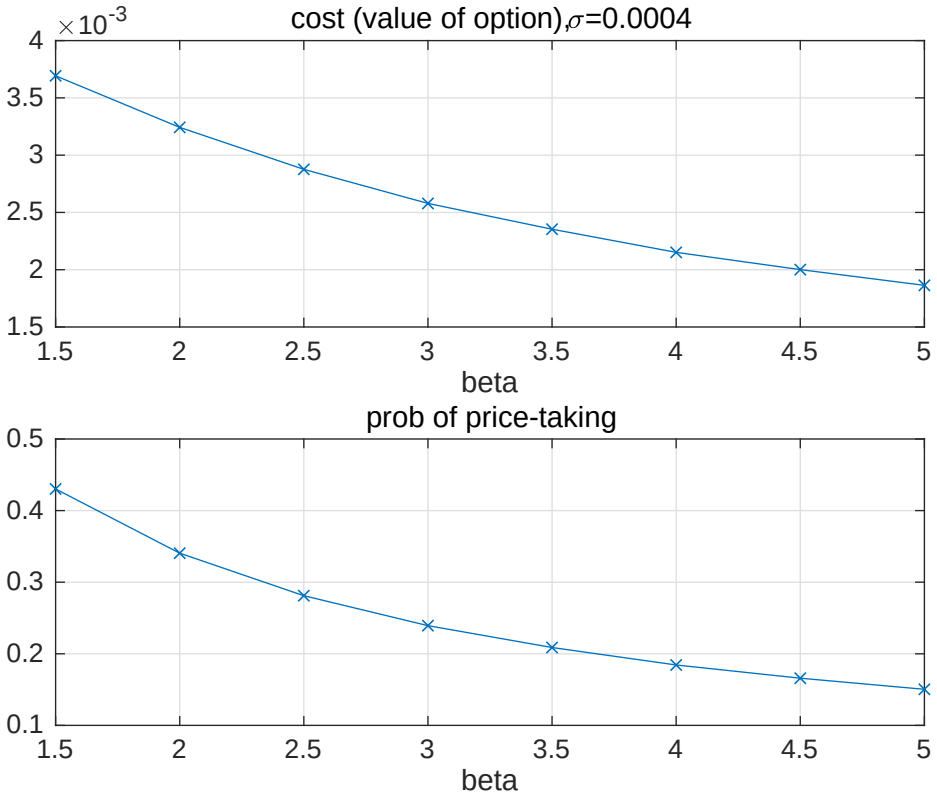


Figure 2. This figure depicts the effect of β on cost of price-inputing and probability of arbitrage.

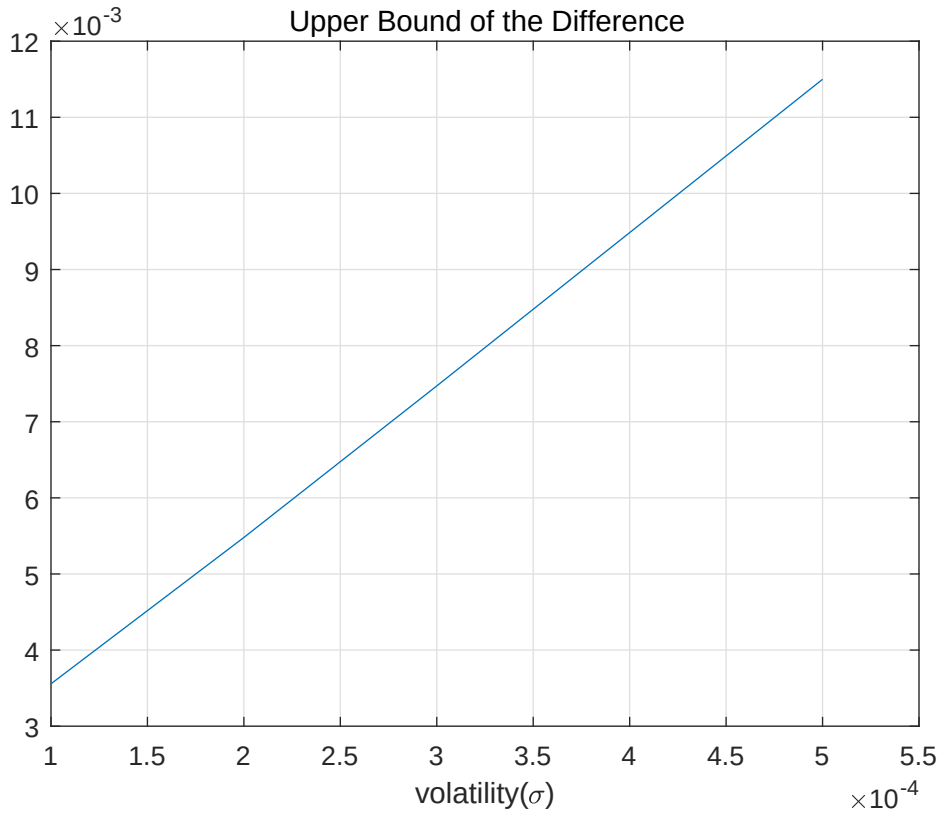


Figure 3. This figure shows the upper bound of difference between the NEST price and the price of an exchange at the same time.