Manipulation of Exponential Moving Average (EMA) Oracles

Michael Bentley

August 9, 2024

1 Introduction

Let the spot price on block n be denoted p_n. An exponential moving average (EMA) of the price at block n is given by

$$a_n = \lambda p_n + (1-\lambda)a_{n-1}, \tag{1}$$

where λ is a weighting parameter determining how much the price on the most recent block, p_{n-1}, influences the EMA versus the prices on previous blocks. When $\lambda = 1$, the EMA is equal to the current price, and when λ is close to zero, the EMA is a slow moving average over many blocks. It is useful to expand the EMA for several terms to see how the weighting of prior values evolves back through time. We have

$$\begin{aligned} a_n &= \lambda p_n + (1-\lambda)(\lambda p_{n-1} + (1-\lambda)a_{n-2}) \qquad (2)\\ &= \lambda p_n + \lambda(1-\lambda)p_{n-1} + (1-\lambda)^2 a_{n-2} \\ &= \lambda p_n + \lambda(1-\lambda)p_{n-1} + (1-\lambda)^2(\lambda p_{n-2} + (1-\lambda)a_{n-3}) \\ &= \lambda p_n + \lambda(1-\lambda)p_{n-1} + \lambda(1-\lambda)^2 p_{n-2} + (1-\lambda)^3 a_{n-3} \\ &= \lambda p_n + \lambda(1-\lambda)p_{n-1} + \lambda(1-\lambda)^2 p_{n-2} + (1-\lambda)^3(\lambda p_{n-3} + (1-\lambda)a_{n-4}) \\ &= \lambda p_n + \lambda(1-\lambda)p_{n-1} + \lambda(1-\lambda)^2 p_{n-2} + \lambda(1-\lambda)^3 p_{n-3} + (1-\lambda)^4 a_{n-4}. \end{aligned}$$

Now let us consider a scenario in which an attacker feels able to manipulate the spot price lower over the most recent m blocks as an attempt to lower the EMA. An expansion of the EMA formula to expose these m terms gives

$$a_n = \lambda p_n + \lambda(1-\lambda)p_{n-1} + \lambda(1-\lambda)^2 p_{n-2} + \cdots + \lambda(1-\lambda)^{m-1} p_{n-(m-1)} + (1-\lambda)^m a_{n-m}, \tag{3}$$

where we assume the first m terms are the potential targets for manipulation, and the last term is the unmanipulated older EMA value. Suppose for a moment that the spot price of the asset prior to any attempt at manipulation is constant, so that $p_n = p$ for all n. In this case, we can assume that $a_{n-m} \approx p$. We have

$$\begin{aligned} a_n &= \lambda p + \lambda(1-\lambda)p + \lambda(1-\lambda)^2 p + \cdots + \lambda(1-\lambda)^{m-1}p + (1-\lambda)^m p \\ &= p \cdot \left[\lambda\left(1 + (1-\lambda) + (1-\lambda)^2 + \cdots + (1-\lambda)^{m-1}\right) + (1-\lambda)^m\right] \\ &= p \cdot \left[\lambda S + (1-\lambda)^m\right], \end{aligned} \tag{4}$$

where S is the finite sum of a geometric series:

$$S = 1 + (1-\lambda) + (1-\lambda)^2 + \cdots + (1-\lambda)^{m-1} = \frac{1-(1-\lambda)^m}{1-(1-\lambda)} = \frac{1-(1-\lambda)^m}{\lambda}. \tag{5}$$

If we substitute back in to equation (4) for S, we have

$$a_n = p \cdot \left[\lambda \cdot \frac{1-(1-\lambda)^m}{\lambda} + (1-\lambda)^m\right] = p. \tag{6}$$

Thus, in the absence of manipulation, the EMA is equal to the constant price, p, as we would expect. Now let us assume that the attacker manipulates the spot price on each of the m blocks by a factor $\mu \in \mathbb{R}_{\geq 0}$. If $\mu \in [0,1)$ then the spot price is lowered. If $\mu > 1$ then the spot price is raised. Making this assumption, we re-write equation (4) as

$$\begin{aligned} a_n &= \lambda p\mu + \lambda(1-\lambda)p\mu + \lambda(1-\lambda)^2 p\mu + \cdots + \lambda(1-\lambda)^{m-1}p\mu + (1-\lambda)^m p \\ &= p\mu \cdot \left[\lambda\left(1 + (1-\lambda) + (1-\lambda)^2 + \cdots + (1-\lambda)^{m-1}\right) + \frac{(1-\lambda)^m}{\mu}\right] \\ &= p\mu \cdot \left[\lambda S + \frac{(1-\lambda)^m}{\mu}\right] \\ &= p \cdot \left[\mu + (1-\mu)(1-\lambda)^m\right], \end{aligned} \tag{7}$$

where we have again substituted for S from equation (5), and then simplified. The expression in square brackets measures the overall degree of manipulation of the EMA. Let us denote this by M. We have

$$M = \mu + (1-\mu)(1-\lambda)^m. \tag{8}$$

Let's explore a couple of brief scenarios. If we assume the attacker can attack over $m = 10$ blocks, achieving a manipulation of 99% of the spot price each block, so that $\mu = 1 - 0.99 = 0.01$, and if we have $\lambda = 0.01$, then we find $M \approx 0.9053$. That is, the manipulation over 10 blocks will reduce the EMA by around 10%. Thus, for small EMA parameter λ the EMA is quite hard to manipulate. However, if λ is increased, this manipulation resistance drops dramatically (due to the exponential nature of the calculation). For example, if we redo the last example, but with $\lambda = 0.1$, then we find $M \approx 0.3552$. In other words, manipulation over 10 blocks will reduce the EMA by around 65%.

We can also rearrange equation (8) to calculate how many blocks an attacker must reduce the spot price by to achieve a particular target level of EMA manipulation. Rearranging to solve for m, we have

$$\begin{aligned} \frac{M-\mu}{1-\mu} &= (1-\lambda)^m \\ \implies \ln\left(\frac{M-\mu}{1-\mu}\right) &= \ln(1-\lambda)^m \\ \implies m &= \frac{\ln(M-\mu)-\ln(1-\mu)}{\ln(1-\lambda)}. \end{aligned} \tag{9}$$

Suppose they want to lower the EMA by 50%, so that $M = 0.5$, again using $\lambda = 0.01$ and $\mu = 0.01$. We find that the number of blocks is $m \approx 70$. Again, re-doing this example with higher EMA parameter $\lambda = 0.1$ gives very different results. We instead find the number of blocks to attack is $m \approx 7$.

2 Curve Example

The last section showed that it is relatively difficult to manipulate an EMA oracle, providing that λ is small enough. Let us take a look at a live example. In the Curve protocol, an EMA oracle is used for several purposes, including pricing of collateral assets for crvUSD. The Curve EMA uses a parameter α in place of λ, with the relationship being $\alpha = 1 - \lambda$. Unlike in the examples above, Curve uses a dynamic calculation for α, to ensure that it grows with the time between recent transactions, Δt. This is designed to make sure α is properly weighted in the EMA across blocks in which no trades have happened (and the price has therefore remained constant). Specifically, α decays exponentially with increasing time between transactions:

$$\alpha = e^{-(\Delta t/\tau)}, \tag{10}$$

where τ is a scaling parameter determining how quickly α tends to zero as the time between transactions grows larger. See definition. According to their GitHub deploy script, the scaling parameter is generally set to $\tau = 10 \cdot 60 \cdot \ln 2 \approx 866$ seconds. This is supposed to reflect what Curve refers to as a '10 minute' EMA. Let us consider the security implications of these choices.

If we assume an attacker is able to manipulate the price on successive blocks, then we have $\Delta t = 12$, which is the block time on Ethereum. Thus, over a single block, we find that $\lambda = 1 - \alpha = 1 - e^{-(12/866)} \approx 0.013763$. Given our analysis above, this would appear to be a sufficiently small parameter to make the Curve EMA difficult to attack.

However, we need to be careful here, because as Δt grows larger, so does λ. For example, if there have been 20 blocks between the last trade and the current trade, then we have $\lambda \approx 0.24$. It is therefore crucial that the latest update to the EMA always uses the price from a previous block, but Δt calculated to the current block. And this is indeed what the Curve protocol does. This approach makes it difficult for an attacker to increase λ for their own advantage without exposing a large arbitrage opportunity.

For example, if the last trade was 20 blocks ago, we calculate λ using $\Delta t = 20 \cdot 12$, but use p_{n-20} when updating the EMA. If the attacker were to try to manipulate p_{n-20}, it would need to leave that spot price exposed for 20 blocks before it would be included in the EMA. Yet it would not be able to do so unless there were no arbitrage in the meantime, which is unlikely to be the case for liquid pairs and large spot price manipulations.

2.1 Further Mitigation

Whilst an EMA can generally be difficult to attack, depending on its parameters, it is possible to further frustrate attackers and make an EMA even more secure with relatively little impact on ordinary users of an AMM. Instead of assuming the attacker can lower the spot price arbitrarily between blocks, as we did above, let us now assume that the maximum that they can lower the spot price is a factor ρ of the spot price on the *prior* block, so that $p_n = \rho p_{n-1}$, and so on. This could be achieved in practice by setting a constraint inside the AMM itself or through a dynamic fee mechanism.

Note that when we do this, it makes it much harder for an attacker to manipulate the EMA, because to even manipulate the spot price now takes many blocks. How many blocks does it take? If the target spot price is q, we need to solve $\rho^N = q$ for N. We have

$$N = \frac{\ln q}{\ln \rho}. \tag{11}$$

Now consider an example where the maximum spot price move in a block is 20% away from the last spot price, so that $\rho = 0.8$. If we want to get to a spot price manipulation of 99%, as in prior examples, we find that the attacker needs to manipulate the spot price on at least $N = 21$ blocks. Thus, constraining the AMM also constrains the potential for EMA oracle attacks.

Let us see how this change impacts our calculations about EMA manipulation. Re-writing equation (4), the EMA is now calculated as

$$\begin{aligned} a_n &= \lambda p \rho^m + \lambda(1-\lambda)p\rho^{m-1} + \lambda(1-\lambda)^2 p\rho^{m-2} + \cdots + \lambda(1-\lambda)^{m-1}\rho p + (1-\lambda)^m p \\ &= \rho p \cdot \left[\lambda\left(\rho^{m-1} + (1-\lambda)\rho^{m-2} + (1-\lambda)^2\rho^{m-3} + \cdots + (1-\lambda)^{m-1}\right) + \frac{(1-\lambda)^m}{\rho}\right] \\ &= \rho p \cdot \left[\lambda S_\rho + \frac{(1-\lambda)^m}{\rho}\right], \end{aligned} \tag{12}$$

where S_ρ is the finite sum of a new geometric series constrained by ρ:

$$S_\rho = \rho^{m-1} + (1-\lambda)\rho^{m-2} + (1-\lambda)^2\rho^{m-3} + \cdots + (1-\lambda)^{m-1}. \tag{13}$$

Multiplying S_ρ through by $(1-\lambda)/\rho$, we have

$$S_\rho\left(\frac{1-\lambda}{\rho}\right) = (1-\lambda)\rho^{m-2} + (1-\lambda)^2\rho^{m-3} + (1-\lambda)^3\rho^{m-4} + \cdots + (1-\lambda)^m\rho^{-1}. \tag{14}$$

Subtracting equation (14) from equation (13), and simplifying, we have

$$S_\rho \left(1 - \left(\frac{1-\lambda}{\rho}\right)\right) = \rho^{m-1} - (1-\lambda)^m \rho^{-1} \tag{15}$$
$$\implies S_\rho = \frac{\rho^m - (1-\lambda)^m}{\rho - (1-\lambda)}.$$

Now substituting this back in to equation (16), we have

$$a_n = p \cdot \left[\lambda\rho \cdot \frac{\rho^m - (1-\lambda)^m}{\rho - (1-\lambda)} + (1-\lambda)^m\right]. \tag{16}$$

As before, the expression in square brackets measures the overall degree of manipulation of the EMA. Let us denote this by M_ρ to distinguish it from M. We have

$$M_\rho = \lambda\rho \cdot \left(\frac{\rho^m - (1-\lambda)^m}{\rho - (1-\lambda)}\right) + (1-\lambda)^m. \tag{17}$$

Let us compare how M_ρ performs compared to M under the scenarios we considered before. Let $\rho = 0.8$, so that the attacker can move the spot price at most 20% on a given block, and again suppose the attacker uses manipulation over $m = 10$ blocks. We again start by assuming $\lambda = 0.01$. We previously arrived at $M \approx 0.9053$, but we now have $M_\rho \approx 0.9379$. If we set $\lambda = 0.1$, we previously found that $M \approx 0.3552$, but we now have $M_\rho \approx 0.5417$. Thus, adding a constraint to the pace of spot price manipulation can significantly increase the difficulty of EMA manipulation.