## Joe V3

# Token Mill

MF
mountainfarmer@lfj.gg

Louis
louis@lfj.gg

Fish
fish@lfj.gg

**Abstract:**

Token Mill offers a secure, flexible solution for permissionless token creation and exchange. It provides token templates, token locking and customizable bonding curves, enabling creators to define tokenomics and price sensitivity to token demand. By shifting liquidity provision to a bonding curve model, the protocol eliminates the need for external liquidity providers, reducing risks like honeypots and liquidity issues.

October 2024

# 1 Product Introduction

### *1.1 Background*

*Permissionless Digital Asset Creation & Peer to Peer Trading.*

Permissionless asset creation and P2P trading are possibly the two most defining core concepts of DeFi as we know it. Those ideas alone have contributed to more innovation and adoption to blockchains than any other application. However, a short conversation with anyone that has participated in a token generation event, fair launch, or someone that 'aped' a random token and you'll quickly learn that participating in this space is more of a minefield than not. The obvious fact is that opening access for more people to create and distribute, which is highly desirable, also creates vectors for misrepresentation, deception, manipulation, and outright theft.

As AMMs have expanded to multiple networks, it is clear that there is disproportionate demand for native assets on each network. While on-chain exchanges are moving toward markets with lower spreads and higher frequency participation, it's important that we don't leave behind this critical market segment where participation looks very different. Years of AMM proliferation and launchpad offerings, and this particular market segment still feels underserved. We believe this asset class deserves tailor-made solutions and attempt a fresh approach with the Token Mill protocol.

## *1.2 Problem Definition*

### *1.2.1 Tokens*

A common issue plaguing participation in newly created token markets actually involves the token itself. More likely than not participants will be subject to honeypots, unrestricted mints, excessive transfer taxes etc. Participants must extend diligence to review contracts or trust the team has implemented their contracts as represented and ensured proper security.

Exchanges can provide a third party solution by offering templates with higher levels of security that have been audited and tested over time. This way protocols can focus on their main product, and certain risks can be removed for participants.

1) Token contract security
2) Token vesting and locks implemented

### *1.2.2 Market Liquidity*

AMMs popularized permissionless asset trading and are still the dominant market type for new and lower cap tokens. For every trade, there is a passive participant that will take the other side of the trade. All liquidity providers subscribe to a common bonding curve/pricing mechanic if they choose to supply liquidity to the pool. Assets often require this pool to offer a two-sided trade and market depth is proportional to the amount of liquidity pooled.

However, most participants aren't actually indifferent on pricing. They generally would rather express a preference for one asset over the other. This leads to new markets being made entirely by the protocol or requires substantial incentives for participants to supply liquidity for two-sided trading. Additionally, token holders take on more risk than those supplying liquidity for trading to sustain over time.

3) Seed liquidity
4) Sustained market liquidity

### *1.3 Token Mill Protocol*

Token Mill is a factory that generates a collection of linked contracts that provide an end-to-end solution for token creators and market participants. Instead of developing a universal solution for all low cap tokens or secondary market launches, Token Mill attempts to serve new token creators specifically. Addressing this market in a focused manner enables LFJ to provide more optionality to creators and better assurances to market participants.

Firstly, the Token Mill facilitates token creation and locking. The token creator is able to create any templated token and define their own cap table and release schedule. Market participants can be assured that the contracts are secure and the token release schedule is enforced.

Secondly, markets are established for the newly created tokens following a token bonding curve. The token creator defines a price-supply bonding curve that will issue and absorb tokens. Since tokens can be deposited into separate markets such as on Liquidity Book, this curve serves as a last resort market maker that expands and contracts circulating supply as a function of demand.

This approach completely reframes liquidity markets for new tokens and removes the need for external liquidity providers. Given this asset class inherently has substantial price discovery and speculation, a healthier market can be engineered by enabling more participants to deliberately express their opinions on price without convoluting it behind curves, fees, and farms. The bonding curve serves the necessary role of providing path-independent pricing across the entire price range and is entirely self-funded.

## 2 Protocol Design

### *2.1 Token Creation*

Token Mill token launches require several steps. Those steps include:

1) Create token from template
2) Create custom bonding curve
3) Transfer ownership of token
4) Send tokens to bonding curve
5) Purchase initial tokens for creator and send to locker
6) Enable trading

#### 2.1.1 Token Factory

The Token Mill factory offers templates for token creators to use for their token launches. These templates are added by the Token Mill owner, and any user can instantly create a token and market using these templates.

Factory Components:

- **Tokens**
- **Token Lockers**
- **Bonding Curves**

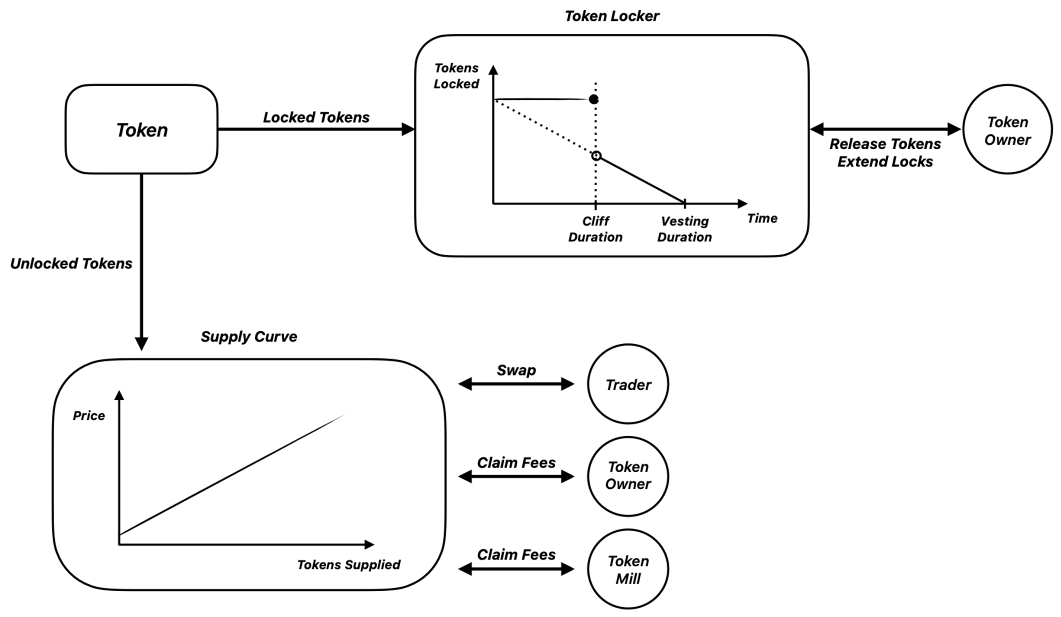

**Figure 1: Token Mill Components**

### *2.1.2 Token Templates and Locker*

Tokens will be created from token templates. Each token template will have different methods and parameters available to the user to customize their token. Token templates with varying features and parameters may be added over time.

The standard token template will be a basic ERC-20 with the following properties:

1) Immutable max supply
2) No mint function / fully minted at creation
3) 18 decimals
4) Basic user inputs
    a) Token owner address
    b) Token symbol and name
    c) Token supply

The token locker is a contract that facilitates token distribution, locking, and vesting. Anyone can send their tokens to the locker and specify a cliff, vesting period and beneficiary, typically the team. Tokens in the locker can only be released by the beneficiary once they are eligible for release.

Locker beneficiary permissions:

- Transfer beneficiary
- Claim tokens directly

### *2.1.3 Team Allocations*

Tokens created through Token Mill cannot be pre-minted before trading begins. This is because the bonding curve can only absorb tokens that have been purchased from it. Therefore, to allocate team tokens, the token creator must buy a portion of the initial supply. While this may seem cumbersome, the cost can be significantly reduced by using a curve with very low initial price points. Additionally, this small cost serves as a deterrent against spam launches.

To avoid front-running of the initial purchase, Token Mill batches this transaction with the token creation process. Once purchased, the tokens are sent to the token locker contract, where they are released according to the vesting schedule set by the creator.

## 2.2 Token Bonding Curve

Tokens supplied for distribution will be priced following two selected price-supply bonding curves: bid and ask. In contrast to typical liquidity pools, pricing in a bonding curve is not a function of the level of reserves in the pool. Instead, bonding curves follow a path-independent pricing function, where the price increases with each additional token supplied to the market. Similarly, tokens are absorbed from the market following the same pricing function.

Before diving deeper, first we define some common terminology:

| | | | |
|---|---|---|---|
| $x$: | $Tokens\ issued\ into\ circulation\ from\ curve$ | $n$: | $Number\ of\ price\ points$ |
| $s$: | $Token\ max\ supply,\ x_{max}$ | $r$: | $Tokens\ available\ for\ release\ from\ locker$ |
| $P_{min}$: | $Smallest\ price$ | $FDV$: | $P(x)\ *\ s$ |
| $P_{max}$: | $Largest\ price$ | $MarketCap$: | $P(x)\ *\ (x\ +\ r)$ |

### 2.2.1 Bonding Curve Creation

The key advantage of Token Mill over its competitors is its flexibility, allowing any pricing function to be used as long as each successive point on the curve is strictly increasing. The Token Mill contract only requires an input array of price points of size $n$ that lie on the curve, leaving the responsibility of defining the curve with the token creator.

Token Mill constructs the bonding curve by connecting each price point with straight lines, creating an approximation of the desired pricing curve. As shown in figure 2, providing more price points improves the accuracy of the approximation, but this comes with increased gas consumption. Fortunately, using $n = 10$ is generally sufficient to approximate the pricing function accurately (Desmos).

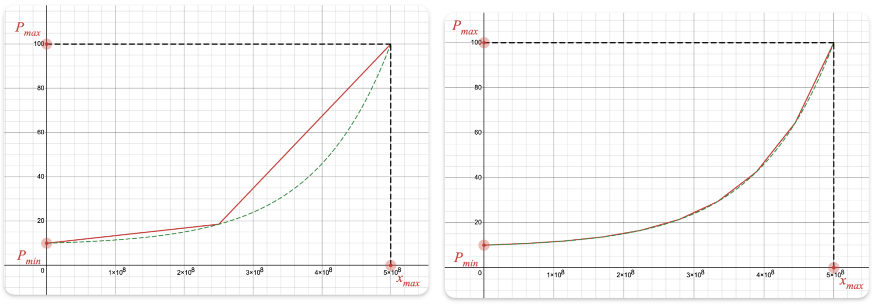

**Figure 2: The effect of approximating (red) a pricing curve (green) when using n=3 vs n=10 price points**

### 2.2.2 Understanding Bonding Curves

The bonding curve is designed to absorb all tokens in circulation and act as the "market maker of last resort". The amount of capital (or $Y$ tokens) that the pool can absorb (capacity) is defined as:

(2.1)
$$C = y(x) = \int_0^x P(x)dx \qquad Capacity,\ Value\ Sink\ from\ x_{0 \rightarrow s}$$

Capacity can also be defined as the amount of $Y$ tokens a creator aims to raise for issuing a total of $s$ amount of $X$ tokens. Creators establish the price-capital relationship for the supply curve through their selection of parameters $x_{max}$, $P_{min}$, $P_{max}$ and array of price points . This relationship determines how sensitive price is to token demand and how many tokens are released for a given change in price. This is illustrated through the figures below:

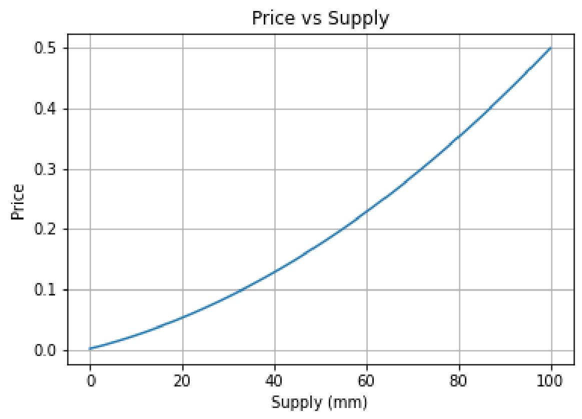

**Figure 3: Supply Curve, Price vs Supply**

*Price vs Supply*

- Pricing function is quadratic
- Price is increasing at an increasing rate from $x_{0\rightarrow s}$
- Price growth of 500x with these parameters

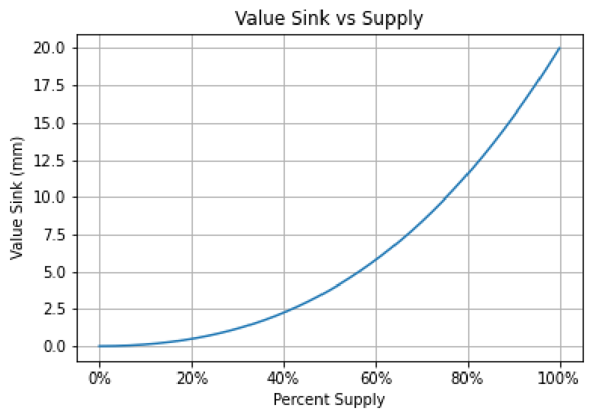

**Figure 4: Supply Curve, Sink vs Supply**

*Sink vs Supply*

- Value sink is the cumulative capital (or $Y$ tokens) absorbed into the pool for a given supply level. Capacity is max value sink.
- Value sink is the integral of the price function, therefore a cubic. The amount of capital required to issue additional shares increases at an increasing rate.
- The first 20% of capital will result in pool issuing over 50% of tokens with these parameters

### *2.2.3 General Form Function Templates*

To help creators design bonding curves, Token Mill offers several general function templates that can be customized with specific parameters. By passing in $n$ values of $x$ (2.2), the creator can generate an array of price points, which can then be used with the Token Mill contract.

(2.2) $$x \in [0, \frac{x_{max}}{n}, \frac{2x_{max}}{n}, ..., \frac{x_{max}}{n}]$$

**Linear**

(2.3) $$P(x) = \frac{x}{x_{max}}(P_{max} - P_{min}) + P_{min}$$ Desmos

**Quadratic**

(2.4) $$P(x) = ax^2 + bx + P_{min}$$ Desmos

Creators have to define capacity $C$ such that:

(2.5) $$x_{max} \cdot \frac{P_{max} + 2P_{min}}{3} \leq C \leq x_{max} \cdot \frac{P_{max} + P_{min}}{2}$$

From $C$, the remaining unknown variables $a$ and $b$ can be deduced:

(2.6) $$a = \frac{3}{x_{max}^3}(x_{max}(P_{max} + P_{min}) - 2C)$$

(2.7) $$b = \frac{2}{x_{max}^2}(3C - x_{max}(P_{max} + 2P_{min}))$$

**Exponential**

(2.8) $$P(x) = \frac{e^x - 1}{e^{x_{max}} - 1}(P_{max} - P_{min}) + P_{min}$$ Desmos

**Logarithmic**

(2.9) $$P(x) = \frac{ln(1+x)}{ln(1+x_{max})}(P_{max} - P_{min}) + P_{min}$$ Desmos

**Power**

Creators have to define an additional non-zero parameter $\alpha$:

(2.10) $$\alpha \in \mathbb{R}^*$$

(2.11) $$P(x) = \frac{e^{\alpha \frac{x}{x_{max}}} - 1}{e^{\alpha} - 1}(P_{max} - P_{min}) + P_{min}$$ Desmos

### *2.2.4 Bid & Ask Curves*

Trading takes place on separate bid and ask curves, where tokens are bought according to the ask curve and sold according to the bid curve. Both curves require an array of prices of size $n$. Token creators have the flexibility to design these curves according to their preferences, as long as the condition $P_{bid}(x) \leq P_{ask}(x)$ for all $x \in [0, s]$ is upheld. A fee is charged when users buy tokens, and is defined as the price difference between ask price and bid price $P_{ask}(x) - P_{bid}(x)$. Users pay ask price $P_{ask}(x)$ while amount equal to the bid price $P_{bid}(x)$ is sent to the bonding curve pool.

One approach to constructing these curves is by using a general form function template to generate an array of ask prices (2.1) and then scaling down each ask price by a factor $(1 - f)$, where $0 < f < 1$, to create the corresponding array of bid prices. In this case, the fee $f$ is fixed and defines the spread between the two curves. However, the fee doesn't have to be fixed. If desired, different fees can be applied at different price points, allowing the creator to customize the curves as needed.

(2.12) $$P_{ask}(x) = [P(x_1), P(x_2), ..., P(x_n)]$$

(2.13) $$P_{bid}(x) = (1 - f)[P(x_1), P(x_2), ..., P(x_n)] \qquad 0 < f < 1$$

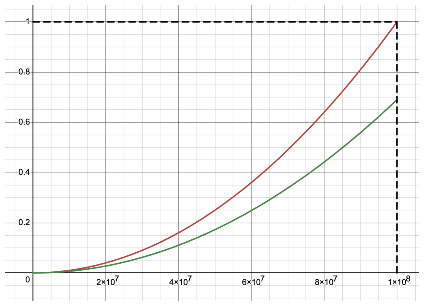

**Figure 5: Bid & ask curves with fixed fee**

## *2.3 Optional Integrations*

Creators can choose to add the following features to help grow their community, with fees allocated to these integrations if implemented.

### *2.3.1 Staking*

Token holders can opt to stake their tokens for a share of fees as allocated by the creator. Fees are streamed continuously to the staking contract and stakers can claim fees in real time. There is no lock up or vesting of staked tokens. Additionally, locked tokens are also staked.

### *2.3.2 Referral Program*

Referral programs are an effective way to incentivize holders to promote the token. Any token holder can be a referrer. Referrers can refer their friends and earn a share of fees from their trades ad infinitum. Referrals apply to all markets and referrers get half of the platform fee (10%).

## *2.4 Fees*

### *2.4.1 Purchase Fees*

Fees are only levied when tokens are purchased and is defined as the spread or price difference between ask price and bid price $P_{ask}(x) - P_{bid}(x)$. Users pay ask price $P_{ask}(x)$ while amount equal to the bid price $P_{bid}(x)$ is sent to the pool to ensure there is sufficient capital to absorb a token sale back to the curve.

### *2.4.1 Customizable Fees*

Token Mill allows you to customize the allocation of fees among various beneficiaries. There are four beneficiaries that can receive a portion of the purchase fee:

- Protocol (Token Mill)
- Creator
- Staking
- Referrer

Token Mill charges a fixed platform fee of 20%. If there is a referrer, they receive half of this fee (10%).  The remaining 80% is allocated according to the creator's preference, which can be split between the creator and staking.

## Appendix

### *Bonding Curve Math*

Token Mill approximates the pricing curve it aims to replicate by connecting each pricing point with straight lines. As a result, the calculations for converting supply to price, and base to quote (and vice versa), differ from the actual math of the original curve.

First we define a few variables:

| | |
|---|---|
| $w = \frac{x_{max}}{n}$ | Size of interval between price points in number of tokens |
| $m = \lfloor \frac{x}{w} \rfloor$ | Index of the last interval |
| $r_x = x \ (mod\ w)$ | Number of $X$ tokens in the last interval |

**Price to Supply**

$$P(x) = p_m + \frac{r_x}{w}(P_{m+1} - P_m)$$

**Base to Quote**

$$y(x) = (\sum_{i=0}^{m-1} w\frac{p_{i+1}+p_i}{2}) + r_x^{\ 2}\frac{p_{m+1}-p_m}{2w} + r_x p_m$$

**Quote to Base**

$$x(y) = w(m + \frac{\sqrt{\mathbf{\Delta}}-p_m}{p_{m+1}-p_m})$$

Where

$$\mathbf{\Delta} = p_m^{\ 2} + \frac{2r_y(p_{m+1}-p_m)}{w}$$

| | |
|---|---|
| $m = \max_{i \in I}(i) + 1$ | Index of the last interval |
| $I = \{i \in [0; n] \mid \sum_{j=0}^{i} w\frac{p_{j+1}+p_j}{2} \leq y\}$ | Subset of all the filled intervals |
| $r_y = y - \sum_{j=0}^{m-1} w\frac{p_{j+1}+p_j}{2}$ | Number of Y token in the last interval |

## *General Form Function Derivations*

Below are the math derivations for the general form functions in section 2.3.3.

**Linear**

$$P(x) = ax + b$$

Let $P(0) = P_{min}$ $$P_{min} = b \quad (1)$$

Let $P(x_{max}) = P_{max}$ $$P_{max} = ax_{max} + P_{min}$$

$$a = \frac{P_{max} - P_{min}}{x_{max}} \quad (2)$$

Inject (1) and (2) back into $P(x)$ $$\therefore P(x) = \frac{x}{x_{max}}(P_{max} - P_{min}) + P_{min}$$

**Quadratic**

$$P(x) = ax^2 + bx + c$$

Capacity $C$ is the capital needed to absorb all $x$ $$C = \int_0^{x_{max}} P(x)\,dx = [\frac{ax^3}{3} + \frac{bx^2}{2} + cx]_0^{x_{max}}$$

$$C = \frac{ax^3_{max}}{3} + \frac{bx^2_{max}}{2} + cx_{max} \quad (1)$$

Let $P(0) = P_{min}$ $$c = P_{min} \quad (2)$$

Let $P(x_{max}) = P_{max}$ $$P_{max} = ax^2_{max} + bx_{max} + P_{min} \quad (3)$$

Inject (2) into (1) $$C = \frac{ax^3_{max}}{3} + \frac{bx^2_{max}}{2} + P_{min}x_{max}$$

$$\frac{bx^2_{max}}{2} = C - x_{max}(\frac{ax^2_{max}}{3} + P_{min})$$

$$b = \frac{2}{x^2_{max}} [C - x_{max}(\frac{ax^2_{max}}{3} + P_{min})] \tag{4}$$

Inject (4) into (3)

$$P_{max} = ax^2_{max} + x_{max} \cdot \frac{2}{x^2_{max}} [C - x_{max}(\frac{ax^2_{max}}{3} + P_{min})] + P_{min}$$

$$P_{max} = ax^2_{max} + \frac{2}{x_{max}} C - \frac{2ax^2_{max}}{3} - 2P_{min} + P_{min}$$

$$P_{max} = \frac{ax^2_{max}}{3} + \frac{2}{x_{max}} C - P_{min}$$

$$\frac{ax^2_{max}}{3} = P_{max} - \frac{2}{x_{max}} C + P_{min}$$

$$a = \frac{3}{x^2_{max}} (P_{max} + P_{min} - \frac{2}{x_{max}} C)$$

$$\therefore a = \frac{3}{x^3_{max}} (x_{max}(P_{max} + P_{min}) - 2C) \tag{5}$$

Rewrite (3)

$$b = \frac{1}{x_{max}} (P_{max} - ax^2_{max} - P_{min})$$

Inject (5)

$$b = \frac{1}{x_{max}} (P_{max} - x^2_{max}[\frac{3}{x^3_{max}} (x_{max}(P_{max} + P_{min}) - 2C)] - P_{min})$$

$$b = \frac{1}{x_{max}} (P_{max} - \frac{3}{x_{max}} (x_{max}(P_{max} + P_{min}) - 2C) - P_{min})$$

$$b = \frac{1}{x_{max}} (P_{max} - 3P_{max} - 3P_{min} + \frac{6C}{x_{max}} - P_{min})$$

$$b = \frac{1}{x_{max}} (- 2P_{max} - 4P_{min} + \frac{6C}{x_{max}})$$

$$\therefore b = \frac{2}{x^2_{max}} (3C - x_{max}(P_{max} + 2P_{min})) \tag{6}$$

We constrain $a \geq 0$ and $b \geq 0$ to make sure that $P(x) \geq 0$ and $P'(x) \geq 0 \; \forall \; x$:

Using (5)

$$\frac{3}{x_{max}^3}(x_{max}(P_{max} + P_{min}) - 2C) \geq 0$$

$$\frac{x_{max}}{2}(P_{max} + P_{min}) \geq C \qquad (7)$$

Using (6)

$$\frac{2}{x_{max}^2}(3C - x_{max}(P_{max} + 2P_{min})) \geq 0$$

$$C \geq \frac{x_{max}}{3}(P_{max} + 2P_{min})) \qquad (8)$$

Combining (7) and (8)

$$\therefore x_{max} \cdot \frac{P_{max}+2P_{min}}{3} \leq C \leq x_{max} \cdot \frac{P_{max}+P_{min}}{2}$$

**Exponential**

$$P(x) = ae^{x} + b$$

Let $P(0) = P_{min}$

$$\therefore b = P_{min} - a \qquad (1)$$

Let $P(x_{max}) = P_{max}$

$$ae^{x_{max}} + b = P_{max} \qquad (2)$$

Inject (1) into (2)

$$ae^{x_{max}} + P_{min} - a = P_{max}$$

$$\therefore a = \frac{P_{max}-P_{min}}{e^{x_{max}}-1} \qquad (3)$$

Inject (1) back into $P(x)$

$$P(x) = ae^{x} + P_{min} - a$$

$$P(x) = a(e^{x} - 1) + P_{min} \qquad (4)$$

Inject (3) into (4)

$$\therefore P(x) = \frac{e^{x}-1}{e^{x_{max}}-1}(P_{max} - P_{min}) + P_{min}$$

**Logarithmic**

$$P(x) = aln(1 + x) + b$$

We use $1 + x$ instead of $x$ to ensure $P(x) \geq 0$

Let $P(0) = P_{min}$ $\quad \therefore b = P_{min}$ (1)

Let $P(x_{max}) = P_{max}$ $\quad \therefore a = \frac{P_{max} - P_{min}}{ln(1+x_{max})}$ (2)

Inject (1) and (2) into $P(0)$ $\quad \therefore P(x) = \frac{ln(1+x)}{ln(1+x_{max})}(P_{max} - P_{min}) + P_{min}$

**Power**

We constrain the power to $\frac{x}{x_{max}}$ to prevent the function from growing too high

$$P(x) = \frac{a^{\alpha\frac{x}{x_{max}}}}{c} + b$$

Where $\alpha \in \mathbb{R}^*$

Let $P(0) = P_{min}$ $\quad \therefore b = P_{min} - \frac{1}{c}$ (1)

Let $P(x_{max}) = P_{max}$ $\quad \therefore c = \frac{a^{\alpha} - 1}{P_{max} - P_{min}}$ (2)

Inject (1) and (2) into

$$P(x) = \frac{a^{\alpha\frac{x}{x_{max}}} - 1}{a^{\alpha} - 1}(P_{max} - P_{min}) + P_{min} \quad (3)$$

$P(x)$ is only defined if and only if

$$a^{\alpha} - 1 \neq 0$$
$$a^{\alpha} \neq 1$$
$$\alpha \cdot log(a) \neq 0$$
$$\alpha \neq 0$$
$$a \neq 1$$

We pick $a = e$ and $\alpha \in \mathbb{R}^*$

$$\therefore P(x) = \frac{e^{\alpha\frac{x}{x_{max}}} - 1}{e^{\alpha} - 1}(P_{max} - P_{min}) + P_{min}$$