# LUSDChickenBonds Shifting Profitability

Daniel Simon

June 15, 2021

## 1 Introduction

The idea behind the shifting functions of LUSDChickenBonds is to allow the "owned" LUSD to act as a moving mass: it can serve as liquidity (in the Curve LUSD-3CRV metapool) when liquidity is needed the most, or provide stability through the Stability Pool when less liquidity is needed. Because of the permissionless nature of the functions, it is crucial that we ensure they can't be used in a way that would result in a loss to the protocol.

### 1.1 Intuition

Let's disregard fees for a second. Our intuition is that single-sidedly depositing LUSD while the pool is light on LUSD, and single-sidedly withdrawing LUSD while the pool is heavy on LUSD are profitable actions under the assumption that LUSD should eventually return to $1.

### 1.2 Informal reasoning

By adding liquidity to a pool (single-sided or not) we are given LP tokens in return, entitling us to a pro-rata share of the pool's reserves of each coin. This means that depositing single-sidedly (into a pool of 2 coins) involves implicitly swapping a portion of our input coin into the other coin. Similarly, withdrawing single-sidedly involves swapping one side of our pro-rata share of the pool into the other coin.

In the case of the LUSD-3CRV metapool: when the pool is light on LUSD, the spot price of LUSD in the pool will be more than $1. Making a single-sided LUSD deposit that moves the pool closer to being balanced amounts to selling part of the LUSD at a favorable rate (when it's "expensive"). On the other hand: when the pool is heavy on LUSD, its spot price will be less than $1. Withdrawing LUSD single-sidedly in this state amounts to buying LUSD at a favorable rate (when it's "cheap").

### 1.3 With fees

The existence of fees somewhat complicates things by reducing the rate at which our coins are swapped during single-sided liquidity changes (or non-proportional in general). In practice, we expect this to make balance-improving single-sided liquidity changes a net loss if the pool is already close to being balanced, such that the price of LUSD is roughly in the range of $1 \pm fee$.

Our approach is to run simulations across a sweep of "A" factors and spot prices to find out more precisely where the boundaries of profitability lie. Based on our findings we will choose spot price thresholds for depositing and withdrawing that ensure profitability under any A that might realistically be set by governance.

## 2 StableSwap Properties

First we introduce a couple of useful properties of StableSwap, which we will later use to show that the results of our simulations can be scaled to any pool size. We've demonstrated that the properties hold through the use of randomized testing, but we expect they could be proven mathematically.

Let's suppose we have the following functions (partially) implementing the StableSwap invariant for the 2-coin case, where $x$ and $y$ are the pool's reserves of coin 1 and 2:

- $D(x, y)$: pool's invariant "D" as a function of the reserves
- $dydx(x, y)$: the spot price $-\frac{dy}{dx}$ between coins 1 and 2
- $x_d(x, y, \Delta x, \Delta y)$: pool's reserve of coin 1 after a deposit of $\Delta x$, $\Delta y$
- $y_d(x, y, \Delta x, \Delta y)$: pool's reserve of coin 2 after a deposit of $\Delta x$, $\Delta y$
- $\frac{\Delta t_d}{t}(x, y, \Delta x, \Delta y)$: LP shares minted in exchange for a deposit of $\Delta x$, $\Delta y$, as a fraction of the (old) total supply t, such that the new total supply is $t' = t(1 + \frac{\Delta t_d}{t})$
- $x_{w1}(x, y, \frac{\Delta t}{t})$: pool's reserve of coin 1 after a single-sided withdrawal of coin 1 that burns fraction $\frac{\Delta t}{t}$ of the (old) total supply of LP shares
- $y_{w1}(x, y, \frac{\Delta t}{t})$: pool's reserve of coin 2 after a single-sided withdrawal of coin 1 that burns fraction $\frac{\Delta t}{t}$ of the (old) total supply of LP shares
- $\Delta x_{w1}(x, y, \frac{\Delta t}{t})$: the amount of coin 1 received when burning fraction $\frac{\Delta t}{t}$ of the (old) total supply to withdraw single-sidedly

All functions implicitly depend on the $A$, $fee$ and $admin\_fee$ parameters, except for $D()$ and $dydx()$, which don't depend on $fee$ or $admin\_fee$.

## 2.1 Property #1: D homogeneity

The function $D()$ is positively homogeneous of degree 1, i.e. for $s > 0$:

$$D(sx, sy) = sD(x, y)$$

Test: `https://github.com/liquity/ChickenBond/blob/b344f8d25ad6bade72092e53c505a8ea5f81b129/tools/sand-buckets/test/pool-prop.test.ts#L84-L94`

## 2.2 Property #2: spot price homogeneity

The function $dydx()$ is positively homogeneous of degree 0:

$$dydx(sx, sy) = dydx(x, y)$$

Test: `https://github.com/liquity/ChickenBond/blob/b344f8d25ad6bade72092e53c505a8ea5f81b129/tools/sand-buckets/test/pool-prop.test.ts#L96-L130`

## 2.3 Property #3: spot price monotonicity

Given property #2, we can express the spot price as a function of only the ratio $\frac{y}{x}$:

$$dydx(x, y) = dydx(\tfrac{x}{x}, \tfrac{y}{x}) = dydx(1, \tfrac{y}{x}) = g(\tfrac{y}{x})$$

We find that the function $g()$ is strictly monotone, which makes it injective. In other words, there is a unique ratio $\frac{y}{x}$ between the pool's reserves for each spot price.

Test: `https://github.com/liquity/ChickenBond/blob/b344f8d25ad6bade72092e53c505a8ea5f81b129/tools/sand-buckets/test/pool-prop.test.ts#L132-L168`

### 2.4 Property #4: deposit homogeneity

The functions $x_d()$, $y_d()$ are positively homogeneous of degree 1:

$$x_d(sx, sy, s\Delta x, s\Delta y) = sx_d(x, y, \Delta x, \Delta y)$$
$$y_d(sx, sy, s\Delta x, s\Delta y) = sy_d(x, y, \Delta x, \Delta y)$$

The function $\frac{\Delta t_d}{t}()$ is positively homogeneous of degree 0:

$$\tfrac{\Delta t_d}{t}(sx, sy, s\Delta x, s\Delta y) = \tfrac{\Delta t_d}{t}(x, y, \Delta x, \Delta y)$$

Test: `https://github.com/liquity/ChickenBond/blob/b344f8d25ad6bade72092e53c505a8ea5f81b129/tools/sand-buckets/test/pool-prop.test.ts#L212-L241`

### 2.5 Property #5: one-coin withdrawal homogeneity

The functions $x_{w1}()$, $y_{w1}()$ and $\Delta x_{w1}()$ are positively homogeneous of degree 1 in the variables $x$, $y$:

$$x_{w1}(sx, sy, \tfrac{\Delta t}{t}) = sx_{w1}(x, y, \tfrac{\Delta t}{t})$$
$$y_{w1}(sx, sy, \tfrac{\Delta t}{t}) = sy_{w1}(x, y, \tfrac{\Delta t}{t})$$
$$\Delta x_{w1}(sx, sy, \tfrac{\Delta t}{t}) = s\Delta x_{w1}(x, y, \tfrac{\Delta t}{t})$$

Test: `https://github.com/liquity/ChickenBond/blob/b344f8d25ad6bade72092e53c505a8ea5f81b129/tools/sand-buckets/test/pool-prop.test.ts#L243-L267`

## 3 Simulation

### 3.1 Marginal profit

Our approach is to simulate single-sided deposits/withdrawals of practically infinitesimal size over a wide range of spot prices and a selection of "A" factors between 10-5000.

#### 3.1.1 Definition for deposits

Let's define the following function for notational convenience, which is a shorthand for the pool's new invariant "D" after single-sidedly depositing $\Delta x$ of coin 1 (in our case: LUSD) into $x$, $y$:

$$D_{d1}(x, y, \Delta x) = D\big(x_d(x, y, \Delta x, 0), y_d(x, y, \Delta x, 0)\big)$$

We are assuming that the LP shares we receive are worth their virtual price in dollars. The definition of virtual price $v$ when $t$ is the total supply of LP shares:

$$v = \frac{D}{t}$$

We get the value of shares received for depositing $\Delta x$ into $x$, $y$ by multiplying the number of shares by the new virtual price after the deposit:

$$\begin{aligned} v_{d1}(x, y, \Delta x) &= t\tfrac{\Delta t_d}{t}(x, y, \Delta x, 0) \cdot \frac{D_{d1}(x, y, \Delta x)}{t\left(1 + \frac{\Delta t_d}{t}(x, y, \Delta x, 0)\right)} \\ &= \tfrac{\Delta t_d}{t}(x, y, \Delta x, 0) \cdot \frac{D_{d1}(x, y, \Delta x)}{1 + \frac{\Delta t_d}{t}(x, y, \Delta x, 0)} \\ &= D_{d1}(x, y, \Delta x)\left(1 - \frac{1}{\frac{\Delta t_d}{t}(x, y, \Delta x, 0)}\right) \end{aligned}$$

We assume that the true price of LUSD is \$1 irrespective of the current Curve spot price, and define marginal profit of a single-sided deposit as:

$$p_{md}(x, y, \Delta x) = \frac{v_{d1}(x, y, \Delta x) - \Delta x}{\Delta x} = \frac{v_{d1}(x, y, \Delta x)}{\Delta x} - 1,$$

in which $\Delta x$ should be small enough not to introduce significant slippage.

We aim to show that this definition of marginal profit is the same for any pool state having the same initial price and relative change in $x$, in other words: that $pmd()$ is scale invariant:

$$p_{md}(sx, sy, s\Delta x) \stackrel{?}{=} p_{md}(x, y, \Delta x)$$

Substituting into the body of $p_{md}(sx, sy, s\Delta x)$:

$$\begin{aligned}\frac{v_{d1}(sx, sy, s\Delta x)}{s\Delta x} - 1 &= \frac{D_{d1}(sx, sy, s\Delta x)\left(1 - \frac{1}{\frac{\Delta t_d}{t}(sx, sy, s\Delta x, 0)}\right)}{s\Delta x} - 1 \\ &= \frac{D\big(x_d(sx, sy, s\Delta x, 0), y_d(sx, sy, s\Delta x, 0)\big)\left(1 - \frac{1}{\frac{\Delta t_d}{t}(sx, sy, s\Delta x, 0)}\right)}{s\Delta x} - 1\end{aligned}$$

Using property #4 for $\frac{\Delta t_d}{t}()$, $x_d()$ and $y_d()$:

$$\frac{D\big(sx_d(x, y, \Delta x, 0), sy_d(x, y, \Delta x, 0)\big)\left(1 - \frac{1}{\frac{\Delta t_d}{t}(x, y, \Delta x, 0)}\right)}{s\Delta x} - 1$$

Using property #1 for $D()$ and simplifying:

$$\begin{aligned}&\frac{sD\big(x_d(x, y, \Delta x, 0), y_d(x, y, \Delta x, 0)\big)\left(1 - \frac{1}{\frac{\Delta t_d}{t}(x, y, \Delta x, 0)}\right)}{s\Delta x} - 1 \\ &= \frac{D\big(x_d(x, y, \Delta x, 0), y_d(x, y, \Delta x, 0)\big)\left(1 - \frac{1}{\frac{\Delta t_d}{t}(x, y, \Delta x, 0)}\right)}{\Delta x} - 1\end{aligned}$$

Replacing subexpression with functions we get the expression we're looking for:

$$\frac{D_{d1}(x, y, \Delta x)\left(1 - \frac{1}{\frac{\Delta t_d}{t}(x, y, \Delta x, 0)}\right)}{\Delta x} - 1 = \frac{v_{d1}(x, y, \Delta x)}{\Delta x} - 1 = p_{md}(x, y, \Delta x)$$

Therefore, the marginal profit for single-sided deposit at a given spot price is independent of pool size.

### 3.1.2 Definition for withdrawals

We once again assume that the LP tokens we burn are worth their virtual price in dollars, and define the value of burnt shares as:

$$v_b(x, y, \tfrac{\Delta t}{t}) = \left(t\tfrac{\Delta t}{t}\right)\frac{D(x, y)}{t} = \tfrac{\Delta t}{t}D(x, y)$$

Assuming LUSD is \$1, we define marginal profit of single-sided withdrawal as:

$$\begin{aligned}p_{mw}(x, y, \tfrac{\Delta t}{t}) &= \frac{\Delta x_{w1}(x, y, \frac{\Delta t}{t}) - v_b(x, y, \frac{\Delta t}{t})}{v_b(x, y, \frac{\Delta t}{t})} \\ &= \frac{\Delta x_{w1}(x, y, \frac{\Delta t}{t})}{v_b(x, y, \frac{\Delta t}{t})} - 1,\end{aligned}$$

in which $\frac{\Delta t}{t}$ should be small enough not to introduce significant slippage.

It can be shown that this is also scale invariant in $x$ and $y$:

$$\begin{aligned}
p_{mw}(sx, sy, \tfrac{\Delta t}{t}) &= \frac{\Delta x_{w1}(sx, sy, \frac{\Delta t}{t})}{v_b(sx, sy, \frac{\Delta t}{t})} - 1 \\
&= \frac{s\Delta x_{w1}(x, y, \frac{\Delta t}{t})}{\frac{\Delta t}{t} D(sx, sy)} - 1 \\
&= \frac{s\Delta x_{w1}(x, y, \frac{\Delta t}{t})}{\frac{\Delta t}{t} sD(x, y)} - 1 \\
&= \frac{\Delta x_{w1}(x, y, \frac{\Delta t}{t})}{\frac{\Delta t}{t} D(x, y)} - 1 \\
&= \frac{\Delta x_{w1}(x, y, \frac{\Delta t}{t})}{v_b(x, y, \frac{\Delta t}{t})} - 1 = p_{mw}(x, y, \tfrac{\Delta t}{t})
\end{aligned}$$

Therefore, the marginal profit for single-sided withdrawal at a given spot price is independent of pool size.

### 3.1.3 Implementation

We choose some arbitrary (large) value $D_0$ for the pool invariant "D", and for each combination of ($dydx_i$, $A_j$) we exploit property #3 to find the unique pair $(x, y)$ for which $D(x, y) = D_0$ and $dydx_{A=A_j}(x, y) = dydx_i$ by binary search. We then evaluate $p_{md}()$ and $p_{mw}()$ using relatively small values for $\Delta x$ and $\frac{\Delta t}{t}$ ($10^{-9}D$ in the case of $\Delta x$ and $10^{-9}$ for $\frac{\Delta t}{t}$).

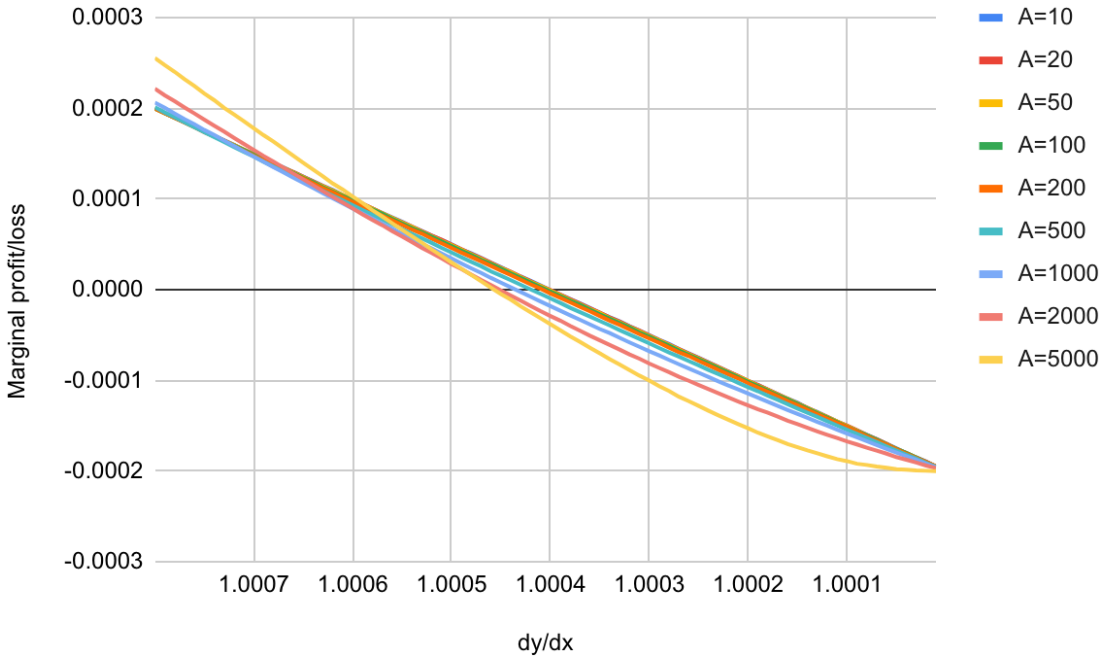
Code:

`https://github.com/liquity/ChickenBond/blob/b344f8d25ad6bade72092e53c505a8ea5f81b129/tools/sand-buckets/src/sheets/marginal-deposit.ts`

`https://github.com/liquity/ChickenBond/blob/b344f8d25ad6bade72092e53c505a8ea5f81b129/tools/sand-buckets/src/sheets/marginal-withdrawal.ts`

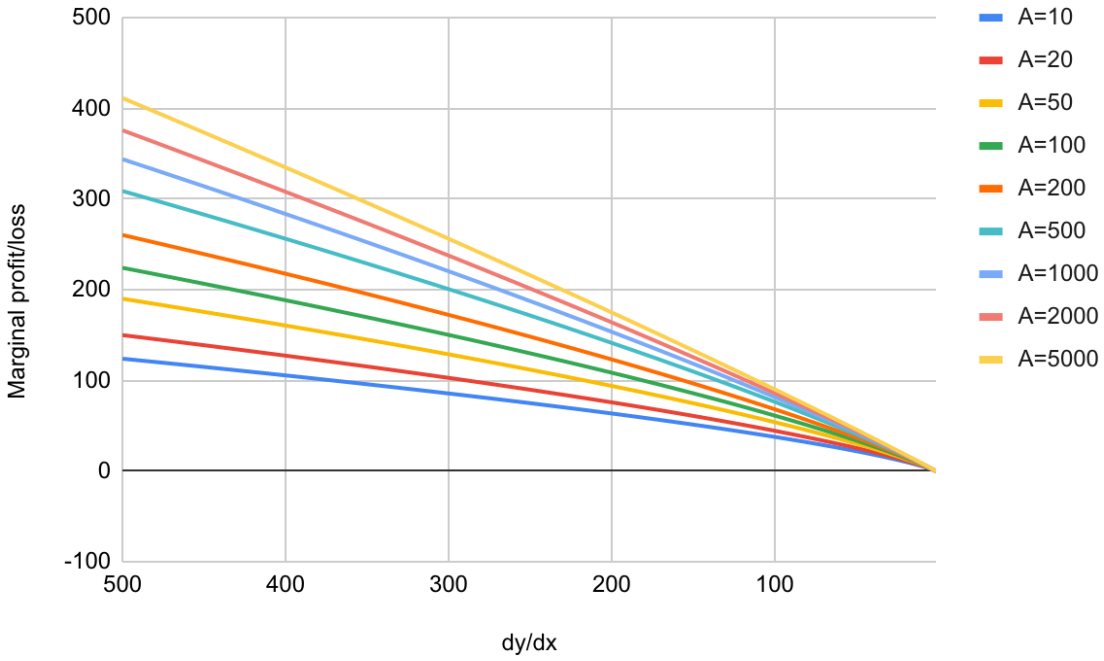
### 3.1.4 Results

We see that depositing becomes profitable around the $dy/dx = 1 + fee$ mark as expected, but that the exact point depends on the variable "A" factor of the pool (adjustable by governance) and is shifted slightly towards higher $dy/dx$ (more imbalance) at higher A's:

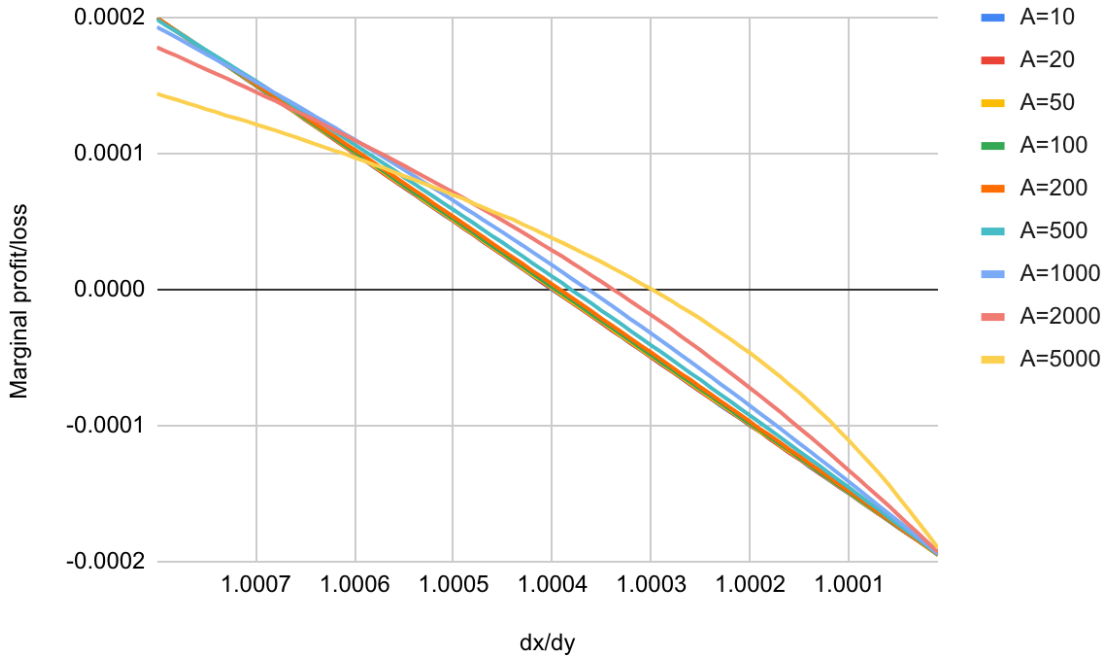



Therefore it makes sense to choose a threshold slightly higher than $1 + fee$ for allowing deposits (i.e. shifting SP $\Rightarrow$ Curve), e.g. 1.0005.

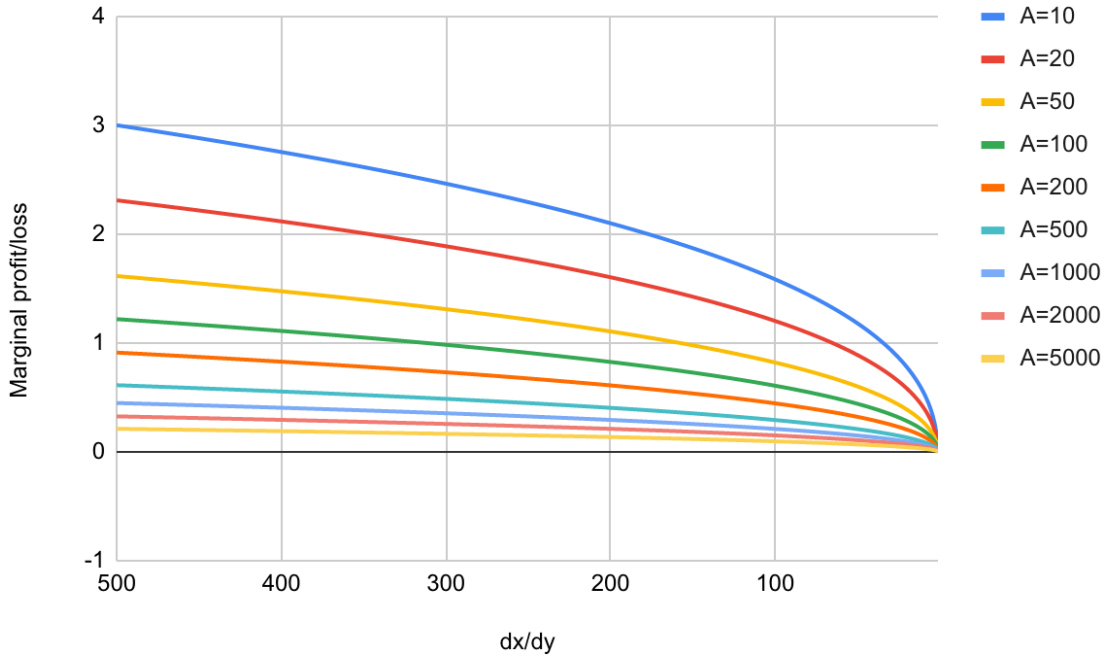
We further find that depositing only becomes more profitable as $dy/dx$ increases into absurd range:




In the case of withdrawal, we see that it also becomes profitable around $dx/dy = 1 + fee$, but the exact point is shifted in the other direction, towards lower $dx/dy$ (less imbalance):




Just as in the case of deposits, withdrawal only becomes more profitable at absurdly high $dx/dy$:




## 3.2 ROI of maximal shifts

We call a shift (either SP ⇒ Curve or Curve ⇒ SP) starting from a particular spot price maximal, if it ends at our chosen threshold for allowing a shift in that particular direction. In other words, any attempt to shift a higher amount would not be allowed due to our final check on the spot price.

It is interesting to note that the ROI of a maximal shift is actually the lowest ROI that can be achieved by a shift, and submaximal shifts are actually more profitable in terms of ROI. This comes

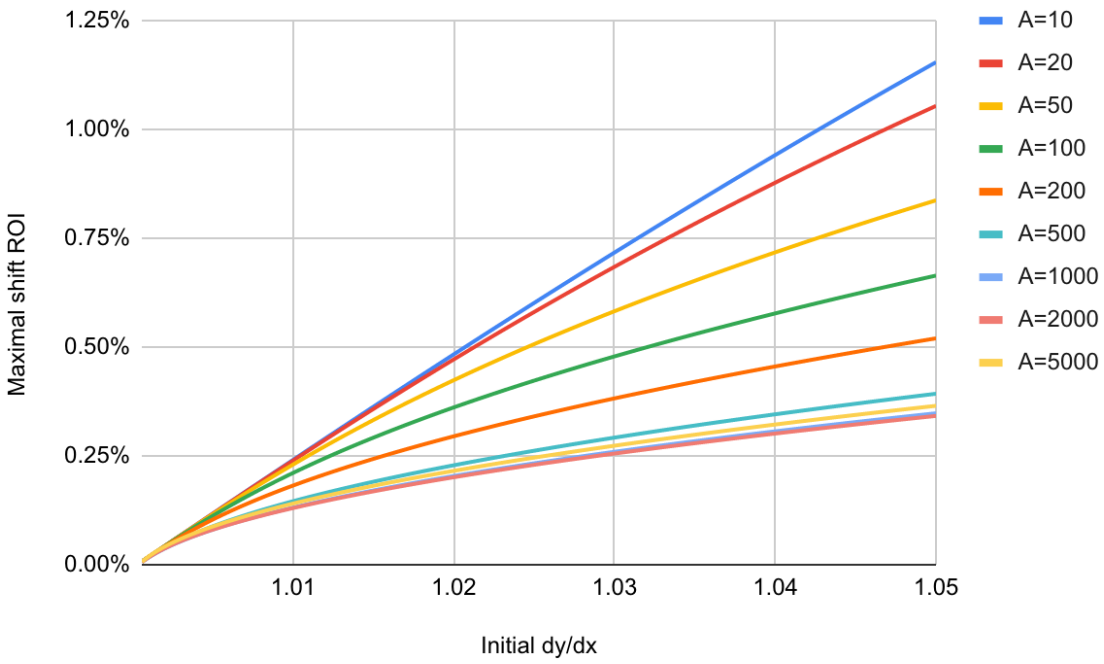
from the fact that marginal profit decreases the closer the spot price is to perfect balance.

Code:

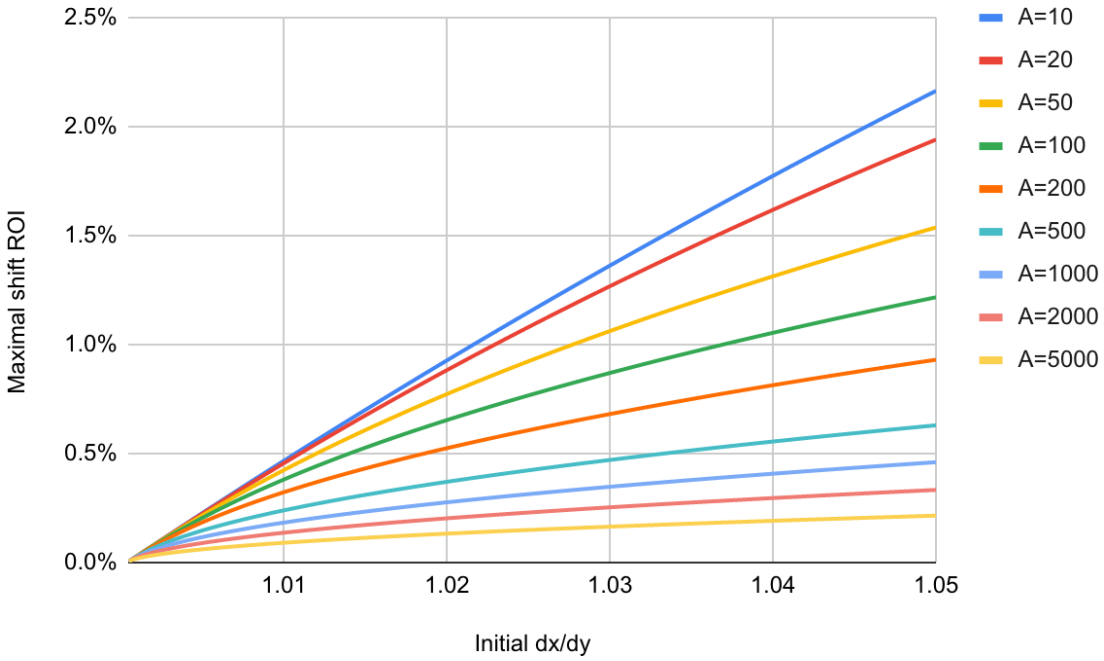
`https://github.com/liquity/ChickenBond/blob/b344f8d25ad6bade72092e53c505a8ea5f81b129/tools/sand-buckets/src/sheets/walled-deposit.ts`

`https://github.com/liquity/ChickenBond/blob/b344f8d25ad6bade72092e53c505a8ea5f81b129/tools/sand-buckets/src/sheets/walled-withdrawal.ts`

### 3.2.1 Deposit (to $dy/dx = 1.0005$)



### 3.2.2 Withdrawal (to $dx/dy = 1.0004$)




## 3.3 Spreadsheets

Link: `https://docs.google.com/spreadsheets/d/1FKwuPD5N6RPTQtYzTWb6NsoEYtpCCZKheYxMplKvD4w/edit#gid=971124440`