Platonic.Systems

*for Project Ardana*

# Audit

*Authors*
Quinn Dougherty quinn.dougherty@platonic.systems
Foo Bar foo.bar@platonic.systems

*Supervisor*
Isaac Shapira isaac.shapira@platonic.systems

November 1, 2021- 19:10

# Table of Contents


**I. Executive summary** **5**
- I.1. Recommendations . . . 5
  - I.1.1. Keep the DeX closed source to prevent forking . . . 5
  - I.1.2. Peg fee structure to governance (i.e. `DANA` holders) to avoid competition . . . 5
  - I.1.3. Monitor development of Cardano ecosystem for **multi-step atomic transactions** to guard against flashloan attacks . . . 5
- I.2. Nonissues . . . 5
- I.3. Insufficient literature . . . 6

**II. Preamble** **7**
  - II.0.1. Desiderata . . . 7
- II.1. Considerations . . . 8
- II.2. Attacks . . . 8

**III. Considerations** **9**
- III.1. Physical and operational security . . . 9
- III.2. Scalar types . . . 9
  - III.2.1. Onchain components . . . 9
  - III.2.2. Offchain components . . . 9
  - III.2.3. TODO after team is prepared to do numerical analysis stuff: finalize this section. . . . 10
- III.3. Root-finding . . . 10
  - III.3.1. Newton's algorithm . . . 11
  - III.3.2. TODO after team is prepared to do numerical analysis stuff: complete this section. . . . 11
- III.4. Throughput . . . 11

**IV. Attacks** **13**
- IV.1. Denial-of-service . . . 13
  - IV.1.1. Conclusion . . . 14
- IV.2. Vampire attack . . . 14
  - IV.2.1. The literature . . . 14
  - IV.2.2. Scenario: reputational damage if we're considered $\Pi'$ . . . 15
  - IV.2.3. Scenario: value siphoned out if we become $\Pi$ . . . 16
  - IV.2.4. Conclusion . . . 16

IV.3. Flashloans . . . 17
IV.3.1. Action: monitoring Cardano for developments in multistep atomic transactions . . . 17
IV.4. Reentrancy . . . 18

**V. Postamble** **19**
V.1. Toward formal verification . . . 19
V.2. Future work . . . 19

**VI. Appendices** **20**
VI.1. Appendix A: invariant polynomial . . . 20
VI.1.1. Derivation of Invariant polynomials . . . 20
VI.1.2. Analysis of roots and of derivatives . . . 22

**VII Bibliography** **23**

# List of beliefs

Belief IV.1.1 (No unique DoS) . . . . . 13
Belief IV.3.1 (No flashloans) . . . . . 17
Belief IV.4.1 (No reentrancy) . . . . . 18

# I. Executive summary

Platonic.Systems has conducted an internal Audit parallel to the engineering efforts building Danaswap, Ardana stablecoins, and Dana governance token mechanisms.

## I.1. Recommendations

Recommendations are assigned a five-valued confidence.

### I.1.1. Keep the DeX closed source to prevent forking

This can play a role in reducing vampire attack risk, and other considerations in . Confidence in importance: very low

### I.1.2. Peg fee structure to governance (i.e. `DANA` holders) to avoid competition

This reduces vampire attack risk, details IV.2.3. Confidence in importance: medium.

### I.1.3. Monitor development of Cardano ecosystem for multi-step atomic transactions to guard against flashloan attacks

With mitigation strategy sketches provided in IV.3.1. Confidence in importance: very high

## I.2. Nonissues

During the audit process research was conducted to rule out the following attack vectors.

1. **Reentrancy**

2. **Flashloans** (modulo IV.3.1)

## I.3. Insufficient literature

As of this writing, the jury is still out on the following considerations or attack vectors.

1. **Transaction-ordering dependence**: waiting on publications or code from IOHK to determine network fee resolution and their impact on miner incentives. (This is blocking any confidence level regarding the frontrunning question). A comment in (Guillemont 2021) suggests that miner-type frontrunning will not be an issue, but our confidence is not high in everything shaking out that way.

2. Filler, need a second item to block a render bug.

# II. Preamble

The audit is a preliminary effort to compensate for the fact that proper formal verification before launch is infeasible.

An audit means many things. Let's be precise about what we mean by audit in this document.

**Definition II.0.1** (Audit)**.** *An audit is a document provided to the community to guide them in taking informed risk.*

**Definition II.0.2** (Community)**.** *A community consists of liquidity providers, investors, swappers, arbitrageurs, governance token holders, and neighboring members/projects of the ecosystem.*

## II.0.1. Desiderata

- **The community is the audience. The community is the customer.**
- I think we want to make an explicit demarcation of risks for whom. Some risks only effect Ardana internally, other risks effect the collective of Dana holders, other risks effect our neighbors in the ecosystem. I think there's a difference between risks we want to eliminate and risks we want to empower the community to take. I'm sort of imagining profiling these as separate *axes* of risk. The idea here is that I'm frustrated with the literature because it almost feels like it uses the word "attack" anytime someone loses money. This is not adequate- sometimes you just gambled and lost, sometimes the "attacker" followed the rules of the mechanism as designed. There's a morality/values question to each attack, and **the audit needs to provide disambiguation and clarity without taking sides on each values/morality question**.
- The audit is **as much as was possible to do before launch time, not exhaustive**.

## II.1. Considerations

In this chapter we look at broad concepts and decisions and provide context into the way the team is thinking about them. This section should add indirect value to the process of taking informed risks.

## II.2. Attacks

In this chapter we profile threat models, attack vectors, vulnerabilities; mostly on the economic and mechanism design levels, but occasionally on the software implementation level.

This audit will take on a bit of a code-is-law opinion; many things which are called "attacks" are in fact people using mechanisms as designed. However, it is still the responsibilty of a platform (such as a DeX) to help the community make informed decisions about risk, even when the risk concerns unforeseen behaviors of a protocol or implementation.

Philosophically, be wary of morally charged language in the overall literature. It often implies that an attack is carried out by a summary enemy of the entire ecosystem, that the ecosystem is victimized, when clearer thinking shows that a small team or platform was the sole victim.

# III. Considerations

## III.1. Physical and operational security

(Shapira 2021)

## III.2. Scalar types

### III.2.1. Onchain components

We use `PlutusTx.Rational`[1] to represent numbers in the Danaswap contract. As of this writing, tolerance (number of decimals needed to evaluate equality) is set to `30`.

For the smart contract, we require calculations which are highly precise and can handle very large numbers and can be reproduced exactly across different hardware. Using FLOPs (floating point operations) is not compatible with these requirements. We are not able to determine exactly how big or how precise our numbers need to be, so we cannot say that FLOPs allow for enough size and precision. We can say, however, that FLOPs are implemented slightly differently on different hardware and results may not be reproducible across different hardware. Additionally, FLOPs are not allowed to be used in Plutus on-chain code. These are the constraints which do not allow us to use `Double` to represent numbers in the smart contract.

### III.2.2. Offchain components

`DanaswapStats` uses vanilla haskell's `Double` type to solve the invariant function.

[1] https://github.com/input-output-hk/plutus/blob/master/plutus-tx/src/PlutusTx/Ratio.hs

### III.2.3. TODO after team is prepared to do numerical analysis stuff: finalize this section.

## III.3. Root-finding

Recall the **invariant equation** from the StableSwap Whitepaper (Egorov 2019, 5). In the formalism provided by our Danaswap Whitepaper (Thomas 2021a, 3), there exists a function $I : S \to \mathbb{R}$ for contract states $S$ such that $I(s) = 0$ is equivalent to the invariant equation. Danaswap borrows everything from StableSwap to vary between constant-product and constant-sum market-making according to a *leverage* parameter, for which we also accept the suggestion found in (Egorov 2019). Sometimes, we need to hold all balances constant to solve for $D$ (which we call *the invariant*, having the semantics of total amount of coins **when** all coins have equal price). Other times, we consider a $k$ and solve for $B(s)_k$ holding everything else (including $D$) constant, when $B : S \to \mathbb{R}^n$ is a function assigning in every state a balance to each of $n$ assets (we will think of an $i \in 1..n$ as an *asset label*).

We define the **invariant polynomial** $n + 1$ times like so

**Definition III.3.1** (Invariant polynomials)**.**

$$I_D := D \mapsto D^{n+1} + (A + n^{-n})n^{2n}(\Pi B(s)_i)D + -An^{2n}(\Pi B(s)_i)\Sigma B(s)_i$$
$$\forall k \in 1..n, I_k := B(s)_k \mapsto B(s)_k^2 + \left(\Sigma_{i \neq k} B(s)_i + (\frac{1}{An^n} - 1)D\right) x_k + \frac{-D^{n+1}}{An^{2n}\Pi_{i \neq k} B(s)_i}$$

The derivations beginning with (Egorov 2019) are in Appendix A: invariant polynomial ($I_D$, $I_k$).

We think the invariant equation is best represented as polynomials set to zero, depending on what you're solving for, for the following reasons

1. **Characterize the roots in terms of existence and uniqueness**. It can be shown that there is exactly one nonnegative real root for $I_D$ and each $I_k$, and we'd like the onchain code to be close to the form that makes this easy to see.

2. **Trivially reason about derivatives**. Without my algebraic choices the derivatives (for Newton's method) are harder to see.

3. (Hypothesis): **Shrink the arithmetic tree size**. Leaving $\chi$ in a blackbox has the advantage of the codebase being able to plug in different leverage coefficients in the future just by supplying the leverage coefficient and its derivative. However, this puts more on the stack than is necessary. I haven't done any formal benchmarking

of this, but I currently believe the invariant polynomials in these forms are simpler trees and should therefore result in lower fees. Note IOHK have not published nor pushed code on a cost semantics / gas model, so we might not be able to reason about this.

4. **Increase our ability to reason about alternatives to Newton's method**. For example, looking at this problem from a companion matrix point of view becomes possible when we have formal polynomials.

### III.3.1. Newton's algorithm

In Curve's implementation of StableSwap, they use Newton's algorithm for root finding, so that's the first iteration of our codebase.

When the derivative can be found in a neighborhood of zero, Newton's method does not enjoy convergence guarantees (Wikimedia 2021, para. 4.1). The probability that invariant polynomial derivatives are in such a neighborhood is tiny, but nonzero, with details in VI.1.

We currently solve in `DanaswapStats` and oblige onchain logic to provide an $\epsilon$-proof that the root is valid.

### III.3.2. TODO after team is prepared to do numerical analysis stuff: complete this section.

## III.4. Throughput

TODO: establish throughput problem with language from (Thomas 2021b).

**Definition III.4.1** (Fairness)**.** *A DeX's concurrency solution is* ***fair*** *if when two people perform an action at the same time, that action is performed for the same price.*

> The Cardano mempool is designed to be "fair." Transactions are processed in a FIFO order regardless or how much in fees they pay (the ledger spec does support a fee market, but cardano-node doesn't take this into account) (Guillemont 2021)

TODO: language to discuss this quote.

**Definition III.4.2** (Fragmented)**.** *A UTXO state model is* ***fragmented*** *when there is more than one state UTXO in play at a time.*

**Definition III.4.3** (Normalized)**.** *A UTXO state model is* ***normalized*** *when there is neither disagreement nor redundancy regarding the data stored by the collection of UTXOs.*

Our UTXO state model design is *fragmented* yet *normalized*. In such a model, a fairness guarantee is impossible: to have a fairness guarantee, each pool would have to update eachother, which isn't possible to do in one transaction.

# IV. Attacks

## IV.1. Denial-of-service

**Definition IV.1.1** (Denial-of-service (DoS))**.** *A* ***denial-of-service*** *or* ***DoS*** *attack is a class of disruption that prevents intended users from reaching a service, usually accomplished by flooding or congesting.*

**Belief IV.1.1** (No unique DoS)**.** *Ardana ecosystem components do not offer a* ***unique*** *Denial-of-service (DoS) vector.*

However, we think Community ought to be made aware of *ambient* vulnerabilities in the broader Plutus and Cardano ecosystem.

We rely on (MLabs 2021) to describe three flavors of onchain DoS vector, which essentially target `Validator`s or `Redeemer`s.

**Definition IV.1.2** (Token dust attack)**.** *An attacker crams hundreds of unique tokens with different* ***`CurrencySymbol`s/`TokenName`s*** *into a single UTXO intending for it's representation to challenge the* ***`16kb`*** *limit. Then, the UTXO is placed in a* ***`Validator`*** *in such a way that one or more* ***`Redeemer`s*** *will need to consume it, blocking transactions on that* ***`Validator`-`Redeemer`*** *pair.*

**Definition IV.1.3** (Datum too big)**.** *In the datum field, an attacker puts an unbounded data structure on a UTXO that happens to demand consumption by a* ***`Redeemer`*** *which is critical to honest users.*

**Definition IV.1.4** (EUTXO concurrency DoS)**.** *An attacker submits a barrage of vacuous transactions consuming blocking EUTXOs.*

(MLabs 2021) points to (IOHK 2020) section on `Min-Ada-Value` as a mechanism that can be leveraged to block Token dust attack, but it's on the developer to set it and its implementation effects honest users.

> Every output created by a transaction must include a minimum amount of `ADA`, which is calculated based on the size of the output (that is, the number of different token types in it, and the lengths of their names). (IOHK 2020).

With similar drawbacks, fees or discincentives could block EUTXO concurrency DoS, where again honest users are impacted by the mechanism.

Neither ourselves nor (MLabs 2021) provide a strategy against Datum too big.

### IV.1.1. Conclusion

Denial-of-service (DoS) vectors are currently a part of Cardano. With respect to these vectors, we do not believe Danaswap nor anything in the Ardana ecosystem is better or worse off (IV.1.1).

## IV.2. Vampire attack

**Definition IV.2.1** (Vampire attack)**.** *Let* $\Pi$ *and* $\Pi'$ *be similar protocols, but* $\Pi$ *launched and attracted investors and customers earlier, and* $\Pi'$ *is somehow derivative of* $\Pi$*. Suppose* $\Pi'$ *competes with* $\Pi$ *such that* $\Pi'$ *makes parameter choices or other measures to become more attractive to investors or customers than* $\Pi$*. A* ***vampire attack*** *is defined as the migration of value (liquidity or other assets) out of* $\Pi$ *into* $\Pi'$*.*

### IV.2.1. The literature

Consult a selection of stories about vampire attacks.

- $\Pi'$ = `SushiSwap`; $\Pi$ = `UniSwap`[1]. SushiSwap was in fact a fork of UniSwap's code, and they provided incentives that directly targeted UniSwap investors and liquidity providers. This is the canonical notion of a vampire attack, with what appears to be the most written about it because of it's scale of impact and how early on the DeFi scene it was found. Our present definition is generalized for analysis that applies outside of the specific conditions here.
- $\Pi'$ = `Swerve`; $\Pi$ = `Curve`[2]. The term "vampire" does not occur in this article, but blaize.tech[3] considers it to be a vampire attack. By forking Curve, Swerve offered

[1] https://youtu.be/UFjXwrCGuog
[2] https://finance.yahoo.com/news/swerve-finance-total-value-locked-075020390.html
[3] https://blaize.tech/services/how-to-prevent-liquidity-vampire-attacks-in-defi/

a platform very similar to Curve's, and became competitive in total value locked (TVL) in a matter of days while people pulled out of Curve. There doesn't appear to be anything unique about Curve and Swerve being stablecoin DeXes.

- $\Pi'$ = `Artion`; $\Pi$ = `Opensea`[4]. At current writing it's too early to tell, but it's possible that Artion by providing a platform competitive with Opensea will be considered to have vampire attacked it. Unfolding events for this to be the case would have to be that Artion is successful at the expense of Opensea. My choice to be influenced by a CoinDesk writer's choice to call this a vampire attack is up for debate, but my intention is to be consistent with the ecosystem and the literature and I don't see grounds to exclude this writer from either.

- See extended notes on forks in (Lee 2020).

#### IV.2.1.1. Major takeaways

- Lack of vampire attack stories in the Cardano ecosystem is, according to my analysis, not a by-construction property of Cardano. I.e. it's a matter of time.

- Forking a codebase is often used as evidence in favor of the accusation that a given $\Pi'$ conducted a vampire attack, though forking is not an intrinsic property of the attack.

### IV.2.2. Scenario: reputational damage if we're considered $\Pi'$

Are there competing DeXs that beat us to market that could accuse us of vampire attacking them?

Imagine if a bunch of Curve investors pull out their liquidity, exchange it for `ADA` on Coinbase, and start playing Danaswap. Would Curve think of that like a vampire attack? The literature has not seen a vampire attack across a distance as great as that between Ethereum and Cardano, but that's only because we're early.

If the literature or ecosystem chooses to view Ardana as a vampire attacker, the project could suffer reputational damage.

[4] https://www.coindesk.com/tech/2021/09/24/andre-cronjes-new-nft-marketplace-is-a-vampire-attack-su

### IV.2.3. Scenario: value siphoned out if we become $\Pi$

Suppose another DeX for stablecoins launches with an incentive structure more attractive to our commmunity than our own. Then, everyone (II.0.2) could choose to migrate to this other DeX. According to the literature, we would be justified in considering this a vampire attack.

Suppose further that, having open sourced the Danaswap repo, this competitor's product is a fork of our own, making supplement components for any aspects that were closed source. If we follow the literature, we would be even more justified in considering this a vampire attack.

#### IV.2.3.1. Mitigations

- **Keeping the `Danaswap` code closed source**. This is a minor payout in risk reduction. We can also make a custom license, make it source available but proprietary, etc.
- **Peg fee parameters to democracy via `DANA` governance token holders**. The Community's preferences are a part of the competitive landscape; if a derivative competitor is closer to our Community's wants and needs than we are, then the Ardana Community will be siphoned out of Ardana. An automatic mechanism to decrease this possibility would look simply like setting policies such as the fee structure with vote inputs from the Community, however, automation shouldn't be the last word; attention and care will have to be paid to make sure people are actually using the mechanism. We see this having a stronger payout in risk reduction.

### IV.2.4. Conclusion

In any kind of market, participants take on the unavoidable risk of competitors showing up with better rates. The factor of code forking presented by the open source software context doesn't change this much, and the factor of fee structure parameters presented by the cryptoeconomic context doesn't either.

Vampire attack is a loose mirage of competitive phenomena: ultimately judged by the ecosystem literature, often coming down to individual journalists or researchers. It is in principle possible to be accidentally accused of vampire attacking. We do not want value siphoned out: there exist some practices to decrease this possibility that mostly amount to community engagement.

## IV.3. Flashloans

Flashloans are associated with something like `$136M`[5] in[6] losses[7].

**Definition IV.3.1** (Flashloan attack)**.** *Let a* ***flashloan*** *be some multi-step transaction that begins with an uncollateralized loan and ends with repayment of that loan, with arbitrary logic in between. Then, a* ***flashloan attack*** *is some method of manipulating prices during such a transaction and profiting.*

Ethereum offers flash loans because they have **multi-step atomic transactions**. There is no such mechanism in Cardano.

**Belief IV.3.1** (No flashloans)**.** *Flashloans will not be entering the Cardano ecosystem.*

As such, Danaswap, Ardana stablecoins, and mechanisms related to Dana governance tokens are not vulnerable to flashloan attacks.

### IV.3.1. Action: monitoring Cardano for developments in multistep atomic transactions

Project Ardana will be monitoring the evolution of Cardano, because we believe that if multistep atomic transactions are introduced flashloans will be shortly around the corner.

#### IV.3.1.1. In this event, the following mitigation strategy sketches will become urgent

- Onchain code only allow interop from one platform and users, not arbitrary platform.
- Lending products ought to require to collateralize in one whole transaction ahead of time before.
- Block price manipulation by disallowing mid-transaction information from updating prices.

[5] https://peckshield.medium.com/bzx-hack-full-disclosure-with-detailed-profit-analysis-e6b1fa9b18fc
[6] https://news.bitcoin.com/defi-protocol-harvest-finance-hacked-for-24-million-attacker-returns-2-5
[7] https://www.coindesk.com/markets/2021/05/20/flash-loan-attack-causes-defi-token-bunny-to-crash-ov

## IV.4. Reentrancy

**Definition IV.4.1** (Reentrant)**.** *A procedure is* ***reentrant*** *if it can be initiated while it's already running or a prior initiation has been interrupted and both runs can terminate, failing to raise an error.*

The infamous "DAO Hack" of 2016 occurred because Solidity allows the programmer to write reentrant smart contracts (Ma et al. 2019, 59–63).

**Belief IV.4.1** (No reentrancy)**.** *Plutus does not afford the freedom to write reentrant contracts.*

We can make a blanket statement that smart contracts in Cardano are invulnerable by construction to reentrancy. This is true because no transaction can be validated by (and it follows can require validation from) two different contracts. If you imagine Alice writes contract $A$ and invokes it (executing the program $\texttt{Alice}_A$) to validate transaction $T$, then Bob invokes $A$ ($\texttt{Bob}_A$) before Alice's invocation terminates, $T$ will be validated by **at most** one of $\texttt{Alice}_A$, $\texttt{Bob}_A$.

As such, reentrancy attacks are not a threat to Danaswap, Ardana stablecoins, or any mechanism relating to Dana governance tokens.

# V. Postamble

## V.1. Toward formal verification

I was hired as a logician on the merit of my coq skills, but formal verification was considered to time-intensive to do before launch. Producing this document was seen as a sort of warmup for FV. Now that we're launched, we will begin a formal verification stage of the project.

Properties we'd like to prove

1. Every transaction **worth `>=0$` to liquidity provider**, for some asset `$`.
2. **Non-indebted pools are never liquidated**.
3. **"No money for nothing"**: no one can arbitrarily withdraw assets from the protocol without depositing something else or paying some fee.
4. **Modular resilience**. In the language of (Genovese 2018), **modular risk** of a composite contract is risk that is greater than the sum of the risks of the individual lego blocks. We would like not just for the Ardana project to be **compositional** (i.e. the sum risk is *no more* than the sum of the risks of the individual lego blocks), but for it to be compositional with respect to actors that may interact with it, arbitrarily.
5. It costs nearly infinite money for an attacker to make `Danaswap`'s beliefs untrue.

## V.2. Future work

Research opportunities that were out of scope for the current document

1. What is the **delta in incentives** off ethereum via **no-multistep-atomic-transactions**?
2. Compare/contrast **transaction-ordering dependence** risks across ethereum to cardano and other blockchains.

# VI. Appendices

## VI.1. Appendix A: invariant polynomial

(Egorov 2019) gives us a way of easing between constant-sum and constant-product market-making by a coefficient $\chi$ called *leverage*, which turns out to be a function of $D$ and $B(s)$, where $B : S \to \mathbb{R}^n$[1] is a function assigning in every state a balance to each of $n$ assets. In what follows, let $x = B(s)$ such that $x_j = B(s)_j$ for each asset label $j$.

$$\chi D^{n-1} \Sigma x_i + \Pi x_i = \chi D^n + \left(\frac{D}{n}\right)^n$$

When $\chi$ is a blackbox, there is very little analysis available regarding the existence and behavior of roots or the convergence of any root-finding algorithm. We will forego any gains of abstracting over leverage coefficients, and let $\chi = \frac{A(\Pi x_i) n^n}{D^n}$ before proceeding.

### VI.1.1. Derivation of Invariant polynomials

First we derive $I_D$, the polynomial in unknown $D$.

[1] Balances are strictly positive, so it's not really $\mathbb{R}^n$, however we enjoy some vector space properties in [@DanaswapWhitepaper, p. 6] so we do not constrain the set.

**Derivation VI.1.1** ($I_D$)**.**

$$\chi D^{n-1}\Sigma x_i + \Pi x_i = \chi D^n + \left(\frac{D}{n}\right)^n$$

$$\frac{A(\Pi x_i)n^n}{D^n}D^{n-1}\Sigma x_i + \Pi x_i = \frac{A(\Pi x_i)n^n}{D^n}D^n + \left(\frac{D}{n}\right)^n$$

$$\frac{A(\Pi x_i)n^n}{D}\Sigma x_i + \Pi x_i = A(\Pi x_i)n^n + \frac{1}{n^n}D^n$$
$$\times D \quad \times D$$

$$An^n(\Pi x_i)\Sigma x_i + (\Pi x_i)D = An^n(\Pi x_i)D + \frac{1}{n^n}D^{n+1}$$
$$-An^n(\Pi x_i)D - \frac{1}{n^n}D^{n+1} \quad - An^n(\Pi x_i)D - \frac{1}{n^n}D^{n+1}$$

$$An^n(\Pi x_i)\Sigma x_i + (\Pi x_i)D - An^n(\Pi x_i)D - \frac{1}{n^n}D^{n+1} = 0$$

$$\frac{-1}{n^n}D^{n+1} + (1 - An^n)(\Pi x_i)D + An^n(\Pi x_i)\Sigma x_i = 0$$
$$\times -n^n \quad \times -n^n$$

$$D^{n+1} - n^n(1 - An^n)(\Pi x_i)D - An^{2n}(\Pi x_i)\Sigma x_i = 0$$

$$D^{n+1} + (A - n^{-n})n^{2n}(\Pi x_i)D + -An^{2n}(\Pi x_i)\Sigma x_i = 0$$

We now have a polynomial in $x \mapsto x^{n+1} + ax + b$ form, for constants $a$ and $b$ which are functions of a balance sheet and the *amplification coefficient* $A$.

Next, we start from a similar place and derive a polynomial in unknown of $B(s)_k = x_k$.

**Derivation VI.1.2** ($I_k$)**.** $\forall k \in 1..n$,

$$\frac{A(\Pi x_i)n^n}{D^n}D^{n-1}\Sigma x_i + \Pi x_i = \frac{A(\Pi x_i)n^n}{D^n}D^n + \left(\frac{D}{n}\right)^n$$

$$\frac{An^n}{D}(\Pi_{i\neq k}x_i)x_k(x_1 + ... + x_k + ... + x_n) + (\Pi_{i\neq k}x_i)x_k = An^n(\Pi_{i\neq k}x_i)x_k + \frac{D^n}{n^n}$$

$$\frac{An^n}{D}(\Pi_{i\neq k}x_i)x_k^2 + \frac{An^n}{D}(\Pi_{i\neq k}x_i)(\Sigma_{i\neq k}x_i)x_k + (\Pi_{i\neq k}x_i)x_k = An^n(\Pi_{i\neq k}x_i)x_k + \frac{D^n}{n^n}$$
$$-An^n(\Pi_{i\neq k}x_i)x_k - \frac{D^n}{n^n} \quad - An^n(\Pi_{i\neq k}x_i)x_k - \frac{D^n}{n^n}$$

$$\frac{An^n}{D}(\Pi_{i\neq k}x_i)x_k^2 + \left((\Pi_{i\neq k}x_i)\left(\frac{An^n}{D}\Sigma_{i\neq k}x_i + 1 - An^n\right)\right)x_k + \frac{-D^n}{n^n} = 0$$
$$\div\frac{An^n}{D}(\Pi_{i\neq k}x_i) \quad \div \frac{An^n}{D}(\Pi_{i\neq k}x_i)$$

$$x_k^2 + \left(\Sigma_{i\neq k}x_i + \frac{D}{An^n} - D\right)x_k + \frac{-D^{n+1}}{An^{2n}\Pi_{i\neq k}x_i} = 0$$

$$x_k^2 + \left(\Sigma_{i\neq k}x_i - D(1 - \frac{1}{An^n})\right)x_k + \frac{-D^{n+1}}{An^{2n}\Pi_{i\neq k}x_i} = 0$$

We now have a quadratic in $x \mapsto x^2 + ax + b$ form, for constants $a$ and $b$ which are each functions of $D$ and the other assets on the balance sheet.

## VI.1.2. Analysis of roots and of derivatives

TODO (notes in longreports/rootfinding/newton-robustness.ipynb)

# VII. Bibliography


10 Egorov, Michael. 2019. "StableSwap - Efficient Mechanism for Stablecoin Liquidity." https://curve.fi/files/stableswap-paper.pdf.

Genovese, Fabrizio Romano. 2018. "Modularity Vs Compositionality: A History of Misunderstandings." https://blog.statebox.org/modularity-vs-compositionality-a-history-of-

Guillemont, Sebastien. 2021. "[FR] - Increase Network Throughput." GitHub issue. 2021. https://github.com/input-output-hk/cardano-node/issues/3247.

IOHK. 2020. "FAQ: Native Tokens (Cardano's Multi-Asset Support Feature)." 2020. https://cardano-ledger.readthedocs.io/en/latest/explanations/faq.html.

Lee, Ian. 2020. "Fork Defense Strategies in DeFi." Bankless Newsletter. 2020. https://newsletter.banklesshq.com/p/fork-defense-strategies-in-defi.

Ma, Richard, Jan Gorzny, Edward Zulkoski, Kacper Back, and Olga V. Mack. 2019. *Fundamentals of Smart Contract Security*. Momentum Press.

MLabs, Team. 2021. "Common Plutus Vulnerabilities." 2021. https://mlabs.slab.com/public/posts/j8pjrj5y.

Shapira, Isaac. 2021. "Securing Ardana Swap." INTERNALLY CIRCULATING / ADD HYPERLINK HERE WHEN PUBLISHED[1].

Thomas, Morgan. 2021a. "Danaswap: A Scalable Decentralized Exchange for the Cardano Blockchain." INTERNALLY CIRCULATING / ADD HYPERLINK HERE WHEN PUBLISHED[2].

———. 2021b. "Transaction Throughput Scalability Strategies for Plutus Smart Contracts." *Journal.Platonic.Systems*. https://platonic.systems/papers/utxo-models-public.pdf.

Wikimedia. 2021. "Newton's Algorithm." 2021. https://en.wikipedia.org/wiki/Newton's_method.


[1] https://INTERNALLY%20CIRCULATING%20/%20ADD%20HYPERLINK%20HERE%20WHEN%20PUBLISHED

[2] https://INTERNALLY%20CIRCULATING%20/%20ADD%20HYPERLINK%20HERE%20WHEN%20PUBLISHED