Understand Volatility of Algorithmic Stablecoin: Modeling, Verification and Empirical Analysis

Wenqi Zhao(✉), Hui Li, and Yuming Yuan

Huobi Research, Hainan, China
{zhaowenqi,lihui0729,yuanyuming}@huobi.com

Abstract. An algorithmic stablecoin is a type of cryptocurrency managed by algorithms (*i.e.*, smart contracts) to dynamically minimize the volatility of its price relative to a specific form of asset, *e.g.*, US dollar. As algorithmic stablecoins have been growing rapidly in recent years, they become much more volatile than expected. In this paper, we took a deep dive into the core of algorithmic stablecoins and shared our answer to two fundamental research questions, *i.e.*, Are algorithmic stablecoins volatile by design? Are they volatile in practice? Specifically, we introduced an in-depth study on three popular types of algorithmic stablecoins and developed a modeling framework to formalize their key design protocols. Through formal verification, the framework can identify critical conditions under which stablecoins might become volatile. Furthermore, we performed a systematic empirical analysis on real transaction activities of the `Basis Cash` stablecoin to relate theoretical possibilities to market observations. Lastly, we highlighted key design decisions for future development of algorithmic stablecoins.

Keywords: Stablecoins · Modeling framework · Empirical analysis

1 Introduction

As cryptocurrencies on blockchain are notoriously known as volatile, *i.e.*, Their prices often fluctuate rapidly, stablecoins are proposed to peg their value to some external assets, *e.g.*, US dollar. In contrast to "unstable" cryptocurrencies, *e.g.*, Bitcoin [19], Ethereum [24], a stablecoin is able to minimize the volatility of its price relative to the pegged asset based on different mechanisms. The most common kind of stablecoins is backed-stablecoin, *i.e.*, The value of a stablecoin is backed by external assets, *e.g.*, commodity, fiat money or cryptocurrency as collateral. For example, the `USDC` stablecoin is backed by US dollar [5]. Unlike backed-stablecoins, algorithmic stablecoins, which are commonly not backed by other assets, have been gaining an increasing level of popularity in recent years due to the capability to stabilize its price via decentralized algorithms (*i.e.*, smart contract) without degrading too much capital efficiency. In general, this

M. Bernhard et al. (Eds.): FC 2021 Workshops, LNCS 12676, pp. 97–108, 2021.
https://doi.org/10.1007/978-3-662-63958-0_8

is realized by controlling the money supply of algorithmic stablecoins, which is similar to printing and destroying money in central banks. In this paper, we mainly focus on algorithmic stablecoins and will use the term interchangeably with "stablecoin" (Backed-stablecoins are not the main target in this work.).

Assuming that a stablecoin is pegged to US dollar, a smart contract is designed to dynamically manage its supply to minimize price volatility. We simply explain the algorithm as follows and will further the discussion later. When the price of the stablecoin exceeds one US dollar, the contract "produces" more coins and distributes them to the market. As a result, the price of the stablecoin should accordingly drop. In cases where the price of the stablecoin is lower than one US dollar, the smart contract decreases the supply of it in order to gradually lift its price back to one dollar. In practice, the aforementioned general algorithm can be instantiated by different models to achieve a more robust control over stablecoins. While many interesting research attempts aim at inventing such models, there is relatively little study on the other side, *i.e.*, Do they really work?

In this paper, we described a fundamental analysis on the volatility of algorithmic stablecoins, both theoretically and empirically. Our attempt of this study is to answer two fundamental research questions, which are:

Research Question 1: Are algorithmic stablecoins volatile *by design*?
Research Question 2: Are algorithmic stablecoins volatile *in practice*?

Our goal of the analysis described in this paper is to provide a more comprehensive understanding on the protocols of stablecoins (at both design and implementation level) with a specific focus on their volatility, which we believe is critical in the optimization of existing stablecoins and creation of potential future designs. We summarize our main contributions as follows.

- We introduced an in-depth protocol analysis on the designs of three popular types of algorithmic stablecoins. Moreover, we developed a general formal modeling and verification framework for stablecoins, which can be used to identify specific hidden criteria under which stablecoins might become volatile.
- We further conducted a systematic empirical study of the `Basis Cash` stablecoin based on real transaction activities on Ethereum and managed to relate theoretical possibilities (that stablecoins might be volatile) to market observations (unexpected volatile prices) between Dec 2020 to Jan 2021.

Paper Organization. The rest of the paper is organized as follows. Section 2 gives a systematic introduction of algorithmic stablecoins. Section 3 presents a formal modeling and verification framework designed for understanding and analyzing the volatility of algorithmic stablecoins in general. Furthermore, Sect. 4 describes an empirical study on one popular project and explains important observations based on real market transactions. In Sect. 5, we discuss related works in the literature and Sect. 6 concludes the whole paper.

2 Background

We classify algorithmic stablecoins into three categories, *i.e.*, rebase-style, seigniorage share and partial-collateral. In this section, we briefly explain key designs of all three types of stablecoins with popular projects as examples.

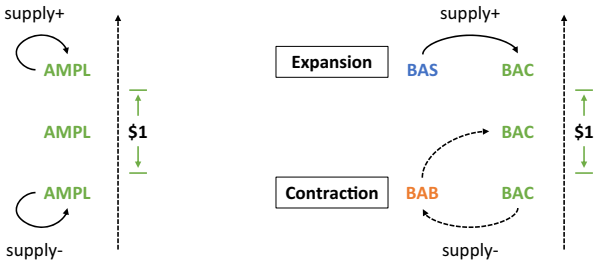


Fig. 1. The algorithmic stablecoins of `Ampleforth` and `Basis Cash`.

2.1 Rebase (`Ampleforth`)

The rebase-style stablecoins manage price-elastic ERC20 tokens, *i.e.*, The total supply of a stablecoin is non-fixed and adjusted adaptively on a routine basis. More specifically, the adjustment is automatically realized via the "rebase" process, which gradually stabilize the price of a target stablecoin near a specific peg, *e.g.*, one US dollar. We use `Ampleforth` [1] as an example for illustration, as shown in the left part of Fig. 1.

By design, the rebasing of `Ampleforth` is activated at 2am UTC on a daily basis. At the time of rebase, new coins are minted and distributed to all accounts proportionally based on their corresponding balances when the price of `Ampleforth` is higher than its peg. Given that the price of `Ampleforth` is \$1.2 with its peg to be \$1 (*i.e.*, 20% relate to peg), an account with 100 coins is rebased to own 120. On the other hand, holding coins might be automatically proportionally burned when the price falls below the peg.

2.2 Seigniorage Share (`Basis Cash`)

The seigniorage share model for algorithmic stablecoins commonly introduces two types of cryptocurrencies, *i.e.*, *coins* as a stablecoin and *shares* as ownership of seigniorage. In principle, shares are used to increase the supply of coins when the price of a coin is above its intended peg. In addition to these two cryptocurrencies, seigniorage-style stablecoins often issue a redeemable bond as an incentive for buyers when the price goes down below the peg. We use the `Basis Cash` [2] stablecoin for further explanation, as shown in the right part of Fig. 1. `Basis Cash` introduces three types of cryptocurrencies:

– `BAC`. `BAC` is the stablecoin and issued by the `Basis Cash` with a peg of \$1.

- `BAS`. `BAS` stands for Basis Shares, which is a seigniorage ERC20 token and provides inflationary gains of `BAC`. The design purpose of `BAS` is to prevent the price of `BAC` from going too high via dynamically increasing its supply. Currently, `BAS` can be earned via participating in yield farming, *i.e.*, deposit liquidity in decentralized finance platforms (*e.g.*, Uniswap [6]).
- `BAB`. `BAB` refers to Basis Bond whose price P_{bab} is mathematically determined by the price of `BAC` P_{bac}, *i.e.*, $P_{bab} = (P_{bac})^2$. Particularly, `BAB` offers an incentive for holders to earn `BAB` in a cost-effective way. The design purpose behind is to push `BAC` back to one dollar when its price falls below $1.

The general protocol of `Basis Cash` is designed to stabilize the price of `BAC` via adaptively controlling the supply of it. This is realized based on the two key mechanisms, *i.e.*, *expansion* and *contraction*, respectively. We simply describe the processes as below.

Expansion. The mechanism of expansion aims at increasing the supply of `BAC` in order to stabilize its price when it rises over the one dollar peg. In the design of `Basis Cash`, expansion is automatically activated in two settings. First, `BAC` will be minted and distributed as a reward to `BAS` holders. That said, for anyone who owns a specific amount of `BAS`, the expansion process proportionally assigns new `BAC` to his or her account. In the second case, owners of `BAB` are allowed to redeem `BAC` with their `BAB` at a 1:1 price, which also result in a quantity growth of `BAC`. Due to the increased supply in both situations, the expansion is expected to gradually make the price of `BAC` to decrease.

Contraction. In contrast to the process of expansion, contraction is designed to shrink the supply of `BAC`. To this end, an incentive is introduced in `Basis Cash` to encourage buyers to exchange `BAB` with `BAC` when the price of `BAC` is below one dollar. In the particular situation, one `BAC` is guaranteed to generate more than one `BAB` based on their price dependency as aforementioned. Moreover, the protocol of `Basis Cash` ensures that a specific amount of `BAB` is able to redeem the same amount of `BAC` when the price of `BAC` grows above $1 and required conditions are met. Based on the design of contraction, th price of `BAC` is anticipated not to fall too far from its peg. Compared to the design of rebase-style algorithmic stablecoins, the contraction mechanism of seigniorage share ones is commonly optional rather than automatically enforced. As shown in the right part of Fig. 1, the two dashed lines indicate that investors are allowed to participate in the contraction phase, or not.

2.3 Partial-Collateral (`Frax`)

In contrast to the two types of algorithmic stablecoins, an emerging class called fractional-algorithmic protocol is recently proposed as a combination of fully-collateral and fully-algorithmic ones. Compared to existing collateral-style stablecoins, *e.g.*, `DAI`, partial-collateral protocols introduce less custodial risks and avoid over-collateralization. On the other hand, it is designed to enforce a relatively tight peg with higher level of stability than purely algorithmic designs. We use the `Frax` project [4] below for illustration.

Particularly, `Frax` is the first attempt to implement the partial-collateral protocol of stablecoins. It introduces a two-token system, *i.e.*, `FRAX` as a stablecoin pegged to \$1 and `FXS` as a governance token, respectively. A collateral ratio $0 \leq r \leq 1$ is dynamically determined very hour with a step of 0.25% in the protocol to control at what percentage of peg the collateral is required to take to stabilize the value of `FRAX`. In cases where $r = 0.5$, \$0.5 must be in other types of stablecoins as collateral to mint a new `FRAX`. It becomes fully collateral when $r = 1.0$ and a pure algorithmic stablecoin if $r = 0$.

The collateral ratio r is 1.0 at genesis. In principle, minting a specific amount n of `FRAX` involves placing $n \times r$ of the value as collateral and burning $n \times (1 - r)$ of the value with `FXS`. As the price goes above its peg, the protocol provides the incentive for investors to mint new `FRAX`. Accordingly, the increased supply of `FRAX` is expected to gradually enforce the price to decrease. In cases where the price falls below the peg, the protocol allows investors to swap a combination of collateral and `FXS` valued \$1 with a single `FRAX` whose value is lower than \$1. Such incentives can potentially produce `FRAX` purchases and rise its price as well.

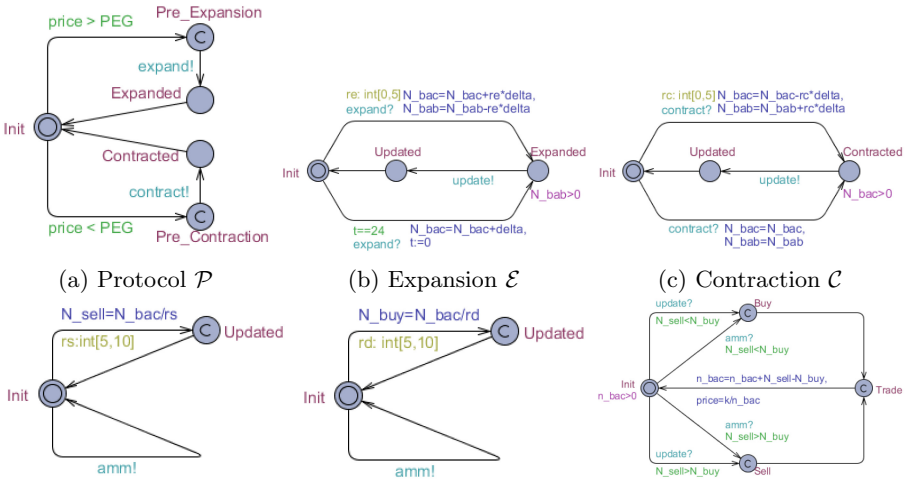
3 Modeling and Verification

3.1 Modeling of Stablecoin

We highlighted a formal modeling framework $\mathcal{M}$ for stablecoins. More formally, $\mathcal{M} \coloneqq \langle \mathcal{P}, \mathcal{E}, \mathcal{C}, \mathcal{S}, \mathcal{B}, \mathcal{X} \rangle$ is a network consisting of six types of timed automata [7], each of which is a tuple $Q \coloneqq \langle S, s_0, X, A, T, I, S_n \rangle$. S is the finite set of states. $s_0 \in S$ is the initial state. X is a set of non-negative real numbers as clock variables. $S_n \subseteq S$ is a set of accepting states. A is a set of actions and I is a set of invariants assigned to states. Given that Φ is constraint function, $T \subseteq S \times \Phi(X) \times 2^X \times A \times S$ is a collection of state transitions $\langle s, a, g, R, s' \rangle$, where s and s' are source and destination states, a is an action, g is the condition to enable the transition and R is the set of clocks to be reset.

Moreover, $\mathcal{M}$ provides communication through four classes of synchronized channels $\Omega \coloneqq \{\omega_e, \omega_c, \omega_x, \omega_u\}$. Specifically, ω_e and ω_c are designed to trigger *expansion* and *contraction* procedures. ω_x simulates market trading activities and generates a new price of stablecoin. ω_u synchronizes updates between $\mathcal{E}$, $\mathcal{C}$ and $\mathcal{X}$. Particularly, we presented a formal model of `Basis Cash` in Fig. 2. The framework is general to other types of stablecoins. Due to page limits, we selected `Basis Cash` because it manifests a typical model and was one of the most popular markets at the time of writing.

- $\mathcal{P}$ models the main protocol with five states, *i.e.*, initial state, `Pre_Expansion` and `Expanded` states when price is above the peg, `Pre_Contraction` and `Contracted` states when price is below the peg. The channels of `expand` (ω_e) and `contract` (ω_c) are activated on two transitions to enable the processes of expansion and contraction.
- $\mathcal{E}$ automata defines a process with a clock t and three states. $\mathcal{E}$ responds to expansion requests from $\mathcal{P}$. An expanding transition is executed to grow the



(a) Protocol $\mathcal{P}$ (b) Expansion $\mathcal{E}$ (c) Contraction $\mathcal{C}$

(d) Seller $\mathcal{S}$ (e) Buyer $\mathcal{B}$ (f) DEX $\mathcal{X}$

Fig. 2. Timed automata model of `Basis Cash` stablecoin.

supply of stablecoins (*i.e.*, global variable `N_bac`). The transition is allowed if t is at an expansion point (*e.g.*, 24:00 UTC). For `Basis Cash`, $\mathcal{E}$ creates two expansion transitions and synchronizes with $\mathcal{X}$ via the `update` channel (ω_u).

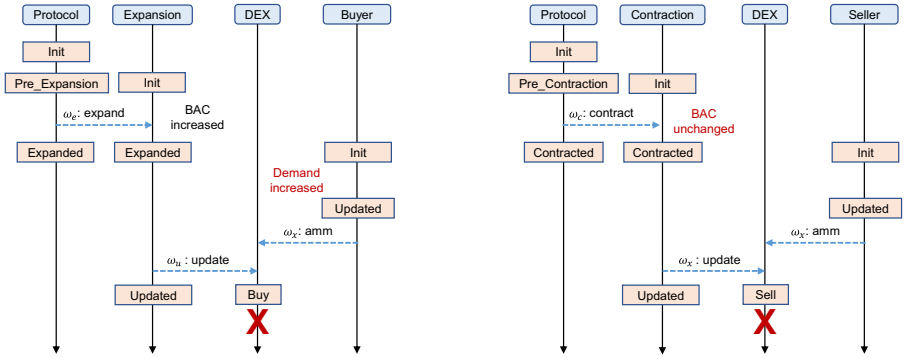
- $\mathcal{C}$ automata abstracts the contraction process. Similar to $\mathcal{E}$, a transition is provided to refine the decrease of supply via updating a global variable. Another transition is designed to model that the supply stays unchanged (investors can choose not to swap `BAB` with `BAC`).
- $\mathcal{S}$ and $\mathcal{B}$ are designed to model the behavior of sellers and buyers in an exchange. They generate random trading requests through the ω_x channel.
- $\mathcal{X}$ introduces an abstract model of decentralized exchanges (DEX) with automatic market making (AMM), *e.g.*, `Uniswap` [6]. $\mathcal{X}$ defines `Sell` and `Buy` states to indicate whether it is a buyer's market (*i.e.*, more sellers than buyers) or seller's market (the other way around). New prices are computed based on AMM and its pool of stablecoins.

3.2 Formal Verification

We further highlight important formal specifications to define stability properties (or non-volatility) of stablecoins with temporal logic [21]. Specifically, stability (non-volatility) is specified through the following two properties (`A` and `G` are quantifiers, *i.e.*, for all paths and for all states of a path in the state space [21]).

$$\texttt{AG}\ (\mathcal{P}.\texttt{Expanded} \wedge \mathcal{E}.\texttt{Updated}) \implies !\mathcal{X}.\texttt{Buy} \qquad \text{(expansion-validity)}$$

$$\texttt{AG}\ (\mathcal{P}.\texttt{Contracted} \wedge \mathcal{C}.\texttt{Updated}) \implies !\mathcal{X}.\texttt{Sell} \qquad \text{(contraction-validity)}$$



(a) Expansion validity (b) Contraction validity

Fig. 3. Counter-examples on non-volatility properties of `Basis Cash` stablecoin.

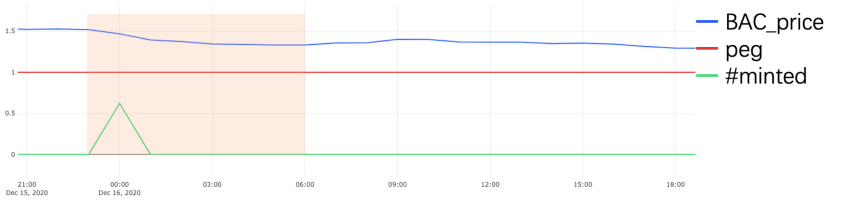
Specifications of Stability (Non-Volatility). Two properties are specified as in expansion-validity and contraction-validity to formalize the resilience against price fluctuation (with `Basis Cash` as an example). As formalized by expansion-validity, in cases where $\mathcal{P}$ is at state `Expansion` and $\mathcal{E}$ is at `Updated` (*i.e.*, expansion has been enforced), $\mathcal{X}$ must not stay at the state of `Buy` for the price to fall, *i.e.*, buyer's market. Similarly, when $\mathcal{P}$ is at `Contraction` and $\mathcal{C}$ is at `Updated`, $\mathcal{X}$ must not be at the state of `Sell`, *i.e.*, seller's market.

Counter-Example Analysis. We verified the model of `Basis Cash` with the `Uppaal` model checker for timed automata [16]. Figure 3 shows two counter-examples of the stability properties, *i.e.*, conditions under which `Basis Cash` might become volatile. Figure 3a describes a trading scenario where expansion validity is violated. Specifically, a demand growth of `BAC` occurs when the expansion process is started to mint and distribute new stablecoins. As a result, DEX goes to the state of `Buy` instead of `Sell` to trigger a counter-example. In terms of contraction-validity, Fig. 3b demonstrates another potential volatility of `Basis Cash`. When the price of `BAC` goes down below its peg, the contraction allows investors to swap `BAB` with `BAC`. However, in cases where the swap does not happen therefore supply of `BAC` stays unchanged, the contraction-validity is violated since DEX goes to the state of `Sell` instead of `Buy` as expected.

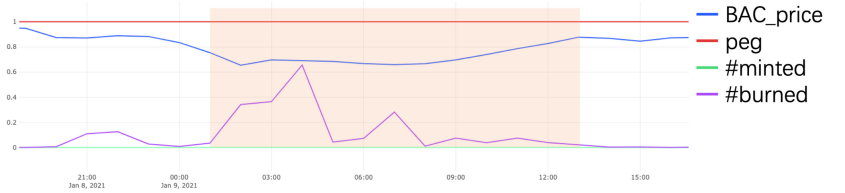
4 Empirical Analysis

Based on the formal modeling and verification of `Basis Cash`, we now describe an empirical analysis with real market observations, as shown in Fig. 4. More specifically, the empirical analysis was based on data at the time of writing from `Dune Analytics` [3]. We open-sourced all the data-retrieving queries at `https://explore.duneanalytics.com/dashboard/winky`.

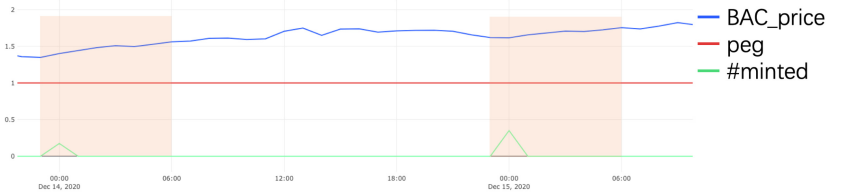
Normal Cases. The first pair of figures, *i.e.*, Fig. 4a and 4b, shows two cases where expansion and contraction worked well to stabilize the price of `BAC`.



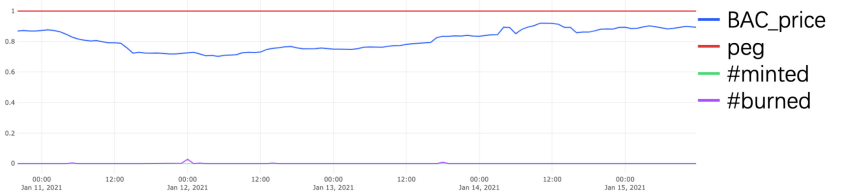
(a) Effective expansion



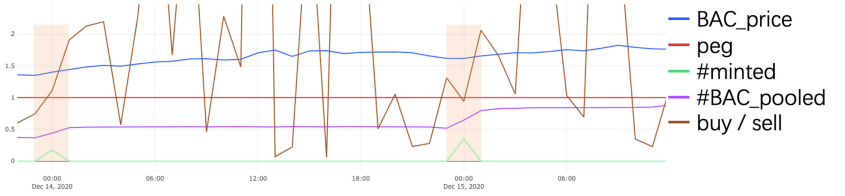
(b) Effective contraction



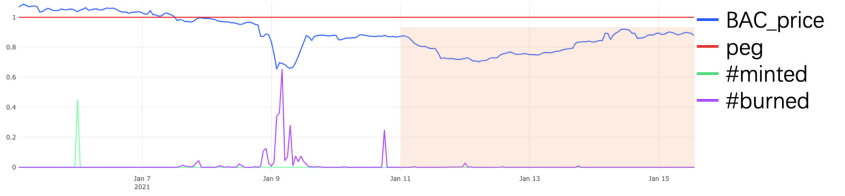
(c) Broken expansion



(d) Broken contraction



(e) Cause of broken expansion



(f) Cause of broken contraction

Fig. 4. Empirical analysis of `Basis Cash`. Unit: 10 million. (Color figure online)

As highlighted in Fig. 4a, as new BAC were minted (indicated by the green hump in the shaded area), its price gradually went down (blue line). Similarly, as a number of BAC were burned (indicated by the purple line) for contraction in Fig. 4b, its price started to rise. Such normal cases perfectly satisfied the design intent of Basis Cash and are different instances of traversing the state space as modeled in Fig. 2.

Broken Expansion. Unfortunately, Basis Cash might manifest an abnormal market as alarmed in Fig. 3 where the price of BAC becomes highly volatile. In our preliminary analysis, we found that such possibilities became real activities. Specifically, Fig. 4c and 4e have explained a broken expansion as inferred in Sect. 3.2 on Dec 14 and 15, 2020. In Fig. 4c, the expansion started at 00:00 with a collection of new BAC minted (green hump). However, its price increased in 7 h from $1.35 to $1.56 (Dec 14) and from $1.62 to $1.76 (Dec 15) instead of sticking to the peg, which amounted to a total growth of 15.72% and 8.40%. Based on Fig. 4e, the broken expansion was attributed to a rapid increase of demand as marked in Fig. 4e (brown line). As a measurement of purchase divided by sales, the brown line stayed above 1 in the most of the time. That said, there were more buyers of BAC in the market than sellers. Back at that time, *i.e.*, a relatively early stage of Basis Cash, yield-farming on BAC-DAI was very popular and led to a extremely high yield rate. Consequently, the demand of BAC was rapidly lifted even at an expansion point. Furthermore, the popularity of BAC was also reflected by the fact that 92% of the newly minted BAC on Dec 14, 2020 went to the yield-farming pool within 2 h after expansion (purple line in Fig. 4e).

Broken Contraction. In further, the potential volatility due to broken contraction was also confirmed in Fig. 4d and 4f. From Jan 11, 2021 to the time of writing, the price of BAC has been staying below its one US dollar peg as displayed in Fig. 4d despite that entries of contraction were continuously open during that period. That said, the mechanism of contraction failed to pull BAC back to or slightly over its peg price. The reason behind was the low participation in contraction at that time, *i.e.*, Many investors were unwilling to burn BAC for BAB due to the fear that they might never be able to redeem. As shown in Fig. 4f, the number of burned BAC (purple humps in the specific shaded area) during that period was much smaller than several days ago.

Design Decisions. Based on the aforementioned empirical analysis, we summarize two high-level important design decisions for algorithmic stablecoins in the future, especially for those adopting a similar design of Basis Cash.

- Compared to the design of extraction or similar mechanism where the supply of an algorithmic stablecoin goes up, the design of contraction is fundamentally more important and challenging. This is because cryptocurrencies are naturally easier to fall than rise. Therefore, a robust design of contraction should be well incentivized with investors' fear and reluctance taken into consideration.
- As two of the important parameters in algorithmic stablecoins, the quantity and cycle of intervention, *i.e.*, by how much adjustment an algorithm should

enforce and how frequent it needs to be triggered, might be potential improvements in the future. While a strong intervention might drive the stablecoin into another polar, a subtle one is probably ineffective. Similar balance is also required when it comes to the cycle of an algorithmic intervention. More reliable and flexible models are highly desired in this context.

5 Related Work

Stablecoins are cryptocurrencies with computing economic designs to achieve a relative stable market value and purchasing capability as well. Research on stablecoins have been attracting both economic and computer science researchers in recent years. Saito *et al.* proposed to stabilize blockchain cryptocurrencies via automatically controlling their supply to absorb both positive and negative demand shocks [22]. From the view of economics, Iwamura *et al.* suggested a new monetary policy for cryptocurrencies to stabilize their values [14]. Caginalp *et al.* argued that existing valuation frameworks were not compatible with cryptocurrencies which hold no underlying value thus new models were needed to the design of stablecoins [11]. To further develop this idea, they proposed a model of cryptocurrencies based on asset flow equations and investigated their stability with different parameterized configurations [10]. The resulting system was able to provide linear stability under specific market conditions. In the context of algorithmic stablecoins, Ametrano described a preliminary design in 2014 called *Hayek Money* which has been implemented by many projects nowadays [8]. In this specific design, price stability is achieved by dynamically rebasing the amount of cryptocurrency. A new paradigm was introduced to control the number of money units in all digital wallets instead of making each unit change its value. To further avoid unanticipated fluctuation of cryptocurrency prices, Sams designed *seigniorage shares* to include an elastic supply rule which adjusts the quantity of coins adaptively according to the changes of market value [23]. Compared to cryptocurrencies like Bitcoin whose supply growth is determined in advance, such scheme is more resilient against the intrinsic uncertainty of cryptocurrencies.

Design review and classification of stablecoins were al discussed in several research papers and industry reports [9,12,13,15,17,18,20]. Pernice *et al.* explored the landscape of stablecoins by proposing a taxonomy based on three types of collateralization, *i.e.*, direct, proxy and self-collateralization [20]. They further highlighted important implications and open questions based on the current development of stablecoins. From the viewpoint as decentralized payment systems, Mita *et al.* presented a similar discussion where stablecoins were categorized based on different types of collateral and intervention [17]. They pointed out that although algorithmic stablecoins introduced decentralization, they were weakly standardized to become practical payment tools. Moin *et al.* decomposed stablecoins in the literature into important building blocks [18]. They analyzed pros and cons of different designs and identified potential future trends as well. Klages *et al.* characterized stablecoins based on their functional risks related with incentive security and economic stability [15].

6 Conclusion

In this paper, we presented an in-depth theoretical and empirical analysis on the volatility of algorithmic stablecoins. We highlighted a formal modeling framework for stablecoins to identified important market criteria under which they might become volatile. Moreover, we related our theoretical findings to transaction activities on stablecoins via a further empirical analysis with real market data. Empirical results showed that potential possibilities predicted in the proposed model were confirmed in practice. Lastly, we highlighted important design decisions for the future development of stablecoin. All data used in this work are available at `https://explore.duneanalytics.com/dashboard/winky`.

References

1. Ampleforth. https://www.ampleforth.org/ (2021)
2. Basis Cash. https://basis.cash/ (2021)
3. Dune Analytics. https://duneanalytics.com/ (2021)
4. Frax. https://frax.finance/ (2021)
5. Tether. http://tether.to (2021)
6. Uniswap. http://uniswap.io (2021)
7. Alur, R., Dill, D.L.: A theory of timed automata. Theor. Comput. Sci. **126**(2), 183–235 (1994)
8. Ametrano, F.M.: Hayek money: the cryptocurrency price stability solution. Available at SSRN 2425270 (2016)
9. Bullmann, D., Klemm, J., Pinna, A.: In search for stability in crypto-assets: are stablecoins the solution? ECB Occasional Paper (230) (2019)
10. Caginalp, C.: A dynamical systems approach to cryptocurrency stability. arXiv preprint arXiv:1805.03143 (2018)
11. Caginalp, C., Caginalp, G.: Opinion: Valuation, liquidity price, and stability of cryptocurrencies. Proc. Nat. Acad. Sci. **115**(6), 1131–1134 (2018)
12. Clark, J., Demirag, D., Moosavi, S.: SoK: demystifying stablecoins. Available at SSRN 3466371 (2019)
13. Hileman, G.: State of stablecoins (2019). Available at SSRN (2019)
14. Iwamura, M., Kitamura, Y., Matsumoto, T., Saito, K.: Can we stabilize the price of a cryptocurrency?: Understanding the design of bitcoin and its potential to compete with central bank money. Hitotsubashi J. Econ. **60**, 41–60 (2019)
15. Klages-Mundt, A., Harz, D., Gudgeon, L., Liu, J.Y., Minca, A.: Stablecoins 2.0: economic foundations and risk-based models. In: Proceedings of the 2nd ACM Conference on Advances in Financial Technologies, pp. 59–79 (2020)
16. Larsen, K.G., Pettersson, P., Yi, W.: Uppaal in a nutshell. Int. J. Softw. Tools Technol. Transf. **1**(1–2), 134–152 (1997)
17. Mita, M., Ito, K., Ohsawa, S., Tanaka, H.: What is stablecoin?: A survey on price stabilization mechanisms for decentralized payment systems. In: 2019 8th International Congress on Advanced Applied Informatics (IIAI-AAI), pp. 60–66. IEEE (2019)
18. Moin, A., Sekniqi, K., Sirer, E.G.: SoK: a classification framework for stablecoin designs. In: Financial Cryptography (2020)
19. Nakamoto, S.: Bitcoin: a peer-to-peer electronic cash system. Tech. rep, Manubot (2019)

20. Pernice, I.G., Henningsen, S., Proskalovich, R., Florian, M., Elendner, H., Scheuermann, B.: Monetary stabilization in cryptocurrencies-design approaches and open questions. In: 2019 Crypto Valley Conference on Blockchain Technology (CVCBT), pp. 47–59. IEEE (2019)
21. Pnueli, A.: The temporal logic of programs. In: 18th Annual Symposium on Foundations of Computer Science (sfcs 1977), pp. 46–57. IEEE (1977)
22. Saito, K., Iwamura, M.: How to make a digital currency on a blockchain stable. Future Gener. Comput. Syst. **100**, 58–69 (2019)
23. Sams, R.: A note on cryptocurrency stabilisation: seigniorage shares. Brave New Coin, pp. 1–8 (2015)
24. Wood, G., et al.: Ethereum: a secure decentralised generalised transaction ledger. Ethereum Proj. Yellow Pap. **151**(2014), 1–32 (2014)