Master Thesis

Melon: An analysis of a blockchain-based asset management infrastructure

Onur Tasyurdu
2014-051-932

Supervised by:
Prof. Dr. Fabian Schär

Credit Suisse Asset Management (Schweiz) Professor for
Distributed Ledger Technologies and Fintech
Center for Innovative Finance, University of Basel

Major in: Finance, Controlling and Banking

Submission date: August 17, 2020

Abstract – In this thesis, we examine whether differences arise from the upgrade of the Melon Protocol by analyzing its architecture in a technical part of this thesis and the activity on Melon in an economic part. We reveal that not much has changed in the architecture, but we observe increased activity. Analyzing the possible drivers, we conclude that the new cryptoassets added to the Melon asset universe – as well as the new decentralized exchanges integrated with the Melon Protocol – contribute to the increased activity on Melon, but they are not the main drivers. Furthermore, we apply a spanning test in the form of a step-down test to investigate whether adding new cryptoassets to the Melon asset universe improves portfolio diversification. Even if the results of the analyzed sub-period demonstrate significant improvements, we conclude that the framework conditions and the lack of the observed data do not allow a valid conclusion.

Contents

List of Figures

List of Tables

List of Abbreviation

AMG	*Asset Management Gas*
AMGU	*Asset Management Gas Unit*
API	*Application Programming Interface*
AuM	*Asset under Management*
CSD	*Central Securities Depository*
DAO	*Decentralized Autonomous Organization*
DApp	*Decentralized Application*
DeFi	*Decentralized Finance*
DEX	*Decentralized Exchange*
DoS	*Denial of Service*
GAV	*Gross Asset Value*
GMVP	*Global Minimum Variance Portfolio*
HWM	*High-Water Mark*
MEB	*Melon Exposed Businesses*
MFP	*Melon Funding Proposal*
MMI	*Melon Manager Interface*
MTC	*Melon Technical Council*
NAV	*Net Asset Value*
TP	*Tangency Portfolio*
TROFs	*Technology Regulated and Operated Funds*
TVL	*Total Value Locked*

University of Basel

Center for Innovative Finance

Plagiatserklärung

Ich bezeuge mit meiner Unterschrift, dass meine Angaben über die bei der Abfassung meiner Arbeit benutzten Hilfsmittel sowie über die mir zuteil gewordene Hilfe in jeder Hinsicht der Wahrheit entsprechen und vollständig sind. Ich habe das Merkblatt zu Plagiat und Betrug vom 22. Februar 2011 gelesen und bin mir der Konsequenzen eines solchen Handelns bewusst.

Pratteln, August 17, 2020

Onur Tasyurdu

1 Introduction

Firms must use technology to transform crucial areas of their businesses and allow adaptation to the rapidly changing environment of the global economy. These fast-growing changes can be observed especially in the asset management industry. Today, technology enables clients to conduct their own research, which reduces the need for human advice, leads to lower-cost asset management products and results in greater scale (PWC 2018). However, in the fund management ecosystem, when setting up a fund or where security between fund managers and investors is not assured, costly financial intermediaries such as custodians, administrators, reporting agents, auditors and many others continue to play important roles. They serve as third parties to mitigate risks in the network (Grant Thornton 2011). When taking the global hedge fund industry as an example, the average set-up costs according to KPMG (2013) are the following: For a small fund, the average set-up costs are $700,000 USD; for a medium-sized fund, the costs rise to $6 million USD; and for a large fund, the average set-up costs can reach $14 million USD. Due to the new regulatory environment in place since 2008, the costs in the industry have increased significantly. It is estimated that hedge funds have spent $3 billion USD on compliance costs since 2008 (KPMG 2013).

Blockchain technology enables new opportunities in the management of assets. Melon, as a project powered by the Melon Protocol, aims to combine the managing of assets with the technology of blockchain. It is based on the Ethereum blockchain and has the goal to facilitate financial transactions through the replacement of intermediary functions required in traditional finance by lines of code. In Melon, anyone is enabled to be a manager who sets up, manages and maintains an on-chain Melon fund. Investors are allowed to invest in these Melon funds, whereby a full transaction history is always provided. This offers more transparency and reduces the risk of manipulation (Melon 2020*g*).

Melon, as a building block of the Decentralized Finance (DeFi), is categorized as an asset management project (DeFi Pulse 2019). DeFi, also referred to as Open Finance, is an alternative financial ecosystem with reduced entry barriers and minimized requirements for trust. The ecosystem is not meant to replace the traditional financial ecosystem, it is about integrating into it to make it more equitable by using open protocols, decentralized applications (DApps) and cryptoassets (see Section 2.2). DeFi is based on smart contracts generally built on top of the Ethereum platform and inherits its advantages as well as its disadvantages. Thus, the Melon Protocol in the *protocol layer* is only as secure as the Ethereum blockchain in the *settlement layer* (Schär 2020). Based on the *settlement layer*, the challenge for DeFi projects is, therefore, to create the necessary economic incentives to generate value for the stakeholders in the use of the underlying protocols.

DeFi is still a niche market (Schär 2020), and various DeFi projects are still trying to establish themselves, including Melon. The objective of this thesis is to examine whether technical or economic differences arise from the upgrade of the Melon Protocol v1.0 to v1.1. Section 2 provides an introduction to the terms such as Ethereum, smart contracts and DeFi. Section 3 offers a brief overview of the traditional asset management industry (see Section 3.1) to better understand the purposes and intentions of Melon (see Section 3.2). Section 4 describes the technical architecture of the Melon Protocol v1.1. Based on the explanation of the *settlement layer* of Melon in Section 2, we continue with the Melon Protocol in the *protocol layer*. Additionally, we introduce the Melon (MLN) token. Section 4.2 is about the Melon Terminal v2.0 in the *application layer*. Section 4.3 discusses the governance structure of Melon. In the first part of Section 5, an economic analysis is provided to answer the following questions: Has the upgrade of the Melon Protocol v1.0 to v1.1 led to an increase in the activity on Melon? If so, what are the possible drivers? The second part of Section 5 applies a mean-variance spanning test in the form of a step-down test to examine whether the inclusion of new cryptoassets to the Melon asset universe with the Melon protocol upgrade to v1.1 led to diversification benefits for Melon managers. And finally, Section 6 provides the conclusions to the thesis.

2 Theoretical Framework

This section provides a brief overview of the Ethereum Blockchain, smart contracts and Decentralized Finance (DeFi). As a programmable and distributed computing platform, Ethereum enables anyone to build open protocols, develop decentralized applications (DApps) and issue tokens by writing smart contracts. Once successfully created, each smart contract has an account state that consists of a nonce, ether balance, storage root and predefined executable code (Wood et al. 2014). These capabilities and functionalities allow the creation of numerous projects, which build an open financial ecosystem, also known as DeFi[1]. One of these DeFi projects is Melon, an open-source protocol for on-chain asset management built on Ethereum. Before describing the architecture of the Melon Protocol in Section 4, it is necessary to discuss relevant terminologies. This chapter provides a basis for the second, more technical part of this thesis.

2.1 Ethereum

With the innovation of Bitcoin, a way was found (Nakamoto 2008) to transfer ownership of virtual Bitcoin units without having to trust a third party, such as a financial institution. Blockchain, based on the distributed ledger technology (DLT), serves as the foundation for the decentralized nature of the Bitcoin system. A useful introduction into the Bitcoin blockchain is provided by Berentsen and Schär (2017), but there are some limitations of Bitcoin, such as the lack of Turing-completeness or the lack of multi-stage contracts. Building on this, Ethereum, which was first described in late November 2013 by Buterin (2013), was created to expand the power of blockchain technology and to use it more broadly. As of August 2020, Ethereum is the second largest cryptoasset after Bitcoin, measured by market capitalization of more than $48 billion USD (Coinmarketcap 2020*a*).

[1] Theoretically, there are other platforms apart from Ethereum which can be used to join the DeFi ecosystem, such as EOS, NEO, Tezos and others (Coinlore 2020).

Ethereum is a public and permissionless blockchain and currently relies on a proof-of-work consensus algorithm, also referred to as Ethash, to secure and maintain the Ethereum blockchain (Wood et al. 2014). The consensus protocol works without essential intermediaries, so that any arbitrary participant can join the open network without relying on any trusted parties. Transactions are validated through the underlying consensus protocol and are stored in a verifiable, immutable and valid way on the blockchain (Wöhrer and Zdun 2018; Luu et al. 2016). However, there are plans to transition to a mechanism based on proof-of-stake, which is referred to as Casper (Buterin and Griffith 2017).

2.2 Smart Contract

As of August 2020, Ethereum is the most-used smart contract platform to build open protocols, develop DApps (State of the DApps 2020) and issue tokens. Open protocols are built from smart contracts and may have implemented their own arbitrary rules for ownership, transaction formats and state transition functions (Buterin 2013). DApps are a new class of applications that have potential use cases across many sectors, including financial services and asset management. The integrated programming capabilities directly into the Ethereum Protocol enable developers around the world to design these DApps which are accessible anywhere in the world (Grayscale Building Blocks 2020). Tokens are also a new type of application. Roth et al. (2019) define tokens as digital units of value, which can represent the ownership of an asset or may include promises for the delivery of goods and services. They are based on a special data structure within the blockchain that tracks the balance of each address. Today, the vast majority tokens are smart contract-based tokens built on Ethereum's ERC20 token standard. ERC20 tokens have implemented a standard Application Programming Interface (API) to ensure the interoperability between tokens on the Ethereum blockchain (EthHub 2019). Of the top 100 tokens by market capitalization, 85% are built on Ethereum (Coinmarketcap 2020*b*). However, Ether (ETH) itself, the native cryptoasset of the Ethereum blockchain, does not comply with this standard. With a different smart contract, wrapping ETH to wrapped ETH (WETH) is possible, so ETH behaves the same as an ERC20 token, and applications that require ERC20 token standards can be used with ETH. The WETH is created by sending ETH to the aforementioned smart contract, which, obviously, should hold the same value in ETH as WETH is created. That is why it should always be possible to convert WETH back to ETH (Binance Academy 2020).

To build and develop open protocols and applications, anyone can write a code based on the Ethereum platform that automatically executes specific actions when certain conditions are met. Because smart contracts for the Ethereum platform run on and are secured by the blockchain, they will be executed strictly as programmed. The smart contract possibilities are due to a built-in Turing-complete programming language in Ethereum, and the commonly used programming language is the high-level language Solidity. Whenever a function of a smart contract – deployed on Ethereum – is called, its code is executed by the Ethereum Virtual Machine (EVM), which can be thought of as a global decentralized computer. In fact, the EVM is rather a network of many computers, also known as nodes, which are in constant communication. All transactions of the network are processed in relative synchrony by these nodes. Verifying the transactions sent from users is done by each node. Nodes can participate on a process called *mining*. These participating nodes are called miners and attempt to add blocks of transactions to the blockchain in order to receive an appropriated reward (Wöhrer and Zdun 2018). As of August 2020 miners will get 2 ETH as a reward, after successfully adding a block to the blockchain. This reward decreased from 3 ETH to 2 ETH after the Constantinople hard fork (ConsenSys 2019).

Another source of miner compensation is ensured by a *gas* model used by Ethereum. Every operation requires computational resources that cost *gas*, where *gas* is paid in ETH. The reward for the miners is charged as a transaction fee, which is based on the computational costs of executing the code. ETH can be seen as the fuel for operating the Ethereum platform (Wöhrer and Zdun 2018), and every transaction has a predefined amount of *gas* that is consumed per computational step. First, a user determines the maximum amount of *gas* she or he is willing to pay, which is known as the *gasLimit*. Next, the *gasPrice* is set by a user who defines the price per unit of *gas* a user is willing to pay. The total ETH costs of a transaction are calculated by multiplying two factors: *gasUsed* and *gasPrice*. If $gasUsed < gasLimit$, the difference will be refunded to the sender of the transaction. By setting a higher *gasPrice*, a transaction is more likely included in the next block by a miner. A second purpose of the *gas model* is an orderly handling of the computational resources. The higher the computations, the more gas is needed, but a user will not flood the system with useless operations unless he is willing to pay for it. This avoids the possibility that the network is jammed by denial-of-service (DoS) attacks, where time-consuming computations by users could lead to an overwhelmed network. It also encourages developers to write quality smart contracts by avoiding poorly programmed and wasteful code[2] (Ethereum Homestead 2020, Wood et al. 2014).

2.3 DeFi

DeFi refers to an open financial ecosystem, also called Open Finance. This financial ecosystem is generally based on open protocols, cryptoassets and DApps, which are based on smart contracts built into a network such as Ethereum. Defi Pulse provides a useful overview of current popular DeFi applications and protocols, which are classified in main categories such as *decentralized lending platforms*, *decentralized exchanges* (DEXs), *decentralized derivatives*, *payments* and *assets*, where Melon is assigned to category *assets* (DeFi Pulse 2019). As of August 2020, the total value locked (TVL) in DeFi applications and protocols amounts to $5.82 billion USD (DeFi Pulse 2020).

An important characteristic of Ethereum is its high interoperability, which enables the selecting and assembling of various building blocks

[2] A more detailed literature to Ethereum and its functionalities is provided by Antonopoulos and Wood (2018)

of a system in multiple combinations (often compared to Lego pieces). A useful illustration of the architecture of the DeFi building blocks is provided in a hierarchical multi-layered framework by Schär (2020).

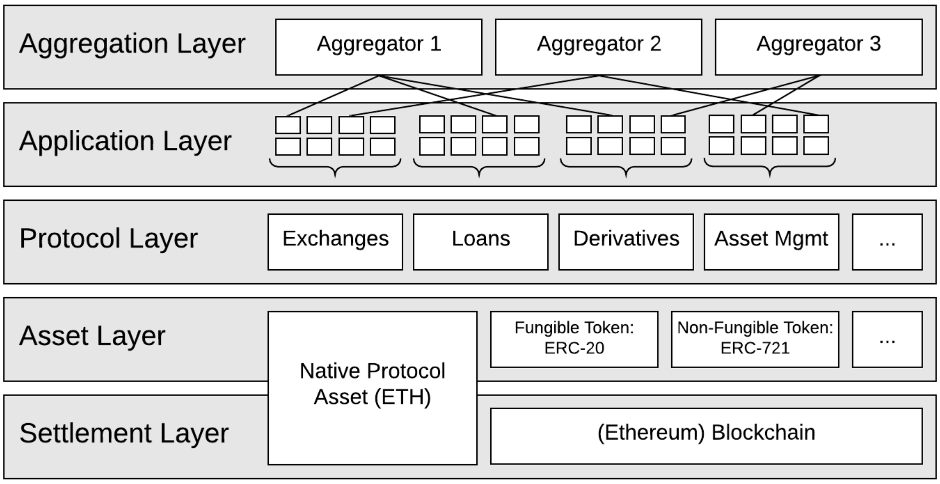


Figure 1: The Decentralized Finance Stack. Taken from Schär (2020).

Figure 1 illustrates five different layers having their own purposes. The blockchain and its native protocol asset build the *settlement layer*. The *asset*, *protocol*, *application* and *aggregation* layers are hierarchical, building on the *settlement layer*. Hierarchical means that the following layers are only as secure as the layers below.

The native protocol asset as well as any additional tokens issued on top of the *settlement layer* are part of the *asset layer*. The *protocol layer* provides standards for specific use-cases such as DEXs or on-chain asset management. The connection from DApps to individual open protocols (both based on smart contracts in the case of Ethereum) is provided by applications created on the *application layer*. The connection is usually abstracted by a web browser-based front end. And at the top of the architecture, the *aggregation layer* has the purpose to connect several applications and protocols to combine relevant information in a clear and concise manner (Schär 2020).

To summarize, numerous financial solutions can be found by connecting open protocols of the *protocol layer* built upon the same public blockchain. The Melon Protocol has integrated several open protocols such as the 0x Protocol and the Kyber Network Protocol (see Section 3.2).

Melon users can access the Melon Protocol through a web browser-based user interface, also referred to as the Melon Terminal (see Section 4.2). The Melon Terminal is a DApp of the *application layer* that facilitates the use of the Melon as well as the integrated protocols. All of these open protocols can be used repeatedly for further combinations to create new projects and expand the DeFi ecosystem.[3]

3 Asset Management

The first part of Section 3 provides an overview of the ecosystem of the traditional asset management to better understand the purposes and intentions of Melon. Melon, powered by the Melon Protocol, is an on-chain asset management infrastructure for an alternative financial ecosystem. It aims to combine the managing of assets with the technology of blockchain. This chapter especially covers relevant areas of traditional asset management, such as setting up and managing a fund, the need for third parties and the role of the regulatory framework, all of which potentially may be impacted by some of the key features provided by the blockchain technology. The second part of Section 3 is about the identification of crucial characteristics of Melon for the Section 4.

[3]However, there is still a lack of interoperability across blockchains. Projects such as Cosmos or Polkadot are trying to achieve full interoperability to interconnect various blockchains.

3.1 Traditional Asset Management

Assets and their management are a strong and an important pillar in a modern economy. Due to the intrinsic complexity of the term *asset management*, there exist numerous definitions. Fausch and Ankenbrand (2019), however, define asset management as "the production and management of investment solutions in the form of collective investment schemes or individual, institutional mandates. The key role for the asset management industry is to provide a link between investors seeking appropriate savings vehicles and the financing needs of the real economy". Vanini (2019) defines asset management as a systematic process of analyzing, trading, lending and borrowing assets of all types throughout the cost-efficient and compliant life cycle, whereas pension funds, institutional investors or private investors are different users of the asset management process. In other words, the asset management process is structured to provide investment solutions to achieve an effective and efficient management of the underlying assets, either by a private person, whose decisions about the management of the underlying assets are made by the owner of the assets, or by the corresponding asset managers, who are acting as a third party on behalf of their clients' assets.

Third party-managed portfolios are either mutual fund companies or discretionary mandates, where in a mandate, the owner of the assets delegates the investment decision to the corresponding asset manager (Vanini 2019). Today, assets under management (AuM[4]) are increasing, and the asset management process is becoming ever more important. PWC (2018) expects that the global AuM will exceed $145 trillion USD by 2025[5], up from $ 98 trillion USD in 2017. PWC also anticipates that the fastest asset growth from 2020 to 2025 will be in developing markets, such as Asia-Pacific at 11.8%, followed by Latin America at 10.4% and Middle East and Africa at 9.5%.

[4]"AuM refers to the market value of assets that an investment company manages on behalf of investors" (Vanini 2019).

[5]"If current growth is sustained, the industry's penetration rate (managed assets, as a proportion of total assets) will expand from 39.6% in 2016 to 42.1% by 2025" (PWC 2018).

To distribute risk, assets are combined into a collective system, where investors can buy and sell shares in funds of mutual funds, ETFs or hedge funds (Vanini 2019). When setting up a fund, it is necessary to hire or outsource independent duties to other third parties in the asset management process, which leads to additional costs for the investors. Intermediaries are attempting to secure the process in a given regulated framework. For example, to ensure a safe clearing and settlement of a trade, a specialized third party, known as a central securities depository (CSD), is needed. To facilitate the settlement of financial assets, other third parties – such as brokers, custodians and payment agents – are involved, which leads to a two-day settlement cycle in the European Union, Hong Kong and South Korea and even to a three-day settlement cycle in the United States, Canada and Japan (Peters and Panayi 2016). Further, a fund administrator provides accounting services such as the portfolio accounting and reporting or the calculation of fund metrics, for example, the net asset value (NAV), gross asset value (GAV) or management and performance fee. Some other key third parties in the asset management process are auditors, reporting agents and back offices.

3.2 An Introduction to Melon

Melon, powered by the Melon Protocol, has the goal to be a "viable, low-cost alternative to the current fund management ecosystem, which has evolved similarly across most legal jurisdictions" (Melon 2020*i*). In other words, it aims to serve as an alternative infrastructure to the traditional asset management infrastructure.

In Melon, anyone is enabled to set up, manage and maintain a new class of investment funds, referred to as technology regulated and operated funds (TROFs). Through the leverage of decentralized technologies, a TROF is established as a series of smart contracts and managed as well as maintained fully on-chain. All of these individually deployed smart contracts are linked together and make up the TROF, whereas individual smart contracts replace intermediary functions required in traditional finance with a code. A TROF is managed by a manager who trades the underlying cryptoassets, whereas an investor can invest in a TROF (Melon 2020*e*). Hereafter, Melon fund is used as a synonym for a TROF (Melon 2020*c*).

Further important components for the maintenance and development of Melon are the Melon community, with various stakeholders such as developer teams, integrated protocols and the Melon asset universe, which are collectively referred to as the Melon Ecosystem. We will introduce the Melon Ecosystem in the next section.

3.2.1 Melon Ecosystem

Melon Community

The Melon community is connected in channels such as Telegram, Twitter and Reddit. With the vision of a decentralized ecosystem, various stakeholders of the community – including Melon users, token-holders and maintainers/developers – are essential for the project. The goal is that the global community promotes the project, representing Melon properly and accurately across the DeFi ecosystem (Zenk 2019*b*).

Developer Teams

Independent teams of developers associated with Melon and working on or building projects on top of Melon include Avantgarde Finance, Midas Technologies and Gorilla Funds. Avantgarde Finance is the lead developer of the Melon Protocol. It proposed a Melon Funding Proposal (MFP[6]), which was accepted by the Melon Council decentralized autonomous organization (DAO) in September 2019 (Melon 2020*a*) (For more about the Melon Council DAO, see Section 4.3). The team got the leadership for the next three years and is aiming to facilitate on-chain asset management and create a better everyday experience for aspiring asset managers and their investors (Zenk 2019*b*). The Swiss-based com-

[6]Teams and projects apply for a funding. If the application is accepted by the Melon Council DAO, teams and projects will be funded with MLN (see Section 4.1.3) (Zenk 2019*c*)

pany Midas Technologies works on the development of an investment mobile app – known as Ash – which allows retail investors to invest in funds through a mobile phone (Melon 2020*a*). Gorilla Funds is a project that has the goal to make investments and redemptions in Melon funds as simple as possible. Every Melon fund manager has its own page on Gorilla Funds with informations they think are relevant for potential investors (e.g., informations about themselves, their qualifications and their investment strategy) (Gorilla Funds 2019).

Protocol Integrations

The Melon Protocol has integrated the following DEX protocols: OasisDEX, 0x v2.1, 0x v3.0, Kyber Network and Uniswap v1. The protocols 0x v3.0 and Uniswap v1 were integrated with the protocol upgrade to v1.1. The DEX protocols enable Melon managers to trade through an aggregated order book, which Melon is providing on its web browser-based front end (see Section 4.2) (Melon 2020*a*).

Melon Asset Universe

The Melon asset universe defines a set of cryptoassets that a Melon manager may trade on the various DEXs. With the upgrade of the Melon Protocol to v1.1, the cryptoassets Aragon Network Token (ANT), Chainlink (LINK), Decentraland (MANA), iExec Token (RLC), Multi-Collateral DAI (DAI) and Ren (REN) were added to the Melon asset universe. The Melon asset universe includes the following ERC20 cryptoassets:

Melon Asset Universe	
Wrapped Ether (WETH)	Augur Reputation Token (REP)
Melon Token (MLN)	USD Coin (USDC)
Maker Token (MKR)	Chainlink (LINK)
Wrapped Bitcoin (WBTC)	Aragon Network Token (ANT)
Kyber Network Crystal (KNC)	MANA Token (MANA)
ZRX Token (ZRX)	Ren Token (REN)
Basic Attention Token (BAT)	RLC Token (RLC)
Single-Collateral DAI (SAI)	Multi-Collateral DAI (DAI)

Table 1: Melon Asset Universe. Source: (Melon 2020*a*).

4 The Melon Protocol

Melonport AG, founded in 2016, is the company that developed the first version of the Melon Protocol. In early March 2019, Melonport released v1.0 of the Melon Protocol – which is referred to as Zahreddino – and handed over its governance to a decentralized governance system. After that transfer, the company dissolved itself (Zenk 2019*a*, Melon 2020*f*). In February 2020, the Melon Protocol v1.0 was upgraded to v1.1.

In this section, we discuss the technical architecture of the Melon Protocol v1.1, building on the introduction of the *settlement layer* in Section 2. Section 4.1 includes a detailed description of the Melon Protocol architecture in the *protocol layer*. Technically speaking, the Melon Protocol is a set of smart contracts that allows Melon managers to emulate an investment fund for on-chain cryptoassets (Zenk 2019*a*).

The Melon Protocol can be seen as having two layers based on different smart contracts. One is the *infrastructure layer* managed by the governance system; this layer is essential for all the Melon funds in the *fund layer*, which are controlled by their respective managers and are participated in by investors (Melon 2020*i*). We discuss in detail the functions and interactions of the various smart contracts of both layers and explain the (economic) purpose of the individual smart contracts. Major changes due to the protocol upgrade are discussed in the respective contracts. Additionally, we review the role of the MLN token. Furthermore, in Section 4.2, we will explain the web browser-based front end of Melon, called Melon Terminal v2.0. Finally, Section 4.3 explains the duties and the influence of the Melon Council.

4.1 Protocol Layer

To begin, Figure 2 provides a brief overview of the architecture of the Melon Protocol. The dotted line represents a Melon fund. Blue boxes outside the Melon fund are contracts of the *infrastructure layer*, grey boxes inside the Melon fund are contracts of the *fund layer* and white boxes refer to investors and DEXs. The black arrows indicate in a simplified form how the contracts, investors and DEXs are connected. The red text and arrows refer to the most important functions of the *fund layer* contracts.

A Melon fund has an architecture of a Hub and several Spokes. The Hub contract is in the center and all the components as Spokes are connected to the Hub and replace third parties. The contracts of the *infrastructure layer* provide important services to the contracts of the *fund layer*.

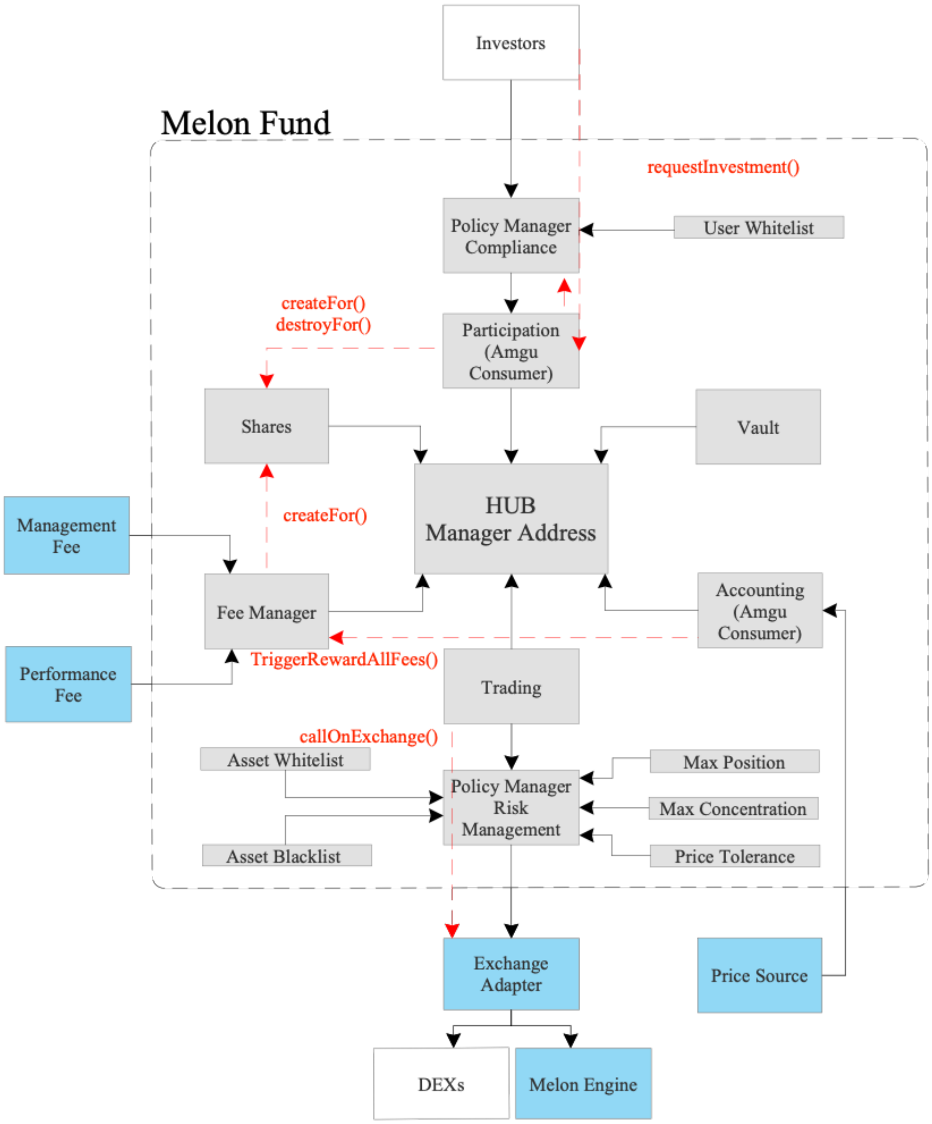


Figure 2: The Melon Protocol Architecture. Own illustration.

The Melon Protocol in the *protocol layer* consists of numerous smart contracts that play an essential but a different role for the robustness of the Melon infrastructure. In the following sections, we examine in detail the architecture illustrated in Figure 2. Table 2 provides an outline of the smart contracts belonging to the corresponding layers.

Infrastructure Layer	**Page**	**Fund Layer**	**Page**
Registry.sol	15	Hub.sol	22
Version.sol	16	Participation.sol	22
FundFactory.sol	16	Accounting.sol	24
ExchangeAdapter.sol	17	FeeManager.sol	24
PriceSource.sol	17	Shares.sol	25
ManagementFee.sol	18	Vault.sol	25
PerformanceFee.sol	19	Trading.sol	25
Engine.sol	19	PolicyManager.sol	26
AmguConsumer.sol	16,20,23,24		

Table 2: Smart Contracts of the *Infrastructure* and *Fund Layer*. Source: Melon (2020*i*).

4.1.1 Infrastructure Layer

The *infrastructure layer* consists of smart contracts that provide important services and functions to the Melon funds. The Melon Council operates and deploys contracts in this layer, and these contracts are deployed once for each version of the Melon Protocol. In the following, we will discuss important smart contracts of the *infrastructure layer*.

Registry
The Registry stores, manages and provides functionalities to maintain registered cryptoassets and DEXs by adding, updating or removing them (Melon 2020*k*). Furthermore, it provides an interface to track the asset universe or all the infrastructural contracts such as the Engine and the Price Source contract (Melon 2020*i*). Therefore, the contract can be seen as central information point for contracts of a specific Melon protocol version (Melon 2020*k*, Jacobs 2020*d*).

Version

The Version is the entry point for a Melon fund manager to create and shut down a fund. The only function the contract includes is the *shutDownFund* function to close a Melon fund. Because the Version contract of a specific Melon protocol version inherits all functions from the Fund Factory contract of that specific version, it can call functions of the Fund Factory contract. The Fund Factory includes seven factory contracts to create components of a Melon fund (see the next section). The factory contracts are exclusive to the Version contract and cannot be changed once they have been set. These factory contracts are the same for the creation of all Melon funds within the same version, but they can differ among various versions. Therefore, Melon funds can access only the features of the version on which they have been deployed. To access new features of a new Melon protocol version, a Melon manager must shut down the fund and deploy a new one (Melon 2020*l*, Jacobs 2020*f*).

Fund Factory

The Fund Factory enables managers to create their own Melon fund by executing the orderly creation of a new fund, linking all aspects of components, settings, routings and permissions. As mentioned earlier, the Fund Factory is inherited by the Version contract. It is important to understand, that the Fund Factory contract is not deployed individually.

The Fund Factory inherits from the Amgu Consumer contract. As of now, we simply must know that this contract determines additional *gas* costs to the Ethereum *gas* when calling certain functions of the Melon Protocol. A more detailed look at this is provided in the section about the Melon Engine.

As in the Melon Protocol v1.0, a Melon fund creation is a nine-step process. The first step initiates the set-up process with the *beginSetup* function. Before calling *beginSetup*, a manager must define parameters; these are the Melon fund name, external DEXs with which a manager wishes to interact, the cryptoasset in which the Melon fund is denominated and the ones the Melon fund accepts for investments. The manager also selects the fee contracts registered with his Melon fund (a management and performance fee), a *feeRate* for each fee contract and the *feePeriod* covering a performance fee measurement period. The modifier *componentNotSet* ensures that the Melon fund set-up process is an one-time execution; in other words, the parameters set cannot be modified (Melon 2020*d*).

When calling the *beginSetup* function, defined parameters as well as contracts of the *infrastructure layer* are mapped to the manager's address. Therefore, the manager has the right to use the services and functions of these contracts. The second through eight steps include the creation of the individual fund components, each of which is deployed by seven independent factory contracts. To complete the set-up process, the manager must call *completeSetup* to register the new Melon fund with the Registry contract. The second through ninth steps consume additional *gas* (Melon 2020*d*).

Exchange Adapter

The Exchange Adapter generalizes the implementation of any concrete DEX. There are different exchange adapter contracts, all of which inherit from the standard Exchange Adapter contract. These generic contracts are deployed once and can be selected by each Melon fund at the time of fund set-up. With the Melon Protocol v1.1, Melon managers now have the option to add DEXs after the fund is deployed. After the selection of, for example DEX a and b out of n DEXs, the specific Melon fund integrates the specific exchange adapter contracts of DEX a and b. The selected exchange adapter contracts serve as an interface and enable interactions between the Melon funds and the DEXs a and b. This interaction is occurring without the manager knowing the specifics of the DEXs.

The Melon Engine has its own exchange adapter, which serves as an interface from a Melon fund to the Melon Engine. It allows a Melon manager to sell MLN to the Melon Engine in exchange for ETH (see the section on Melon Engine) (Melon 2020*b*).

Price Source

The Price Source is the price reference for the contracts of the Melon ecosystem. The prices of all cryptoassets are stored in the Price Source contract and are accessible for other contracts. Generally, there is a price update once a day, except for invalid prices, where there can be several updates. As a result, the Price Source contract asks for the current prices from the Kyber Network (Melon 2020*j*).

Management Fee

The Management Fee includes the logic to calculate the management fee of each individual Melon fund. To ensure a fair basis for the calculation of the performance fee, the management fee must always be calculated first. Regardless of the performance of the fund, management fee rates are paid annually. If, for example an investor redeems his cryptoassets, the weighted proportion of a year that has passed is at the same time the basis to weight the management fee of that specific Melon manager. This weight is multiplied by the number of fund shares managed by the Melon manager since the beginning of this period. This allows the investor to redeem his cryptoassets, while a Melon manager automatically receives his rewards. It is important to understand in the coming discussion in Section 4.1.2 that a Melon manager is rewarded in fund share tokens [7] and not in cryptoassets managed by the fund. When claiming management fee rewards, the holdings of the investor's quantity of fund shares remains constant. As a consequence, the value of each fund share will decrease.

To calculate the management fee, the time-weighted, pre-dilution share quantity must be known. It is defined as follows:

$$PDf = (Tn)\left(\frac{te}{ty}\right)(fm) \tag{1}$$

Where PDf is the pre-dilution quantity of shares, Tn is the number of current outstanding shares, te is the number of seconds elapsed since the previous conversion event, ty is the number of seconds in a year and fm is the management fee rate. The calculation of the number of shares

[7] Share tokens represent a proportional portion of a Melon fund (see Section Shares Component)

that must be created to cover the management fees earned during the conversion period is defined as follows:

$$SMFe = \frac{PDf\ Tn}{Tn\ \text{-}\ PDf} \tag{2}$$

Let us assume the management fee rate is set to 2%, and 100 shares are managed for six months by the Melon manager. This results in 1.0101 fund shares for the manager (Melon 2020*c*, Orthwein 2018).

Performance Fee

The Performance Fee includes the logic to calculate the performance fee of each individual Melon fund. The performance fee accrues over time and depends on the performance of the Melon fund. The calculation of this fee is based on the difference between the NAV per share (net of management fees) and the high-water mark (HWM). A performance fee is rewarded if the difference to the HWM is positive. The HWM is the maximum share valuation of all measurement periods at the end of each *feePeriod*.

Let us assume a manager has set the *feePeriod* parameter to 90 days, but an investor wants to redeem his cryptoassets after 45 days. In this case, if the current NAV exceeds the fund HWM, a performance fee is due. The fee is weighted by the redemption share quantity's proportion to the Melon fund's total share quantity (Melon 2020*c*).

Melon Engine: Buy-and-Burn Model

The Engine collects ETH paid by Melon users who have called a payable function of the Melon Protocol. This ETH are additional costs to the Ethereum *gas* costs. Melon funds that hold MLN tokens have the option to offer the Melon Engine those MLN tokens to buy the collected ETH, where if some conditions are met, the Melon manager will get a premium-adjusted amount of ETH. Melon Engine will send the purchased MLN tokens to an inaccessible address to burn them and to further decrease the supply of MLN. The idea behind the buy-and-burn model is to create incentives for MLN holders by decreasing the total supply of MLN tokens when the Melon network is used (El Isa 2018). A more detailed look at the model is provided in the following.

Payable functions of the Melon Protocol are the second through ninth steps in the set-up process, the function *requestInvestment*, *cancelRequest*, *executeRequest* (see the section on the Participation Component) and *triggerRewardAllFees* (see the section on the Accounting component). These functions have implemented the *amguPayable* modifier, which is defined in the Amgu Consumer contract. This contract determines the initial gas amount, called *asset management gas* (amg), for the payable functions. The total amg costs of payable functions are calculated by multiplying two factors: *amguUsed* and *amguPrice*. *Asset management gas unit* (amgu) refers to the units of amg, whereas *amguPrice*, which is denoted in MLN, defines the price per unit of amg. Other than in Ethereum, a user cannot define the price per unit of amg a user is willing to pay. The amgu price is fixed but is adjustable by the Melon Council (see Section 4.3). As of now, we have the amg costs in MLN, but payable functions are paid in ETH. Price Source retrieves the prices of ETH and the MLN token from the Kyber Network to determine the amount of ETH required for the payment of the calculated amg (Melon 2020*h*).

There is a consecutive 30-day freeze period maintained by the Engine contract, where the collected ETH is frozen. When that freeze period ends, the frozen ETH will be liquidated. Depending on the amount of liquid ETH, the Melon Engine applies a premium (Melon 2020*h*).

- Ether in the Melon Engine $<$ 1 ETH, premium $=$ 0 %
- $1 \leq$ Ether in the Melon Engine < 5, premium $=$ 5 %
- $5 \leq$ Ether in the Melon Engine < 10, premium $=$ 10 %
- Ether in the Melon Engine ≥ 10, premium $=$ 15 %

The number of ETH a Melon fund manager will receive by selling xMLN tokens to the Melon Engine is defined as follows:

$$ETH = xMLN \cdot ETH/MLN\ Price \cdot (1 + premium) \qquad (3)$$

Melon fund managers now have the opportunity to sell MLN to the Melon Engine to obtain ETH with a potential adjusted premium. Only Melon managers can trade with the Melon Engine. This is ensured by allowing only the Vault contract to call the *sellAndBurnMLN* function. This function triggers the function *ethPayoutForMlnAmount* and sends the (premium-adjusted) quantity of ETH to the corresponding Melon fund for receiving the previously determined quantity of MLN tokens. In a final step, the Melon Engine will permanently and irreversibly send these purchased tokens to an inaccessible address (0x0000000000000000000000000000000000000000). This process is known as the *burning* of MLN tokens, which will lead to a decrease in the total MLN supply. An overview of the process is depicted in Figure 3 (Melon 2020*h*):

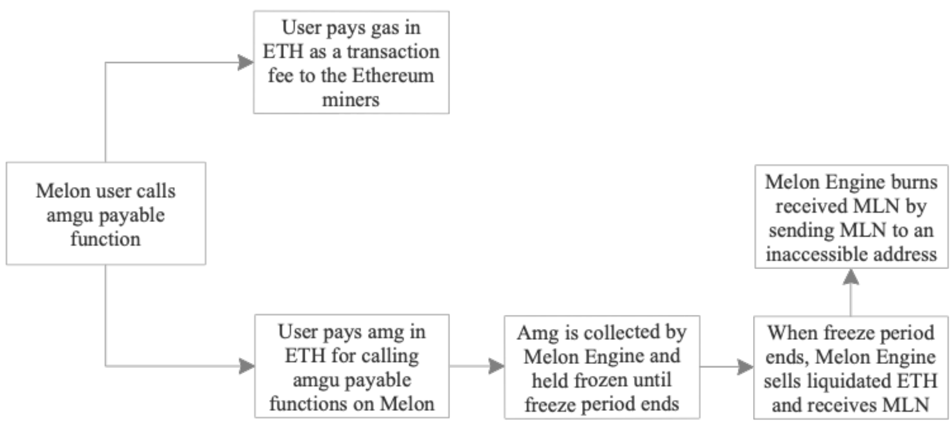


Figure 3: Melon Engine Process. Own illustration.

4.1.2 Fund Layer

A Melon fund has an architecture of a Hub and Spokes. The Hub smart contract is in the center, and all of the components deployed individually over several steps are connected to the Hub as Spokes. These components replace third parties, so that the duties of a central party are taken over by a code. The individual Melon manager can add and choose his own tools and parameters within the individual components (e.g., fee structure; trading exchanges; risk management tools; compliance tools; or the asset universe). The Hub and the several Spokes are explained in detail in the following sections.

Hub
The Hub is the core contract of each individual Melon fund. This contract maintains the specific Spokes of the fund in terms of their permissions. When calling the *beginSetup* function, the individual Hub is created with the account of the fund manager as the deployer. The address of the fund manager is permanently stored in the Hub. After deploying, that Hub creates a link to the specific Spokes, which correspond to that specific Hub and are registered with it. Later on, functions are called within the Spokes. If a Spoke wants to interact with another Spoke, the Hub knows about the allowed interactions between the corresponding Spokes. There are individual Spokes that have programmatic access to information in specific other Spokes. This permits only specific Spokes to invoke specific functions on specific target Spokes. The centralized architecture ensures that the Hub knows everything about all of the components. In the following, we will discuss the individual Spokes (Melon 2020*c*, Jacobs and Casey 2020*a*).

Participation Component
The Participation organizes the entire Melon fund interface with investors and manages or delegates all functionality pertaining to fund investment and redemption activities. When compliance is ensured (see the section on Policy Manager), the first component with which external investors interact is the Participation component (Melon 2020*c*).

The Participation component knows about the cryptoassets that are allowed for investments in a Melon fund. A manager can define which cryptoassets are enabled (disabled) for investments in the own Melon fund. Subsequently, investors can request an investment with the respective cryptoasset(s) by calling the *requestInvestment* function. The Participation component inherits the Amgu Consumer contract, and both function calls charge amg. Investment requests can either be executed or cancelled, with both scenarios requiring amg as well. The *cancelRequest* function can be invoked if at least one of the following conditions is met: invalid investment cryptoasset price, expired request or the fund is shut down. If an investor wishes to request an investment, that investor must specify an investment price. The investor must wait for the next price feed update before the action is executed. If the price of the cryptoasset exceeds the investor's request price after the next price feed update, the investment execution will fail. With regard to the expiry period, the duration is set to one day (Melon 2020*c*, Jacobs 2020*c*).

When an investment is executed, the *executeRequestFor* function ensures that several conditions are met; for example, the invested cryptoasset has a valid price and the fund is not shut down. The final step for a successful investment includes the transfer of the invested cryptoasset to the Vault contract and the receipt of the commensurate quantity of newly created Melon fund share tokens by the investor. These tokens represent the proportional holdings of the fund's cryptoassets. Functions of the Shares component as well as the share token are discussed in the section on the Shares component (Melon 2020*c*, Jacobs 2020*c*).

After an investment is executed, investors can redeem at any time. The function *redeemQuantity* is invoked by the *redeem* function. Before a share-commensurate quantity of all cryptoassets of a fund is transferred to the investor, the function *redeemWithConstraints* ensures that the management and performance fees have been calculated and allocated to the manager's address. Depending on the number of share tokens an investor holds, he will receive a proportionate amount of the underlying cryptoassets of the fund, even if he had invested with another cryptoasset.

In a final step, the investor's share tokens will be destroyed, and the investor will receive his cryptoassets (Melon 2020*c*, Jacobs 2020*c*).

Accounting Component

Accounting defines the accounting rules implemented by the fund. These are, for example, the calculation of fees, GAV/NAV and the pricing of cryptoassets. The component aims to replace the administrator as a third party of the traditional asset management industry. The AuM of a Melon fund is calculated by using prices of the Price Source contract. This enables managers and investors to access various metrics. It is important to understand that these metrics are not calculated in real time. As mentioned earlier, the Price Source contract updates the prices once a day, so the calculations are based on this daily valid prices.

When a manager claims his rewards, the *triggerRewardAllFees* function of the Accounting component triggers the *rewardAllFees* function of the Fee Manager component. *TriggerRewardAllFees* inherits from the Amgu Consumer contract and calculates Melon gas. The function *rewardAllFees* is discussed in the next section (Melon 2020*c*, Jacobs 2020*a*).

Fee Manager Component

Fee Manager manages the management and performance fees and is responsible for the calculation of the different fees as well as the correct execution order of the fee contracts. As mentioned earlier, the fee parameters *feeRate* and *feePeriod* are established before the *beginSetup* function is called. A manager will first receive management fees for the management services rendered. Subsequently, the manager is rewarded for any performance achievements with a performance fee.

Triggering *rewardAllFees* by calling the *triggerRewardAllFees* function initiates the claiming of accrued fees for the manager. *RewardAllFees* triggers the calculation functions of the management and performance fee contracts to calculate the fees. Consequently new share tokens in the Shares component are created and allocated to the manager's account. Thus, positions of a Melon fund do not have to be liquidated to pay fees (Melon 2020*c*, Jacobs 2020*b*).

Shares Component
Shares manages the shares of a Melon fund that represent the proportionate holdings of an investor. To represent ownership, Melon uses a token referred to as a share token. The fungible ERC20 token is not tradable, so the Melon fund manager knows about his investors. A market price fluctuation of the underlying cryptoassets within a Melon fund will adjust the price of the share tokens of that Melon fund (Melon 2020*c*).

Share tokens are created when the Participation or Fee Manager components call the *createFor* function of the Shares component, which then calls the internal function *mint*. A proportionate quantity of share tokens is created by either an investment in a fund or the payment of management and performance fees. After the *share* tokens are issued, the invested cryptoassets are transferred to the Vault. For redeeming the underlying cryptoassets, the Participation component calls the *destroyFor* function of the Shares component, which then calls the internal function *burn* to burn the corresponding *share* tokens (Melon 2020*c*, Jacobs 2020*e*).

Vault Component
The Vault stores the cryptoassets of a Melon fund and serves as a custodian, and the proportionate ownership of an investor is represented by the share token. The Trading component is delegatecalled to increase or decrease the asset balance for trading. If an investor wants to withdraw cryptoassets from the Vault, the *withdraw* function must be called (Melon 2020*c*, Jacobs and Casey 2020*c*).

Trading Component
Trading serves as an interface and allows Melon fund managers to interact with DEXs integrated with the Melon Protocol as well as with the Melon Engine. The function *callOnExchange* is the general interface of the Melon fund to registered DEXs for trading cryptoassets. It allows for the call of exchange functions through adapters. A Melon manager is only allowed to call the *takeOrder* function on the Melon Terminal v2.0. Trading is delegatecalled to increase or decrease the balance of the Vault component. However, before specific trading interactions are allowed,

this function calls the Policy Manager component to ensure that the exchange trade adheres to the policies configured for the Melon fund (Melon 2020*c*, Jacobs and Casey 2020*b*). Depending on the policies, an action will return *true* for allowed actions and *false* for disallowed actions; these are discussed in the next section.

Policy Manager Component

Policy Manager manages the various risk management and compliance policies. Risk management policies regulate the trading of fund positions and compliance policies pertain to investor investments. Each manager can set risk engineering rules regarding investment guidelines and compliance rules regarding investor participation. These policies are individual contracts and are registered with the Policy Manager component.

All contracts, with the exception of the User Withelist, govern the risk management of fund positions. The User Whitelist contract sets the compliance rules of a Melon fund; this contract enables a Melon manager to define specific investor addresses which are then allowed to invest or prevented from investing in a specific Melon fund. A manager can remove addresses from the User Whitelist. This will not affect an invested address' current invested status or ability to redeem, but will prevent future investments (Melon 2020*c*).

Risk management policies include the contracts Max Position, Max Concentration, Price Tolerance, Asset Whitelist and Asset Blacklist. An Asset Whitelist specifies the cryptoassets in which a Melon fund will ever be able to invest in; conversely, an Asset Blacklist specifies the cryptoassets in which a Melon fund will never be able to invest in. Managers are able to add further cryptoassets to the Asset Blacklist, but they cannot remove those they have put on the list. Cryptoassets added to the Asset Whitelist can be removed, but new cryptoassets cannot be added (Melon 2020*c*). Max Concentration prevents a Melon manager from investing a large proportion of the funds in a single position. If the parameter is set to 20%, the manager is allowed to invest, at most, 20% of the funds in one position. Max Position defines the quantity of cryptoasset positions in which a manager can invest. If the parameter is set to two, that specific

Melon fund can contain, at most, two different cryptoassets. To prevent a high deviation between the order price and the current reference price as provided by the price feed, a price tolerance is specified. If the parameter is set to 5%, the order price may deviate, at most, 5% from the last price feed update provided by the Kyber Network. If the deviation is higher than the set parameter, the trade is not permitted and the transaction is reverted. The various Policies are listed in the following:

Policies	**Description**
Max Position	maximum number of positions a fund may have
Max Concentration	a position (or positions in a single category) cannot actively exceed x% of the NAV
Price Tolerance	order price is not allowed to deviate more than x% from a reference price provided by the price feed
Asset Whitelist	list of cryptosassets eligible for investments by a manager
Asset Blacklist	list of cryptoassets that a manager may not hold or invest in
User Whitelist	list of investors which are permitted to invest in the fund

Table 3: Policies. Source: Melon (2020*c*).

4.1.3 Melon Token

Melon (MLN) is the native token of the Melon Protocol. The only utility MLN provides is to buy ETH from the Melon Engine, as described in the section on the Melon Engine. However, there are others who hold the MLN token to participate in the Melon infrastructure.

Every year, the Melon Protocol mints a fixed number of 300,600 MLN tokens, which are added into a pool, called the Melon inflation pool. In January 2019, the initial supply of MLN was 932,613. In March 2019, after the minting of 300,600 MLN tokens in the first year, the total supply was 1,233,213. At the time of this writing in August 2020, the supply of

MLN tokens is 1,524,127 (Etherscan 2020). There are two reasons for the declining number of MLN tokens after the minting in Year 2 (1,524,127 – 1,233,213 $<$ 300,600). First, as mentioned earlier, approved proposals by the Melon Council DAO are funded through this Melon inflation pool so that they will be supported. Furthermore, Melon Council members are also rewarded by this pool. However, unspent tokens can be burned by the Melon Council at the end of the year, which leads to a decline in the total MLN token supply. Second, the Melon network is based on a gas model, as has been demonstrated with the buy-and-burn model in the section on the Melon Engine (Zenk 2018).

4.2 Application Layer

This section discusses the Melon Protocol in the *application layer*. As a web browser-based front end, the Melon Terminal v2.0 serves as an interface to ensure easier use of the Melon Protocol.

4.2.1 Melon Terminal

In March 2019, together with the Melon Protocol v1.0, the v1.0 of the Melon interface was released. The interface is a desktop application and is called Melon Manager Interface (MMI) (Zenk 2019*a*). In February 2020, with the release of the Melon Protocol v1.1, the Melon Terminal v2.0 was released by Avantgarde Finance. The new user interface runs on the new protocol version and enables managers as well as investors to access the Melon Protocol through a web browser-based front end. Melon Terminal v2.0 is rebuilt from the ground up and has the goal to make the interface much faster, more stable and easier to use compared to the MMI (Avantgarde Finance 2020).

Managers and investors will need to have an Ethereum account and some ETH to pay for Ethereum *gas* and amg. The account can be connected to, for example, Metamask, Frame or Fortmatic (Melon Terminal 2020).

The Melon Terminal provides functionalities for managers to set up and manage a Melon fund and enables them to interact with several DEX protocols. Melon Terminal allows only funds to be managed that are not shut down and run on the newest Melon protocol version. The only function that the Melon Terminal provides for funds deployed on an older version is to redeem the invested cryptoassets. Further, the manager has an overview of the managed fund, proportions of fund shares, pending investment requests and the account balances of the invested cryptoassets (Melon Terminal 2020).

The Melon Terminal provides transparency about on-chain fund metrics, so investors can look for a Melon fund that meets their needs. But investors must take into account that the calculations of the metrics are based on the daily update prices and are not made in real time.

Furthermore, the trade and investment history of each Melon fund is recorded, and the set policies and parameters are viewable. For investments in WETH, the Melon Terminal provides a function to wrap ETH to WETH (Melon Terminal 2020).

The Melon Terminal provides various types of trading. The first is the trading on decentralized order book exchanges, for example, trading on 0x v2.1 or 0x v3.0. As mentioned earlier, a manager can only call the *takeOrder* function. Therefore, a manager can only choose an offer from the order book. A manager can also trade on smart contract-based liquidity pools such as on Uniswap v1 or on reserve aggregation protocols, for example, on the Kyber Network. A more detailed description of these approaches can be found in Schär (2020).

4.3 Governance

The objective of the Melon governance framework is to provide a decision-making process to develop and maintain the Melon Protocol. The Melon governance system is referred to as the Melon Council, which is responsible for protocol upgrades, resource allocations and adjustments of network parameters.

The council is a DAO, whose operations are powered by aragonOS[8] (Melon 2020*f*). A DAO is a new type of organizational design to automate governance and decision-making. This alternative is based on a code, commonly on a series of smart contracts, that have implemented governance rules into its code (Jentzsch 2016).

The Melon Council consists of five Melon Technical Council (MTC) members and two Melon Exposed Business (MEB) representatives. MEB are persons, who must "either (i) run more than 1% of the total assets under management on the Melon protocol and measured in ETH terms, and (ii) individually or as part of its business provably be invested in a Melon fund or exposed to Melon" (Melon 2018). The MEB has the opportunity to elect two MEB representatives into the Melon Council, which represent the Melon users. The MTC provides technical expertise with regards to the Melon Protocol. New MTC members can be elected by a two-thirds majority of the Melon Council (Melon 2018).

The Melon Council is based on a multisig[9] schema. In a vote, a majority of more than 50% of the votes is required.

The only parameter the Melon Council can adjust is the amgu price. Only two scenarios can lead to an adjustment of the amgu price: first, if there is an overnight spike in network usage, and second, if the MLN/USD price is extremely volatile (Melon 2020*h*).

4.4 A Review and Remarks on the Melon Protocol

After introducing the Melon infrastructure in Section 3.2 as well as discussing the *protocol layer*, *application layer* and the governance framework of the Melon Protocol in Section 4, we want to briefly summarize the technical upgrades of the Melon Protocol.

[8]For further informations on aragonOS: https://hack.aragon.org/docs/aragonos-intro

[9]In a multisig schema, multiple keys are required to authorize a transaction (Berentsen and Schär 2017, pp. 184).

The major changes are the new cryptoassets added to the Melon asset universe, the new DEX protocols integrated with the Melon Protocol and the release of the Melon Terminal v2.0. Another of the few technical changes is that Melon funds now have the possibility to add DEXs after the Melon fund is created.

Before continuing with the economic part of this thesis, we offer our concluding remarks on the technical part. We noticed that the policies are still deployed individually for each Melon fund. When examining the fee contracts, it can be seen that the contracts are deployed on the *infrastructure layer*, and each individual Melon fund can make use of them. Deploying the policies on the *infrastructure layer* would reduce the number of contracts for each Melon fund as well as the costs for the deployment of a Melon fund. As a consequence, the usage of the Melon Protocol would be more user-friendly for a Melon manager.

We consider the Price Source contract as a critical component. Since Melon relies on the Kyber Network, it has a centralized point of failure. Such a case was seen in December 2019 as a result of Ethereum's Istanbul hard fork (Melon Council 2019), where the Price Source contract did not receive valid prices. As a consequence, managers were unable to trade, and investors were unable to invest. We also consider the daily price updates to be critical. On the one hand, prices and metrics are not available in real time on the Melon Terminal, and on the other hand, investors cannot withdraw their deposits at any time, since they must wait for the next valid price update from when they made an investment request.

Finally, we would like to express our remarks on the amgu price. The buy-and-burn model relies strongly on the ETH/MLN fluctuations, if the amgu price is not adjusted. We see here another centralized point of failure, because it is not predictable how much a fund costs today and how much it will cost tomorrow if the parameter is not adjusted. In the next section, will see that the amgu price was adjusted only once.

5 Economic Analysis

At this point, we have already described the smart contracts and their functions in the Melon Protocol v1.1. We have also introduced the Melon Terminal v2.0 as well as the governance framework of Melon. Building upon the previous sections, we analyze and discuss whether any economic differences arose from the upgrade of the Melon Protocol v1.0 to v1.1. We offer a descriptive analysis in the first part and a mean-variance spanning test as an empirical test in the second part.

5.1 Descriptive Analysis

The first part focuses on a descriptive analysis to examine whether the Melon protocol upgrade to v1.1 led to an increase in activity in terms of amg earned and collected by the Melon Engine, new Melon managers and Melon funds, new investors and investments as well as in terms of trading activity. Furthermore, if an increase in activity was observable, we analyze the potential drivers. It is assumed that either the new cryptoassets added to the Melon asset universe or the new DEXs integrated with the Melon Protocol are the possible drivers.

5.1.1 Data Description

This section presents the data used for the descriptive statistics. Most of the data sample used in this section is queried with the Melon API from The Graph[10]. Although the Melon Protocol updates its prices taken from the Kyber Network once a day, sometimes there are several price updates (e.g., when the price was not valid). We corrected this by taking the latest daily update. The historical daily closing prices for all cryptoassets denominated in ETH are taken from CoinGecko[11], whereas the

[10]see The Graph 2019

[11]Coingecko 2020 calculates prices based on the pairings collected from various exchanges. For calculations, they use a global volume-weighted average price formula.

volume of the MLN token in USD is taken from CoinMarketCap[12]. This section, which was prepared in Excel and processed in RStudio[13], covers the period from March 01, 2019 to July 16, 2020. We have taken into account that July 2020 is not representative for the most figures because data for only half of the month were obtained.

5.1.2 Descriptive Results

Melon Network History

Depending on how the amgu price is set by the Melon Council DAO, how the ETH/MLN price develops and how actively the Melon Protocol is used, the amount of amg that the Melon Engine earns and collects varies. Figure 4 illustrates, how much amg the Melon Engine earned and collected since the Melon Protocol was deployed to the Ethereum mainnet. Especially in June 2020 we see that the amount of amg the Melon Engine earned and collected increased.

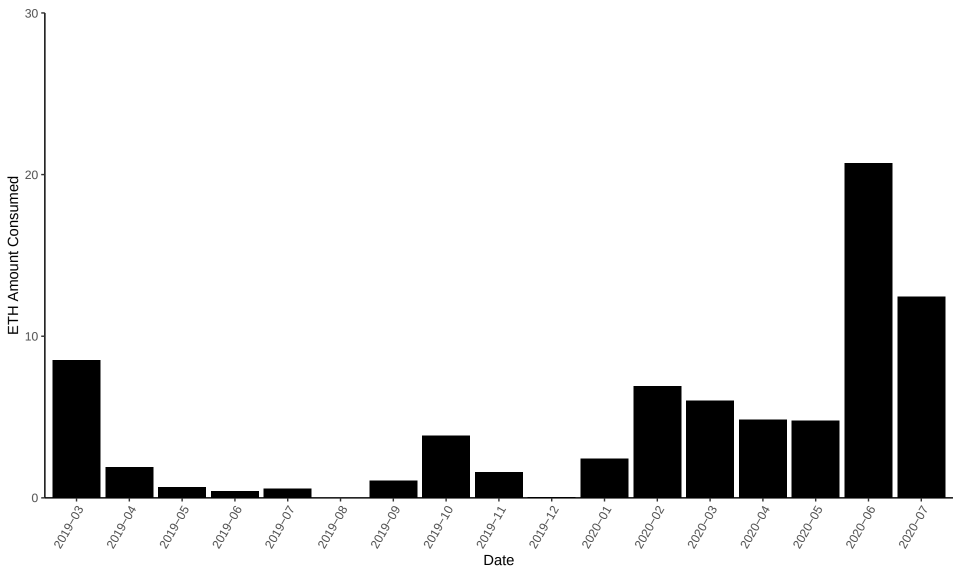


Figure 4: Amg Earned and Collected by the Melon Engine in ETH. Source: Melon API, own illustration.

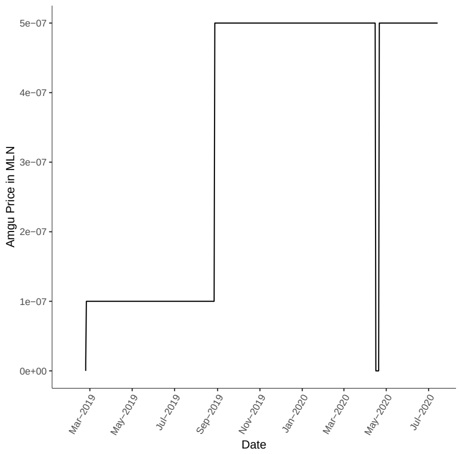
[12]see CoinMarketCap 2020*a*
[13]see RStudio 2020

In the following, we illustrate in Figure 5 four factors that have an impact on how much amg the Melon Engine earns and collects. It is legitimate to assume, that the volume of an asset is related to its price. Therefore, we want to examine in Table 4 and 5 how much the volume of the MLN token generated by the buy-and-burn model contributes to the weekly average volume of the MLN token.

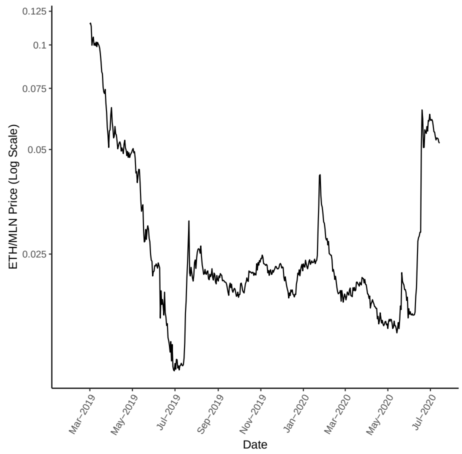
Figure 5a indicates that the amgu price has not been adjusted since August 28, 2019. From April 16 to April 20, 2020, the amgu price was at 0 for a short time, but this is ignored. Therefore, the amgu price can be seen as a constant in this thesis. Consequently, the amount of amg consumed by the Melon Engine depends on how the ETH/MLN price develops (Figure 5b) and how actively the Melon Protocol is used.

In Figure 5b we see that the ETH/MLN price exhibits enough volatility to influence the amount of ETH earned and collected by the Melon Engine. If we look at the MLN and ETH price in Figure 5c separately and at Table 12 in the Appendix, we can confirm the high correlation coefficient of 0.53. At the beginning of Figure 5c, the prices move in opposite directions. It is assumed that this has to do with the previous price increase of the MLN token before the deployment of the Melon Protocol v1.0. The prices run fairly similarly until June 2020, when there was an increase in the MLN price although the ETH price remained stable. At this point, if we look at the market capitalization of both cryptoassets, it is legitimate to assume that the MLN price follows the ETH price (Coinmarketcap 2020*a*). Therefore, it is important to examine influences on the MLN price fluctuations.

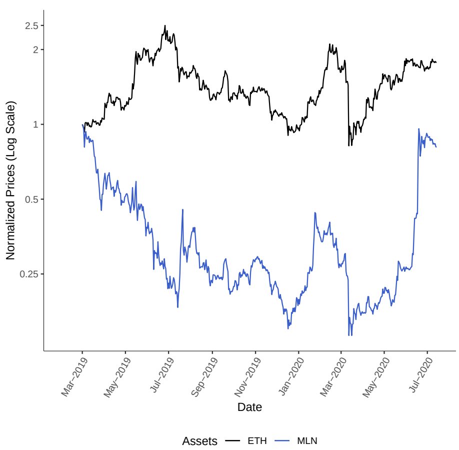
It is plausible to assume that the MLN volume is related to the MLN price, because there is often a price movement with a higher volume. Figure 5d illustrates the daily volume of the MLN token. We see, except for August 2019, the volume was quite low until November 2019. From November 2019 until March 2020, a significantly higher volume was observed, but it declined again in April 2020. As of May 2020, the volume has increased again.



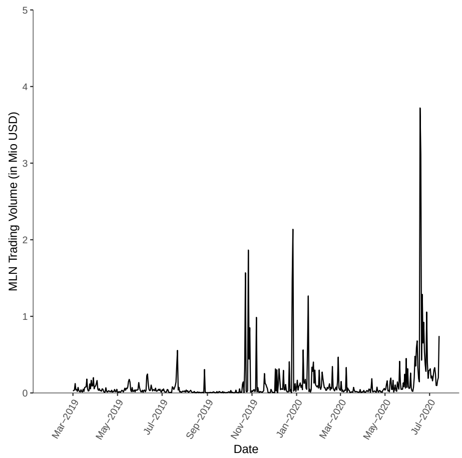
(a) Amgu Price Development



(b) ETH/MLN Price Development



(c) ETH and MLN Price Developments



(d) MLN Volume

Figure 5: Price and Volume Histories. Sources: Melon API (Figure 5a), CoinGecko (Figure 5b/5c) and CoinMarketCap (Figure 5d), own illustrations.

Table 4 indicates some statistics about the MLN token volume. In general, we can say that there are outliers when the mean is higher than the median, and the more the mean is deviated from the median, the larger are the outliers. We see in Table 4 that the mean compared to the median is three times higher, so there must be outliers. This is confirmed when looking at the variance of 91,322,724,679, the maximum volume of \$3,720,073 USD, whereas the mean is only \$112,279 USD. The outliers can also be observed in Figure 5d. To analyze how much of this apparent activity is driven by the buy-and-burn model of the Melon Protocol, we look at Table 5.

1st Qu.	Var	Median	Mean	3rd Qu.	Min	Max
14.069	91,322,724,679	35,274	112,279	77,901	1,040	3,720,073

Note: Daily observations (in USD). 1st Qu. and 3rd Qu. refer to the 25% and 75% quantile.

Table 4: Daily MLN Volume Statistics in USD. Source: CoinMarketCap, own calculations.

Table 5 illustrates how much MLN was sold in interactions between Melon managers and the Melon Engine. At the time of the analysis, 2,301.2 MLN tokens were exchanged for 77.6 ETH. For the data sample, only significant quantities were taken into account; small quantities of up to two MLN were not considered. We make this restriction because the small amounts comprise less than 0.3% of the MLN tokens traded with the Melon Engine. When several transactions were carried out on consecutive days, the amount of MLN sold on those days was added (e.g., March 26 and April 30, 2019). We took the daily closing prices of MLN to calculate the total volume of this transactions in USD. We then calculated the weekly average volume of the MLN token to make the data sample less sensitive to outliers. These two values were calculated to determine the relative proportion of Melon Engine transactions in the respective weekly average volume. This value is represented in the third column. The last column illustrates how high the premiums were for the respective interactions.

We see that the premiums (with the exception of May 2020) have always been at least 10% after the Melon protocol upgrade. In July 2020, a premium of 15% was even achieved for the first time. This indicates that after the release of the Melon Protocol v1.1, more ETH was accumulated by the Melon Engine. We can see that in March 2020 the buy-and-burn model accounted for the largest proportion of the weekly average volume, with 4.91%. The four highest proportions are observed in October 2019 and March, April and July 2020, with three of the four under the Melon Protocol v1.1. Additionally, we must consider that, at the same time, the overall volume of the MLN token was increased (see Figure 5d). Thus, there is more activity needed on the Melon Protocol to exhibit a significantly higher proportion of the weekly average volume of MLN. The results of this analysis illustrate an increase in the absolute and relative proportion of the buy-and-burn model in the weekly average volume. However, we must consider that such events take place too rarely to draw a representative conclusion.

Date	MLN Sold	MLN Sold rel. to w.av.vol.	Premium
26. Mar 2019	127	1.33%	10%
30. Apr 2019	42.14	1.41%	5%
31. May 2019	43.3	0.81%	0%
03. Jul 2019	31.83	0.35%	0%
02. Oct 2019	96	2.79%	5%
05. Nov 2019	120	0.32%	5%
29. Jan 2020	144	0.50%	5%
16. Mar 2020	445	4.91%	10%
16. Apr 2020	280.35	1.50%	10%
17. May 2020	206	0.59%	5%
16. Jun 2020	201.7	0.169 %	10%
16. Jul 2020	568	2.23 %	15%

Note: MLN Sold relative (rel.) to weekly (w.) average (av.) volume (vol.)

Table 5: Burning Event Statistics of the Melon Engine. Sources: Melon API (MLN Sold, Premium), CoinGecko (MLN Price) and CoinMarketCap (Weekly Average Volume of MLN), own calculations.

Melon Managers and Melon Funds

We know from the previous section that activity has increased and more ETH was earned and collected by the Melon Engine. Now, we want to examine whether activity has increased in terms of new Melon managers and Melon funds. Figure 15 in the Appendix illustrates the development of the number of active and inactive Melon funds. We define an active fund as a fund that is not shut down and is deployed on the newest Melon protocol version, whereas an inactive fund is deployed on an older one or is shut down. This definition is chosen because the Melon Terminal v2.0 allows only active funds to participate in trades or investments. The only action that investors can do with an inactive fund is to redeem their shares. We can see that the upgrade of the Melon Protocol to v1.1 led to an increase in the number of newly deployed, active funds.

Figure 6 illustrates how the AuM is distributed over all Melon funds deployed since the launch of the Melon Protocol v1.0. We distinguish between funds deployed on v1.0 and v1.1 to determine, whether funds set-up on v1.0 or v1.1 have managed more AuM. The figure indicates that, until October 2019, the AuM of all Melon funds was low compared to the AuM at the end of the year. After the upgrade of the Melon Protocol to v1.1, the AuM in funds deployed on v1.0 has declined and, in May 2020, less than 40 ETH was managed by funds deployed on v1.0. We see that, by the beginning of May 2020, more AuM was managed by funds deployed on v1.1 than ever before by funds deployed on v1.0; on July 16, 2020, that figure was even more than twice as much

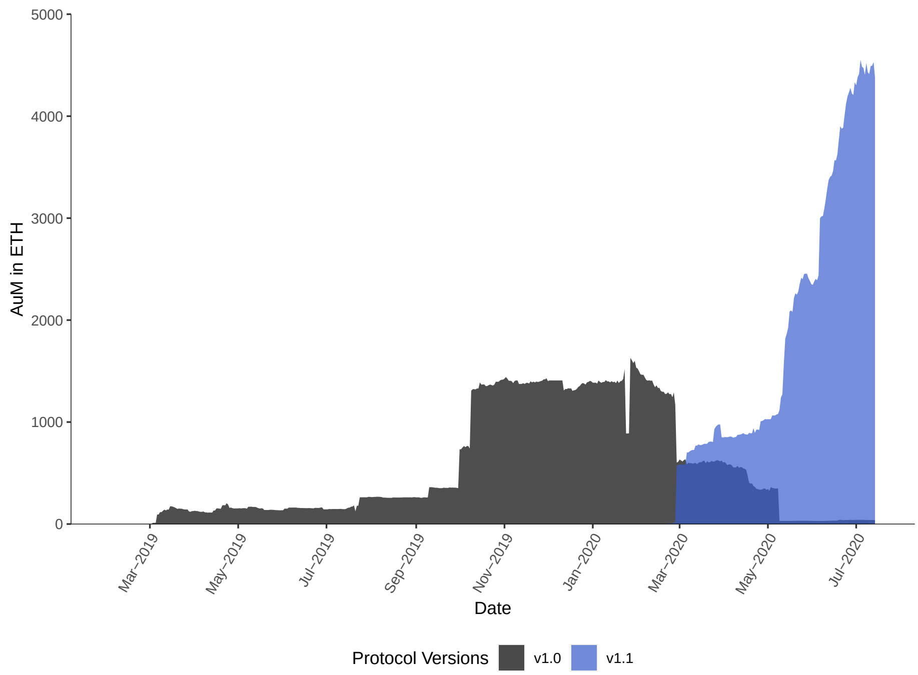


Figure 6: Melon AuM Distribution over Melon Funds Deployed on v1.0 and v1.1. Source: Melon API, own illustration.

After seeing in Figure 6 how much AuM is managed by Melon funds deployed on the different protocol versions, we now examine the distribution of this AuM over active Melon funds. Figure 7 illustrates that, at the time of writing on July 16, 2020, an AuM of 4,456 ETH is distributed over about 180 active Melon funds. Only active funds are considered, because less than 1% is managed by inactive funds.

The Gini coefficient equals 0.95, which indicates an inequality in the distribution of the AuM. This is indicated in Figure 7, where 50% of the cumulative share of active funds have no or a negligible volume. Additionally, we can observe that more than 90% of the funds manage less than 10% of the total AuM. We conclude that even though the AuM managed by Melon funds has increased significantly after the Melon protocol upgrade, the majority is managed by very few Melon funds.

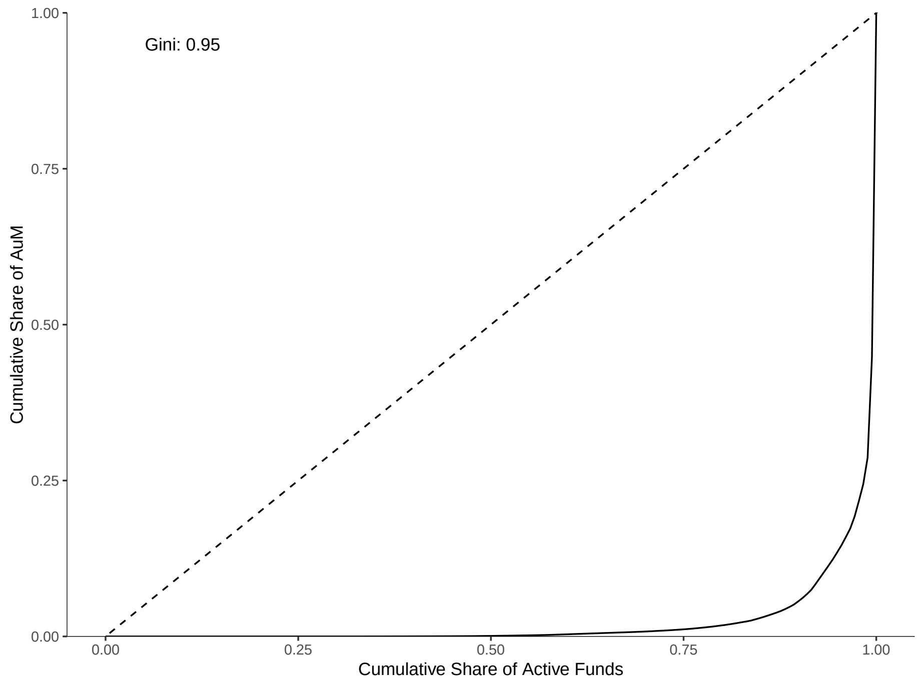


Figure 7: Melon AuM Distribution over Active Melon Funds. Source: Melon API, own illustration.

Investors and Investments

In this section, we consider the investors and their investments. In this section, we wish to check whether the activity has increased in terms of new investors and investments. Figure 8 illustrates that new investor addresses per month have increased after the Melon protocol upgrade. In the months of May and June 2020, approximately the same number of investor addresses were created as in the entire period of Melon v1.0.

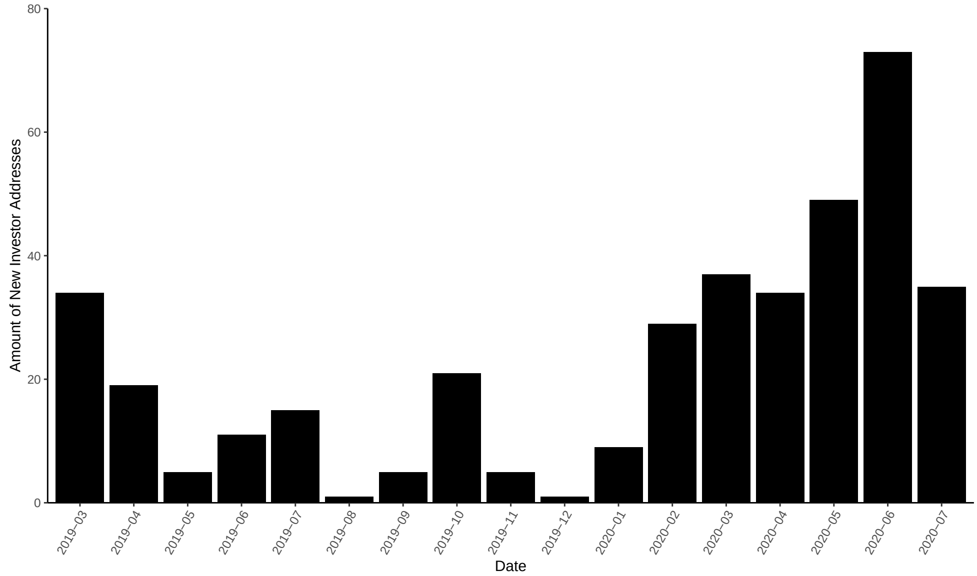


Figure 8: Amount of New Investor Addresses per Month. Source: Melon API, own illustration.

Table 6 illustrates some statistics about the investors and their investments in Melon funds. We see that despite the different periods, each metric is higher in v1.1 than in v1.0. In only five months on v1.1, more than three times as much ETH was invested in Melon funds than in the entire period of v1.0. On a monthly basis, more than seven times as much ETH was invested. The number of investor addresses has more than doubled, and the number of investments has almost tripled. There is a clear increase in activity on the part of investors and their investments. However, it must be taken into account that, during its initial months, the Melon Protocol v1.0 had to gain ground.

	I	**N of Inv.**	**⌀ Inv.p.I**	**T Inv.**	**⌀ Inv.a.p.I**	**⌀ Inv.a.p.m**
v1.0	101	171	1.69	1,496.74	8.75	130.15
v1.1	235	508	2.16	4,641.95	9.14	928.39

Note: The amounts are in ETH. Investors (I), number of investments (N of Inv.), average investments per investor (⌀ Inv.p.I), total investments (T Inv), average investment amount per investor (⌀ Inv.a.p.I), average investment amount per month (⌀ Inv.a.p.m). We took 11.5 months for v1.0 and 5 months for v1.1.

Table 6: Investors and Investment Statistics. Source: Melon API, own calculations.

Trading Activity
Figure 9 illustrates how the activity of the Melon managers has increased on the different DEXs. We see that there has been a massive increase in trading volume on Melon since March 2020. In the month of March alone, more volume was generated than over the entire period up to the Melon protocol upgrade. In June 2020, the highest volume since the launch of the Melon Protocol can be observed. Therefore, in terms of the volume of trades generated by the Melon funds, we can conclude, that there has been a massive increase in the volume of trades compared to the period before the protocol upgrade of v1.0 to v1.1. So far, we have looked at whether there was a change in activity but not at what the drivers might be for these results.

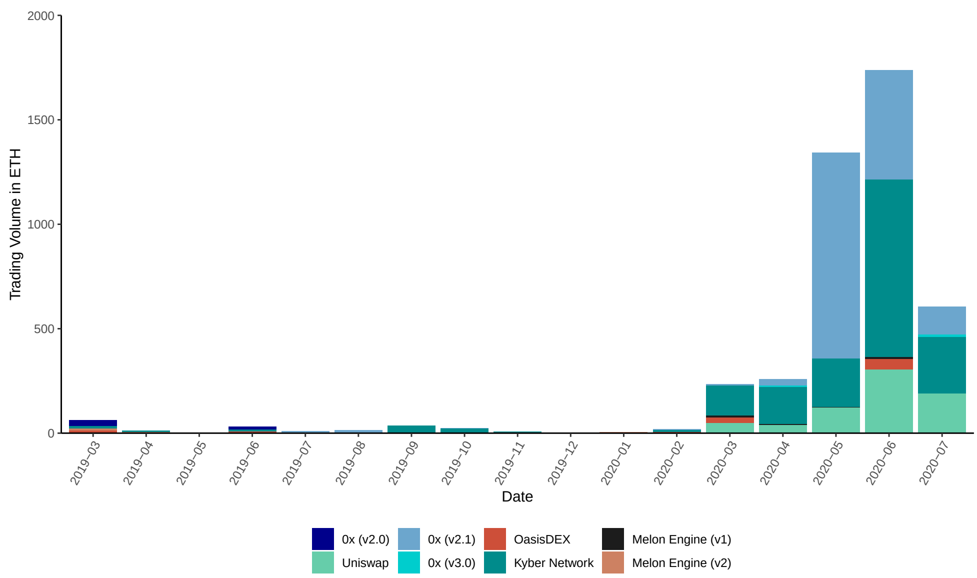


Figure 9: Trading Activity on Melon per Month. Source: Melon API, own illustration.

5.1.3 Possible Drivers

After discussing the increase in the activity on Melon, it is of further interest to understand what the possible drivers might be. This section examines whether the new cryptoassets added to Melon asset universe or the new DEXs integrated with the Melon Protocol might be possibilities.

New Cryptoassets

Figure 10 illustrates the trading counts per month on Melon. We distinguish between old cryptoassets, those added to the Melon asset universe with v1.0, and new cryptoassets, those added to the Melon asset universe under v1.1. We already introduced the tokens in Section 3.2.1. When we examine the old cryptoassets, it is observable that the trading counts increased after the Melon protocol upgrade in February 2020. We can see that the new cryptoassets comprise more than a third of the monthly trading counts. Especially in June 2020, many trades were made with new cryptoassets in absolute terms.

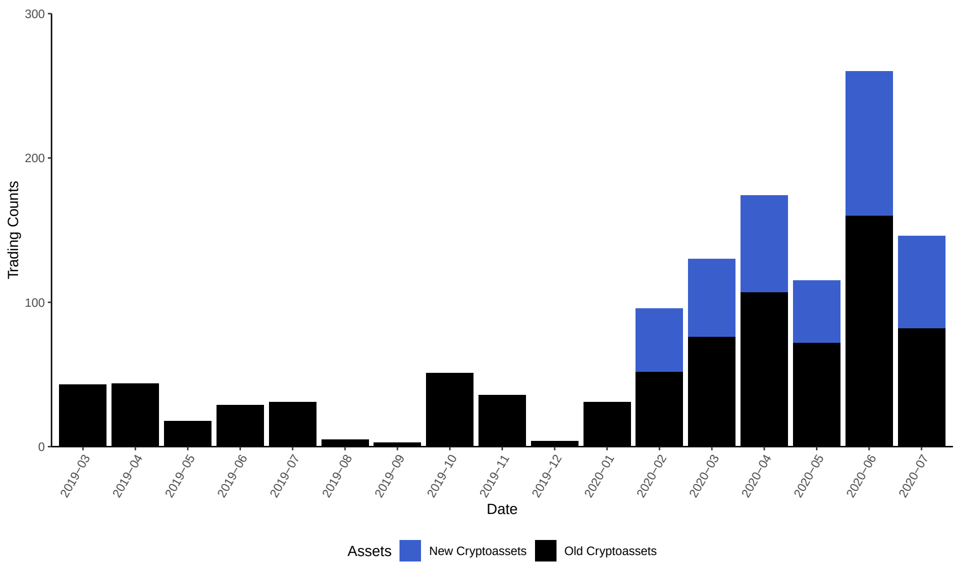


Figure 10: Trading Counts on Melon per Month. Source: Melon API, own illustration.

In Figure 11, we can see that this is not true for the trading volume of new cryptoassets on Melon. The new cryptoassets do not account for a large proportion of the increased volume per month. Even if the absolute proportion of new cryptoassets in the total volume has increased significantly each month (with the exception of July), it is not the same in relative terms. Nevertheless, it must also be noted here that these proportions cannot be ignored. Especially in June 2020, for example, the new cryptoassets make up 15% to 20% of the total volume.

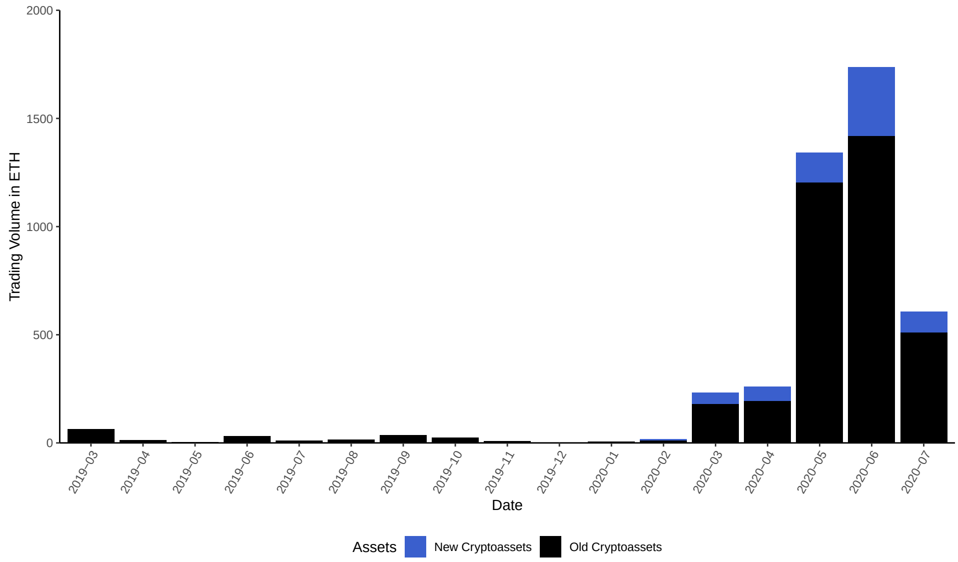


Figure 11: Trading Volume on Melon per Month. Source: Melon API, own illustration.

After examining both figures, we conclude that the volume and the counts of the new cryptoassets do not have the same impact on the increased activity. The figures indicate that the relative terms of the counts are higher compared to the volume, but nevertheless, a noticeable proportion of the volume can be observed. However, as already indicated in Figure 7, the AuM is inequitably distributed.

The Melon API provides the opportunity to retrieve on-chain data stored in the Ethereum blockchain to check, which cryptoassets were included in the highest transactions executed by the Melon fund managers. The data indicate that the 10 transactions with the highest volume were executed by the same Melon fund. This Melon fund, called Rhino Fund, sold 1.100 ETH for BTC between May 19 and June 29, 2020. Therefore, we can say that new cryptoassets are traded, but the volume generated by all Melon fund managers is more likely determined by the preferences of a few, or even of only one Melon fund manager. We conclude that the new cryptoassets contribute to the increased activity on Melon, but they are not the main drivers.

New DEXs

After analyzing the cryptoassets, we examine the new DEXs integrated with the Melon Protocol v1.1. Figure 10 illustrates the trading counts per month on the various DEXs. We can see that many trades were made on the new DEXs, especially on Uniswap v1 (see Figure 9). As we have already seen in the section on the new cryptoassets, the highest number of trades were made in June. If we take into account that the data were only obtained until mid-July, then the trades have increased continuously with the exception of May 2020. The trades on the new DEXs make up one-quarter to one-half of the trading counts, a significant proportion of the overall counts.

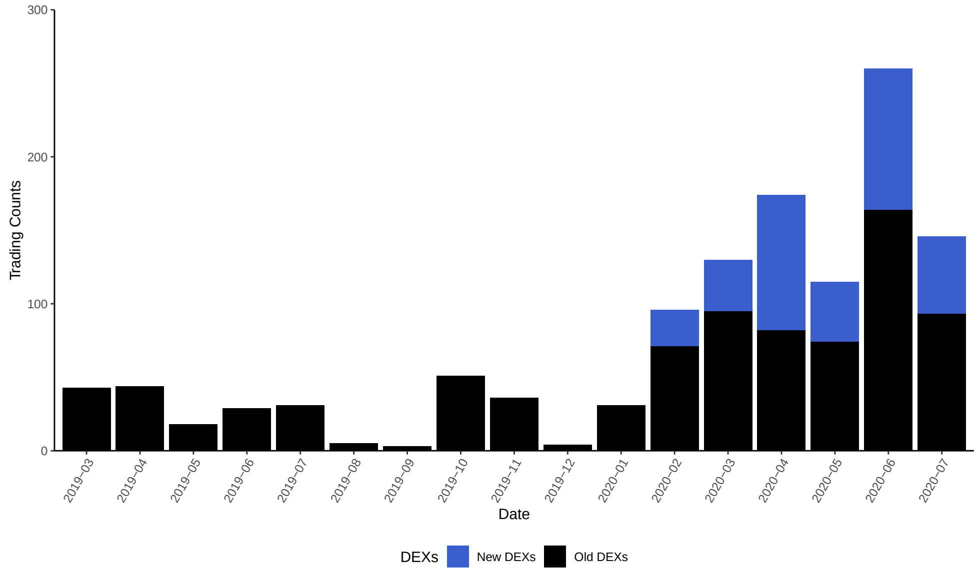


Figure 12: Trading Counts on DEXs Integrated with the Melon Protocol per Month. Source: Melon API, own illustration.

Figure 13 illustrates that the volume generated on the DEXs is not driven mainly by the new DEXs. We can observe a monthly increase in absolute terms of the total volume generated on the new DEXs, but not in relative terms. As in the section on new cryptoassets, we cannot ignore these results. New DEXs made up a similar proportion of 15% to 20% in June 2020.

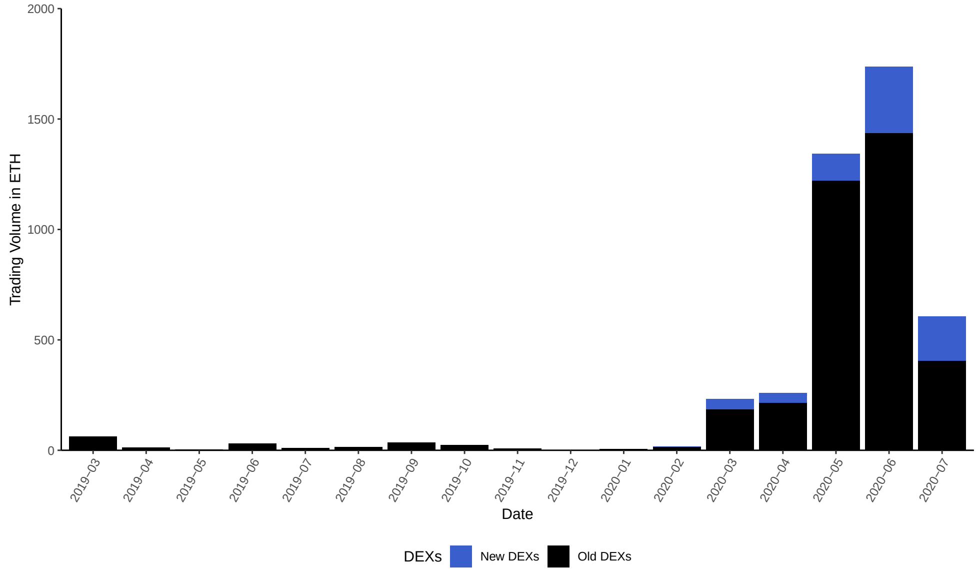


Figure 13: Trading Volume on DEXs Integrated with the Melon Protocol per Month. Source: Melon API, own illustration.

After looking at both figures, we can make the same assumptions, such as were made in the section on new cryptoassets. If we refer to Figure 7 again, we can assume that Figure 13 is distorted as well. If the few Melon managers with the most AuM prefer a specific DEX, these managers will also generate the highest volume on that specific DEX. To check this, we retrieve on-chain data stored in the Ethereum blockchain. The data indicate that the 10 transactions with the highest volume executed by the Rhino Fund were all executed on 0x v2.1. Therefore, we can say that the new DEXs are used for trades, but the volume generated by the Melon fund managers on the DEXs integrated with the Melon Protocol is more likely determined by the preferences of a few, or even of only one Melon fund manager. We conclude that the new DEXs contribute to the increased activity on Melon, but they are not the main drivers.

We examined only the volume and the trading counts, but even better information could have been obtained by looking at how many different Melon managers have traded on the respective DEXs and how high the volume was. Nevertheless, we would like to take a closer look at the DEXs and examine a typical trade on the respective DEX to, perhaps, draw more accurate conclusions. In the following, we will represent some statistics about the DEXs in Table 7 and a boxplot for each DEX in Figure 14.

If we look at Table 7 and Figure 14, we can observe how a typical trade on the DEXs differ from each other. The statistics refer to the time after the new DEXs were integrated. The main focus of the investigation is on the Kyber Network, Uniswap v1 and 0x v2.1. The reason for this limitation is that 0x v3.0 and OasisDEX have small number of trades compared to the other DEXs, and the total volume of both DEXs generated by Melon fund managers comprise less than 2.5% of the sum of the total volume generated by Melon fund managers on the Kyber Network, Uniswap v1 and 0x v2.1.

If we examine our main DEXs, we see that 0x v2.1 has the same total volume as the Kyber Network and twice as much as Uniswap v1 but a lower number of trades. If we look more closely at 0x v2.1, we see a very high variance compared to the Kyber Network and Uniswap v1 and a mean that is more than 19 times higher than the median. This suggests that, with a low number of trades, it is more likely that few managers have created transactions with high volumes, and the remaining transactions were quite low (as we have seen on the previous section). A similar pattern can be observed with OasisDEX and 0x v3.0 apart from the lower variance of 0x v3.0. This is the reason why we restrict our focus again and take a closer look at the Kyber Network and Uniswap v1.

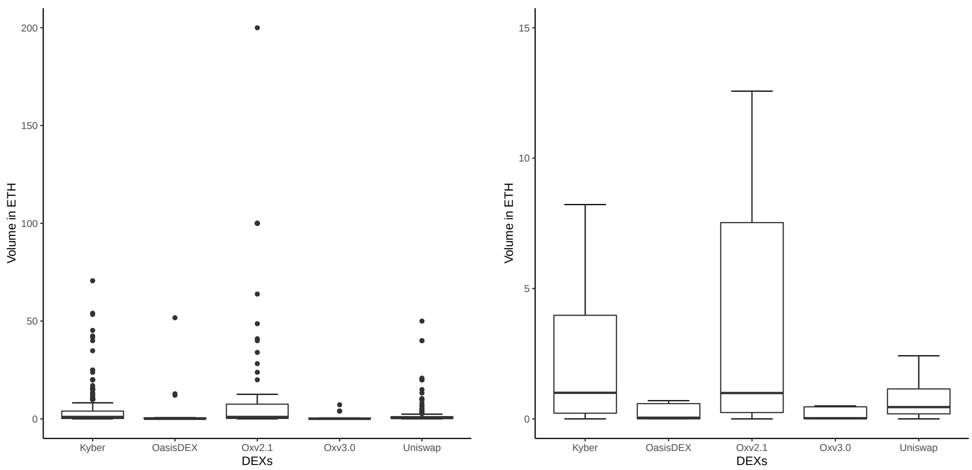
The Kyber Network and Uniswap v1 have the highest number of trades compared to the other DEXs, with the Kyber Network having a higher number than Uniswap v1 and twice the total volume. If we look at the variance, the Kyber Network has twice as much variance in transactions as Uniswap v1. Here, too, the mean is a multiple of the median, which seems that few managers perform high transactions. This fact is, however, not as marked as for the other DEXs. If we compare the Kyber Network and Uniswap v1, then it is likely that because of the higher variance and total volume, larger transactions are made on the Kyber Network, where Uniswap v1 is compared to the Kyber Network used for smaller transactions. However, we would like to note that this does not mean that there are no large transactions on Uniswap v1 and no small transactions on the Kyber Network.

	N	TV	Var	Mean	Median	Min	Max
Kyber Network	467	1,681.91	55.05	3.60	1.00	0.00	70.64
Uniswap v1	325	706.66	28.81	2.17	0.45	0.00	50.00
0x v2.1	88	1,685.96	1,503.17	19.16	0.99	0.00	200.00
0x v3.0	18	20.47	4.54	1.14	0.03	0.00	7.22
OasisDEX	14	77.79	196.45	5.56	0.04	0.00	51.72

Note: Terms are in ETH. N refers to number of trades after the protocol upgrade, TV refers to the total volume generated by Melon fund mangers on these DEXs.

Table 7: DEX Statistics. Source: Melon API, own calculations.

Figure 14 illustrates the described pattern in Table 7 in a boxplot for each DEX. We can see that 0x v2.1 has the highest outliers. We know from the section on the new cryptoassets and DEXs that one fund made 10 transactions with a total volume of 1,100 ETH. This is not representative in the figure, because the fund manager executed nine out of the top 10 transactions each with a volume of 100 ETH, which are represented by the same outlier.



(a) Boxplots with Outliers (b) Boxplots without Outliers

Figure 14: Boxplots of the DEXs. Source: Melon API, own illustration.

Now, we examine the volume of the top 10 trades generated on our main

DEXs. We look at the on-chain data we retrieved in sections on the new cryptoassets and DEXs. The results illustrate that the volume of the top 10 transactions executed by Melon fund managers on 0x v2.1 is 1,100 ETH; on the Kyber Network we observe a volume of 432 ETH; and on Uniswap v1 the volume of the top 10 transactions executed by Melon fund managers is 266 ETH. In May 2020 alone, the volume generated by the Rhino Fund comprise more than 50% of the total monthly volume. This is a further confirmation that the sections on the new cryptoassets and DEXs are not representative because one fund manager executes high volume trades.

One reason for trades with higher volumes on 0x v2.1 might be its approach of a decentralized order book exchange, where Melon managers can select an order (we know from see Section 4.2.1 that Melon managers can only take orders) they would like to match (Schär 2020). This option does not exist when using the Kyber Network or Uniswap v1, where a trade with a high volume on Uniswap v1 might lead to a high slippage (Adams 2018).[14]

It is also possible that the overall increased activity on Melon may be due to the increased interest in the DeFi ecosystem. Before the Melon protocol upgrade on February 2020, the TVL in DeFi reached $ 1.18 billion USD, and as of mid-July 2020 $ 2.61 billion USD was locked in DeFi related smart contracts. As of August 15, 2020, the TVL in the DeFi ecosystem amounts to $ 5.82 billion USD (DeFi Pulse 2020).

So far, we have examined whether the new cryptoassets or the new DEXs might be the possible drivers for the increased activity. We concluded that both of them contribute to the increased activity on Melon, but they are not the main drivers. The following section examines the new cryptoassets from another perspective. We investigate whether the new cryptoassets statistically lead to diversification benefits, from the perspective of a mean-variance spanning test.

[14] We must consider that the Kyber Network has integrated Uniswap v1 reserves as one of their on-chain liquidity reserves, thus trades considered to be conducted via the Kyber Network actually use Uniswap's reserves.

5.2 Mean-Variance Spanning Analysis

In the second part of the economic analysis, we investigated whether the new cryptoassets provide other benefits for users of the Melon Protocol. For this, we applied a so-called mean-variance spanning test in the form of the step-down test according to Kan and Zhou (2012). The classic mean-variance spanning test is a joint test that illustrates whether the efficient frontier of the benchmark assets spans the efficient frontier of the benchmark assets and the test assets under normality assumptions. We applied the step-down test introduced by Kan and Zhou (2012) to examine whether the shift of the efficient frontier is caused by the global minimum-variance portfolio (GMVP) or by the tangency portfolio (TP).

5.2.1 Methodology

In this section, we examined whether the inclusion of a new set of assets (new cryptoassets added to the Melon asset universe with the Melon protocol upgrade to v1.1) to a diversified portfolio of risky assets (the Melon asset universe of the Melon Protocol v1.0) improves the risk and return trade-off of an efficient portfolio. From the perspective of a mean-variance spanning test, an efficient portfolio is a portfolio that has the highest expected return for a given level of risk. In more general terms, we investigated the effect that adding a new set of assets to the diversified portfolio of risky assets has on the efficient frontier. For this, we examined the expansion of the efficient frontier of the cryptoassets included in the Melon asset universe defined in the Melon Protocol v1.0.

Mean-Variance Spanning Test

Huberman and Kandel (1987) were the first to introduce a mean-variance spanning statistic test, which evaluates the contribution to the portfolio diversification of various asset classes. The mean-variance frontier is a set of all portfolios that is expected to create the maximum possible return for a given level of risk. The test is a regression-based approach and assumes, that investors make portfolio investment decisions only based on mean and variance.

The starting point is to define R_{1t} as a K-vector with the returns of the K benchmark assets and R_{2t} as an N-vector with the returns of the N test asset. The total returns of $K + N$ assets are defined as $R_{\mathbf{t}} = \left[R'_{1t}, R'_{2t}\right]'$. It is assumed that the risk-free interest rate is equal to 0, which means that an investor must be fully invested in either the K portfolio or the $K + N$ portfolio. The expected returns of $R_{\mathbf{t}}$ are denoted as

$$\mu = \mathrm{E}[R_t] \equiv \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \tag{4}$$

and the covariance matrix of the $K + N$ risky assets is defined as

$$\mathrm{V} = \mathrm{Var}[R_t] \equiv \begin{bmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{bmatrix}, \tag{5}$$

where V is nonsingular. The matrix contains the variance of the benchmark assets and the test assets (represented by V_{11} and V_{22}) and the covariance of the benchmark assets and the test assets (represented by V_{12} and V_{21}). By using the OLS approach to project R_{2t} on R_{1t}, we estimate the following regression model

$$R_{2t} = \alpha + \beta R_{1t} + \epsilon_t, \quad t = 1, 2, ..., T, \tag{6}$$

where T is the time vector, ϵ_t are the residual terms, $\beta = V_{21}V^{-1}{}_{11}$, $\alpha = \mu_2 - \beta\mu_1$ and $\delta = 1_N - \beta 1_K$. 1_N is an N-vector of ones. It is assumed that ϵ_t is independent and identically normal distributed and the following equation holds

$$\mathrm{E}[\epsilon_t] = 0_{\mathrm{N}} \quad \text{and} \quad \mathrm{E}[\epsilon_t R'_{1t}] = 0_{\mathrm{NxK}}, \tag{7}$$

whereas 0_{NxK} is an N by K matrix of zeros and 0_N is an N-vector of zeros. We formulate the joint spanning hypothesis as

$$H_0: \quad \alpha = 0_{\mathrm{N}}, \quad \delta = 1_{\mathrm{N}}\text{-}\beta 1_{\mathrm{K}} = 0_{\mathrm{N}} \tag{8}$$

The joint spanning hypothesis checks whether there is a linear combination of the K benchmark assets that has the same mean but a lower variance than the N test assets. This would mean that the test asset is dominated (spanned) by the benchmark assets. As a consequence, the contribution of the test asset on the efficient frontier is statistically insignificant. In this case, as indicated by Kan and Zhou (2012), the test assets in the two efficient portfolios – the TP and (GMVP) – on the mean-variance frontier would have a zero portfolio-weight ($\alpha = 0$ for TP and $\delta = 0$ for GMVP). If the null hypothesis is rejected, then the test asset is not dominated (spanned) by the benchmark assets. In other words, the test asset can expand the investment opportunity set and lead to diversification benefits for investors, from the perspective of a mean-variance spanning test.

The test results of the mean-variance spanning test consists of the Wald (W), Likelihood Ratio (LR), and the Lagrange Multiplier (LM) tests. As mentioned earlier, we did not apply the classic mean-variance spanning test, where $\alpha = 0$ and $\delta = 0$ were checked simultaneously. Instead, we analyzed $\alpha = 0$ and $\delta = 0$ separately on a limited scale in the form of a step-down test. Therefore, we recommend the paper of Kan and Zhou (2012) for a detailed instruction of the W, LR and LM tests.

Step-Down Test

In Kan and Zhou (2012), we can see that testing $\alpha = 0$ and $\delta = 0$ in equation (8) is a joint test, which might lead to inaccurate results. A shortcoming of the joint test is that δ is estimated more accurately than α, which influences the weighting of both conditions in equation (8). As a consequence, the spanning tests' results might indicate significant changes in the TP, even though there are no significant changes in reality.

Even if there are significant changes in the GMVP in reality, the spanning tests do not lead to significant results (Schmitz and Hoffmann 2020). Kan and Zhou (2012) introduced the step-down test to check the acceptance or rejection of both conditions separately. Hence, $\alpha = 0$ is tested at a 5% significance level by

$$F_1 = \left(\frac{T\text{ - }K\text{ - }N}{N}\right)\left(\frac{|\bar{\sum}|}{|\hat{\sum}|} - 1\right) \sim F_{\text{N, (T - K - N)}}, \tag{9}$$

where F_1 has a central F-distribution with N and T - K - N degrees of freedom under the null hypothesis. $\hat{\sum}$ and $\bar{\sum}$ are the unconstrained and constrained estimates of $\sum$, where $\sum = V_{22}$ - $V_{21}{V_{11}}^{-1}V_{12}$. Constraint is defined as $\alpha = 0_N$. Conditional on $\alpha = 0_{\text{N}}$, we check with a second F-test (F_2-test) a 5% significance level whether the condition $\delta = 0$ holds

$$F_2 = \left(\frac{T\text{ - }K\text{ - }N + 1}{N}\right)\left(\frac{|\tilde{\sum}|}{|\bar{\sum}|} - 1\right) \sim F_{\text{N, (T - K - N + 1)}}, \tag{10}$$

where $\tilde{\sum}$ is the constrained estimate of $\sum$ with the joint constraints of $\alpha = 0$ and $\delta = 0$. F_2 has a central F-distribution with N and T - K - N + 1 degrees of freedom under the null hypothesis.

The step-down checks the cause of the shift of the efficient frontier. By testing both conditions of equation (8) separately, we know whether the rejection of the hypothesis is caused by the significantly different TP of portfolios K and $K + N$ (due to F_1), or the rejection is caused by the significantly different GMVP of portfolios K and $K + N$ (due to F_2). Another feature of the step-down test is that it also is applicable to small samples.

5.2.2 Hypotheses

In the previous sections, we explained the structure of the empirical analysis. In this chapter, we formulate the null hypotheses of the second part of the economic analysis.

As we know from the previous sections, the Melon added six new cryptoassets to the Melon asset universe with the Melon protocol upgrade to v1.1. These new cryptoassests build a new investments opportunity set that a Melon manager can use to construct a portfolio. By applying the step-down test from Kan and Zhou (2012), this thesis aims to provide, firstly, an answer to the question of whether the new cryptoassets that were added to the Melon asset universe provide diversification benefits to the pre-existing Melon asset universe. Secondly, this thesis aims to determine what causes the diversification benefits. We examine the period from March 01, 2019 to July 16, 2020 and the sub-period from the Melon protocol upgrade to v1.1 to July 16, 2020.

$H_{\mathbf{0}}$ **1**: The efficient frontier consisting of the pre-existing Melon asset universe as the benchmark assets spans the efficient frontier consisting of the same benchmark assets with the addition of test assets as the new cryptoassets added to the Melon asset universe with v1.1.

$H_{\mathbf{0}}$ **2**: The TP is the cause of spanning the efficient frontier of the pre-existing Melon asset universe as the benchmark assets with the addition of the new cryptoassets added to the Melon asset universe with v1.1.

$H_{\mathbf{0}}$ **3**: The GMVP is the cause of spanning the efficient frontier of the pre-existing Melon asset universe as the benchmark assets with the addition of the new cryptoassets added to the Melon asset universe with v1.1.

5.2.3 Data Description

First, we must define the K benchmark and N test assets for the step-down test. The dataset consists of nine cryptoassets added to the Melon asset universe with the launch of the Melon Protocol v1.0 as the benchmark assets and six newly added to cryptoassets to the Melon asset universe with the Melon protocol upgrade to v1.1 as the test assets. The benchmark assets are composed of WBTC, WETH, MLN, MKR, KNC, ZRX, BAT, REP and USDC. The test assets are composed of LINK, ANT, MANA, REN, RLC and WSaiDai.[15]

For the step-down test, we need the returns of the cryptoassets. The daily closing prices in USD for all cryptoassets of the Melon asset universe from March 01, 2019 to July 16, 2020 are taken from CoinGecko. One reason for choosing the daily closing prices is that daily returns provide more observations for such a short time frame. Moreover, the continuous trading availability of cryptoassets can be better captured with higher data frequency. For each cryptoasset, there is a total of 504 observations. We applied the step-down test for two periods: The first period – the entire period with 504 observations from March 01, 2019 to July 16, 2020 – checked whether the inclusion of the test assets to the Melon asset universe with v1.0 would have led to diversification benefits for Melon managers. The second sub-period – with 152 observations from February 16, 2020 to July 16, 2020 – indicates whether the results are also true for the inclusion of the test assets to the Melon asset universe with v1.1. In a first step, we calculated the log returns $r_{\text{i,t}} = \log \left[\frac{P_{\text{i,t}}}{P_{\text{i,t-1}}}\right]$ for the entire dataset. After calculating the log returns, we can report and discuss statistical results.

According to the summary of descriptive statistics in Tables 8 (entire period) and 9 (sub-period), the data of this thesis reveal high means and standard deviations. The lowest standard deviation has the USDC with 0.36% (0.19%) followed by WSaiDai, with a standard deviation of 0.8% (0.89%) in Tables 8 and 9. Additionally, the two tables indicate that, aside from the stablecoins, WBTC and WETH have the lowest standard deviation with 5.64% (5.55%) and 5.02% (6.59%), respectively. Aside from the stablecoins, in both tables, three of the four cryptoassets with the highest standard deviation are new cryptoassets added to the Melon

[15]Because Dai is a new version of Sai, we will weight both cryptoassets by their marketcap to get the weighted Sai and Dai price as a test asset.

asset universe (ANT, iExec, Ren).

At the same time, the cryptoassets have, in both tables, the highest mean of the top five cryptoassets with the highest mean. This indicates, that the new cryptoassets added to the Melon asset universe might increase the returns but also the overall volatility of a portfolio constructed by the pre-existing cryptoassets of the Melon asset universe. We can see that both metrics are higher in the sub-period in Table 9 than in the entire period in Table 8. One of the reasons might be the decline in the cryptomarket in March 2020. In both tables, a high kurtosis as well as mostly negative skewed distributions are observable. Compared to the normal distribution, the cryptoassets in both tables have longer left tails. The empirical Jarque-Bera test[16] indicates, in both tables, that the null hypothesis for the normality assumption is rejected at a 5% level. This demonstrates that all cryptoassets of the Melon asset universe are – for the given periods – not normally distributed.

	Mean	**SD**	**Min**	**Max**	**SK**	**K**	**JB**
0x	0.09%	5.65%	-43.01%	37.24%	-0.01	14.57	2,794.44
Augur	0.09%	5.77%	-56.82%	46.22%	-0.61	29.95	15,192.64
BAT	0.10%	5.66%	-59.46%	24.03%	-2.04	27.80	13,186.43
KNC	0.41%	7.07%	-64.12%	35.81%	-0.80	18.94	5,357.44
MKR	-0.07%	6.73%	-88.10%	44.20%	-3.63	66.08	84,163.70
MLN	-0.04%	7.17%	-51.36%	56.05%	0.02	17.29	4,262.79
USDC	0.00%	0.36%	-1.98%	1.98%	-0.04	9.47	873.98
WBTC	0.18%	5.64%	-49.21%	44.73%	-1.29	32.08	17,791.82
WETH	0.11%	5.02%	-55.55%	17.49%	-2.67	33.39	19,874.41
ANT	0.24%	7.14%	-72.93%	24.14%	-1.82	25.46	10,807.01
LINK	0.59%	6.78%	-66.22%	47.87%	-0.83	26.56	11,644.68
MANA	-0.01%	5.94%	-65.36 %	23.41%	-2.57	32.30	18,472.49
iExec	0.23%	7.24%	-75.69%	31.47%	-1.80	27.32	12,617.32
WSaiDai	0.00%	0.80%	-4.43 %	4.65%	0.25	8.31	593.81
Ren	0.47%	7.88%	-71.78 %	26.79%	-1.23	17.20	4,335.56

Note: Statistics are based on daily log returns. SD, SK, K and JB stand for standard deviation, Skewness, Kurtosis and Jarque-Bera.

Table 8: Entire Period Statistics. Source CoinGecko, own calculations.

[16]The Jarque-Bera test is – based on the sample skewness and kurtosis – a test for normality (Jarque and Bera 1980).

	Mean	SD	Min	Max	SK	K	JB
0x	0.21%	7.36%	-43.01%	37.24%	-0.33	14.53	822.49
Augur	0.24%	7.26%	-56.82%	22.44%	-2.97	27.98	4,065.58
BAT	-0.10%	7.14%	-59.46%	22.23%	-3.62	34.53	6,453.78
KNC	0.89%	9.03%	-64.12%	24.08%	-2.13	20.25	1,946.88
MKR	-0.19%	10.39%	-88.10%	44.20%	-3.32	39.12	8,317.25
MLN	0.55%	8.53%	-51.36%	56.05%	0.52	22.98	2,468.41
USDC	0.00%	0.19%	-0.98%	0.95%	-0.17	10.59	355.96
WBTC	-0.04%	5.55%	-49.21%	13.77%	-4.43	43.24	10,469.50
WETH	-0.07%	6.59%	-55.55%	17.49%	-3.85	35.86	7,024.26
ANT	0.43%	9.40%	-72.93%	24.12%	-3.08	27.05	10,807.01
LINK	0.43%	7.86%	-66.22%	20.99%	-3.93	36.62	11,644.68
MANA	-0.22%	7.92%	-65.36 %	23.41%	-3.67	32.76	18,472.49
iExec	0.36%	9.35%	-75.69%	20.78%	-3.45	30.99	12,617.32
WSaiDai	0.01%	0.89%	-4.43 %	3.23%	-0.20	8.44	593.81
Ren	0.73%	9.16%	-71.78 %	22.25%	-3.26	28.29	4,335.56

Note: Statistics are based on daily log returns. SD, SK, K and JB stand for standard deviation, Skewness, Kurtosis and Jarque-Bera.

Table 9: Sub-Period Statistics. Source CoinGecko, own calculations.

Tables 12 and 13 in the Appendix reveal high positive correlations between the various cryptoassets of the Melon asset universe. Aside from the stablecoins, the highest (lowest) correlation coefficient of the pre-existing cryptoassets in Table 12 is 0.73 (0.37) and 0.95 (0.44) in Table 13, respectively. Again, aside from the stablecoins, the correlations between the new cryptoassets added to the Melon asset universe and the pre-existing cryptoassets in Tables 12 and 13 are between 0.34 and 0.65 (0.53 and 0.87). Therefore, the high coefficients of the correlation matrix in Tables 12 and 13 allow us to assume that low diversification benefits were generated from the inclusion of the new cryptoassets to the Melon asset universe. Since adding assets to the existing asset universe causes a shift of the efficient frontier to the left, it is of further interest to test this shift for significance. This is done in the following section.

5.2.4 Step-Down Test Results

Tables 10 and 11 reveal the results of the step-down procedure proposed by Kan and Zhou (2012). The F-test indicates the classic mean-variance spanning test as a joint hypothesis test of $H_{\mathbf{0}}$ **1**, whereas F_1 and F_2 of the step-down test indicate whether the acceptance or rejection of $H_{\mathbf{0}}$

1 is caused by $H_\mathbf{0}$ **2**, $H_\mathbf{0}$ **3** or both. Table 10 indicates that the F-test rejects spanning for ANT, LINK and WSaiDai at a 5% significance level for the entire period. The F-test reveals that a Melon manager can improve the efficient frontier and benefit from the diversification effect mostly by including WSaiDai in a portfolio consisting of the benchmark assets. A high F-test of 31.411 indicates lower variance of the test asset relative to the benchmark assets. ANT and LINK have a F-test value of 3.347 and 3.120, respectively.

However, the cause of the rejection differentiates between the cryptoassets. When the rejection of $H_\mathbf{0}$ **1** is caused by the rejection of $H_\mathbf{0}$ **2**, then the TP of the benchmark assets is statistically different from that of the benchmark assets plus test assets. The rejection of $H_\mathbf{0}$ **2** can be observed when examining the test results of LINK. The result of the F_1 test suggests that the TPs between K and K + LINK are statistically different. Consequently, it may be concluded that LINK is weighted in the K + LINK TP. For $H_\mathbf{0}$ **3**, the GMVPs between K and K + LINK are statistically not different, therefore we cannot reject $H_\mathbf{0}$ **3**.

When the rejection of $H_\mathbf{0}$ **1** is caused by the rejection of $H_\mathbf{0}$ **3**, then the GMVP of the benchmark assets is statistically different from that of the benchmark assets plus test assets. The results of the F_2 test of ANT and WSaiDai reveal this occurrence. They indicate, that the GMVPs of K and K + ANT as well as of K and K + WSaiDai are statistically different. Thus, a Melon manager can improve the GMVP of the K benchmark assets by adding ANT or WSaiDai as N test assets to the benchmark portfolio. The results demonstrate that this is not true for the TP. This implies that ANT and WSaiDai, as the test assets, are not weighted in the TP of K benchmark assets.

When reviewing Table 11 to determine whether the results in Table 10 are also true for the sub-period, we see that the $H_\mathbf{0}$ **1** of all new cryptoassets added to the Melon asset universe with v1.1 were not rejected. Thus, the TPs as well as the GMVPs between K and $K + N$ are statistically not different. As a consequence, we cannot reject $H_\mathbf{0}$ **2** and $H_\mathbf{0}$ **3** in Table 11. The results for the sub-period reveal that spanning exists, and a Melon manager cannot profit from the diversification benefits by including the test assets into the benchmark portfolio. In other words, the test asset does not have a significant effect on the mean-variance frontier of the benchmark portfolio.

Except for MANA, we can observe lower F-test values and higher p-values in the sub-period. One reason could be the higher correlation coefficients between the benchmark assets and the test assets. As a consequence, the diversification benefits of the test assets may have decreased.

	$\hat{\alpha}$	$\hat{\delta}$	**F-test**	**p**	**F1**	**p**	**F2**	**p**
ANT	0.001	1.788	3.347	0.036	0.309	0.579	6.394	0.012
iExec	0.001	- 0.888	0.892	0.410	0.195	0.659	1.592	0.208
LINK	0.005	0.846	3.120	0.045	4.667	0.031	1.560	0.212
MANA	-0.001	0.148	0.279	0.757	0.473	0.492	0.085	0.771
Ren	0.004	-0.373	0.957	0.385	1.695	0.194	0.218	0.641
WSaiDai	0.000	0.796	31.411	0.000	0.003	0.954	62.946	0.000

Table 10: Step-Down Test with New Cryptoassets Entire Period. Source: CoinGecko, own calculations.

	$\hat{\alpha}$	$\hat{\delta}$	**F-test**	**p**	**F1**	**p**	**F2**	**p**
ANT	0.003	1.155	0.298	0.743	0.381	0.538	0.217	0.642
iExec	0.002	0.465	0.101	0.904	0.168	0.683	0.035	0.852
LINK	0.005	0.585	1.166	0.315	2.224	0.138	0.106	0.746
MANA	-0.003	0.742	0.538	0.585	0.855	0.357	0.222	0.638
Ren	0.005	-0.245	0.578	0.562	1.146	0.286	0.010	0.921
WSaiDai	0.000	0.215	0.170	0.844	0.002	0.965	0.340	0.561

Table 11: Step-Down Test with New Cryptoassets Sub-Period. Source: CoinGecko, own calculations.

We can see that the rejection of H_0 **1** might bias the investment decisions of a Melon manager. The cause of spanning is identifiable after applying the step-down test. The additional information from the step-down test is of particular importance if H_0 **1** is not rejected because of H_0 **2** *and* H_0 **3**, but because of H_0 **2** *or* H_0 **3**. This is the case for all rejected hypotheses in Table 10.

5.2.5 Limitations

The step-down test is an extension of the classical mean-variance spanning test. We know, that the mean-variance spanning test assumes normal distributed returns. The Jarque-Bera test reveals that our data sample is not normally distributed. This leads to the assumption that the results may be distorted. Kan and Zhou (2012) propose a so called Generalized Method of Moments (GMM) Wald test for not normally distributed returns, which was not applied in this thesis.

Furthermore, we must be aware that these results are valid only for the chosen periods. Although monthly returns are desirable, the testing results would be constrained by the limited sample. The cryptoasset market and especially the Melon Protocol in the DeFi space are a new phenomenon that is still improving, so there is a lack of data that might not be substantially representative.

We must consider that the results of this study might be biased because the benchmark assets are not well-diversified. Since we analyzed the Melon Protocol in this thesis, we considered only cryptoassets of the Melon asset universe. The high correlation between the cryptoassets of the Melon asset universe increases the risk in the portfolio by reducing the diversification effect. Nevertheless, the test promises more significant results if the correlation coefficients between the cryptoassets of the Melon asset universe decrease over time or when new asset classes are added to the Melon asset universe. Several studies reveal the low correlation between cryptoassets and traditional asset classes (Eisl et al. 2015, Briere et al. 2015).

We also must take into account that we did not consider the effect of transaction costs or liquidity issues, which might have a significant effect on the results. There is already some literature on spanning tests concerning transaction costs or liquidity issues (e.g., Schmitz and Hoffmann 2020).

6 Conclusion

This thesis analyzed the technical and economic changes after the upgrade of the Melon Protocol v1.0 to v1.1. The main difference between the versions is the addition of new cryptoassets to the Melon asset universe, the integration of the new DEXs and the possibility to add further exchanges after the fund is deployed. After the technical part discussed the individual smart contracts and their functions, possible inefficiencies and dependencies were elaborated upon. We found that the number of smart contracts to set up a Melon fund can be reduced, for example, in the Policy Manager component. We discussed the oracle dependence on the Kyber Network and identified the Price Source as a centralized point of failure. We have also indicated what the impact on Melon managers and investors might be if the Melon Council DAO fails to adjust the amgu price in case of strong ETH/MLN fluctuations. As a consequence, we regarded the influence of the Melon Council DAO as critical, especially because Figure 5a, in the economic part of this thesis, indicates that the amgu price was once adjusted despite ETH/MLN price fluctuations.

In the descriptive results of the economic part, we demonstrated that activity increased after the Melon Protocol was upgraded to v1.1. First, we saw that more amg was earned and collected by the Melon Engine. We saw in Table 5 that the absolute and relative proportion of the buy-and-burn model in the weekly average volume has increased. However, we must consider that such events take place too rarely to draw a representative conclusion. Then we illustrated that more AuM is managed by Melon fund managers, that these managers generate more volume on the DEXs integrated with the Melon Protocol, that more investor addresses were created, and more investments were made compared to the period before the Melon Protocol was upgraded to v1.1.

Then we examined whether the new cryptoassets or DEXs are possible drivers for this increased activity. We came to the conclusion that the new cryptoassets as well as DEXs contribute to the increased activity on Melon, but they are not the main drivers of it. Then we considered the distribution of the AuM over the Melon funds to assume that these results might be distorted, because Figure 7 illustrates that only a few Melon fund managers manage a large portion of the AuM of Melon. Thus, their preferences could have an influence on the proportion of new cryptoassets and DEXs in the total volume. To check this, we retrieved on-chain data stored in the Ethereum blockchain and illustrated that the Rhino Fund has generated more than 50% of the volume in May 2020. The fund used 0x v2.1 (which is an old DEX) for the trades and sold WETH for WBTC (which are old cryptoassets).

In a final step, we examined a typical trade on the DEXs integrated with the Melon Protocol to confirm our assumption that the contribution of the new cryptoassets and DEXs in the increased activity might be distorted. We have illustrated with statistics that one Melon manager has executed large trades on 0x v2.1, where managers who used the Kyber Network and Uniswap v1 executed more trades with lower volume compared to 0x v2.1. We assumed that one reason might be the different approaches the DEXs are using. Additionally, we must consider that the Kyber Network has integrated Uniswap v1 reserves as one of their on-chain liquidity reserves, thus trades considered to be conducted via the Kyber Network actually use Uniswap's reserves. And finally, we also must take into account that the increased activity on Melon might be due to the increased interest in the DeFi ecosystem.

Based on daily log returns, we demonstrate with a mean-variance test, in the form of a step-down, whether the inclusion of the new cryptoassets to the Melon asset universe had a significant impact on the efficient frontier. We examined the period between March 01, 2019 and July 16, 2020 as well as the sub-period between February 16, 2020 until July 16, 2020. The results of the empirical analysis reveal that, for the former period, the cryptoassets ANT, LINK and WSaiDai offer diversification benefits for Melon managers. However, the results of the latter period indicate that none of the newly added cryptoassets provide diversification benefits for Melon managers. Due to overall conditions and the lack of data, the empirical results are regarded as critical.

References

Adams, H. (2018), 'Uniswap Whitepaper'. URL https://hackmd.io/@Uniswap/HJ9jLsfTz, accessed August 15, 2020.

Antonopoulos, A. M. and Wood, G. (2018), 'Mastering Ethereum: Building SmartContracts and DApps', O'Reilly Media, Inc. URL https://github.com/ethereumbook/ethereumbook, accessed August 15, 2020.

Avantgarde Finance Ltd (2020), 'Introducing the Melon Terminal'. URL https://medium.com/melonprotocol/introducing-the-melon-terminal-a81a9c5e766e, accessed August 15, 2020.

Berentsen, A. and Schär, F. (2017), 'Bitcoin, Blockchain und Kryptoassets', *BoD - Books on Demand, Norderstedt* .

Binance Academy (2020), 'Wrapped Ether (WETH)'. URL https://www.binance.vision/glossary/wrapped-ether, accessed August 15, 2020.

Briere, M., Oosterlinck, K. and Szafarz, A. (2015), 'Virtual currency, tangible return: Portfolio diversification with bitcoin', *Journal of Asset Management* **16**(6), 365–373.

Buterin, V. (2013), 'A Next-Generation Smart Contract and Decentralized Application Platform'. URL https://github.com/ethereum/wiki/wiki/White-Paper, accessed August 15, 2020.

Buterin, V. and Griffith, V. (2017), 'Casper the Friendly Finality Gadget'.

CoinGecko (2020), 'Market Data'. URL https://www.coingecko.com/de/methodik, accessed August 15, 2020.

Coinlore (2020), 'Smart Contract Platforms'. URL https://www.coinlore.com/de/smart-contract-platforms, accessed August 15, 2020.

Coinmarketcap (2020*a*), 'Top 100 Cryptocurrencies by Market Capitalization'. URL https://coinmarketcap.com, accessed August 15, 2020.

Coinmarketcap (2020*b*), 'Top 100 Tokens by Market Capitalization'. URL https://coinmarketcap.com/tokens/, accessed August 15, 2020.

ConsenSys (2019), 'The Thirdening: What You Need To Know'. URL https://media.consensys.net/the-thirdening-what-you-need-to-know-df96599ad857, accessed August 15, 2020.

DeFi Pulse (2019), 'DeFi on Ethereum – What is it?'. URL https://defipulse.com/blog/defi-on-ethereum/, accessed August 15, 2020.

Defi Pulse (2020), 'Total Value Locked (USD) in DeFi'. URL https://defipulse.com, accessed August 15, 2020.

Eisl, A., Gasser, S. and Weinmayer, K. (2015), 'Caveat emptor: Does bitcoin improve portfolio diversification?', *Available at SSRN 2408997*.

El Isa, M. (2018), 'Melonomics Part 2: The Melon Engine'. URL https://medium.com/melonprotocol/melonomics-part-2-the-melon-engine-48bcb0dae65, accessed August 15, 2020.

Ethereum Homestead (2020), 'Account Types, Gas, and Transactions'. URL https://ethereum-homestead.readthedocs.io/en/latest/contracts-and-transactions/account-types-gas-and-transactions.html, accessed August 15, 2020.

Etherscan (2020), 'Melon Token'. URL https://etherscan.io/token/0xec67005c4e498ec7f55e092bd1d35cbc47c91892, accessed August 15, 2020.

EthHub (2019), 'ERC20'. URL https://docs.ethhub.io/built-on-ethereum/erc-token-standards/erc20/, accessed August 15, 2020.

Fausch, J. and Ankenbrand, T. (2019), 'IFZ/AMP Asset Management Study 2019: An Overview of Swiss Asset Management'. URL https://sfama-cms.cdn.prismic.io/sfama-cms%2F0ff76f12-ba8e-41a9-acc5-e5c6ca15b0c1_ifz+amp+studie+2019_final.pdf, accessed August 15, 2020.

Gorilla Funds (2019), 'Introducing Gorilla Funds'. URL https://medium.com/melonprotocol/introducing-gorilla-funds-b1d6cb3ffe5d, accessed August 15, 2020.

Grant Thornton (2011), 'How do you start a hedge fund? The new era of hedge fund creationand operational management'. URL https://www.managedfunds.org/wp-content/uploads/2012/03/Starting-a-hedge-fund-GrantThornton-Stonegate-Capital-Dec-2011.pdf, accessed August 15, 2020.

Graph, T. (2019), 'Melon'. URL https://thegraph.com/explorer/subgraph/melonproject/melon, accessed August 15, 2020.

Grayscale Building Blocks (2020), 'An introduction to Ethereum'. URL https://grayscale.co/wp-content/uploads/2020/02/Grayscale-Building-Blocks-Ethereum-February-2020.pdf, accessed August 15, 2020.

Huberman, G. and Kandel, S. (1987), 'Mean-variance spanning', *The Journal of Finance* **42**(4), 873–888.

Jacobs, T. (2020*a*), 'Accounting.sol'. URL https://github.com/melonproject/protocol/blob/master/src/fund/accounting/Accounting.sol, accessed August 15, 2020.

Jacobs, T. (2020*b*), 'Feemanager.sol'. URL https://github.com/melonproject/protocol/blob/master/src/fund/fees/FeeManager.sol, accessed August 15, 2020.

Jacobs, T. (2020*c*), 'Participation.sol'. URL https://github.com/melonproject/protocol/blob/master/src/fund/participation/Participation.sol, accessed August 15, 2020.

Jacobs, T. (2020*d*), 'Registry.sol'. URL `https://github.com/melonproject/protocol/blob/master/src/version/Registry.sol`, accessed August 15, 2020.

Jacobs, T. (2020*e*), 'Shares.sol'. URL `https://github.com/melonproject/protocol/blob/master/src/fund/shares/Shares.sol`, accessed August 15, 2020.

Jacobs, T. (2020*f*), 'Version.sol'. URL `https://github.com/melonproject/protocol/blob/master/src/version/Version.sol`, accessed August 15, 2020.

Jacobs, T. and Casey, S. (2020*a*), 'Hub.sol'. URL `https://github.com/melonproject/protocol/blob/master/src/fund/hub/Hub.sol`, accessed August 15, 2020.

Jacobs, T. and Casey, S. (2020*b*), 'Trading.sol'. URL `https://github.com/melonproject/protocol/blob/master/src/fund/trading/Trading.sol`, accessed August 15, 2020.

Jacobs, T. and Casey, S. (2020*c*), 'Vault.sol'. URL `https://github.com/melonproject/protocol/blob/master/src/fund/trading/Trading.sol`, accessed August 15, 2020.

Jarque, C. M. and Bera, A. K. (1980), 'Efficient tests for normality, homoscedasticity and serial independence of regression residuals', *Economics letters* **6**(3), 255–259.

Jentzsch, C. (2016), 'Decentralized autonomous organization to automate governance', *White paper, November* .

Kan, R. and Zhou, G. (2012), 'Tests of mean-variance spanning', *Annals of Economics and Finance* **13**(1), 139–187.

KPMG (2013), 'The cost of compliance'. URL `https://home.kpmg/content/dam/kpmg/pdf/2014/07/Cost-of-Compliance.pdf`, accessed August 15, 2020.

Luu, L., Chu, D.-H., Olickel, H., Saxena, P. and Hobor, A. (2016), Making smart contracts smarter, *in* 'Proceedings of the 2016 ACM SIGSAC conference on computer and communications security', pp. 254–269.

Melon (2018), 'Melon Council Statutes'. URL `https://docs.google.com/document/d/1tLEa2rM-mmzHQkv41i0E9o0g4aHGrEyYvMfNjEPid2s/edit`, accessed August 15, 2020.

Melon (2020*a*), 'Ecosystem overview'. URL `https://melonprotocol.com/docs/ecosystem/`, accessed August 15, 2020.

Melon (2020*b*), 'Exchange Adapters'. URL `https://docs.melonport.com/chapters/exchanges.html#exchange-adapters`, accessed August 15, 2020.

Melon (2020*c*), 'Fund'. URL `https://docs.melonport.com/chapters/fund.html#fund`, accessed August 15, 2020.

Melon (2020*d*), 'Fund Factory'. URL `https://docs.melonport.com/chapters/factory.html#factory`, accessed August 15, 2020.

Melon (2020*e*), 'Governance'. URL `https://docs.melonport.com/chapters/governance.html#governance`, accessed August 15, 2020.

Melon (2020*f*), 'Governance'. URL `https://melonprotocol.com/docs/governance/`, accessed August 15, 2020.

Melon (2020*g*), 'Melon as a solution'. URL `https://melonprotocol.com/docs/solution/`, accessed August 15, 2020.

Melon (2020*h*), 'Melon Engine: buy-and-burn model'. URL `https://docs.melonport.com/chapters/melon_engine.html#melon-engine-buy-and-burn-model`, accessed August 15, 2020.

Melon (2020*i*), 'Melon Protocol Reference'. URL `https://docs.melonport.com/#melon-protocol-reference`, accessed August 15, 2020.

Melon (2020*j*), 'Pricing'. URL https://docs.melonport.com/chapters/prices.html#pricing, accessed August 15, 2020.

Melon (2020*k*), 'Registry'. URL https://docs.melonport.com/chapters/registry.html#registry, accessed August 15, 2020.

Melon (2020*l*), 'Version'. URL https://docs.melonport.com/chapters/version.html#version, accessed August 15, 2020.

Melon Council (2019), 'Melon Pricefeed Upgrade'. URL https://medium.com/melonprotocol/melon-pricefeed-upgrade-81c5c8febae2, accessed August 15, 2020.

Melon Terminal (2020), 'Melon Terminal'. URL https://melon.avantgarde.finance/, accessed August 15, 2020.

Nakamoto, S. (2008), 'Bitcoin: A Peer-to-Peer Electronic Cash System'. URL https://bitcoin.org/bitcoin.pdf, accessed August 15, 2020.

Orthwein, J. (2018), 'Management and Performance Fees in a Melon Fund'. URL https://medium.com/melonprotocol/management-and-performance-fees-in-a-melon-fund-f1df2a26abd9, accessed August 15, 2020.

Peters, G. W. and Panayi, E. (2016), Understanding modern banking ledgers through blockchain technologies: Future of transaction processing and smart contracts on the internet of money, *in* 'Banking beyond banks and money', Springer, pp. 239–278.

PWC (2018), 'Asset & Wealth Management Revolution: Embracing Exponential Change'. URL https://www.pwc.com/gx/en/asset-management/asset-management-insights/assets/awm-revolution-full-report-final.pdf, accessed August 15, 2020.

Roth, J., Schär, F. and Schöpfer, A. (2019), 'The Tokenization of Assets: Using Blockchains for Equity Crowdfunding', *Available at SSRN 3443382* .

RStudio (2020), 'The R Project for Statistical Computing'. URL https://www.r-project.org/, accessed August 15, 2020.

Schmitz, T. and Hoffmann, I. (2020), 'Re-evaluating cryptocurrencies' contribution to portfolio diversification–a portfolio analysis with special focus on german investors', *arXiv preprint arXiv:2006.06237* .

Schär, F. (2020), 'Decentralized Finance: On Blockchain- and Smart Contract-based Financial Markets'.

State of the DApp (2020), 'DApp Statistics'. URL https://www.stateofthedapps.com/stats, accessed August 15, 2020.

Vanini, P. (2019), 'Asset management', *Available at SSRN 2710123* .

Wood, G. et al. (2014), 'Ethereum: A secure decentralised generalised transaction ledger', *Ethereum project yellow paper* **151**(2014), 1–32.

Wöhrer, M. and Zdun, U. (2018), Smart contracts: security patterns in the ethereum ecosystem and solidity, *in* '2018 International Workshop on Blockchain Oriented Software Engineering (IWBOSE)', IEEE, pp. 2–8.

Zenk, J. (2018), 'Melonomics Part 3: Counting Melons'. URL https://medium.com/melonprotocol/melonomics-part-3-counting-melons-7632afad844c, accessed August 15, 2020.

Zenk, J. (2019*a*), 'MELON V1.0: ZAHREDDINO'. URL https://medium.com/melonprotocol/melon-v1-0-zahreddino-60105f51988d, accessed August 15, 2020.

Zenk, J. (2019*b*), 'MFP #2 Avantgarde Finance'. URL https://github.com/melonproject/MFP/blob/MFP2/Pending/MFP2/MFP2-avantgarde-finance.md, accessed August 15, 2020.

Zenk, J. (2019*c*), 'MFP: Melon Funding Proposals'. URL https://github.com/melonproject/MFP, accessed August 15, 2020.

7 Appendix

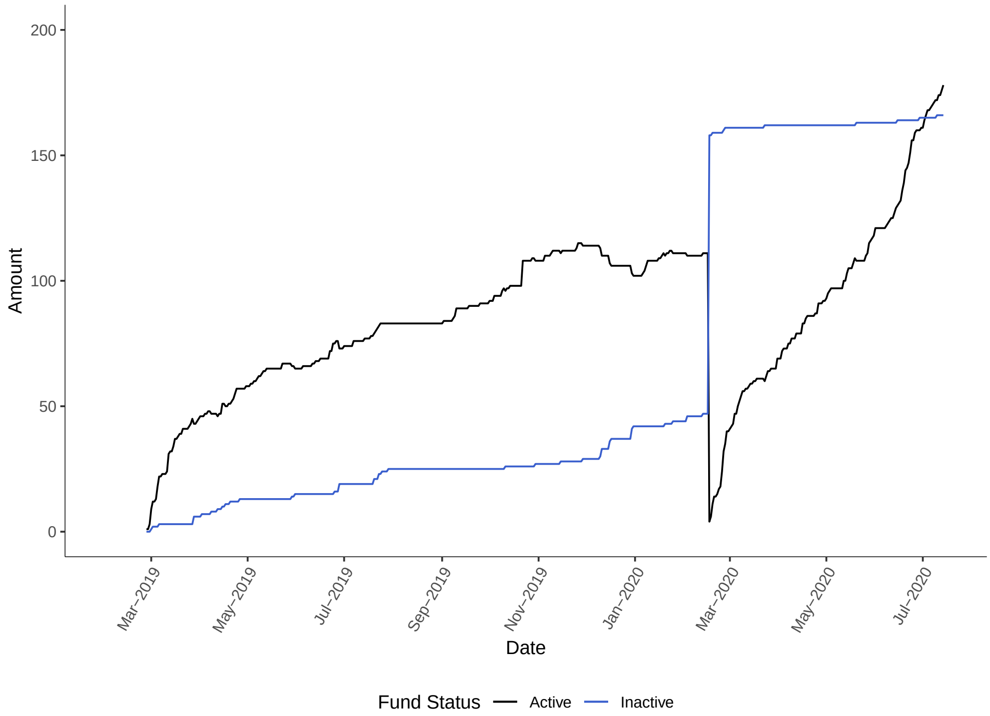


Figure 15: Development of Active and Inactive Melon Funds. Source: Melon API, own illustration.

	0x	**Augur**	**BAT**	**KNC**	**MKR**	**MLN**	**USDC**	**WBTC**	**WETH**	**ANT**	**LINK**	**MANA**	**iExec**	**WSaiDai**	**Ren**
0x	1.00														
Augur	0.55	1.00													
BAT	0.65	0.55	1.00												
KNC	0.49	0.49	0.54	1.00											
MKR	0.56	0.56	0.60	0.51	1.00										
MLN	0.42	0.44	0.42	0.39	0.46	1.00									
USDC	-0.07	-0.06	-0.05	-0.03	-0.09	-0.08	1.00								
WBTC	0.46	0.48	0.47	0.38	0.56	0.37	-0.18	1.00							
WETH	0.64	0.63	0.67	0.54	0.73	0.53	-0.14	0.67	1.00						
ANT	0.48	0.47	0.46	0.42	0.51	0.40	-0.16	0.45	0.61	1.00					
LINK	0.47	0.45	0.54	0.41	0.53	0.35	-0.11	0.43	0.62	0.43	1.00				
MANA	0.59	0.56	0.65	0.52	0.58	0.41	-0.08	0.49	0.64	0.50	0.56	1.00			
iExec	0.51	0.48	0.54	0.46	0.53	0.39	-0.04	0.47	0.60	0.49	0.50	0.53	1.00		
WSaiDai	-0.05	-0.09	-0.12	-0.07	-0.15	-0.07	0.09	-0.04	-0.03	-0.08	-0.14	-0.13	-0.05	1.00	
Ren	0.47	0.47	0.49	0.41	0.51	0.34	-0.05	0.43	0.54	0.42	0.48	0.48	0.42	-0.10	1.00

The table presents the pairwise correlation coefficients of the benchmark assets and test assets based on daily log returns from March 1, 2019 to July 16, 2020

Table 12: Melon Asset Universe Correlation Matrix Entire Period.

	0x	Augur	BAT	KNC	MKR	MLN	USDC	WBTC	WETH	ANT	LINK	MANA	iExec	WSaiDai	Ren
0x	1.00														
Augur	0.68	1.00													
BAT	0.74	0.78	1.00												
KNC	0.61	0.67	0.68	1.00											
MKR	0.59	0.70	0.74	0.66	1.00										
MLN	0.44	0.59	0.53	0.50	0.50	1.00									
USDC	0.07	-0.10	-0.12	-0.08	-0.04	-0.09	1.00								
WBTC	0.69	0.76	0.86	0.67	0.76	0.59	-0.04	1.00							
WETH	0.70	0.78	0.87	0.71	0.79	0.61	-0.09	0.95	1.00						
ANT	0.60	0.66	0.73	0.66	0.63	0.53	-0.09	0.80	0.81	1.00					
LINK	0.66	0.71	0.81	0.63	0.73	0.55	-0.09	0.84	0.87	0.70	1.00				
MANA	0.72	0.76	0.86	0.67	0.72	0.59	-0.10	0.85	0.87	0.73	0.81	1.00			
iExec	0.64	0.69	0.77	0.65	0.67	0.59	-0.09	0.77	0.79	0.71	0.81	0.78	1.00		
WSaiDai	-0.19	-0.28	-0.34	-0.21	-0.38	-0.25	0.21	-0.28	-0.28	-0.27	-0.32	-0.32	-0.32	1.00	
Ren	0.59	0.69	0.72	0.65	0.66	0.60	-0.07	0.76	0.77	0.68	0.75	0.71	0.71	-0.25	1.00

The table presents the pairwise correlation coefficients of the benchmark assets and test assets based on daily log returns from February 16 to July 16, 2020

Table 13: Melon Asset Universe Correlation Matrix Sub-Period.