SUI Project

▼ Current coin filtering process for liquidity analysis:

1. **Select Initial Coins:**
    - Retrieve all SUI coins.
    - Sort by:
        - Market Cap
        - Verification Status (isVerified)
        - Fully Diluted Valuation (FDV)
        - Holder Count
    - Select the **top 30,000** coins.
2. **Apply Filters:**
    - Include only coins with **price > 0**.
    - Ensure coins meet **at least one** of the following conditions:
      a)
      Market Cap is not empty.
      b)
      FDV > 1,000,000 and **Holder Count > 1,000**.
3. **Identify Active SUI Pools:**
    - Filter for coins with **active SUI pools**.
4. **Filter Pools by Liquidity:**
    - Include pools with **USD pool amount ≥ 20,000**.
5. **Check for Tick Availability:**
    - Retain only pools where **ticks** are available.
6. **Filter By Slippage**
   Based on that, other filtration strategies can be added, for example, USD

amount that reaches 2% slippage > 1000$.
For example, the only following coins satisfy the above criteria:

```
wUSDC
DEEP
HIPPO
CETUS
NAVX
NS
SCA
afSUI
vSUI
WBTC
ETH
BUCK
MEMEFI
WSOL
LOFI
```

Methodology

The main file of the project is **sui-analysis.ipynb**. The file performs the following tasks:

Fetch coin data

1. Fetch all pages of the response from **Blockberry API: getCoins** (https://docs.blockberry.one/reference/getcoins) into a dataframe.
2. Sort by `marketCap`, `isVerified`, `fdv`, and `holdersCount`, and select the first 30,000 coins.
3. Save the data into a CSV file.

This process takes a long time due to the Blockberry API's bandwidth limit—one request every 15 seconds.

Their message on the website:

"*Temporary hard limits on bandwidth have been introduced due to high load: we recommend sending requests no more than once every 15 seconds. Our engineers are actively working to change this. Stay in touch, we will let you know.*"

Because of this limitation, the initially loaded CSV is reused in further iterations.

Load coins CSV

1. Load the coins from the CSV file.
2. Filter the coins that have a `price > 0` and meet one of the following conditions:
   a)
   `marketCap` is not empty
   b)
   `fdv > 1,000,000` and `holdersCount > 1,000`

Get current SUI price

The SUI price is used in most calculations, so it is fetched from **Blockberry API: getCoinByCoinType** (https://docs.blockberry.one/reference/getcoinbycointype).

Fetch data on all pools for the selected coins

For liquidity analysis, the Cetus protocol was used. Other exchanges like Turbos and BlueMove have no liquidity in custom tokens - only stablecoins.

For this part, the **fetch_pools.ts** file is called as a subprocess. This file performs the following tasks:

1. Uses **initCetusSDK**, specifically the **getPoolByCoins()** method (https://cetus-1.gitbook.io/cetus-developer-docs/developer/via-sdk/features-available/get-clmm-pools) to fetch data on all pools for the coin and SUI.
2. Calculates **usdPoolAmount** using the formula:

   (SUI amount * currentPrice + coinB amount) * suiPrice, considering the decimals of the coins.
3. Saves the data as JSON files in the **pools** folder.

The **fetch_pools** function is then used to load these JSON files into a Python dataframe (picking the most recent file for each coin). Only coins with active SUI

pools are kept at this step.

Select pools

The fetched pools are combined into a single dataframe and filtered: only pools with **usdPoolAmount ≥ 20,000** are kept.

Fetch ticks for pools

For this part, the **fetch_liquidity_bypool.ts** file is called as a subprocess. This file performs the following tasks:

1. Uses the **getPool()** method from **initCetusSDK** to fetch tick data for a pool by its address.
2. Calculates liquidity depth using the **getCoinAmountFromLiquidity()** method(https://cetus-1.gitbook.io/cetus-developer-docs/developer/via-sdk/features-available/liquidity-and-coin-amounts-calculation). This involves iterating through all ticks to compute accumulated liquidity, accumulated liquidity in USD, and accumulated amounts of coins (initialized in the **liquidity.ts** file).
3. Calculates trade sizes for slippage thresholds (currently 0.5% and 2%) using the **calculateRates()** method (https://cetus-1.gitbook.io/cetus-developer-docs/developer/via-sdk/features-available/preswap). This involves iterating through all ticks to determine the input and output amounts of coins (initialized in the **price_impacts.ts** file).
4. Saves the data as JSON files in the **ticks** folder.

The **fetch_ticks** function is then used to load these JSON files into the **PoolTicks** structure (picking the most recent file for each coin).

Draw graphs for liquidity

For each pool, the following visualizations are created:

1. **Main info (Table) -** Main information about the pool

Main info

Date	2025-03-18 18:32:06
Current price	0.043629774927257566
Amount of CETUS	17285299.8649
Amount of SUI	671251.2584
TVL	$3132115.6743
USD Amount of CETUS	$1664882.7950
USD Amount of SUI	$1467232.8793

2. **Trade Sizes and Slippage (Table)** - Input and output amounts of coins (in original tokens and USD) for slippage thresholds in both directions.

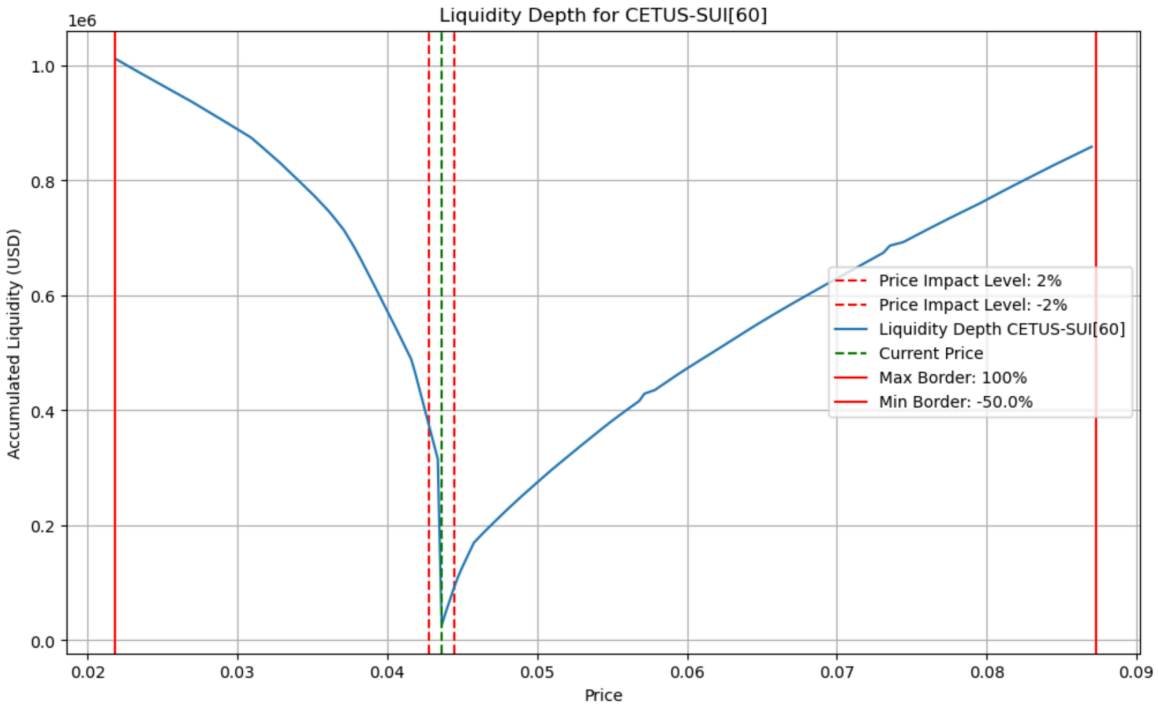
Trade Sizes Reaching 0.5% Slippage

Direction	Amount In	Token In	USD In	Amount Out	Token Out	USD Out
↓ A to B ↓	2.48135e+06	CETUS	$242504.36	107720	SUI	$241292.60
↑ B to A ↑	13303.4	SUI	$29799.63	303581	CETUS	$29669.19

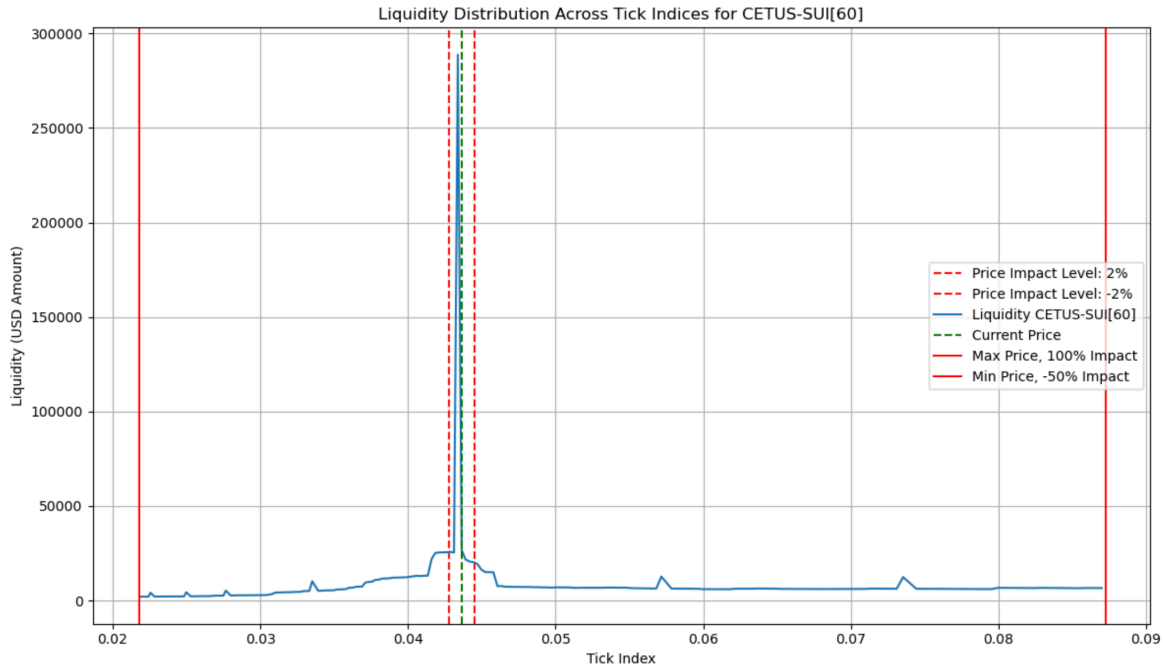
Trade Sizes Reaching 2% Slippage

Direction	Amount In	Token In	USD In	Amount Out	Token Out	USD Out
↓ A to B ↓	3.52097e+06	CETUS	$344106.99	152511	SUI	$341625.43
↑ B to A ↑	37338.7	SUI	$83638.76	846447	CETUS	$82723.85

3. **Liquidity Depth (Graph)** - Accumulated liquidity depth in USD in both directions from the current price. This indicates the 2% price impact level and the minimum (-50%) and maximum (+100%) reasonable price boundaries.



4. **Liquidity Distribution (Graph)** - Liquidity at each tick index.

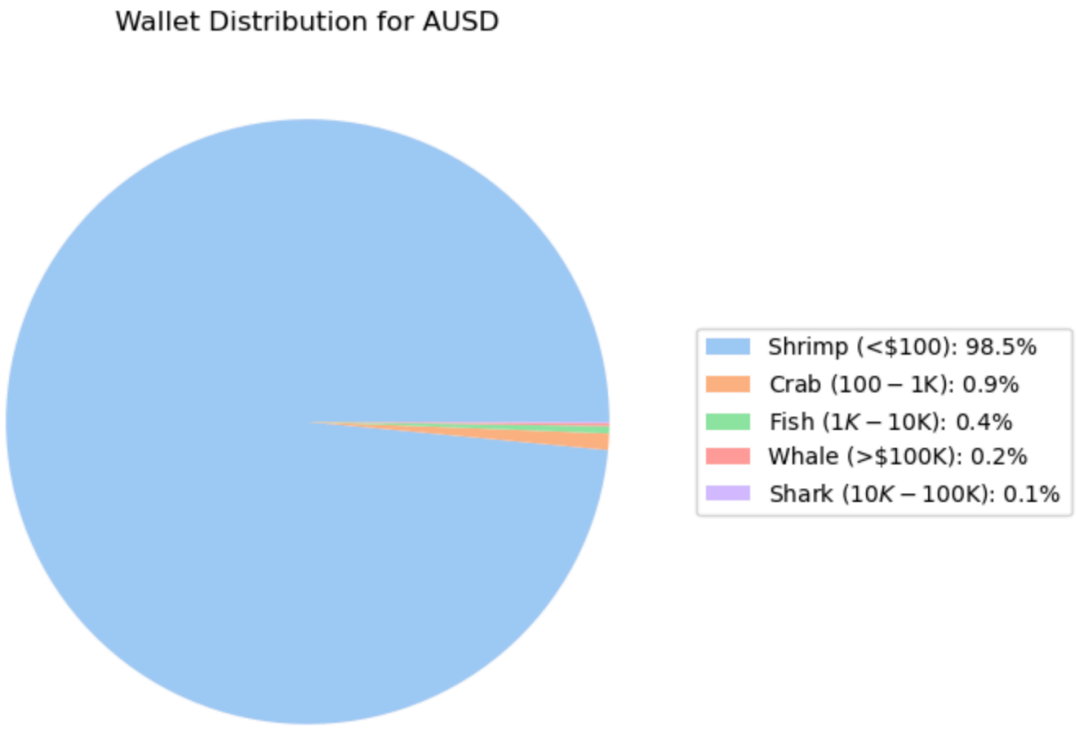


Analyse holders

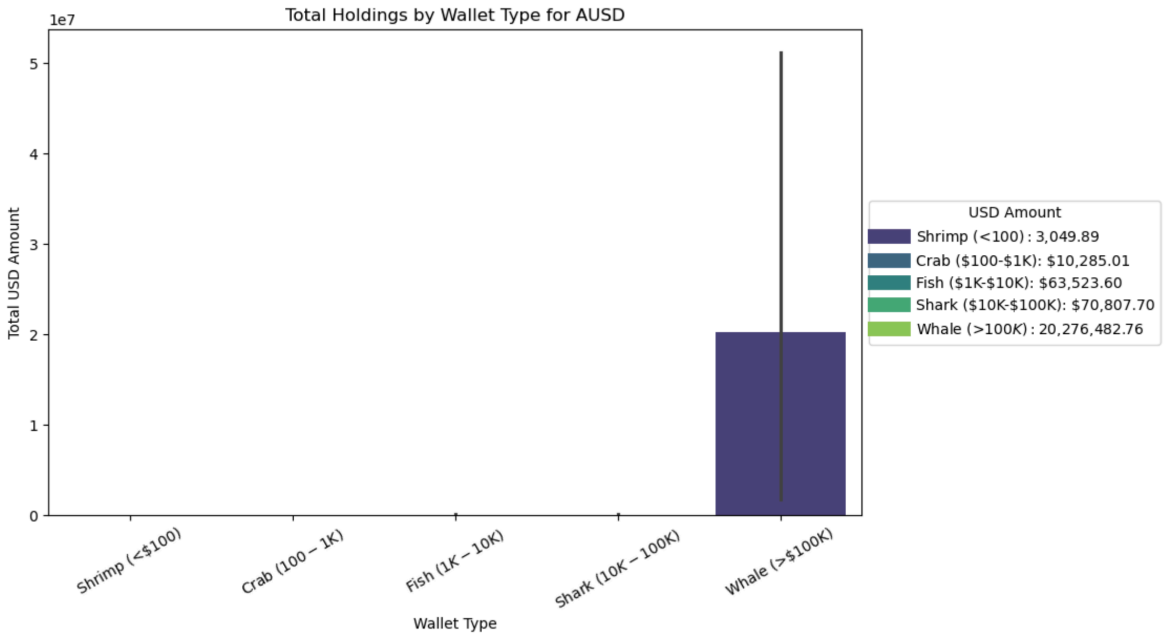
Holders data is fetched for coins using Blockberry API's **getHoldersByCoinType()** method (https://docs.blockberry.one/reference/getholdersbycointype).

Unfortunately, this process is very slow due to the API's bandwidth limits and the maximum page size of 50 rows per page. As examples, the following visualizations are created for a few selected coins:

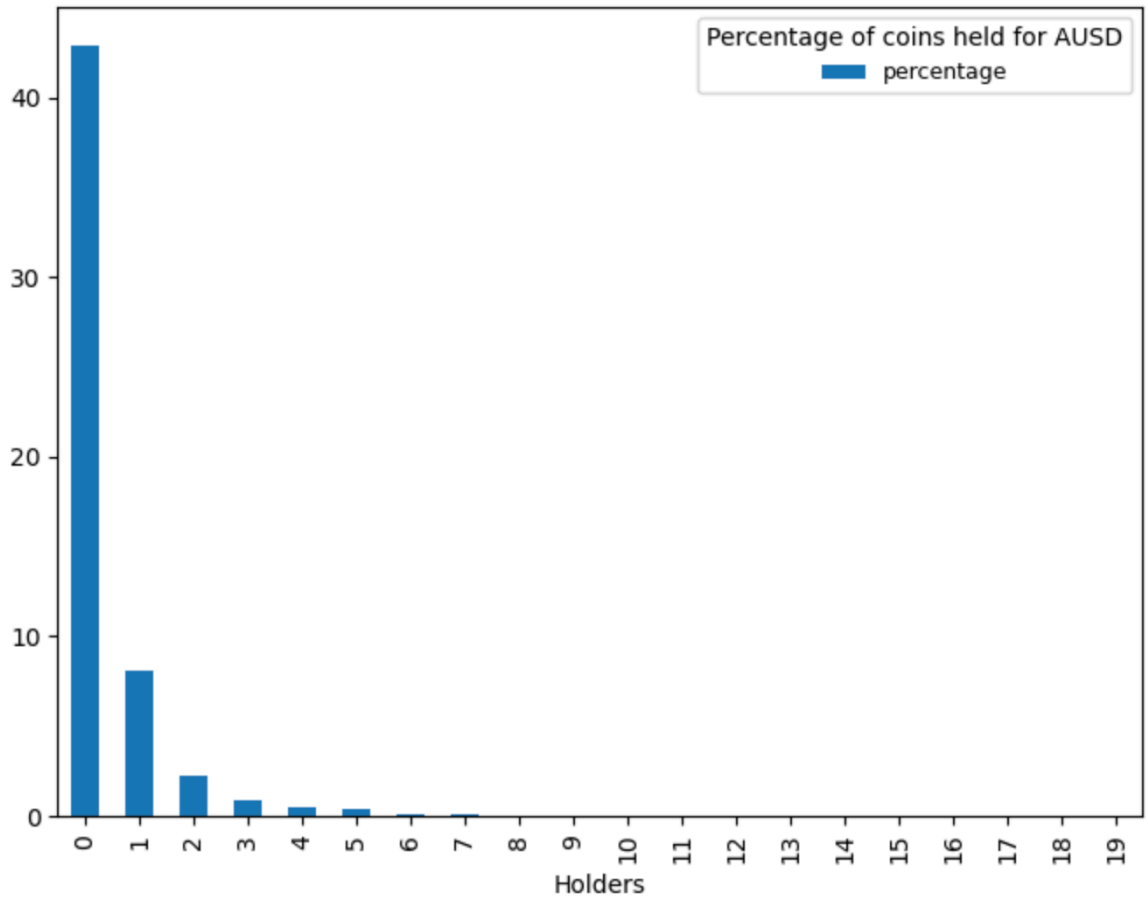
1. **Wallet Distribution (Pie Chart)** - The number of holders by wallet type.



2. **Holdings by Wallet Type (Bar Chart)** - The USD value of a coin held by different wallet types.



3. **Percentage of Coins Held (Bar Chart)** - The percentage of coins held by the top 20 holders.


Percentage of coins held for AUSD
percentage
40
30
20
10
0
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
Holders