Trading Portfolio Analysis Service Public Documentation

Maria Stuebner, Michael Tulskikh,
Nicholas Bentivoglio, and Wesley Rollings.

AstraBit

Research and Development
Updated: September 2025

Abstract

This guide is intended as documentation of the mathematics behind AstraBit's latest Portfolio Analysis system. The topics explored within this document range from calculating the Sharpe ratio to computing a portfolio's holding period return to graphing the Efficient Frontier to finding a user's optimal portfolio. Indeed, the features of the Portfolio Analysis system are split into (1) metrics and (2) graphs. The metrics include measure of risk and return such as standard deviation, beta, alpha, and holding period return. The graphs include features like the efficient frontier, the optimal portfolio, individual asset positions, and the Capital Allocation Line (CAL).

Contents

0 Introduction

Modern Portfolio theory is the investment theory that aims to assemble an asset portfolio that maximizes expected return for a given level of risk. [Har83] One of the assumptions of this theory is that investors are risk-averse, meaning for a given level of expected return, investors will prefer the least risky portfolio. [RC88] AstraBit's new Portfolio Analysis system seeks to implement modern portfolio theory in the digital asset space, providing users with portfolio metrics and graphs that build a clearer picture of where their portfolio is currently and how an optimized portfolio would look. Indeed, the features of the Portfolio Analysis system are split into (1) metrics and (2) graphs.

0.1 Portfolio Analysis System: Metrics

The metrics section is focused on precise measures of risk, return, and combinations of the two. We calculate these metrics both for a users' individual assets and portfolio as well as for certain market benchmarks, such as the S&P 500 and AstraBit's new proprietary index, the ASTRA100 (expanded upon in Section 6). The goal is to show users how their assets and portfolio line up with the performance of both the legacy market and the digital asset market. Thus, when possible, each metric is calculated on the following five levels

- asset (or bot) level,
- portfolio level,
- coin level,
- ASTRA100 level, and
- S&P 500 level.

Since our users' portfolios consist of bots, rather than specific assets, we actually perform the asset level calculation at the bot level. The difference here is that modern portfolio theory typically evaluates assets on the stock or coin levels, for example, whereas we define asset to be an individual bot, which can trade the same coin as another bot, but with a different strategy and parameters. Below we present the list of metrics currently calculated for a user's portfolio from v1.0, including two features that were released as part of v1.1: an asset correlation matrix and an asset allocation table that shows annualized expected return, standard deviation, and Sharpe ratios, all corresponding to the values shown on the Efficient Frontier graph.

- Daily Mean Return
- Beta (β)
- Capital Asset Pricing Model (CAPM)
- Correlation Matrix
- Asset Allocation Table
- Downside Deviation
- Holding Period Return
- Jensen's Alpha (α_J)
- Pure Alpha (α_P)
- Sharpe Ratio
- Sortino Ratio
- Standard Deviation

These metrics are sorted by measures of risk, return, and combined measures, and their implementations are explained in Sections 1, 2, and 3 of this document.

0.1.1 Scoring System

In addition to providing users with performance measures for their bots and portfolios, we have also implemented a scoring system that provides users with context for the displayed measures. Each calculated metric is individually score, and we compile these scores and specifically display three overall scores: (1) an overall performance score, (2) an overall risk score, and (3) an overall combined score. These scores are represented as percentages, on a scale from 0 to 100%. While the performance and combined score consider 100% to be the maximum score, the risk score is inverted, since, under the assumptions of modern portfolio theory, 0% is the maximal risk score possible.

The scoring feature is discussed in more depth in Section 5, but the main purpose of the scoring is to provide users with AstraBit's evaluation of the numerical values they will see calculated on their dashboards. The AstraBit score is an attempt to responsibly provide users with industry context and our own analysis beside the measures of performance that are already provided for them. In calculating scores, we rely both on internally discussed benchmarks and scores as well as on market comparisons and evaluations, considering both the legacy and digital markets.

0.1.2 Market Comparisons

Out of the twelve metrics that we calculate, five of them involve a market comparison: (1) beta, (2) CAPM, (3) Jensen's alpha, (4) downside deviation, and (5) the Sortino ratio. The calculation of the last two requires the minimum accepted return (MAR), which is set around a market benchmark.

In order to provide users with the most flexibility in analyzing their portfolios, we give them the opportunity to choose whethere to calculate these metrics against the legacy market (via the S&P 500) *or* the digital market (via the ASTRA100). This is implemented via a toggle feature, and provides for a more detailed analysis of the calculated metrics.

0.2 Portfolio Analysis System: Graphs

On top of the metrics and scoring system, we also introduce a new graph in v1.0 of the Portfolio Analysis system: the Efficient Frontier graph. The efficient frontier is a concept from modern portfolio theory, calculated as a curve that displays the lowest risk for a given level of expected return given the assets in a user's portfolio. We decided to name our graph after the efficient frontier because our own calculated efficient frontier is the starting point for all the other features displayed. Indeed, the theme of portfolio optimization is echoed by all the features of our graph:

- annualized Sharpe ratios,
- bot positions,
- capital allocation line,
- current portfolio position,
- efficient frontier,
- optimal portfolio position,
- portfolio weighting, and
- risk-free rate.

The features above are discussed in more depth in Section 4 of this document. Following a theoretical derivation of the efficient frontier, we present various arguments for how we have chosen to calculate the curve, and expand upon the ways in which the other features are calculated and graphed.

1 Metrics: Measures of Return

Each calculation in this document is performed for an inputted period, with a period start date and period end date. Let us define a global variable N, equal to the total number of days in the period. We will refer to this variable through this document.

1.1 Holding Period Return

The holding period return (HPR) is given by

$$\text{HPR} = \frac{P_1 - P_0 + D_1}{P_0},$$

where P_0 is the initial purchase price of the instrument, P_1 is the price received for the instrument at its maturity, and D_1 is the cash distribution paid by the instrument at its maturity (i.e., interest). For v1.0, we are implementing the holding period return with $D_1 = 0$, i.e. with no interest. Thus, the HPR becomes the difference between the close and open price divided by the open price.
We also calculate an annualized HPR (AHPR) by:

$$\text{AHPR} = (1 + \text{HPR})^{\frac{N}{\text{Period}}} - 1$$

where N is the number of trading days in a year, and Period refers to the number of days over which the HPR is calculated.

1.1.1 Bot-level

On the bot level, the price at maturity (P_1) is given by the USD value of the bot after the last trade on the inputted end date while the initial price (P_0) is given by the USD value of the initial allocation of the bot, i.e. the USD value of the bot before the first trade on the start date.

1.1.2 Portfolio level

On the portfolio level, the price at maturity (P_1) is given by the USD value of the total portfolio after the last bot trade on the inputted end date while the initial price (P_0) is given by the USD value of the initial allocation of the portfolio, i.e. the sum of all of the USD values of the bots in the portfolio before the first trade on the start date.

1.1.3 Coin level

For coins, we use the close price of the specific coin on the end date as the price at maturity (P_1) and the open price of the coin on the start date as the initial price (P_0). In the case of missing candle data, we use the closest candles to the start and end dates.

1.1.4 ASTRA100 level

For the ASTRA100 holding period return, we use the price of the index on the end date as the price at maturity (P_1) and the price of the index on the start date as the initial price (P_0).

1.1.5 S&P 500 level

Similarly, for the S&P 500 holding period return, we use the price of the index on the end date as the price at maturity (P_1) and the price of the index on the start date as the initial price (P_0).

1.2 Mean Return

The annualized mean return represents the annualization of the mean daily return. The difference between how the mean return is calculated across levels has to do with what daily returns are used. In all cases, we annualize by compounding the daily return according to the formula:

$$AR = (1 + \overline{R})^N - 1$$

where N is the number of trading days (generally 365 for crypto assets, and 252 for traditional assets).

1.2.1 Bot-level

The mean return of a bot is given by the mean of daily returns R_i

$$\overline{R_{\text{bot}}} = \frac{1}{N} \sum_{i=1}^{N} (R_{\text{bot}})_i,$$

where $(R_{\text{bot}})_i$ is the daily return of day i, given by

$$(R_{\text{bot}})_i = \frac{(V_{\text{bot}})_i - (V_{\text{bot}})_{i-1}}{(V_{\text{bot}})_{i-1}}$$

where $(V_{\text{bot}})_i$ is the total value of the bot on day i. We consider $(V_{\text{bot}})_0$ to be the initial allocation of the bot.

1.2.2 Portfolio level

The mean return of a portfolio is given by the weighted average of the expected returns of individual assets, calculated by the above formula. The weights for this average are given by the proportion of the total dollar value traded by individual bots to the total dollar value of a user's portfolio.

Portfolio weights More specifically, we calculate daily portfolio weights, which are then used to calculate daily portfolio returns, which are subsequently averaged for a portfolio mean return. For a given input date, we collect the initial USD allocation for the bot as well as the USD PnL for each series of trades that make up a total "performance". We can thus regularly update each bot allocation so that we can also update the portfolio weights, i.e. the proportion that each bot allocation makes up out of the total portfolio allocation. However, because we currently update portfolio statistics daily, we also only update the portfolio weights daily. We thus get the end of day bot allocation for each bot, and then divide this value by the sum of all the end-of-day bot allocations (to get the total end-of-day portfolio allocation) in order to get each entry in weights vector. Let B be the number of bots in the portfolio for the given period of time.

$$w_i = \frac{D_i}{\sum_{i=1}^{B} D_i},$$

where D_i represents the end-of-day USD value of the ith bot allocation. Then, we have the formula

$$\overline{R_{\text{portfolio}}} = \sum_{i}^{B} w_i \overline{R_i}.$$

1.2.3 Coin level

The mean return of a coin is given by the mean of daily returns

$$\overline{R_{\text{coin}}} = \frac{1}{N} \sum_{i=1}^{N} (R_{\text{coin}})_i,$$

where $(R_{\text{coin}})_i$ is the daily return of day i. We use the coin's daily holding period return as it's daily return, which is given by the close price (P_i^c) on day i of the candle minus the open price (P_i^o) on day i divided by the open price:

$$(R_{\text{coin}})_i = \frac{P_i^c - P_i^o}{P_i^o}.$$

1.2.4 ASTRA100 level

The mean return for the ASTRA100 index is given by the mean of daily returns

$$\overline{R_{\text{ASTRA}}} = \frac{1}{N} \sum_{i=1}^{N} (R_{\text{ASTRA}})_i,$$

where $(R_{\text{ASTRA}})_i$ is the daily return value for day i.

1.2.5 S&P 500 level

The mean return for the S&P 500 is given by the mean of daily returns

$$\overline{R_{\text{S\&P}}} = \frac{1}{N} \sum_{i=1}^{N} (R_{\text{S\&P}})_i,$$

where $(R_{\text{S\&P}})_i$ is the daily return value for day i.

1.3 Risk-Free Rate

The risk-free rate (r_f) is computed using daily data from the 10-Year Treasury Yield. We convert annual values to daily values using the inverse compounding formula

$$\text{daily value} = \left(1 + \frac{\text{annual value}}{100}\right)^{\frac{1}{365}} - 1,$$

and then we average over the inputted time period for the final r_f value.

1.4 Minimum Accepted Return (MAR)

While we are not currently displaying the minimum accepted return (MAR), it is essential in calculating the Sortino ratio and downside deviation. The MAR and is computed differently on the asset-level and portfolio-level. On the asset level, and in particular on the bot-level, we use the minimum of the CAPM return of the coin traded by the bot and the risk-free rate.

1.4.1 Bot-level

The minimum accepted return for individual bots is calculated as the minimum of the risk-free rate and the CAPM of the coin traded by the bot:

$$\text{MAR}_{\text{bot}} = \min(r_f, \text{CAPM}(R_{\text{coin}})).$$

1.4.2 Portfolio level

The minimum accepted return for portfolios is the CAPM of the ASTRA100 index.

$$\text{MAR}_{\text{portfolio}} = \text{CAPM}_{\text{ASTRA}}$$

We also considered two other alternatives: the mean return of the S&P 500 or the mean return of the ASTRA100 index. However, since the CAPM of the ASTRA100 index combines the returns of the S&P 500 with the risk of both the ASTRA100 and the S&P 500, we decided to use this approach.

1.4.3 Coin level

The minimum accepted return for coins is the mean return of the ASTRA100 index:

$$\text{MAR}_{\text{coin}} = \overline{R_{\text{ASTRA}}}.$$

1.4.4 ASTRA100 level

The minimum accepted return for the ASTRA100 index is the mean return of the S&P 500:

$$\text{MAR}_{\text{ASTRA}} = \overline{R_{\text{S\&P}}}.$$

1.4.5 S&P 500 level

The minimum accepted return for the S&P 500 is the risk-free rate:

$$\text{MAR}_{\text{S\&P}} = r_f.$$

1.5 Monthly Return Table

The Monthly Returns (MR) for all relevant market indexes as well as coin trading pairs are calculated using a formula very similar to the Holding Period Return (HPR), and displayed in a table representing buy and hold return percentages based on historical data, with an additional column for the yearly buy and hold return.

To calculate the data for a specified year j, and month $1 \leq i \leq 12$ we use the formula:

$$MR_{i,j} = \frac{P_{1,i,j} - P_{0,i,j}}{P_{0,i,j}}$$

where $MR_{i,j}$ is the buy and hold return for month i of year j, where $P_{0,i,j}$ is the purchase price of the instrument at the beginning of month i, and $P_{1,i,j}$ is the price received for the instrument at the end of month i.
In addition, the yearly return (YR) is calculated as:

$$YR_j = \frac{P_{1,12,j} - P_{0,1,j}}{P_{0,1,j}}$$

These calculations allow for the matrix

$$[MR_{i,j}|YR_j]$$

to be displayed on the trade analysis page of relevant market indexes or coin pairs.

1.5.1 Coin Level

For coins, we use the close price of the coin at the end of month i as $P_{1,i,j}$, and the open price of the coin at the beginning of month i as $P_{0,i,j}$.

1.5.2 ASTRA100 level

For the ASTRA100 monthly return table, we use the price of the index at the end of month i as $P_{1,i,j}$, and the price of the index at the beginning of month i as $P_{0,i,j}$.

1.5.3 S&P 500 Level

For the S&P 500 monthly return table, we use the price of the index at the end of month i as $P_{1,i,j}$, and the price of the index at the beginning of month i as $P_{0,i,j}$.

2 Metrics: Measures of Risk

2.1 Beta

The general formula for beta is given by the covariance of asset returns and base market returns divided by the variance of market returns. Mathematically, this is given by

$$\beta_A = \frac{\text{Cov}(R_A, R_{BM})}{\text{Var}(R_{BM})},$$

where R_A is the list of asset returns and R_{BM} is the list of base market returns. Below, we specify which market returns are used for which calculation entities.

2.1.1 Bot level

On the bot-level, we use the ASTRA100 index as our benchmark:

$$\beta_{\text{bot}} = \frac{\text{Cov}(R_{\text{bot}}, R_{BM})}{\text{Var}(R_{BM})},$$

where R_{BM} can be switched between the returns of the ASTRA100 and the S&P 500.

2.1.2 Portfolio level

A user's portfolio beta is given by a weighted sum of individual bot beta's, using the portfolio weights defined in Section 1.2. Altogether, we have

$$\beta_{\text{portfolio}} = \vec{w} \cdot \vec{\beta} = \sum_{i=1}^{B} w_i \cdot \beta_i,$$

where β_i is the beta of the ith bot, computed as defined above.

2.1.3 Coin level

On the coin level, beta is similar to the bot beta, except that R_{bot} is replaced by R_{coin}:

$$\beta_{\text{coin}} = \frac{\text{Cov}(R_{\text{coin}}, R_{BM})}{\text{Var}(R_{BM})},$$

where R_{BM} can be switched between the returns of the ASTRA100 and the S&P 500.

2.1.4 ASTRA100 level

Beta for the ASTRA100 is what we call the "ASTRABeta", calculated as follows

$$\beta_{\text{ASTRA}} = \frac{\text{Cov}(R_{\text{ASTRA}}, R_{BM})}{\text{Var}(R_{BM})}.$$

2.1.5 S&P 500 level

Beta for the S&P 500 uses a very similar formula

$$\beta_{S\&P} = \frac{\text{Cov}(R_{\text{S\&P}}, R_{BM})}{\text{Var}(R_{BM})}$$

2.2 Covariance

Let X and Y be two jointly-distributed random variables. We define the covariance of X and Y as

$$\text{cov}(X, Y) = \text{E}[X - \text{E}[X]] \cdot \text{E}[Y - \text{E}[Y]].$$

2.2.1 Portfolio level

Let R_i represent the daily returns of the ith bot in a user's portfolio, and let there be B total bots. Then, we can compute the covariance between the ith and jth bots as

$$\text{cov}(R_i, R_j) = \text{E}[R_i - \text{E}[R_i]] \cdot \text{E}[R_j - \text{E}[R_j]].$$

Calculating the covariance across all bots, we get a $B \times B$ matrix Σ

$$\Sigma_{ij} = \text{cov}(R_i, R_j).$$

Note that Σ is a symmetric matrix ($\Sigma_{ij} = \Sigma_{ji}$):

$$\Sigma_{ij} = \text{cov}(R_i, R_j) = \text{E}[R_i - \text{E}[R_i]] \cdot \text{E}[R_j - \text{E}[R_j]] = \text{E}[R_j - \text{E}[R_j]] \cdot \text{E}[R_i - \text{E}[R_i]] = \text{cov}(R_j, R_i) = \Sigma_{ji}.$$

We return the matrix Σ.

2.3 Correlation

Let X and Y be two jointly-distributed random variables. We define the correlation of X and Y as

$$\rho_{X,Y} = \text{corr}(X, Y) = \frac{\text{cov}(X, Y)}{\sigma_X \sigma_Y} = \frac{\text{E}[X - \text{E}[X]] \cdot \text{E}[Y - \text{E}[Y]]}{\sigma_X \sigma_Y},$$

if $\sigma_X \sigma_Y > 0$. In words, the correlation between X and Y is the covariance between X and Y divided by the standard deviation of X (σ_X) and the standard deviation of Y (σ_Y). We immediately observe from the formula above that correlation is intimately related to covariance.

2.3.1 Portfolio level

Let R_i represent the daily returns of the ith bot in a user's portfolio, and let there be B total bots. Then, we can compute the correlation between the ith and jth bots similar to the covariance, as

$$\text{corr}(R_i, R_j) = \frac{\text{E}[R_i - \text{E}[R_i]] \cdot \text{E}[R_j - \text{E}[R_j]]}{\sigma(R_i)\sigma(R_j)}.$$

Calculating the correlation across all bots, we get a $B \times B$ matrix C

$$C_{ij} = \text{corr}(R_i, R_j).$$

Note that C is again a symmetric matrix ($C_{ij} = C_{ji}$):

$$C_{ij} = \text{corr}(R_i, R_j) = \frac{\text{E}[R_i - \text{E}[R_i]] \cdot \text{E}[R_j - \text{E}[R_j]]}{\sigma(R_i)\sigma(R_j)} = \frac{\text{E}[R_j - \text{E}[R_j]] \cdot \text{E}[R_i - \text{E}[R_i]]}{\sigma(R_j)\sigma(R_i)} = \text{corr}(R_j, R_i) = C_{ji}.$$

We return the matrix Σ.

2.4 Standard Deviation

Let X be a discrete random variable. The expected value is defined as the average

$$\mu = \mathrm{E}[X] = \frac{1}{N}\sum_{i=1}^{N} x_i,$$

The standard deviation of a random variable X is defined as

$$\sigma^2 = \mathrm{E}[(X - \mathrm{E}[X])]^2 = \mathrm{E}[(X - \mu)]^2.$$

For our returns R, we have

$$\sigma(R) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\Big(R_i - \mu\Big)^2}.$$

2.4.1 Bot level

For individual bots, we use daily bot returns as defined in previous sections, so we have

$$\sigma(R_{\mathrm{bot}}) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\Big((R_{\mathrm{bot}})_i - \mu\Big)^2},$$

where μ is the average value of the $(R_{\mathrm{bot}})_i$'s.

2.4.2 Portfolio level

Standard deviation on the portfolio level is calculated a bit differently. Continuing with the interpretation for standard deviation as portfolio volatility, we are no longer looking at the standard deviation of one set of returns but rather that of multiple sets of returns. Standard deviation on the portfolio level is calculated using the covariance matrix ($\Sigma(R_i)$) and the portfolio weights (as defined in the section 1.2), so we have

$$\sigma(R_{\mathrm{portfolio}}) = w^T \Sigma(R_i) w.$$

2.4.3 Coin level

For individual coins, we use daily coin returns as defined in previous sections, so we have

$$\sigma(R_{\mathrm{coin}}) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\Big((R_{\mathrm{coin}})_i - \mu\Big)^2},$$

where μ is the average value of the $(R_{\mathrm{coin}})_i$'s.

2.4.4 ASTRA100 level

For the ASTRA100, we use daily ASTRA100 index, so we have

$$\sigma(R_{\mathrm{ASTRA}}) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\Big((R_{\mathrm{ASTRA}})_i - \mu\Big)^2},$$

where μ is the average value of the $(R_{\mathrm{ASTRA}})_i$'s.

2.4.5 S&P 500 level

For the S&P 500, we use daily S&P 500 index values as defined in previous sections, so we have

$$\sigma(R_{\text{S\&P}}) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\Big((R_{\text{S\&P}})_i - \mu\Big)^2},$$

where μ is the average value of the $(R_{\text{S\&P}})_i$'s.

2.5 Downside Deviation

In most cases, downside deviation is computed similarly to standard deviation, except that only the distance between points below an established Minimum Accepted Return (MAR) are factored into the calculation. Thus, letting R denote a list of daily returns, we have

$$\sigma_{\text{downside}}(R) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\min\Big(R_i - \text{MAR}, 0\Big)^2}.$$

2.5.1 Bot level

On the bot level, we use the above formula and set $R_i = (R_{\text{bot}})_i$, the daily return on day i.

2.5.2 Portfolio level

Downside portfolio volatility is calculated differently from the other entities since we are no longer looking at the downside deviation of one set of returns but rather that of multiple sets of returns. Similar to how we expand upon standard deviation through the covariance matrix, we can expand upon downside deviation into a semicovariance matrix, defined as

$$(\Sigma_{\text{semi}})_{ij} = \sigma_{\text{downside}}(R_i)\cdot\sigma_{\text{downside}}(R_j)$$

Downside portfolio volatility (Σ_{semi}) is calculated using the semicovariance matrix and the portfolio weights (as defined in Section 1.2), so we have

$$\text{Downside Portfolio Volatility} = w^T\Sigma_{\text{semi}}(R_i)w.$$

2.5.3 Coin level

On the coin level, we use the formula and set $R_i = (R_{\text{coin}})_i$, the daily return on day i.

2.5.4 ASTRA100 level

For the ASTRA100, we use the formula and set $R_i = (R_{\text{ASTRA}})_i$, the daily return on day i.

2.5.5 S&P 500 level

On the S&P 500 level, we use the formula and set $R_i = (R_{\text{S\&P}})_i$, the daily return on day i.

3 Metrics: Measures of Risk and Return

3.1 Sharpe Ratio

The Sharpe ratio is a measure of return over risk. It is computed as the expected return minus the risk-free rate over volatility. The main difference in how the Sharpe ratio is calculated across entities lies in how expected return and volatility are calculated.

3.1.1 Bot level

The bot-level Sharpe ratio is calculated using the formula below, where the expected return is calculated as the mean daily return ($\mathrm{E}[R_{\text{bot}}] = \overline{R_{\text{bot}}}$), and the standard deviation ($\sigma(R_{\text{bot}})$) is calculated as specified in the Section 2.4.

$$\text{Sharpe Ratio} = \sqrt{N} \cdot \frac{\overline{R_{\text{bot}}} - r_f}{\sigma(R_{\text{bot}})}.$$

We multiply by $\sqrt{N}$, where N is the number of days, in order to annualize the ratio since we use daily returns.

3.1.2 Portfolio level

The portfolio level Sharpe ratio is calculated using the formula below, where the expected return is calculated as the mean daily return ($\mathrm{E}[R_{\text{portfolio}}] = \overline{R_{\text{portfolio}}}$), and the portfolio volatility ($\Sigma(R_i)$) is calculated using the portfolio weights and the covariance matrix (calculated in Section 2.2).

$$\text{Sharpe Ratio} = \sqrt{N} \cdot \frac{\overline{R_{\text{portfolio}}} - r_f}{w^T \Sigma w}.$$

We again multiply by $\sqrt{N}$ to annualize the ratio since we use daily returns.

3.1.3 Coin level

The coin level Sharpe ratio is calculated similar to the bot-level, with the expected return given by the mean daily return ($\mathrm{E}[R_{\text{coin}}] = \overline{R_{\text{coin}}}$):

$$\text{Sharpe Ratio} = \sqrt{N} \cdot \frac{\overline{R_{\text{coin}}} - r_f}{\sigma(R_{\text{coin}})}.$$

We again multiply by $\sqrt{N}$ to annualize the ratio since we use daily returns.

3.1.4 ASTRA100 level

The ASTRA100 level Sharpe ratio is calculated similar to the bot-level, with the expected return given by the mean daily index ($\mathrm{E}[R_{\text{ASTRA}}] = \overline{R_{\text{ASTRA}}}$):

$$\text{Sharpe Ratio} = \sqrt{N} \cdot \frac{\overline{R_{\text{ASTRA}}} - r_f}{\sigma(R_{\text{ASTRA}})}.$$

We again multiply by $\sqrt{N}$ to annualize the ratio since we use daily returns.

3.1.5 S&P 500 level

The S&P 500 level Sharpe ratio is calculated similar to the bot-level, with the expected return given by the mean daily index ($\mathrm{E}[R_{\mathrm{S\&P}}] = \overline{R_{\mathrm{S\&P}}}$):

$$\text{Sharpe Ratio} = \sqrt{N} \cdot \frac{\overline{R_{\mathrm{S\&P}}} - r_f}{\sigma(R_{\mathrm{S\&P}})}.$$

We again multiply by $\sqrt{N}$ to annualize the ratio since we use daily returns.

3.2 Sortino Ratio

The Sortino ratio is a measure of return over downside risk. It is computed as the expected return minus the risk-free rate over downside volatility. The Sortino ratio considers only the risk of the "downside", penalizing only risky bad returns. We measure how "bad" a return is by comparing it to a threshold called the Minimum Accepted Return, or the MAR, as specified in Section 1.4.

3.2.1 Bot level

The bot-level Sortino ratio is calculated using the formula below, where the expected return is calculated as the mean daily return ($\mathrm{E}[R_{\mathrm{bot}}] = \overline{R_{\mathrm{bot}}}$), and the downside deviation ($\sigma(R_{\mathrm{bot}})$) is calculated as specified in Section 2.5.

$$\text{Sortino Ratio} = \sqrt{N} \cdot \frac{\mathrm{E}[R_{\mathrm{bot}}] - r_f}{\sigma_{\mathrm{downside}}(R_{\mathrm{bot}})}$$

.

We multiply by $\sqrt{N}$, the number of days, in order to annualize the ratio since we use daily returns.

3.2.2 Portfolio level

The portfolio level Sortino ratio is calculated using the formula below, where the expected return is calculated as the mean daily return ($\mathrm{E}[R_{\mathrm{portfolio}}] = \overline{R_{\mathrm{portfolio}}}$), and the portfolio volatility ($\Sigma(R_i)$) is calculated as in Section 2.5, using the portfolio weights and the semicovariance matrix (Σ_{semi}):

$$(\Sigma_{\mathrm{semi}})_{ij} = \sigma_{\mathrm{downside}}(R_i) \cdot \sigma_{\mathrm{downside}}(R_j)$$

Thus, altogether, the Sortino ratio is given by

$$\text{Sortino Ratio} = \sqrt{N} \cdot \frac{\mathrm{E}[R_{\mathrm{portfolio}}] - r_f}{w^T \Sigma_{\mathrm{semi}} w}.$$

3.2.3 Coin level

The coin level Sortino ratio is calculated similar to the bot-level, with the expected return given by the mean daily return ($\mathrm{E}[R_{\mathrm{coin}}] = \overline{R_{\mathrm{coin}}}$), and the downside deviation ($\sigma(R_{\mathrm{coin}})$) is calculated as specified in Section 2.5.

$$\text{Sortino Ratio} = \sqrt{N} \cdot \frac{\mathrm{E}[R_{\mathrm{coin}}] - r_f}{\sigma_{\mathrm{downside}}(R_{\mathrm{coin}})}.$$

3.2.4 ASTRA100 level

The ASTRA100 level Sortino ratio is calculated similar to the bot-level, with the expected return given by the mean daily index ($\mathrm{E}[R_{\mathrm{ASTRA}}] = \overline{R_{\mathrm{ASTRA}}}$), and the downside deviation ($\sigma(R_{\mathrm{ASTRA}})$) is calculated as specified in Section 2.5:

$$\text{Sortino Ratio} = \sqrt{N} \cdot \frac{\mathrm{E}[R_{\mathrm{ASTRA}}] - r_f}{\sigma_{\text{downside}}(R_{\mathrm{ASTRA}})}.$$

3.2.5 S&P 500 level

The S&P 500 level Sortino ratio is calculated similar to the bot-level, with the expected return given by the mean daily index ($\mathrm{E}[R_{\mathrm{S\&P}}] = \overline{R_{\mathrm{S\&P}}}$), and the downside deviation ($\sigma(R_i)$) is calculated as specified in Section 2.5:

$$\text{Sortino Ratio} = \sqrt{N} \cdot \frac{\mathrm{E}[R_{\mathrm{S\&P}}] - r_f}{\sigma_{\text{downside}}(R_{\mathrm{S\&P}})}.$$

3.3 Capital Asset Pricing Model

The Capital Asset Pricing Model gives us the capital asset expected return for a given asset R_i. The expected return $\mathrm{CAPM}(R_i)$ is equal to the risk-free return plus the beta of the asset times the market risk premium, as given by the following formula:

$$\mathrm{CAPM}(R_i) = r_f + \beta \cdot (\mathrm{E}[R_M] - r_f).$$

We use the holding period return of the market index as the expected return of the market, so $\mathrm{E}[R_M] = \mathrm{HPR}_M$. On the bot, portfolio, and coin levels, we have implemented two different market comparisons: against the ASTRA100 ($\mathrm{E}[R_M] = \mathrm{HPR}_{\mathrm{ASTRA}}$) and the S&P 500 ($\mathrm{E}[R_M] = \mathrm{HPR}_{\mathrm{S\&P}}$).

3.3.1 Bot-level

On the bot level, we use bot beta (β_i) as specified in Section 2.1:

$$\mathrm{CAPM}(R_i) = r_f + \beta_i \cdot (\mathrm{E}[R_M] - r_f).$$

3.3.2 Portfolio level

On the portfolio level, we use portfolio beta (β_i) as specified in Section 2.1:

$$\mathrm{CAPM}(R_{\text{portfolio}}) = r_f + \beta_{\text{portfolio}} \cdot (\mathrm{E}[R_M] - r_f).$$

3.3.3 Coin level

For coins, we use the coin beta (β_i) as specified in Section 2.1:

$$\mathrm{CAPM}(R_{\text{coin}}) = r_f + \beta_{\text{coin}} \cdot (\mathrm{E}[R_M] - r_f).$$

3.3.4 ASTRA100 level

For the ASTRA100 we use the ASTRABeta (β_{ASTRA}) as specified in Section 2.1:

$$\mathrm{CAPM}(R_{\mathrm{ASTRA}}) = r_f + \beta_{\mathrm{ASTRA}} \cdot (\mathrm{E}[R_M] - r_f).$$

3.3.5 S&P 500 level

For the S&P 500, we use the ($\beta_{\text{S\&P}}$) as specified in Section 2.1:

$$\text{CAPM}(R_{\text{S\&P}}) = r_f + \beta_{\text{S\&P}} \cdot (\text{E}[R_M] - r_f).$$

3.4 Jensen's Alpha

Jensen's alpha measures the difference between an asset's returns and the CAPM expected return of the asset, as given by the formula below:

$$\alpha_J(R_i) = \text{E}[R_i] - \text{CAPM}(R_i).$$

As in the calculation of the CAPM, we use the holding period return of the market index as the expected return of the market. On the bot, portfolio, and coin levels, we have implemented two different market comparisons: against the ASTRA100 and the S&P 500.

3.4.1 Bot-level

On the bot level, we calculate pure alpha by:

$$\alpha_J(R_i) = \text{E}[R_i] - \text{CAPM}(R_i).$$

3.4.2 Portfolio level

On the portfolio level, we calculate pure alpha by:

$$\alpha_J(R_{\text{portfolio}}) = \text{E}[R_{\text{portfolio}}] - \text{CAPM}(R_{\text{Portfolio}}).$$

3.4.3 Coin level

For coins, we calculate pure alpha by:

$$\alpha_J(R_{\text{coin}}) = \text{E}[R_{\text{coin}}] - \text{CAPM}(R_{\text{coin}}).$$

3.4.4 ASTRA100 level

For the ASTRA100, we calculate pure alpha by:

$$\alpha_J(R_{\text{ASTRA}}) = \text{E}[R_{\text{ASTRA}}] - \text{CAPM}(R_{\text{ASTRA}}).$$

3.4.5 S&P 500 level

For the S&P 500, we calculate pure alpha by:

$$\alpha_J(R_{\text{S\&P}}) = \text{E}[R_{\text{S\&P}}] - \text{CAPM}(R_{\text{S\&P}}).$$

3.5 Pure Alpha

Pure alpha measures the difference between the holding period return of an asset and the holding period return of the chosen benchmark for that asset, as given by the formula below:

$$\alpha_P(R_i) = \text{HPR}[R_i] - \text{HPR}[R_B],$$

where R_i represents the returns of the asset and R_B the returns of the benchmark for that asset.

3.5.1 Bot-level

On the bot level, we use the returns of the coin traded by that bot as the benchmark (R_B):

$$\alpha_P(R_i) = \text{HPR}[R_i] - \text{HPR}[R_{\text{coin}}],$$

3.5.2 Portfolio level

On the portfolio level, we use a weighted coin holding period return as the benchmark. We get this by calculating the holding period return of each coin traded by a bot in the portfolio and then calculate the weighted average of these HPR's, weighing by initial allocation to the bot trading that coin over the total initial portfolio allocation. This gives us a sense of the holding period return if that same initial allocation had been invested in the respective coins instead of invested in bots trading that coin. The calculation of pure alpha is thus

$$\alpha_P(R_{\text{portfolio}}) = \text{HPR}[R_{\text{portfolio}}] - \Sigma_{\text{coin}}\ w_{\text{coin}} \cdot \text{HPR}[R_{\text{coin}}],$$

where w_{coin} is the initial allocation of the coin over the total initial allocation across all coins.

3.5.3 Coin level

For coins, we use the returns of the ASTRA100 index as the benchmark (R_B):

$$\alpha_P(R_{\text{coin}}) = \text{HPR}[R_{\text{coin}}] - \text{HPR}[R_{\text{ASTRA}}].$$

As such, we compare an individual coin's performance to the performance of the top 100 coins.

3.5.4 ASTRA100 level

For the ASTRA100, we use the chosen benchmark R_{BM} as the benchmark R_B:

$$\alpha_P(R_{\text{ASTRA}}) = \text{HPR}[R_{\text{ASTRA}}] - \text{HPR}[R_{BM}].$$

3.5.5 S&P 500 level

For the S&P 500, we use the chosen benchmark R_{BM} as the benchmark R_B

$$\alpha_P(R_{\text{S\&P}}) = HPR[R_{S\&P}] - HPR[R_{BM}].$$

4 Graphs: the Efficient Frontier

4.1 The Parabolic (or Hyperbolic) Efficient Frontier

Consider a portfolio of n assets. Let μ_i be the expected return on asset i, $i = 1, 2, ..., n$ and σ_{ij} be the covariance between the returns of assets i and j, $1 \leq i, j \leq n$. Let

$$\mu = (\mu_1, \mu_2, ..., \mu_n)' \quad \text{and} \quad \Sigma = [\sigma_{ij}].$$

Σ is called the covariance matrix for the assets and is symmetric ($\Sigma_{ij} = \Sigma_{ji}$) and positive semi-definite (all of its eigenvalues are non-negative). For the purpose of the theory laid out in this section, we make the stronger assumption that Σ is positive definite (all of its eigenvalues are positive). Let x_i denote the proportion of wealth to be invested in asset i and let $x = (x_1, x_2, ..., x_n)'$. In terms of x, the expected return of the portfolio μ_p and the variance of the portfolio σ_p^2 are given by

$$\mu_p = \mu' x \quad \text{and} \quad \sigma_p^2 = x' \Sigma x.$$

Let $l = (1, 1, ..., 1)'$; i.e., l is an $n-$vector of ones. Since the components of x are proportions, they must sum to one; i.e., $l'x = 1$. The constraint $l'x = 1$ is usually called the budget constraint [Bes10].

The goal is to choose a value for x which gives a large value for μ_p and a small value for σ_p^2. These two goals tend to be in conflict. Suppose we have two portfolios, both having the same expected return but the first having a small variance and the second having a large variance. The first portfolio is obviously more attractive because it bears less risk for the same expected return. This is the key idea behind H. Markowitz's definition of an efficient portfolio.

We can solve this optimization problem in three ways:

1. minimizing variance
2. maximizing expected return
3. combined optimization.

Minimizing variance. Consider the following definition.

Definition 4.1 *A portfolio is* **variance-efficient** *if for a fixed μ_p, there is no other portfolio which has a smaller variance σ_p^2.*

Definition 4.1 implies that a portfolio is efficient if for some fixed μ_p, σ_p^2 is minimized. Thus the efficient portfolios are solutions of the optimization problem

$$\min\{x'\Sigma x \mid \mu' x = \mu_p, l'x = 1\}. \tag{4.1}$$

Maximizing expected return. Consider the following definition.

Definition 4.2 *A portfolio is* **expected return-efficient** *if for a fixed σ_p^2, there is no other portfolio with a larger μ_p.*

Definition 4.2 implies that a portfolio is efficient if for some fixed σ_p^2, μ_p is maximized. Thus the efficient portfolios are solutions of the optimization problem

$$\min\{\mu' x \mid x'\Sigma x = \sigma_p^2, l'x = 1\}. \tag{4.2}$$

Combined optimization problem. There is a third optimization problem which also produces efficient portfolios. It is a more convenient formulation than (4.1) or (4.2), and we will use it to develop our portfolio optimization. Let t be a scalar parameter and consider the problem

$$\min\{-t\mu' x + \frac{1}{2}x'\Sigma x \mid l'x = 1\}. \tag{4.3}$$

For $t \geq 0$, the parameter t balances how much weight is placed on the maximization of $\mu' x$ (equivalently, the minimization of $-\mu' x$) and the minimization of $x'\Sigma x$. If $t = 0$, (4.3) will find the minimum variance portfolio. As t becomes very large, the linear term in (4.3) will dominate and portfolios will be found with higher expected returns at the expense of variance. Before solving (4.3), we define

$$h_0 = \frac{\Sigma^{-1} l}{l'\Sigma^{-1} l}, \quad h_1 = \Sigma^{-1}\mu - \frac{l'\Sigma^{-1}\mu}{l'\Sigma^{-1} l}\Sigma^{-1} l.$$

We also define the variables $\alpha_0, \alpha_1, \beta_0, \beta_1$, and β_2:

$$\alpha_0 = \mu' h_0, \quad \alpha_1 = \mu' h_1,$$

$$\beta_0 = h_0'\Sigma h_0, \quad \beta_1 = h_1'\Sigma h_0, \quad \text{and} \quad \beta_2 = h_1 \Sigma h_1.$$

We finally get the following solution for the optimization problem in (4.3)

$$\sigma_p^2 - \beta_0 = \frac{(\mu_p - \alpha_0)^2}{\alpha_1} \qquad (4.4).$$

The algebraic relationship between σ_p^2 and μ_p is a parabola, represented graphically below. The "nose" of the efficient frontier corresponds to the minimum variance portfolio ($t = 0$) where the investor wants the smallest risk and is not interested in expected return. Points on the efficient frontier below the minimum variance point correspond to portfolios which are not efficient and therefore only the top half of the efficient frontier is used. [Mer72]

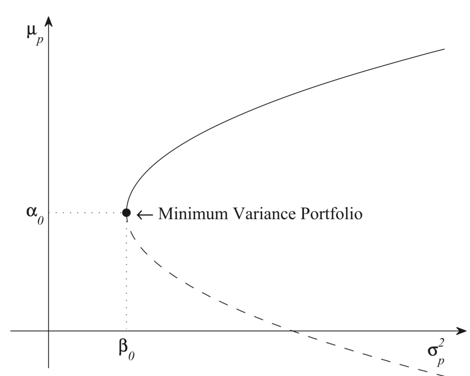

Figure 1: The above graph is from p. 26 of [Bes10].

So far, we have chosen to think of the efficient frontier in (σ_p^2, μ_p) space; i.e. mean-variance space. Sometimes it is helpful to think of it as a curve in (σ_p, μ_p) space; i.e., mean-standard deviation space:

$$\sigma_p^2 - \frac{(\mu_p - \alpha_0)^2}{\alpha_1} = \beta_0$$

show that the efficient frontier depends on the difference of the squares of the two variables μ_p and σ_p. Thus, in (σ_p, μ_p) space, the graph of the efficient frontier is a hyperbola.

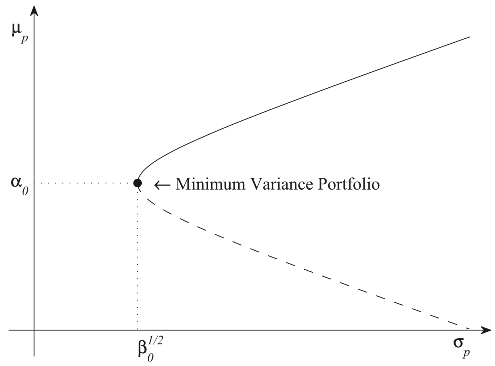

Figure 2: The above graph is from p. 28 of [Bes10].

4.2 The Piecewise Parabolic (or Hyperbolic) Efficient Frontier

Why use constraints? We have so far seen the model problem

$$\min\{-t\mu' x + \frac{1}{2}x'\Sigma x \mid l'x = 1\}, \tag{4.5}$$

The parameter $t \geq 0$ quantifies the risk aversion of the investor. Although (4.5) is very useful for developing the basic concepts for portfolio optimization, it is not particularly suitable in practice. There are two main reasons for this. First, the solution of (4.5) may result in excessive long and short selling. An example of this is an optimal solution of (4.5) with $x_1 = 1000$, $x_2 = -1000$, $x_3 = 1$, $x_4 = 0$, ..., $x_n = 0$. This means that the investor would sell 1000 times his wealth in asset 2 in order to purchase 1000 times his wealth in asset 1, which is completely unrealistic. Also, there are generally legal requirements restricting short sales. One way of precluding short sales is to impose non-negativity restrictions ($x \geq 0$) on the problem. [Bes10]

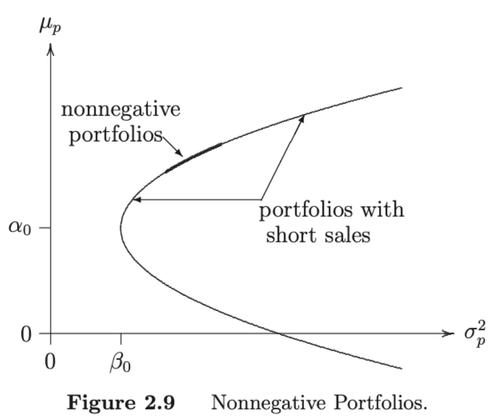

Figure 3: The above graph is taken from p. 39 of [Bes10]

Constrained optimization. The general constraint model problem is given by

$$\begin{aligned} \text{minimize} \quad &: \quad -t\mu'x + \frac{1}{2}x'\Sigma x \\ \text{subject to} \quad &: \quad a_i'x \leq b_i, i = 1, 2, ..., m, \\ & \quad\;\; a_i'x = b_i, i = m+1, m+2, ..., m+q \end{aligned}$$

4.3 AstraBit Implementation

No Short-Selling. If we want to only add constraints so that our weights are non-negative and add up to one, we get the following constrained optimization problem:

$$\begin{aligned} \text{minimize} \quad &: \quad -t\mu'x + \frac{1}{2}x'\Sigma x \\ \text{subject to} \quad &: \quad -x_i \leq 0, i = 1, 2, ..., m, \\ & \quad\;\; l'x = 1. \end{aligned}$$

Solving this optimization problem out leads to a piecewise efficient frontier, which is the result of multiple parabolic efficient frontiers defined on intervals, as shown below.

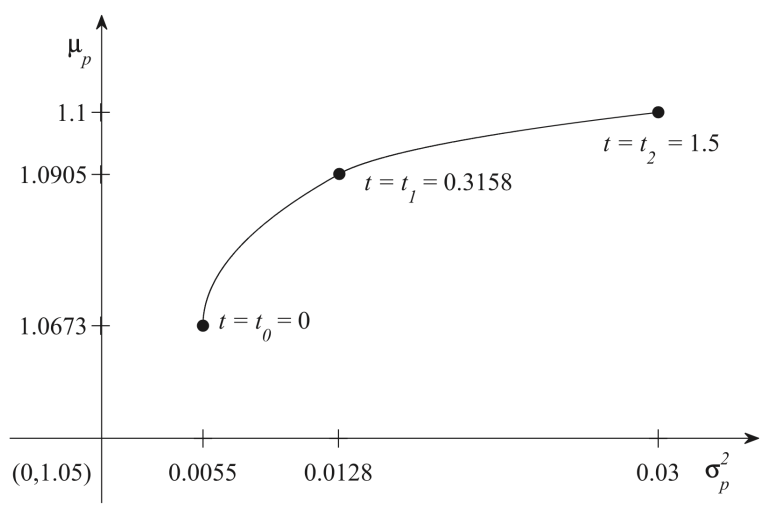

Figure 4: This figure is from p. 150 of [Bes10].

At AstraBit, we have decided to not allow for short-selling or inverse-selling in v1.0 of our Portfolio Analysis system, so our implementation of the efficient frontier also returns a piecewise parabolic function. We calculate the curve using the 'annualized daily means of the bots in the user's portfolio that have been trading for at least 30 days, and we display them against the corresponding annualized standard deviations. Both values are displayed in percentage format (%).

Furthermore, the user can hover over each point shown on the efficient frontier, and a box will pop up that displays the annualized daily mean return (%), annualized standard deviation (%), annualized Sharpe ratio of that point, and portfolio weighting at that point.

4.4 Capital Allocation Line

We now vary the previous model. In addition to the n risky assets, we will now suppose there is an additional asset with special properties. This asset will be risk free and as such will have a zero variance and a zero covariance with the remaining n risky assets. For the risk free asset, we use the 10 Year Treasury value. Let x_{n+1} denote the proportion of wealth invested in the risk free asset and let r_f denote its return. The expected return of this portfolio is

$$\mu_p = \mu_1 x_1 + \mu_2 x_x + \cdots + \mu_n x_n + r x_{n+1} = (\mu', r_f) \begin{bmatrix} x \\ x_{n+1}. \end{bmatrix}$$

Its variance is

$$\sigma_p^2 = x'\Sigma x = \begin{bmatrix} x \\ x_{n+1}. \end{bmatrix}' \begin{bmatrix} \Sigma & 0 \\ 0' & 0 \end{bmatrix} \begin{bmatrix} x \\ x_{n+1}. \end{bmatrix}.$$

The covariance matrix for this $n+1$ dimensional problem is

$$\begin{bmatrix} \Sigma & 0 \\ 0' & 0 \end{bmatrix},$$

for which the last row and column contain all zeros corresponding to the risk free asset. We thus have the following optimization problem

$$\begin{aligned} \text{minimize} \quad &: \quad -t(\mu', r_f) \begin{bmatrix} x \\ x_{n+1}. \end{bmatrix} + \frac{1}{2} \begin{bmatrix} x \\ x_{n+1} \end{bmatrix}' \begin{bmatrix} \Sigma & 0 \\ 0' & 0 \end{bmatrix} \begin{bmatrix} x \\ x_{n+1} \end{bmatrix} \\ \text{subject to} \quad &: \quad -l'x + x_{n+1} = 1. \end{aligned}$$

Solving the above problem, we get

$$\mu_p - r_f = \sigma_p [(\mu - rl)' \Sigma^{-1} (\mu - rl)]^{\frac{1}{2}}$$

which can be rewritten in the following form:

$$\mu_p = r_f + \left[\frac{(\mu - r_f)}{\sigma} \right] \sigma_p.$$

In mean-standard deviation space, the efficient frontier is a line. It is called the Capital Asset Line (CAL) and is illustrated in Figure 5 (below). Investors move up and down the Capital Asset Line according to their aversion to risk. For $t = 0$, all wealth is invested in the risk free asset and none is invested in the risky assets. As t increases from 0, the amount invested in the risk free asset is reduced whereas the holdings in the risky assets increase.

4.5 Optimal Portfolio Position

In order to get the optimal portfolio, we maximize for the Sharpe ratio across all possible portfolio positions. Since the efficient frontier already represents a set of the most "efficient", or optimal positions, we can actually maximize only over the efficient frontier itself: the optimal portfolio will always lie on the efficient frontier curve! Indeed, this maximization leads to a position that is called the *tangency portfolio*, given that the CAL is tangent to the hyperbolic efficient frontier exactly at this point. The optimal portfolio, as well as the Capital Allocation Line (CAL), are shown on the graph of the hyperbolic efficient frontier below.

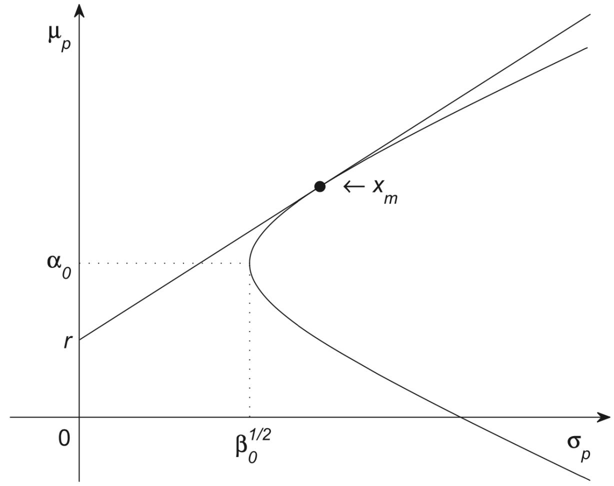

Figure 5: This figure is from p. of [Bes10]

Similar to the points composing the efficient frontier, the optimal portfolio also has a hover feature: users can hover over the point in order to see the breakdown of annualized daily mean return (%), annualized standard deviation (%), annualized Sharpe ratio of that point (which is also the maximal Sharpe ratio), and portfolio weighting at that point. The portfolio weighting is particularly important, as it will indicate to the user which bots contribute more heavily to an optimized portfolio.

4.6 Current Portfolio Position

Along with the optimized measures, such as the efficient frontier, Capital Allocation Line, and optimal portfolio position, we also plot the current position of the user's portfolio. It is important for us that the user have the ability to compare their current portfolio both algebraically and graphically to the optimal portfolio. The coordinates are simply

$$(\text{annualized daily mean return (\%)}, \text{annualized standard deviation (\%)}).$$

However, we also show the annualized Sharpe ratio of that point (which is the annualized version of the Sharpe ratio shown on the dashboard), and portfolio weighting at that point (these are the weights used throughout the calculation of metrics like portfolio standard deviation and beta).

4.7 Bot Positions

We also plot the position of individual bot positions. This is achieved by plotting the positions corresponding to 100% weighting one individual bot. The coordinates are similar to above, except that we plot the individual bot annualized daily mean return (%) and annualized standard deviation (%). We show the $100\% - 0\% - \cdots - 0\%$ weighting and the annualized Sharpe ratio of that point (which is the annualized version of the Sharpe ratio shown on the bot dashboard).

4.8 Risk-Free Rate

We lastly also plot the risk-free rate. This is computed using daily data from the 10-Year Treasury Yield, as explained in Section 1.3. The point $(0, r_f)$ is also graphically the y-intercept of the CAL.

5 Scoring System

5.1 Scorer Structure

While it is important to calculate each metric value, we also want to give our users context for what the metric value actually means. For example, we can tell the user that the downside deviation of their portfolio is equal to 0.43, or to 2.65, but what does that actually mean? In order to responsibly display these metrics, we also "score" all our metrics, giving each a score on a scale from 0 to 100, with 0 representing the worst score and 100 the best.

Each metric is scored via a `Scorer` structure, i.e. via a class that inherits the `Scorer` interface. For example, any request for the score of the Sharpe ratio of an entity is sent through `SharpeScorer`. We have a total of 11 scorers for this particular project, listed below:

- `BetaScorer`
- `CAPMScorer`
- `CompoundedReturnScorer`
- `DownsideDeviationScorer`
- `HoldingPeriodReturnScorer`
- `JensenAlphaScorer`
- `MeanReturnScorer`
- `SharpeScorer`
- `SortinoScorer`
- `StandardDeviationScorer`

Each `Scorer` inherits five methods:

- `calculate_score()`: calculates the total score
- `calculate_market_score()`: calculates the market score
- `calculate_fundamental_score()`: calculates the fundamental score, calls either `calculate_lower_score()` or `calculate_upper_score()` depending on the midpoint
- `calculate_lower_score()`: calculates the lower score
- `calculate_upper_score()`: calculates the upper score

The logic is the following: the only method that is directly called is `calculate_score()`. This method then calls on both `calculate_market_score()` and `calculate_fundamental_score()` to calculate the market score and fundamental scores for that metric, respectively. The market score gives the metric a score in context, considering the market values for that metric (both the S&P 500 value and the ASTRA100 value). The fundamental score, on the other hand, gives the metric a score solely rooted in certain fixed benchmarks that Astrabit has set.

Depending on the where the metric lies in the context of the benchmarks we set, we either call `calculate_lower_score()` or `calculate_upper_score()`. If the metric is greater or equal to the midpoint we have set, `calculate_upper_score()` is called. Otherwise, `calculate_lower_score()` is called. We then take a weighted average of the market score and fundamental score to get the overall score. The weighting used in this weighted average is determined by the metric. For metrics like the Sharpe ratio, we weight the fundamental score higher since there are generally accepted values for a "good" or "bad" Sharpe ratio. For other metrics, like standard deviation, we rely more heavily on the market score since the metric only makes sense in context.

5.2 Scoring Functions

We use a sigmoidal scoring function in order to bound the calculated metrics and provide a score between 0% and 100%. The graph below shows a few example sigmoid functions, all of which return a score between 0% and 100%. For the Sharpe ratio, we might use a more sensitive sigmoid, such as the purple curve below, to indicate less tolerance for a value outside of a certain range, while for return, we might use a curve like the green one below, that is slower to grow towards the asymptotes.

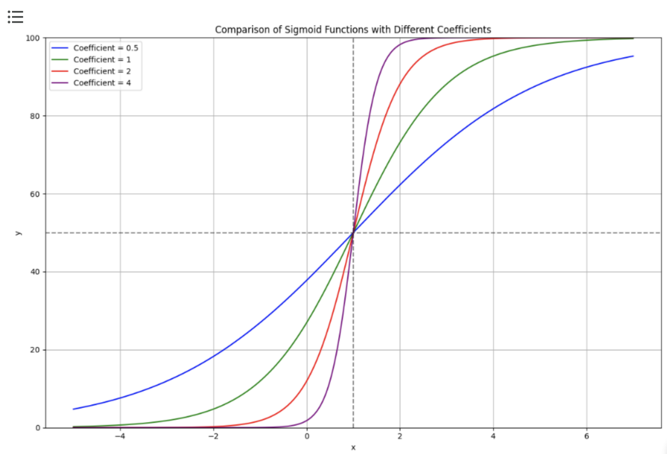

Figure 6: The graphs above were created in Python by Wes Rollings.

Since we use a midpoint-cutoff, we actually allow for two different sigmoidal functions per metric, one corresponding to the "upper score", and one corresponding to the "lower score". In order to well-define a sigmoidal function, you either need to define a coefficient, as shown in the graph above, or you can define a midpoint and another point through which the curve passes. This second point is essential in our process of defining benchmarks in the following two sections. Whether they are the 10%-50%/50%-90% benchmarks defined in the context of the fundamental score, or the 50%-75% benchmarks defined in the market score, the two points through which the sigmoid passes are fundamental to calculating the correct coefficient used in scoring.

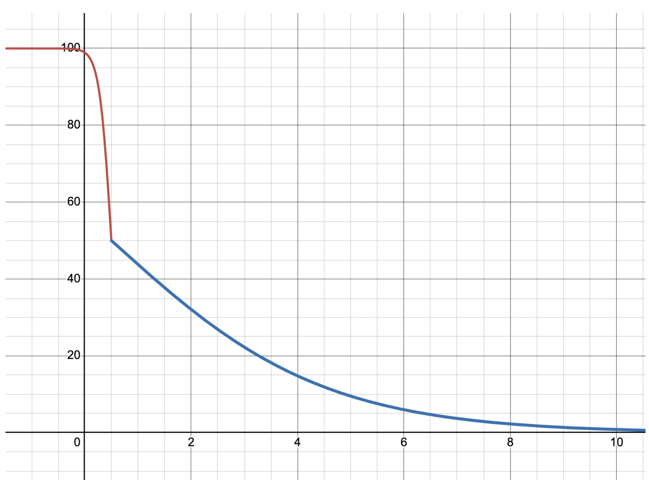

Figure 7: The graph above was created in Desmos by Maria Stuebner.

It is also important to make the distinction between increasing and decreasing sigmoid scoring functions. For a measure like standard deviation, which is a measure of volatility, we want to score low values higher and high values lower. Thus, in these cases, we use a piecewise sigmoid function similar to the one shown in Figure 7 (above). Note also the difference in coefficient between the upper score and lower score: a comparison against the midpoint indicates whether the value scored using the function to the left (red) or the function to the right of the midpoint (blue).

5.3 Market Score

We calculate the market score by first asking the following two questions: what is the market average (averaging the metric value of the S&P 500 and the ASTRA100)? and which of the two indices has the "worst" metric value? We call the solution to the first question the market average and the solution the second the worst value. We then set the worst value as the midpoint (i.e. we give it a score of 50%), and we give the market average a score of 75%.

Depending on the score, the "worst" value could be either the minimum or maximum of the two. For example, in the case of the Sharpe ratio, the "worst" value is minimum of the S&P 500 and ASTRA100 values, but in the case of a risk measure like standard deviation, the "worst" value is actually the maximum of the S&P 500 and ASTRA100 values (higher risk is bad).

5.4 Fundamental Score

The fundamental score is an evaluation based on AstraBit-set benchmarks that we consider independent of market performance. Our benchmarks are informed by general market consensus and our own evaluations. For example, independent of how the market is performing, the market generally considers a Sharpe ratio of 0.0 to be bad, 1.0 neutral, and 2.0 good. [Sch19] We have quantified the qualifiers "bad", "neutral", and "good" by assigning them to corresponding percentages: 10%, 50%, and 90%. Indeed, for each metric, we have defined three values, corresponding to

- a score of 10%
- a score of 50%
- a score of 90%.

We have set these benchmarks after our own evaluations and according to market research; they are periodically reviewed. Note that for metrics that are optimal when minimized, such as standard deviation, the "best" value is the minimal one of the three, while for other metrics that are optimal when maximized, such as the Sharpe ratio, the "best" value is the maximal of the three.

5.5 Weighted Average

As mentioned, once we have calculated a metric's market score and fundamental score, we take a weighted average to determine its overall score. The weighting is determined based on how market dependence of each metric score. For example, as noted above, since there is more general consensus around what a "Sharpe" ratio looks like, we weigh the fundamental score more highly (75%). On the other hand, since standard deviation and downside deviation only really make complete sense in context, we weight the market score more highly (80%).

6 ASTRA100: Digital Market (Crypto) Index

Indices are used as a consolidated, single source of information and measure of the performance of an entire security market. Early in the development of AstraBit's Portfolio Analysis system, we recognized the need for a reliable, broad, and representative sampling of crypto assets to enable AstraBit to more accurately:

1. gauge market sentiment;
2. measure and model return, systematic risk, and risk-adjusted performance;
3. develop a proxy for the overall digital asset class to serve as a foundation for a more reliable asset allocation model;
4. provide a reliable benchmark for AstraBit's individual users' actively managed crypto portfolios; and
5. create a broad portfolio model that will serve for such future investment products as index funds, ETFs, index options, etc.

6.1 Key Factors in Determining an Appropriate Sample Size.

As its name suggests, the ASTRA100 is a market capitalization-weighted index of the 100 most highly capitalized crypto assets (as determined by market float) available to trade. The weight of each constituent asset is determined by dividing its individual market capitalization by the aggregate total market capitalization of all 100 assets in the index.

In determining the most appropriate sample size from the thousands of crypto assets available in the market, we determined that a market capitalization approach would best enable AstraBit to make more accurate inferences about the population without having to study the entire population, which is decidedly impractical given the transient market nature of "meme coins." We aim to have the sample accurately represent the population so that conclusions drawn from the sample can be reasonably and accurately generalized to the population and, perhaps most importantly, the investing public, whose predominant focus is generally limited to Bitcoin and the major Altcoins.

Through research, we determined that the top 100 assets making up the ASTRA100 constitute a 99.5% share of total market capitalization. This sufficiently large sample enables us to estimate population parameters accurately. The excluded assets (coins) fall acceptably within the margin of error AstraBit is willing to tolerate in our results.

The primary advantage of using market capitalization weighting is that the constituent coins are held in proportion to their values in the target market. Generally, the primary disadvantage of market-cap-weighted indices is that constituent assets whose prices have disproportionately risen the most (or fallen the most) have a greater (or lower) weight. Thus, its weight increases as an asset's price increases relative to other assets in the index.

Whilst in standard equity and securities markets, the market-cap-weighted method can lead to over-weighting equities that may also be overvalued (in light of their fundamentals), we believe this problem is ameliorated when applied to crypto assets due to the speculative nature of crypto and dearth of fundamentals when analyzing individual coins. Thus, a market-cap-weighted index may better represent the highly volatile nature of the assets themselves and the speculative behavior of market participants, thereby providing a more accurate gauge of digital asset market sentiment.

Apropos of the above, using a market-cap-weighted method is akin to a momentum investment strategy, which better reflects most crypto market participants' predominant speculative investment paradigm.

6.2 Market Capitalization Weight

The weight of each constituent asset is determined by its market float; a function of both the price and the number of coins of the constituent asset available to the investing public.

$$w_i^M = \frac{Q_i P_i}{\sum_{j=1}^{N} Q_j P_j},$$

where

$$\begin{aligned} w_i &= \text{weight of the } i^{th} \text{ asset,} \\ Q_i &= \text{the number of coins outstanding of the } i^{th} \text{ asset,} \\ P_i &= \text{price of the } i^{th} \text{ asset, and} \\ N &= \text{number of securities in the index.} \end{aligned}$$

We now apply the formula for a market capitalization index to the ASTRA100:

$$\text{ASTRA100} = \frac{\sum_{i=1}^{N} Q_i P_i}{D}$$

where

$$\begin{aligned} Q_i &= \text{the number of coins outstanding of the } i^{th} \text{ asset,} \\ P_i &= \text{price of the } i^{th} \text{ asset,} \\ N &= \text{number of securities in the index, and} \\ D &= \text{the value of the divisor.} \end{aligned}$$

The divisor is chosen by setting the total index value on January 1, 2024 to be the value of the S&P 500 on January 1, 2024. An example of normalizing a market cap index is shown in the table below.

EXHIBIT 1 – ASTRA100X Index Value Computation example

ASSET	Coins Outs.	BOP Price	BOP Mkt Cap.	BOP weight %	EOP Price	EOP Mkt Cap.	EOP Weight %
A	21 mm	$50.00	$1,050 mm	35%	$60.00	$1,260 mm	38.12%
B	10 mm	$150.00	$1,500 mm	50%	$152.00	$1,520 mm	45.99%
C	500 mm	$0.50	$250 mm	8.33%	$0.75	$375 mm	11.35%
D	100 mm	$2.00	$200 mm	6.67%	$1.50	$150 mm	4.54%
TOTAL			**3,000 mm**	**100%**		**3,305 mm**	**100%**
INDEX Value			**1,000**			**1,101.67**	
Divisor			3 mm			3 mm	

Divisor remains constant = 3,000,000
BOP = Beginning of Period
EOP = End of Period

6.3 ASTRA100 Management: Rebalancing and Reconstitution

Two crucial questions pertaining to the maintenance of the ASTRA100 are:

- timing and frequency of rebalancing; and
- reexamination of constituent asset selection.

Rebalancing poses less concern for market capitalization weight indices as they largely rebalance themselves. Nevertheless, the weights of the constituent assets will be reviewed quarterly on the third business day of the first week of each new quarter.

Reconstitution (i.e., the process of changing the constituent assets in the ASTRA100) will be done bi-annually on the first business day of the second week of January and again on the first business day of the second week of July. At these times, the ASTRA100 committee will review the constituent assets, re-apply the initial criteria for inclusion in the index, and determine which assets to retain, remove, or add to the ASTRA100.

The main problem with reconstituting a market-cap-weighted index is ensuring that the weights of all the other constituent assets are appropriately adjusted to maintain the index's market capitalization. Reconstitution also poses corollary concerns with ensuring the accurate recalculation of important measures of risk and return (e.g., Beta, CAPM, Alpha, etc.).

6.4 Modeling Returns, Systematic Risk, Risk-Adjusted Performance

Beta represents the systematic risk of an asset with respect to the entire market and forms the most critical factor in the Capital Asset Pricing Model (CAPM). The ASTRA100 in the CAPM consists of individual crypto assets aggregating to a representative sample size of 99.5 percent of the market. Due to the adequacy of the sample size, we use the ASTRA100 to measure and model:

1. digital asset market systematic risk;
2. digital asset market returns;
3. digital asset risk premiums;
4. etc.

The ASTRA100 serves as a market proxy and is designed to allow investors to juxtapose the risk-adjusted performance of their actively managed portfolios against a passive alternative with the same level of systematic risk. Additionally, the ASTRA100 is designed to highlight Alpha at the asset and overall portfolio levels more accurately.

The ASTRA100 is a critical proxy for the overall digital asset class and portfolio allocation model as it provides historical data that assist in modeling the risks and returns of the component digital assets in each AstraBit user's portfolio. This will, in turn, enable AstraBit to create an ASTRA100 index fund that can be incorporated into an investor's portfolio to expose them to the overall market and diversify their targeted investment approach through the use of other AstraBit products, such as bots, third-party algorithms, etc. Perhaps most importantly, the ASTRA100 serves as a benchmark against which they can measure the performance of their own actively managed portfolio.

6.5 ASTRA100 Proprietary Intrinsic Value Filter

One of the main challenges of developing the market capitalization-weighted ASTRA100 was the creation and burning of tokens/coins that resulted from practices such as:

1. Follow-on offering of treasury coins (e.g., XRP) that can skew the market cap weighting due to an increase in coins outstanding;
2. Minting of new tokens by protocols as part of their staking/fixed income APYs;
3. Burning of existing tokens by protocols as part of a deflationary, reverse split that reduces the number of coins outstanding;
4. Etc.

Due to the highly speculative nature of the Digital Asset market space, our primary objective with the ASTRA100 is to incorporate a value filter that also factors in price changes to ensure that market capitalization cannot overly skew the value of the index while ignoring the importance of price.

As such, the ASTRA100 incorporates a proprietary Value Filter to smooth the effects of sudden increases/decreases in market capitalization due to increases/decreases in coins outstanding. This Value filter incorporates three (3) components designed to:

1. Ensure price is also fairly and reasonably weighted and factored into any shift of the Index Value;
2. Ensure sudden changes in outstanding coins are incorporated in such a way as to reflect market supply/demand factors better;
3. Ensure a more transparent approach to market participant behavior and the value placed by a larger population of market participants on any given asset in the index;

These three (3) key components help ensure an index value that better represents the overall market by limiting, to some degree, the impact of one large participant manipulating the value of any given asset in the index and skew the index unreasonably.

6.6 Excluded Assets

In constructing the ASTRA100 Index, our goal was to include a majority of the liquidity in the crypto space, representing broader market supply & demand factors for non-fiat-backed assets. Exclusions from this index include:

1. Stablecoins (e.g., USDT, USDC, etc.) that derive their value exclusively from a peg to the US Dollar and are primarily used as a base currency against which to trade, and limiting tax consequences (realized gains, etc.) triggered when moving from digital assets back into fiat. These assets do not qualify as holistic digital assets for the Index.
2. Wrapped tokens (e.g., WETH, WBTC, etc.) are derivative tokens with their price pegged to the original, underlying asset. These assets ultimately skew the index's value as they have a duplicative market effect. The underlying base coins are included (e.g., WBTC has a market cap of approx. 12.9B, which would put it in the top 20 tokens as of the index construction, while the BTC market cap is 1.75T), presenting a fairer index valuation.

Once these two categories are eliminated, the remainder of the index is only comprised of coins and tokens that are their own asset and not derivative/duplicative. This has the effect of shrinking the total applicable market cap, thereby allowing the index to capture a more accurate digital asset composition and representation of the market capitalization.

6.7 ASTRA100 Tracks $\sim 96\%$ of Crypto Market Capitalization

Based on the analysis of the top 100 cryptocurrencies, after removing stablecoins (USDT, USDC, DAI, FDUSD, PYUSD, TUSD) and wrapped tokens, the Total Adjusted Market Capitalization is approximately $2,579.0 billion ($2.579 trillion). The calculation of this value is given by

$$\mathrm{TAC}_m = \mathrm{GC}_m - (\mathrm{S}_t + \mathrm{W}_t)$$

where

$$\begin{aligned}
\mathrm{TAC}_m &= \text{Total Adjusted Capitalization (\$2,579.0 billion)} \\
\mathrm{GC}_m &= \text{Gross Market Cap of Top 100 (\$2,752.297 billion)} \\
\mathrm{S}_t &= \text{Stablecoins} \\
\mathrm{W}_t &= \text{Wrapped Tokens}
\end{aligned}$$

This represents the crypto market cap exclusive of stable assets backed by fiat currencies.

Bitcoin and Ethereum make up a significant portion of this total, with Bitcoin alone accounting for approximately $1.742 trillion of the total market cap (as of Nov. 14, 2024).

This means stablecoins and wrapped tokens account for about 6.29% of the total market cap in the top 100 cryptocurrencies. Most of this value comes from major stablecoins like USDT (Tether) with $126.6B and USDC with $36.8B in market cap. The percentage calculation is:

$$\frac{\$163.4\ \mathrm{B}}{\$2.858\ \mathrm{T}} \cdot 100 = 5.92\%$$

As of Nov. 14, 2024, the total market cap of the entire crypto market is $2.858T, with stablecoins and wrapped tokens accounting for approximately 5.18% of the total market. The Top 100 digital assets, excluding stablecoins/wrapped tokens is $2.752T. Therefore, the Astra100 tracks approximately 96.3% of the entire crypto market

$$\frac{\$2.752\ \mathrm{T}}{\$2.858\ \mathrm{T}} \cdot 100 = 96.3\%$$

The Astra100 shows the dominance of the top non-stablecoin cryptocurrencies in the overall market, with Bitcoin and Ethereum alone accounting for a significant portion of this percentage.

References

[Bes10] M.J. Best. *Portfolio Optimization*. Chapman & Hall/CRC finance series. Taylor & Francis, 2010.

[Har83] D.R. Harrington. *Modern Portfolio Theory and the Capital Asset Pricing Model: A User's Guide*. Prentice-Hall, 1983.

[Mer72] Robert C. Merton. An analytic derivation of the efficient portfolio frontier. *The Journal of Financial and Quantitative Analysis*, 7(4):1851–1872, 1972.

[RC88] A. Rudd and H.K. Clasing. *Modern Portfolio Theory: The Principles of Investment Management*. Andrew Rudd, 1988.

[Sch19] Charles Schwab. Calculate the Sharpe Ratio to Gauge Risk, 2019.