IMPERIAL

Imperial College London

Department of Mathematics

Alpha Generation in Commodity Markets: A Genetic Programming Approach with Maritime Trade Data

Author: Zhengyang Li (CID: 02054603)

A thesis submitted for the degree of

MSc in Mathematics and Finance, 2024-2025

Declaration

The work contained in this thesis is my own work unless otherwise stated.

Acknowledgements

I am very grateful for the support from Dr. Paul Bilokon for this thesis, who provides me with valuable guidance and inspirations for working on this topic. Also, I would like to express my gratitude to the IMF PortWatch project group. The dataset they build is of vital help to the project.

Abstract

The commodity market has garnered increasing interest from investors in recent years. Unlike equities, commodities exhibit relatively inefficient price discovery due to their fragmented market structures and unique supply-demand dynamics. These dynamics are influenced by a wide range of factors, including economic trends, geopolitical risks, and physical constraints such as storage and transportation. For quantitative investment in this area, alpha generation is defined as the development of certain quantitative signals to predict returns using data, or to form a trading strategy for obtaining excess returns beyond a benchmark. In this thesis, we generated alphas for commodities using Genetic Programming (GP), a machine learning algorithm which performs symbolic regression through evolutionary computation. A novel dataset of maritime trade data is utilised in a framework where potential alphas can be selected using GP and further finetuned, and experiments are conducted to apply new mechanisms of multi-objective fitness functions and penalty terms. Results show that Genetic Programming on maritime trade data could generate valid alphas for Brent oil, and the multi-objective fitness function facilitates GP by reducing drawdown and alleviating over-fitting. This method outperforms traditional formulaic alphas and Sharpe Ratio-based GP to a certain extent, with better performance metrics and good economic interpretations.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Informally traded since ancient times, commodity futures have been playing a fundamental role in the global economy [1]. From the 19th century, standardised modern commodity futures emerged in the US market [2], which gradually becomes a worldwide ecosystem of producers, consumers and traders we see today. Entering the 21st century, the commodity market has experienced a series of volatile shifts induced by demand-supply dynamics, economic trends and geopolitical events, with a recent "V-shape" movement from 2020 to mid-2022 driven by the COVID-19 pandemic and the Russia-Ukraine war. Close to the time when this thesis is written, the commodity future prices from 2025 tend to be more stable with a smaller fluctuating range, and the newly elected US administration aims to reduce the crude oil price to \$50/bbl or lower for managing inflation [3].

With all the circumstances stated above, the commodity market has seen rising interest in recent years. Although the inflation is declining from the high periods a few years ago, it is still persistent above expectation [4]. In this sense, the profits in the commodity market would stand, where historical returns of the Bloomberg Commodity Index have been 20% higher during periods when the year-over-year inflation is above 2% than below [4]. From another perspective, the uprise of algorithmic trading [5], big data and artificial intelligence [6] make more alternative data and methods manifest themselves, which creates new opportunities for manipulating the inefficiencies in the market. Finally, the unique and sophisticated dynamics of commodities render it a good asset class for diversification, which in some scenarios are more robust than equities. Hence, the development of commodity trading strategies possesses great potential for gaining profits and managing risks.

In quantitative finance, alpha generation is defined as the development of certain quantitative signals to predict returns using data, or to form a trading strategy for obtaining excess returns beyond a benchmark. Historically, alpha generation has been relying on traders' intuition and heuristic-based indicators from economic theories [7]. However, the advancement of machine learning and algorithmic trading has enabled the development of data-driven alpha discovery methods [8], leveraging statistical and machine learning techniques to identify exploitable market inefficiencies.

Among the machine learning methods, Genetic Programming (GP) has been a popular choice for formulating quantitative strategies based on interpretable mathematical structures. It is an automated and evolutionary algorithm which could perform symbolic regression on input features and maximise an objective function. A number of research has been conducted to discover alphas for the stock market strategies using the Information Coefficient (IC) [7] [9] or Sharpe Ratio [10] as the objective function, but the number of public research on the commodity market is relatively low.

Therefore, this thesis aims to generate alphas in the commodity market with maritime trade data using Genetic Programming, with two innovations being developed in this process. Firstly, a novel dataset, IMF PortWatch [11] is utilised for features, along with commodity future prices. This is a relatively new dataset which provides satellite-based, real-time updating data on maritime trade, which includes port activity, chokepoint trade estimate, disruption and climate scenarios. Attempts are made to discover the relationship between commodity future prices and these maritime trade data, due to the fact that such prices are directly or indirectly influenced by transportation and storage conditions.

Secondly, we introduce a multi-objective fitness function together with more sophisticated penalty mechanisms in the Genetic Programming approach of alpha discovery. Specifically, the proposed multi-objective fitness function balances the profitability (Sharpe Ratio, PnL) as well as risk factors (drawdown). In this way, the output of Genetic Programming and concomitant fining-tuning of parameters will orchestrate alphas which are more robust and have lower possibilities to be over-fitting. These innovations, together with feature engineering, automatic/manual alpha selection, alpha parameter fine-tuning, and efficacy testing, form a systematic framework of alpha generation.

The thesis is organised as follows: Chapter 2 provides background knowledge on commodity pricing, alpha generation in commodity markets, performance metrics used and Genetic Programming. Then, Chapter 3 elucidates the data and core mathematical and computational methods utilised in this thesis, from mathematical operations in feature engineering, the systematic framework of alpha generation with Genetic Programming, alpha selection and fine-tuning, and the proposed multi-objective fitness function. Chapter 4 presents the experiments conducted using the methods stated in previous sections, and the results of the developed strategies. Finally, the results are discussed and evaluated in Chapter 5, followed by Chapter 6 which concludes this thesis. The additional information, tables and figures are listed in the Appendix.

It is the ambition of this thesis to explore the novel IMF PortWatch dataset for alpha discovery in the commodity market and research on possible mathematical structures of alphas using this data. Also,it hopes to contribute to the existing research by experimenting Genetic Programming with modifications, developing a replicable and insightful framework of alpha generation for academic researchers and practitioners in the industry.

Chapter 2

Background

2.1 Theories in Commodity Pricing

First of all, some mathematical and economical theories of commodity pricing are introduced. These approaches aim to determine the fair price of the commodities by traditional factor models, combining mathematical structures with economic ideas. Although there is no such formula that could always predict the actual price of a commodity, these efforts are instrumental for understanding the characteristics of the commodity market and designing effective trading strategies.

2.1.1 Capital Asset Pricing Model (CAPM)

It is argued that models which explain commodity prices with risk factors and market returns mostly follow the paradigm of Capital Asset Pricing Model (CAPM) [1]. For instance, Grauer and Litzenberger [12] suggest that futures contracts can be priced by expected spot price with social risk and exposure to inflation. The CAPM used in this framework can be formulated as [12, Equation 17, page 81]

$$\tilde{R}_i = \alpha_i + \beta_i \tilde{R}_m + \epsilon_i, \tag{2.1.1}$$

where $\tilde{R}_i$ and $\tilde{R}_m$ are excess returns of the asset i and the market, and $\beta_i = \frac{\mathrm{Cov}[\tilde{R}_i, \tilde{R}_m]}{\mathrm{Var}[\tilde{R}_m]}$. The residuals satisfy $\mathrm{E}[\epsilon_i] = 0$ and $\mathrm{Cov}[\epsilon_i, \tilde{R}_m] = 0$.

However, there is also academic literature which argues that the beta coefficients in CAPM are not statistically significant [1], and that it is different from other capital assets, since commodities have consumable, perishable characteristics and are more susceptible to geopolitical environments.

2.1.2 The Theory of Storage

Continuing the arguments in the previous section, unlike equities, commodities are physical goods which need to be transported, stored and consumed. Therefore, inventory level also plays an essential role in the pricing of commodities.

When the inventory is scarce, holders of the inventory would receive implicit benefits called the convenience yield, which deteriorates when the inventory level increases [13]. In this sense, the nearby (spot) price of commodity will be pushed higher than the deferred commodity future price, which is referred as backwardation. The opposite scenario, where the spot price of commodity is lower than the future price, is called contangoment.

Storage costs also influence the level of inventories for commodities, thus influencing the prices. It is argued that high storage costs would keep the inventories levels relatively low and the convenience yield relatively high, while low storage costs would keep the inventories levels relatively high and the convenience yield relatively low. Research has been made to establish models which explain commodity prices with inventory levels, and verify the links between basis (difference between spot and future prices), spot volatility and inventory levels [13].

2.1.3 The Hypothesis of Hedging Pressure

Initially proposed by Cootner [14], the hypothesis of hedging pressure suggests that the commodity prices are influenced by hedging pressures from both commodity consumers/producers and speculators. These participants of the commodity market who either have a long (short) position in the underlying commodity would hedge their risks in the futures market (by taking the opposite directions). Therefore, the open interests for these hedgers, or the hedging pressure they cause, would affect the term structure as well as the futures price movements of the commodities.

2.1.4 A Factor Pricing Model

In this section, a factor model proposed by Zaremba *et al.* [15, Equation 1, page 2] will be demonstrated as an example of applying the pricing theories discussed above. The model is formulated as follows:

$$R_{i,t} = \alpha_i + \beta_{i,\mathrm{EW}}\, EW_t + \beta_{i,\mathrm{TS}}\, TS_t + \beta_{i,\mathrm{HP}}\, HP_t + \varepsilon_{i,t}, \tag{2.1.2}$$

where $\beta_{i,\mathrm{EW}}$, $\beta_{i,\mathrm{TS}}$ and $\beta_{i,\mathrm{HP}}$ are factor beta parameters, EW_t, TS_t, HP_t are factor returns, and $\varepsilon_{i,t}$ is model residuals.

EW is an equal-weighted long-only portfolio of all commodities, which indicates the overall price movements of the global commodity market.

TS is a term structure factor motivated by the Theory of Storage (section 2.1.2), which corresponds to how the most backwardated commodities would outperform the most contangoed commodities. The cross-sectional portfolio is selected by ranking the rolling yield (natural logarithm of the ratio between the nearby future price and the second nearest future price, Equation 2.2.5), where one longs the commodities with the highest past average rolling yields, and shorts the ones with the lowest past average rolling yields.

HP is the hedging pressure factor (from section 2.1.3), where one constructs such a portfolio by longing the commodities with the lowest average hedging pressure for hedgers ($HP_H = \frac{Long_H}{Long_H + Short_H}$ [15, page 3], referring to the long/short interest for hedgers) and the highest hedging pressure for speculators ($HP_S = \frac{Long_S}{Long_S + Short_S}$ [15, page 3], referring to the long/short interest for speculators). On the other hand, one shorts the the commodities with the highest average hedging pressure for hedgers and the lowest hedging pressure for speculators.

2.2 Alpha Generation in Commodity Markets

2.2.1 Formulation of the Problem

Given a commodity future with the price time series P_t over the discrete trading timestamps indexed by t, and a feature matrix $\mathbf{X}_t \in \mathbb{R}^{t \times N}$, which comprises trade or price data up to each trading timestamp (from row 0 to t) for time t and a total number of N features, the alpha (or trading signal) is defined as a functional mapping $f : \mathbb{R}^{t\times N} \to \mathbb{R}$, such that $\alpha_t = f(\mathbf{X}_t)$, where α_t refers to the trade position at time t.

In this way, the problem in alpha research is to discover such a mapping f which results in good performance of the trading signal. This alpha can be derived using a range of formulaic mathematical operations, including arithmetic and rolling computations, and in recent years, it has also been developed through machine-driven nonlinear mappings, such as deep learning and reinforcement learning.

2.2.2 Alphas Categories in Commodity Markets

In this section, some main categories of alphas in commodity markets are discussed, including the traditional formulaic alphas such as momentum, mean-reversion, and term structure, as well as the recent uprise of Machine Learning driven strategies.

Momentum Strategies

In a broad sense, a considerable proportion of common alphas in quantitative finance is in the form of momentum and mean-reversion [16]. The alpha strategies capture the phenomena that either the asset will continue its trend, or statistically revert to its average level. The momentum strategies usually examine how well the asset has performed in the past, and expect that it will continue to outperform/under-perform. For a price series P_t, examples could include [16, Equation 2, page 73]

$$\alpha_t^{(1)} = \log(P_{t-1}/P_{t-2}), \qquad \alpha_t^{(2)} = P_t - \frac{1}{T}\sum_{i=1}^{T} P_{t-i}. \tag{2.2.1}$$

The momentum could be based on price (Equation 2.2.1), or on other alpha factors (which is referred to as alpha momentum). There are already a wide range of research of proving the momentum property in the commodity market, which mainly use price (returns) [17] [18] [19], price-based alpha factors (from linear regression and asset pricing models) [16], performance of commodity funds [20], and macroeconomic indicators [19].

However, the scale and diversity of mathematical formulas of momentum strategies are not abundant in the open literature. While there is a large number of research in the stock market and commodity prices, no major research is established in momentum strategies particularly for maritime trade data. Therefore, it is a meaningful topic to explore whether such momentum effects are present using maritime trade data, and how more mathematically sophisticated structures could result in better performance of such strategies.

Mean Reversion Strategies

Apart from momentum strategies, mean-version strategies are more related to statistical concepts such as statistical arbitrage. It exploits price fluctuations or mispricings and expects the price level will revert to the statistical mean level. It usually has a negative sign to the returns on which it is based. For example, formulas could be constructed as [16, Equation 3, page 73]

$$\alpha_t^{(3)} = -\log(P_t/P_{t-1}), \qquad \alpha_t^{(4)} = \frac{1}{T}\sum_{i=1}^{T} P_{t-i} - P_t. \tag{2.2.2}$$

This phenomenon has also been observed and studied for a broad range of asset classes. For instance, it is argued that commodity futures (energy) has price-inefficiencies in the short term for mean-reversion strategies [21], and it has been extensively used in stock markets [22] [23] [24].

Mathematically, the concept of Z-score is closely related to applying the mean-reversion property in quantitative finance. It is defined as [25, Section 11.62.1, page 121]

$$\zeta(P_t) = \frac{P_t - \mu_P}{\sigma_P}, \tag{2.2.3}$$

where we set $\mu_P = \frac{1}{T}\sum_{i=t-T}^{t} P_i$, and $\sigma_P^2 = \frac{1}{T}\sum_{i=t-T}^{t}(P_i - \mu_P)^2$. If the price value deviates far from its historical mean level, adjusted by its standard deviation, it would be considered to be attractive to enter a certain position. A Z-score with absolute value above 2 is noteworthy, and above 3 is exceptional. Adjusting portfolio positions via Z-score is another way how one could perform mean-reversion strategies.

In mathematical finance, an important stochastic model for investigating mean-reversion (momentum) process is the Vasicek Model, which can be formulated as [25, Equation 99, page 123]:

$$dX(t) = \varkappa(\theta - X(t))\,dt + \sigma\,dW(t), \tag{2.2.4}$$

where $\varkappa > 0$ is the mean reversion speed, θ is the long-term mean level, and σ is the volatility and $W(t)$ is a standard Brownian motion. One could conduct statistical tests to examine whether a time series is mean-reverting by methods such as estimating the Hurst exponent.

Term Structure Strategies

The term structure strategies are driven by the theory of storage (section 2.1.2), where the scarcity in supply will lead to backwardation (futures lower than spots), while low demand will lead to contango (futures higher than spots). The typical strategy involves setting long-short portfolio where one longs the most backwardated commodities and shorts the most contangoed ones. Different term structures could be expressed mathematically by indicators such as the roll yield:

$$\text{Roll Yield} = \log\left(\frac{P_{front}}{P_{next}}\right), \tag{2.2.5}$$

where P_{front} and P_{next} represent front month and next month prices.

Machine Learning Strategies

In recent years, the advancement of Machine Learning (ML) and Artificial Intelligence (AI) inspires a range of new alpha strategies beyond the traditional formulaic alphas. They are usually based on ML algorithms which learn to develop alphas from labels or objective functions.

These approaches differ from formulaic or rule-based methods due to the fact that they are machine and data driven which do not totally rely on human brain-power, but make machines learn to mimic, understand or execute the trading logic. Also, they are harder to interpret than strategies based on economic theories because of the complexity of the algorithm, randomness in initialisation and optimisation, as well as the non-linear natures of the operators used. We will introduce some prominent research in this field as follows.

Tree Models:
One important class of models for alpha generation using ML is tree models with ensemble learning. For regression task, these models could learn nonlinear interactions and have better flexibility than neural networks for prediction since it does not make predictions based on model parameters, but from a partitioned input feature space.

There are two major categories of tree models, more traditional random forests and gradient boosting variants such as XGBoost and LightGBM. Random forest uses Bagging in decision trees to reduce variance and enhance model generalisation power [26], and gradient boosting models sequentially train trees for residuals from predictions to correct errors [27]. It has shown promising performance in predicting stock future returns, using engineered features from prices and other market data.

Deep Learning Models:
As a spotlight of the scientific community in recent years, Deep Learning has attracted much attention and shaped new dynamics in quantitative finance, with a number of research applying these methods in the financial market [6]. It constructs the Machine Learning model by layers of connected neurons with a set of parameters, where the input can be computed by linear operations and non-linear activation functions to predict the response [28]. The model parameters can be trained using gradient descent methods with respect to a loss function, and automatic differentiation could facilitate efficient execution.

In this manner, one could train models from rudimentary Multilayer Perceptrons (MLPs) for predicting price movements, order book flows and other labels of financial interests. Furthermore, more complicated structures can be utilised, such as Recurrent Neural Network (RNN) and Long Short-Term Memory network (LSTM), where the models capture the temporal patterns of the data by iterative structure and memory gate mechanism [29].

Reinforcement Learning Models:
Reinforcement Learning (RL) is a type of Machine Learning algorithm where an agent learns to make decisions by interacting with an environment. It offers a natural paradigm where trading actions can be conducted via sequential decision-making. A Reinforcement Learning agent has the goal to maximize the expected future rewards over time through the feedback it receives from the environment, and its observations of the state of the environment [30]. It has been applied in areas such as hedging risks [31], conducting portfolio management [32], optimising asset liquidation [33], as well as generating alphas by proposing formulas or generating stock investment decisions based on data and rewards [9].

2.2.3 Performance Metrics

In this section, a number of performance metrics for alpha signals are introduced. The developed trading strategies in the experiment section will be evaluated and discussed using these metrics as criteria.

Sharpe Ratio

Sharpe Ratio is one of the most important risk-award metrics in quantitative investment. It measures the expected level of (excess) returns against its stability (standard deviation). It is defined as [34, Section 2.4, page 9]:

$$\mathrm{SR}_{t-1} = \frac{\mathrm{E}_{t-1}[R_t - R_t^f]}{\mathrm{Std}_{t-1}[R_t - R_t^f]}. \tag{2.2.6}$$

In commodity research, it is of convention to use the absolute price changes as returns, i.e., $R_t = P_t - P_{t-1}$, compared with equities where one often uses relative changes or log returns. R_t^f denotes the risk free returns or the benchmark for evaluation. In this thesis, we alleviate this by assuming a zero risk free rate.

Regarding the time horizon of the Sharpe Ratio, if the returns series are measured at a benchmark frequency (such as hourly or daily), with Sharpe Ratio $\mathrm{SR}^{benchmark} = \frac{\mu}{\sigma}$, where μ and σ are expectation and standard deviation of the excess returns for the benchmark excess returns series. Considering a longer time horizon such as one year, if the longer time horizon has the relationship $\mathrm{E}[R^e_{year}] = \mathrm{E}[\sum_{t=1}^T R_t^e]$ and $\mathrm{Var}[R^e_{year}] = \mathrm{Var}[\sum_{t=1}^T R_t^e]$ with the benchmark, we have [34, Section 2.6, page 11]

$$\mathrm{SR}^{year} = \frac{\mathrm{E}[\sum_{t=1}^T R_t^e]}{\sqrt{\mathrm{Var}[\sum_{t=1}^T R_t^e]}} = \frac{T\mathrm{E}[R_t^e]}{\sqrt{T\mathrm{Var}[R_t^e]}} = \sqrt{T}\frac{\mu}{\sigma}, \tag{2.2.7}$$

assuming that the excess returns are independent and identically distributed.

Drawdown

Drawdown tracks the performance of a strategy with respect to its past maximum. The high water mark is defined as the so-far best value the portfolio has achieved [34, Section 2.7, page 12]: $\mathrm{HWM}_t = \max_{s \le t} V_s$, where V_s stands for the value of a portfolio. The absolute drawdown is subsequently defined as the distance between the current value and the high water mark:

$$\mathrm{DD}_t = \mathrm{HWM}_t - V_t. \tag{2.2.8}$$

Profits and Losses (PnL)

The profits and losses (PnL) are computed using the accounting formulation, that the change in value of the portfolio is [35, Section 3.2, page 12]

$$\Delta V_{t+1} = V_{t+1} - V_t = H_t(P_{t+1} - P_t) - \kappa |P_{t+1}(H_{t+1} - H_t)|, \tag{2.2.9}$$

where H_t is the position of the portfolio (a naive implementation would be $H_t = \alpha_t$), and κ is the coefficient for trading costs. Therefore, the cumulative PnL can be computed as

$$\mathrm{PnL}_t = \sum_{i=1}^t \Delta V_i = \sum_{i=1}^t H_{i-1}(P_i - P_{i-1}) - \kappa |P_i(H_i - H_{i-1})|, \tag{2.2.10}$$

assuming that $V_0 = 0$ and $H_0 = 0$.

Information Coefficient

Information Coefficient (IC) is a common performance metric used in the alpha research of the stock market. It is computed as the average level of Pearson's correlation between the alpha signal α_t and the returns series R_t [7, Section I, page 2]:

$$\mathrm{IC}(f, \mathbf{X}_T, R) = \mathrm{E}_t[\rho(\alpha_t, R_t)] = \frac{1}{T}\sum_{t=1}^T \rho(\alpha_t, R_t). \tag{2.2.11}$$

It often measures a pool of stocks, where $\alpha_t, R_t \in \mathbb{R}^n$, and reflects whether the alpha signal is a good indicator of the returns.

Profit per Trade

Profit per trade is defined as the ratio between the final value of the portfolio and the total number of trades conducted. Since in the real market transaction costs cannot be neglected, this metric provides insights in whether the developed strategy is monetisable and cost-efficient. The mathematical formulation is

$$\mathrm{PPT} = \frac{\mathrm{PnL}_T}{\sum_{i=1}^{T} |H_i - H_{i-1}|}. \tag{2.2.12}$$

Holding Period

Holding period measures the volume weighted average of the duration of holding the asset, i.e., it can be computed as

$$\mathrm{VWAHP} = \frac{\sum_i v_i d_i}{\sum_i v_i}, \tag{2.2.13}$$

where v_i is the volume of trades and d_i is the duration for which this volume was held.

2.3 Genetic Programming

In this section, Genetic Programming will be introduced in various aspects, from the regression task, objective (fitness) function, evolution procedure to termination criteria.

Genetic Programming (GP) is derived from Genetic Algorithms proposed by John Holland in the 1970s, which evolves solutions to complex problems by rules inspired from biological evolution [36]. This extension (GP), developed by Koza in the 1990s, focuses on evolving computer algorithms which are represented by trees and optimised through an objective (fitness) function [37]. By specifying an initial population of candidate programs, GP evolves the programs by a range of operations including crossover and mutation to generate offspring. This procedure is guided by the fitness function and eventually obtains the best program (solution) by stochastic optimisation.

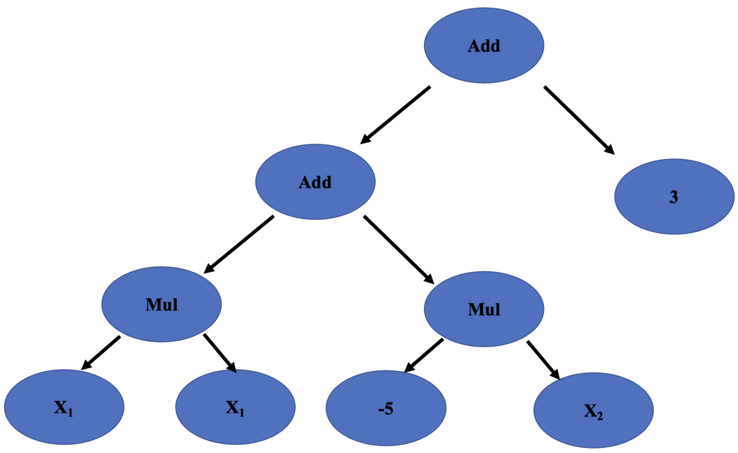

Figure 2.1: Tree representation of $X_1^2 - 5X_2 + 3$

2.3.1 Symbolic Regression

Symbolic regression is a machine learning task to ascertain the underlying mathematical relationship between an input (feature) and a response (target) [38]. As shown in Figure 2.1, a mathematical expression can be represented as a tree structure where the leaf nodes are input features, and the root nodes are operators. The operators include basic arithmetic operators (such as 'add' with an arity of 2), common mathematical functions (such as 'absolute value' with an arity of 1, 'sin' with an arity of 1), and rolling functions (such as rolling mean and variance with an arity of 1).

In this way, formulaic alpha expressions can be regressed from the input features by the evolution of these tree programs in a symbolic manner. Starting from a random population of expressions on features, the successive generations are selected based on their fitness function scores, evolving to the true relationship from the parent generations [37]. When the best individual in the last generation is attained, it can be used to predict the target based on new input features (which is referred as the execution of the program).

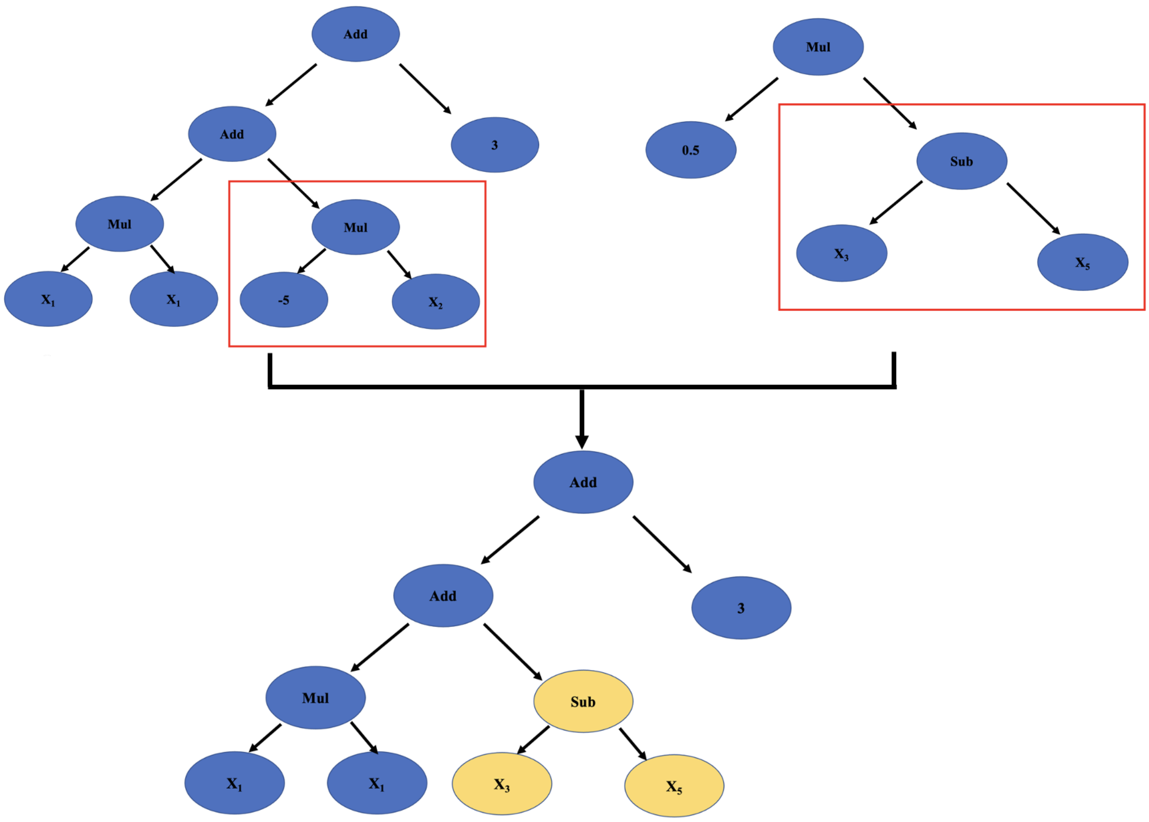

Figure 2.2: Crossover of tree programs

2.3.2 Operator Pool

In this section, the pool of operators used in GP for this thesis is listed. It contains three main categories: arithmetic operators, common mathematical operators and rolling/lag operators. Different pools are formed from these catering different feature organisations.

- **Arithmetic operators**
  Addition: $x + y$; Subtraction: $x - y$; Multiplication: $x \cdot y$;
  Division: x/y (with safe-guarding in code for $y = 0$);

- **Common mathematical operators**
  Sign: $\mathrm{sign}(x)$, Exponential: $\exp(x)$; Logarithm: $\log(x)$;
  Maximum: $\max(x, y)$; Minimum: $\min(x, y)$; Absolute value: $|x|$;

- **Rolling / lag operators**

  – Rolling mean:
  $$\mathbf{E}_{\tau,j}[\mathbf{X}_t] \;=\; \mathbf{E}\Big[\tilde{\mathbf{x}}^j_{t,\tau}\Big] \;=\; \mathbf{E}\Big[(x^j_t, x^j_{t-1}, x^j_{t-2}, \ldots, x^j_{t-\tau+1})\Big]\;;$$

  – Rolling standard deviation:
  $$\mathbf{Std}_{\tau,j}[\mathbf{X}_t] \;=\; \mathbf{Std}\Big[\tilde{\mathbf{x}}^j_{t,\tau}\Big] \;=\; \mathbf{Std}\Big[(x^j_t, x^j_{t-1}, x^j_{t-2}, \ldots, x^j_{t-\tau+1})\Big]\;;$$

  – Rolling normalization:
  $$\mathbf{Norm}_{\tau,j}[\mathbf{X}_t] \;=\; \mathbf{Norm}\Big[\tilde{\mathbf{x}}^j_{t,\tau}\Big] \;=\; \frac{\tilde{\mathbf{x}}^j_{t,\tau} - \mathbf{E}\Big[\tilde{\mathbf{x}}^j_{t,\tau}\Big]}{\mathbf{Std}\Big[\tilde{\mathbf{x}}^j_{t,\tau}\Big]}.$$

  – Rolling maximum:
  $$\mathbf{Max}_{\tau,j}[\mathbf{X}_t] \;=\; \mathbf{Max}\Big[\tilde{\mathbf{x}}^j_{t,\tau}\Big] \;=\; \mathbf{Max}\Big[(x^j_t, x^j_{t-1}, x^j_{t-2}, \ldots, x^j_{t-\tau+1})\Big]\,.$$

  – Rolling minimum:
  $$\mathbf{Min}_{\tau,j}[\mathbf{X}_t] \;=\; \mathbf{Min}\Big[\tilde{\mathbf{x}}^j_{t,\tau}\Big] \;=\; \mathbf{Min}\Big[(x^j_t, x^j_{t-1}, x^j_{t-2}, \ldots, x^j_{t-\tau+1})\Big]\,.$$

  – Rolling correlation (with feature/index k):
  $$\mathbf{Cor}_{\tau,j,k}[\mathbf{X}_t] \;=\; \mathbf{Cor}\Big[\tilde{\mathbf{x}}^j_{t,\tau},\, \tilde{\mathbf{x}}^k_{t,\tau}\Big] \;=\; \mathbf{Cor}\Big[(x^j_t, x^j_{t-1}, \ldots, x^j_{t-\tau+1}),\, (x^k_t, x^k_{t-1}, \ldots, x^k_{t-\tau+1})\Big]\,.$$

  – Lag operator:
  $$\mathbf{Lag}_{\tau,j}[\mathbf{X}_t] \;=\; x^j_{t-\tau}.$$

2.3.3 Fitness Function

Fitness function is the objective function which serves as the baseline of the optimisation in Genetic Programming. It evaluates how well the proposed expression fits the regression task, and outstanding programs with better fitness function scores will be selected for the evolution. In a statistical sense, Mean Squared Error (MSE) and R^2 are common fitness functions for symbolic regressions, which are defined as follows:

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{2.3.1}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{2.3.2}$$

where y_i are true values of the response, and $\hat{y}_i$ are the predictions from the proposed expression.

Also, in the context of alpha research, specific functions can also be used as fitness functions. For example, IC and Sharpe Ratio from section 2.2.3 are also used in the literature [7] [10], for it either captures the correlation of the alpha with the asset returns, or an award-risk ratio for the returns of the strategy. In the following Chapter 3, we will further discuss the choice of these fitness functions, along with developing a new multi-objective fitness function for GP..

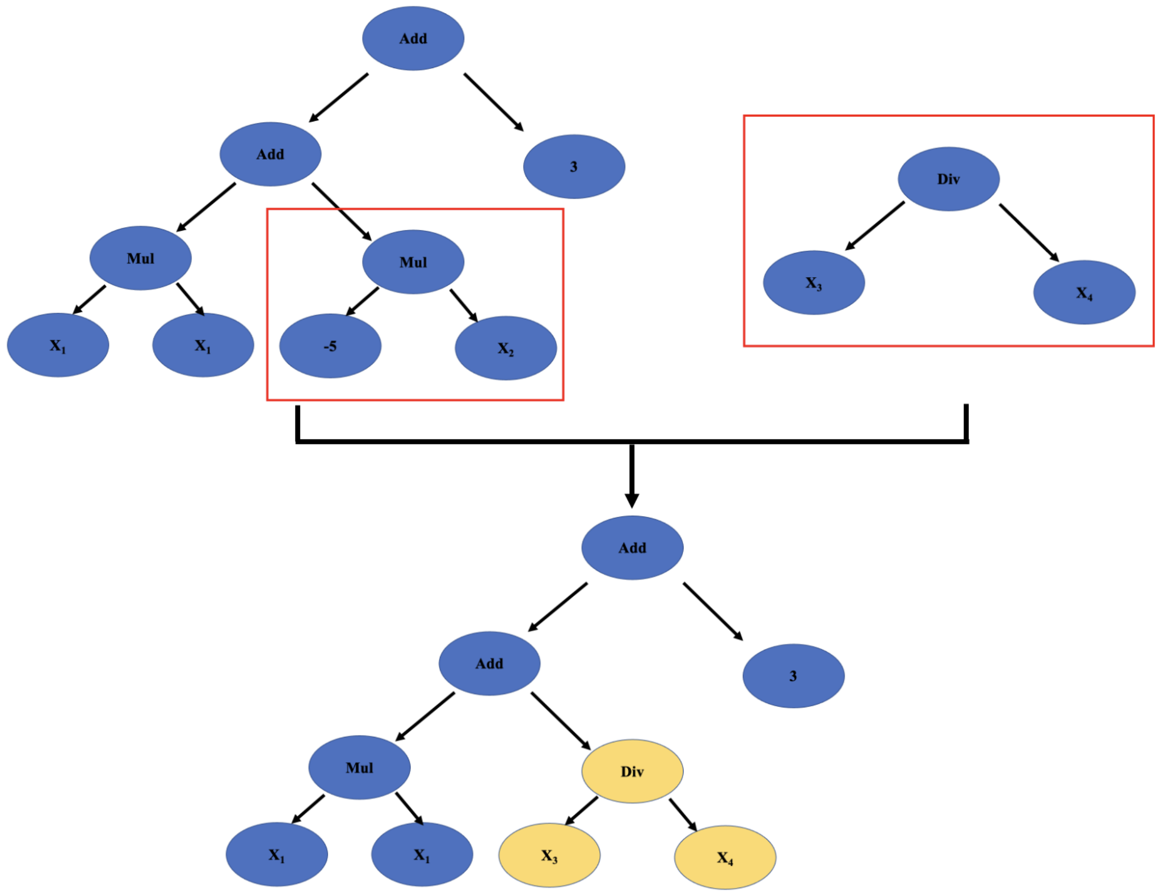

Figure 2.3: Subtree mutation of tree programs

2.3.4 Evolution

Initialisation

The evolution of the GP starts by initialisation. A pool of formulas is initiated following a certain initialisation method. These formulas in the first generation are unaware of the fitness function, and then will serve as the starting point for the evolution [38].

Selection

The selection of each generation is conducted by tournaments. A fixed number of random programs (formulas) in that generation is selected, and compared based on the fitness function, the fittest individual will be selected to move to the next generation [38]. The size of the tournament would affect the convergence speed and computation time, since a larger tournament size will select the fittest program more efficiently, while a smaller size keeps more diversity of programs and takes longer time for the computation.

Crossover

As for the evolution after the fittest individuals are selected in the tournament, a range of operations could be made to generate the actual next generation programs. The most common method is crossover, where a random subtree of the survivor of a tournament (parent) will be replaced [39]. The donor of that new sub-tree is also randomly selected from the survivor of another subsequent tournament, as shown in Figure 2.2.

Mutation

Apart from crossover, there is also a range of mutation methods in the evolution [40]. Subtree mutation refers to randomly choosing a subtree of the survivor of a tournament (parent), and replacing it by another randomly generated subtree (displayed in Figure 2.3). Hoist mutation takes such a subtree, further selects a subtree of that subtree, and merges that secondary level subtree into the primary subtree level, as shown in Figure 2.4. This method is used to reduce genetic components in the evolution. Point mutation, which is the most common mutation in GP [38], is defined as replacing a random node by another randomly selected node to keep the diversity of the programs (Figure 2.5).

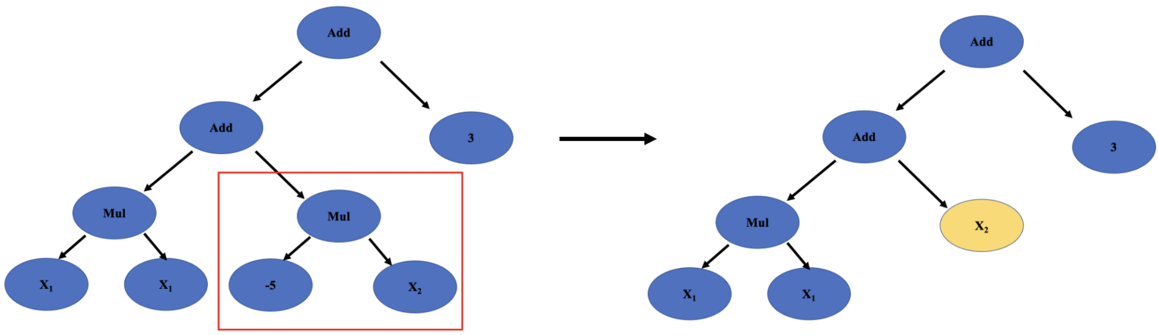

Figure 2.4: Hoist mutation of tree programs

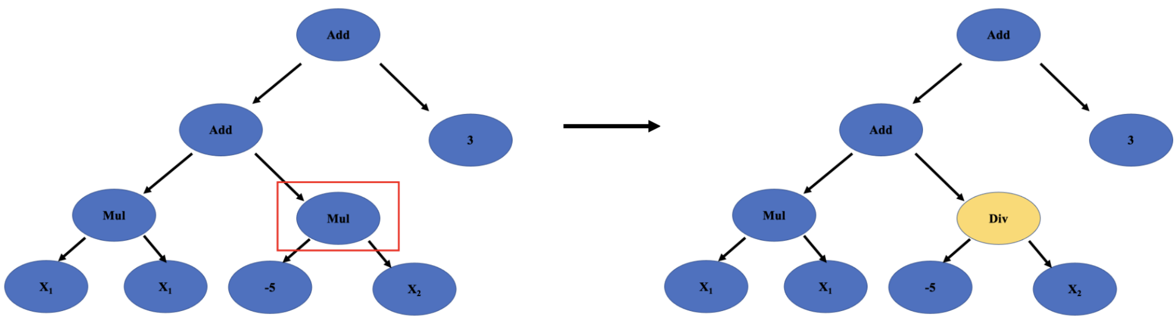

Figure 2.5: Point mutation of tree programs

2.3.5 Termination

Finally, the GP yields the final output, i.e., the best individual by two cases [37]. Firstly, the maximum number of generation is reached, and the best performing individual from the last generation is returned. Otherwise, there is certain program(s) whose fitness score exceeds a preset hyperparameter stopping_criteria, and the GP is early-stopped and returns the best individual.

Chapter 3

Data and Methodology

3.1 Maritime Trade Data

In this thesis, we use the IMF PortWatch [11] as the source for maritime trade data. It is a relatively new dataset of maritime trade flows (officially launched on November 15, 2023) managed by a joint initiative between IMF and University of Oxford, with a particular focus on trade disruptions and shocks.

It utilises satellite-based data to obtain information on port activities, trade flows, cargo quantities and capacities of chokepoints, disruptions as well as climate scenarios [41]. It aims to provide insights in global supply chains, impact of port/chokepoint disruptions on trades and real-time shipping dynamics for policy-makers, business stakeholders and general public. The data of main focus investigated in this thesis is listed as follows.

3.1.1 Daily Port Activity Data and Trade Estimates

This data contains the daily count of port call, the approximations of import and export volumes (metric tons) of more than 1700 ports in the world [42]. The data is obtained using Automatic Identification System (AIS) of ship movements on a real-time basis [42]. The key contents in this dataset are:

- Date information: date, year, month, day (time span: 2019/02/20 to 2025/02/21);
- Port information: port id, port name;
- Port call: number of different types of ships entering the port (container ships, dry bulk carriers, general cargo ships, ro-ro ships, tankers);
- Import volume: The total import volume (metric tons) of different types of ships;
- Export volume: The total export volume (metric tons) of different types of ships.

A detailed description of this data is presented in Appendix A.1, and a visualisation of the ports in the dataset is displayed in Figure 3.1.

3.1.2 Daily Chokepoint Quantity and Volume Estimates

This data contains the daily transit calls and approximations of trade volumes of 24 chokepoints globally [43]. The data is obtained using the same source as above, and instead of focusing on specific ports, it gets information from chokepoints such as Suez Canal, Cape of Good Hope, Dover Strait and Malacca Strait (displayed in Figure 3.1). The key contents in this dataset are:

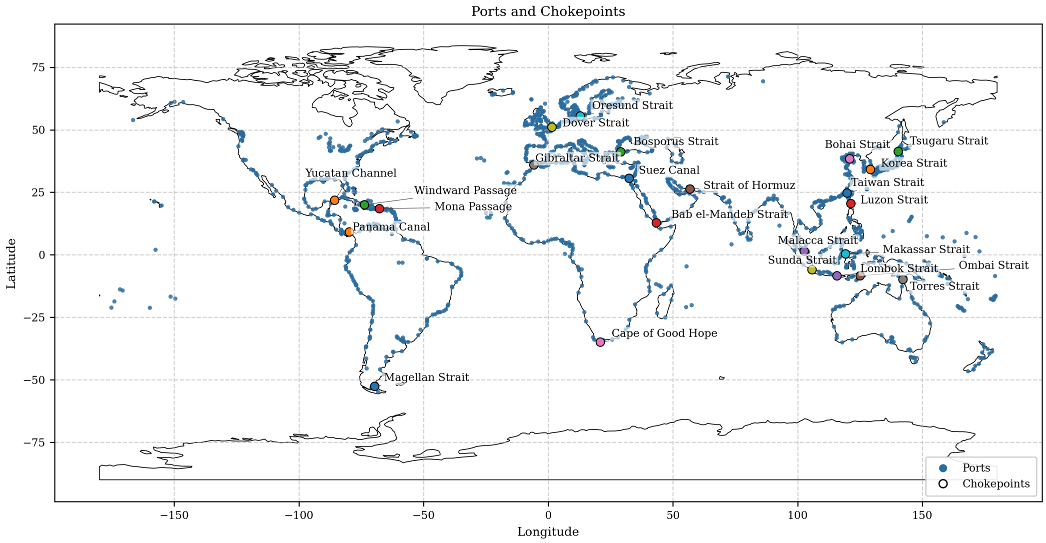

Figure 3.1: Ports and chokepoints from IMF PortWatch data

- Date information: date, year, month, day (time span: 2019/02/20 to 2025/02/21);
- Chokepoint information: chokepoint id, chokepoint name;
- Ship quantity: number of different types of ships transiting at the chokepoint (container ships, dry bulk carriers, general cargo ships, ro-ro ships, tankers);
- Trade volume: The total trade volumes (metric tons) of different types of ships.

In particular, the chokepoint trade volumes are referred as chokepoint capacities, since this is the column name in the actual dataset. A detailed description of this data and the chokepoints is presented in Appendix A.2 to Appendix A.4, and some chokepoint capacities (trade volume estimates) in the dataset is displayed in Figure 3.2.

3.1.3 Disruptions

The disruptions to maritime trade caused by natural disasters are obtained by the Global Disaster Alert and Coordination System (GDACS) service, including earthquakes, tsunamis, tropical cyclones and floods [44]. Other types of disruptions are also present such as geopolitical tensions and infrastructure failures. These events could potentially affect the global supply-demand dynamics of the commodities, and eventually drive the change in commodity future prices. The key contents in this dataset are:

- Date information: date, year (time span: 2019/02/20 to 2025/02/21);
- Event information: event name, event type, event description;
- Location information: latitude, longitude;
- Port information: Affected ports.

A detailed description of this data is presented in Appendix A.5.

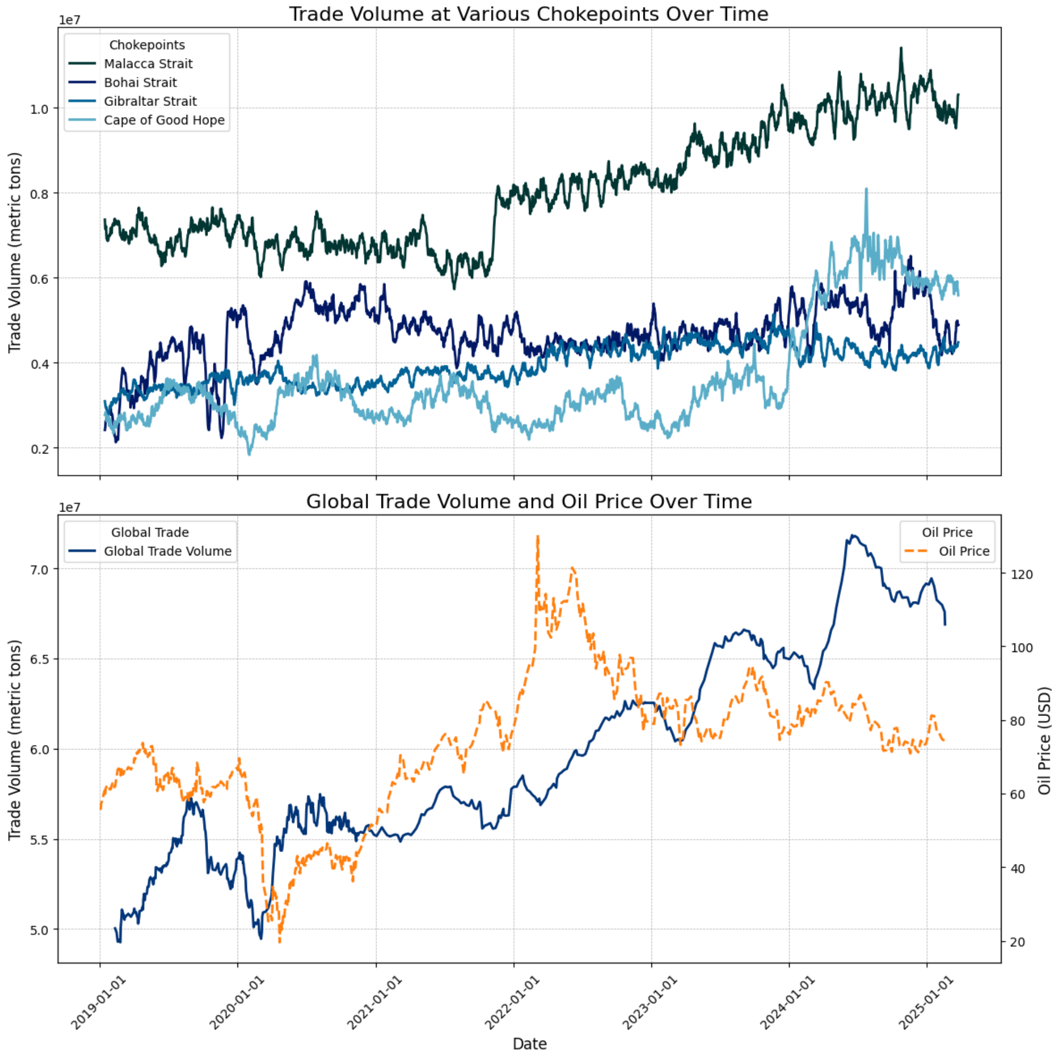

Figure 3.2: Chokepoint capacities over time of global maritime trade

3.2 Commodity Futures Data

A range of commodity futures data are investigated in this thesis. The data is provided by Dr Bilokon [45], and two most mainstream energies, oil and natural gas are selected for alpha discovery. The data units are tick sizes, and the tick size and contract specifications are listed in Table 3.1 below. For alpha generation based on the IMF PortWatch data, only the sections where the IMF data is available are used, and for attempts which are only based on commodity future prices, all available data is used.

3.3 Feature Engineering

In this section, the methods used to engineer features from the maritime trade data are discussed. These engineered features will be used as the input for the alpha discovery framework stated in this chapter.

Port Activity Data and Trade Estimates:
For port activity data, since the dimension of the port is very high (above 1700 for one time stamp), one needs to perform some mathematical operations to reduce the dimension and obtain a summary from the raw data. Three methods are applied accordingly: weighted

Table 3.1: Tick sizes and contract specifications for various oil and natural gas futures

Contract	Code	Contract Size	Tick Size	Time Frame
Dutch TTF Natural Gas	TTF	672-745 MWh	€0.005/MWh	2013-2025
UK NBP Natural Gas	GWM	1000 therms	£0.001/therm	2007-2025
NYME Natural Gas	NG	10,000 MMBtu	\$0.001/MMBtu	1996-2025
RBOB Gasoline	RB	42,000 gal	\$0.0001/gal	2006-2025
NY Harbor ULSD	HO	42 000 gal	\$0.0001/gal	1996-2025
Low Sulphur Gasoline	G	100 mt	\$0.25/mt	2005-2025
Brent Crude Oil	B	1 000 bbl	\$0.01/bbl	2005-2025
WTI Crude Oil Futures	CL	1 000 bbl	\$0.01/bbl	1996-2025

average, Iterated Principal Component Analysis (IPCA) and clustering.

3.3.1 Weighted Average

Let the import and export vectors at time t be $\mathbf{I}_t = (I_{1,t}, I_{2,t}, ..., I_{p,t}) \in \mathbb{R}^p$ and $\mathbf{E}_t = (E_{1,t}, E_{2,t}, ..., E_{p,t}) \in \mathbb{R}^p$ respectively, where the components denote the import or export volumes at different ports. A summary statistic of the import and export dynamics would be a weighted average:

$$\mu_t^I = \sum_i w_i^I I_{i,t}, \qquad \mu_t^E = \sum_i w_i^E E_{i,t}. \tag{3.3.1}$$

3.3.2 Iterated Principal Component Analysis (IPCA)

Principal Component Analysis (PCA) is an important method for dimension reduction in mathematics, and has a wide range of applications in physics, biology, astronomy, finance and other disciplines. It can be formulated in various ways, including using a smaller set of basis to represent the data in the vector space minimising the Mean Squared Error [26], or ascertaining an affine subspace with a smaller dimension where the projected points could best approximate the data [46].

However, the causality of time series data in finance renders this problem more complicated, that financial data is usually received in stream or batches temporally, so that performing PCA either on subsets of data or whole data at once leads to problems of numerical instability or future data leakage.

Therefore, an Iterated Principal Component Analysis (IPCA) is proposed to process the data in subsets temporally, where the eigenvectors of the newly arrived data subset are inherited and refined from the previous subsets by the Ogita and Aishima algorithm [47]. In this way, one can obtain causal and more mathematically robust principal components for financial analysis. The mathematical formulations will be discussed below.

Regular PCA

The dataset is defined as $\mathbf{X} := \{\mathbf{x}^{(i)}\}_{i=1}^N$, where $\mathbf{x}^{(i)} \in \mathbb{R}^p$ is the port activity data at one timestamp of dimension p. These data could be represented by a set of orthonormal basis $\boldsymbol{\phi}_j \in \mathbb{R}^p$ as

$$\mathbf{x}^{(i)} = \sum_{j=1}^{p} a_j^i \boldsymbol{\phi}_j, \tag{3.3.2}$$

where the coordinates are computed as $a_j^i = \boldsymbol{\phi}_j^T \cdot \mathbf{x}^{(i)}$. To reduce the dimension, we have [26, Equation 11.2, page 122]

$$\widehat{\mathbf{x}}_m^{(i)} = \sum_{j=1}^{m} a_j^{(i)} \boldsymbol{\phi}_j + \sum_{j=m+1}^{p} b_j \boldsymbol{\phi}_j, \tag{3.3.3}$$

where b_j are independent of i. Hence, the target is to minimise the error made by this dimension reduction, i.e.,

$$\Delta \mathbf{x}^{(i)} = \mathbf{x}^{(i)} - \widehat{\mathbf{x}}_m^{(i)} = \sum_{j=m+1}^{p} \left[a_j^{(i)} - b_j \right] \boldsymbol{\phi}_j. \tag{3.3.4}$$

By performing optimisation for both b_j and $\boldsymbol{\phi}_j$, the optimised Mean Squared Error (MSE) could be expressed as [26, Equation no. 1, page 124]

$$\text{MSE} = \sum_{j=m+1}^{p} \boldsymbol{\phi}_j^{\star T} \mathbf{C}_{\mathbf{x}} \boldsymbol{\phi}_j^{\star} = \sum_{j=m+1}^{p} \lambda_j, \tag{3.3.5}$$

where λ_j are the eigenvalues of $\mathbf{C}_{\mathbf{x}}$, the covariance matrix of the data, and $\boldsymbol{\phi}_{j^\star}$ are its eigenvectors. Therefore, one can perform PCA by using the Singular Value Decomposition (SVD) of the covariance matrix of the centered data $\mathcal{X}_{N\times p} := \left(\mathbf{I} - \frac{1}{N}\mathbf{1}\mathbf{1}^T\right)\mathbf{X}$. Then SVD is conducted to obtain [26, page 128]

$$\hat{\mathcal{X}}_{N\times p} = \mathbf{U}_k \boldsymbol{\Sigma}_k \mathbf{V}_k^T =: \mathbf{X}_{\text{PCA}} \mathbf{V}_k^T, \tag{3.3.6}$$

for the reduced rank k, where the eigenvector matrices correspond to the covariance matrix $\mathbf{C}_{\mathbf{x}} = \frac{1}{N}(\mathcal{X}^T \mathcal{X})_{p\times p}$.

Iterated PCA

We adhere to the practices stated in the paper of Dr. Paul Bilokon [48] for the implementation of Iterated PCA. The Iterated PCA begins with approximating the eigenvalues of a symmetric matrix $\mathbf{C} \in \mathbb{R}^{N\times N}$ based on a precomputed eigenvector matrix $\hat{\mathbf{U}} \in \mathbb{R}^{N\times N}$. Then, the Ogita and Aishima algorithm is applied to iteratively refine the eigenvectors $\hat{\mathbf{U}}$ based on the eigenvalues and eigenvector matrix stored from previous computations. The pseudo code of this algorithm is displayed in Algorithm 3.3.3 [48].

It is argued that the algorithm above has monotone and quadratic convergence properties under regularity conditions [47]. In this way, one can computes the new arriving subset of the financial data using this algorithm to obtain causal and numerically robust principal components, which could serve as the input features for alpha generation.

3.3.3 Clustering

Since the dimension for ports is considerably high, the explained variance of the principal components is relatively low. Hence, a pre-processing of the data is conducted by performing a k-means clustering on the port activities and applying IPCA on the mean value for each cluster. In this way, it is shown that the explained variance for the first few principal components are larger with better interpretations.

The k-means clustering algorithm is formulated as follows [26, page 111]:

- Initialise a number of N clusters randomly;

- Calculate the centroids of the clusters by $m_i = \frac{1}{|c_i|}\sum_{j\in c_i}\mathbf{x}^{(j)}$ for $i = 1, ..., N$;

- Reassign the data points $\mathbf{x}^{(j)}$ to the nearest centroid, i.e., $p = \text{argmin}_i ||\mathbf{x}^{(j)} - m_i||^2$;

- Iterate the procedures above until the objective function $W(C) = \frac{1}{2}\sum_{i=1}^{N}\frac{1}{|c_i|}\sum_{j,k\in c_i}\|\mathbf{x}^{(j)} - \mathbf{x}^{(k)}\|^2$ converges.

Using the methods above, we have transformed the features of the port activity data in a number of ways, including weighted average, IPCA principal components, and IPCA principal components on the clustered data.

Algorithm 1 Algorithm of Iterated PCA

Require: Covariance matrix $\mathbf{C}$, eigenvector matrix $\hat{\mathbf{U}}$, tolerance *tol*

iteration_count $\leftarrow 0$
while True **do**
  iteration_count $+= 1$
  $\mathbf{R} \leftarrow \mathbf{I} - \hat{\mathbf{U}}^\top\hat{\mathbf{U}}$
  $\mathbf{S} \leftarrow \hat{\mathbf{U}}^\top\mathbf{C}\hat{\mathbf{U}}$
  for $i = 1$ to n **do**
    $\tilde{\lambda}_i \leftarrow s_{ii}/(1 - r_{ii})$
  end for
  $\tilde{\mathbf{D}} \leftarrow \text{diag}(\tilde{\boldsymbol{\lambda}})$
  $\delta \leftarrow 2(\|\mathbf{S} - \tilde{\mathbf{D}}\|_2 + \|\mathbf{C}\|_2\|\mathbf{R}\|_2)$
  for $i = 1$ to n **do**
    for $j = 1$ to n **do**
      $\tilde{e}_{ij} \leftarrow \begin{cases} \frac{s_{ij}+\tilde{\lambda}_j r_{ij}}{\tilde{\lambda}_j - \tilde{\lambda}_i}, & \text{if } |\tilde{\lambda}_i - \tilde{\lambda}_j| > \delta \\ r_{ij}/2, & \text{otherwise} \end{cases}$
    end for
  end for
  $\hat{\mathbf{U}}' \leftarrow \hat{\mathbf{U}} + \hat{\mathbf{U}}\tilde{\mathbf{E}}$
  $\epsilon \leftarrow \|\hat{\mathbf{U}}' - \hat{\mathbf{U}}\|_2$
  if $\epsilon < tol$ **then**
    break
  end if
  $\hat{\mathbf{U}} \leftarrow \hat{\mathbf{U}}'$
end while
return $\hat{\mathbf{U}}'$

Chokepoint Quantity and Volume Estimates:
For chokepoint quantity and volume estimates, similar operations are conducted for features engineering. First of all, the weighted average is applied to quantity and capacity features analogous to section 3.3.1. Also, IPCA procedure is conducted directly on the chokepoint data, since there are only 24 of them, which is much smaller than the port activity data.

In addition, momentum features and mean-reversion features are also computed based on the (weighted) capacities and IPCA results, following the mathematical formulations in section 2.2.2. All these processed or raw features are organised for different experiments and serve as inputs of GP.

3.4 Genetic Programming Framework

The Genetic Programming framework utilised in this thesis is derived from the package gplearn [49], and some modifications and extensions are made to cater the specific needs of this project. The framework utilised for the experiments of this project will first be discussed, and then the changes made to the original gplearn package are introduced.

3.4.1 Framework Description

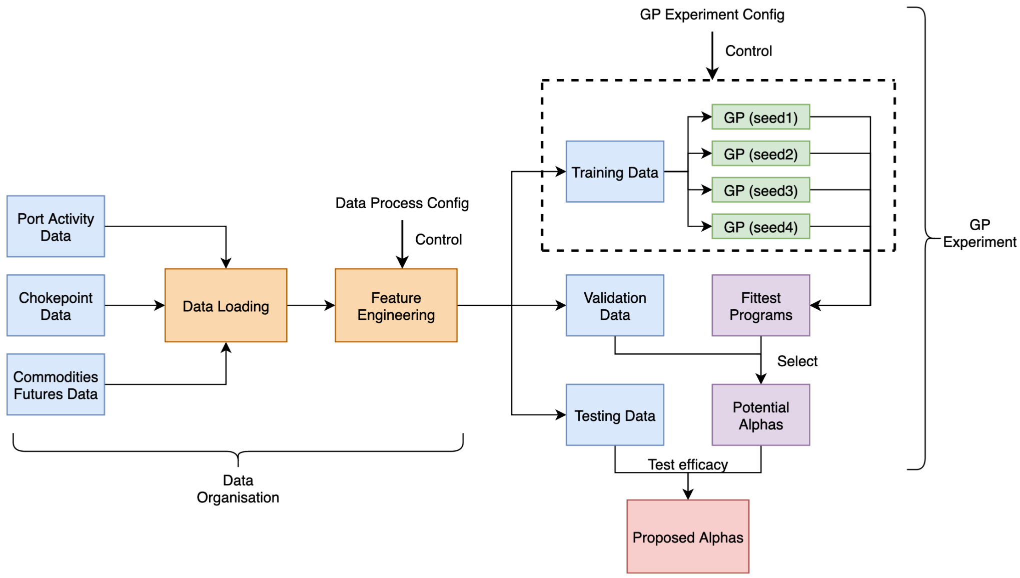

Figure 3.3: Genetic Programming Framework

From Figure 3.3, the different sections of the GP framework are displayed. Firstly, the data used for analysis, including port activity data, chokepoint trade data and commodities futures data are loaded into the program. Then, these data are pre-processed by feature engineering following the methods described in section 3.3. Then this data is split into three sections, the training data, the validation data and the testing data. The training and validation data are in-sample, which means that they are used for finding alpha structures and refine them. The testing data is out-of-sample, which means that it is only used for final evaluation, testing the validity of the structures proposed.

The training data is used to train the GP programs. For one instance of the GP experiment, a number of seeds are used to get a variety of program evolutions, fully exploring the alpha function space and reduce bias caused by randomness. Then the fittest programs are output from the GP process. In most cases, the GP is stable, and the converged fittest program from the last generation serves as the output. These output fittest programs are then selected based on the relative performance of the training fitness score and the validation fitness score, that the selected programs should exhibit good results from the training process, and also have relatively good performance on the validation data among all outputs. However, in some particular cases when the function space is sparse and the programs is hard to converge, or it goes through a harmful mutation in the very end, one might need to select the potential alphas from the programs in the previous generations. This is useful for preventing over-fitting problems.

After the potential alphas are selected based on the training and validation scores, these alphas are verified by computing the metric scores from section 2.2.3 using the testing data, giving unbiased out-of-sample results for the proposed alphas.

Two configurations could dynamically control this framework. As shown in Figure 3.3, Data Process Config could control how the input data is organised, such as the location of data storage, the data source and commodity types used, the range for training, validation, testing split, and what methods the feature engineering adheres to. GP Experiment Config controls the details of the GP experiment process, including experiment name and date, seed generation, initialisation settings, tournament size, probabilities for crossover and mutation operations, genetic function pool and fitness function.

3.4.2 Modifications to Gplearn

A range of modifications are made to the original gplearn package [49], which allows more sophisticated functionalities.

Program Length Penalty

A more flexible and effective program length penalty mechanism is developed for better convergence of GP experiments. Instead of applying the old "parsimony_coefficient" keyword argument directly on program length, the length penalty term could be defined inside fitness functions. This change also makes (by modification) the fitness functions accept more flexible inputs such as program length and other hyperparameters. The length penalty is mathematically formulated as

$$\tilde{f}(p) = \begin{cases} f(p), & \text{if } \mathrm{len}(p) \leq K_1, \\ f(p) - \alpha_1(\mathrm{len}(p) - K_1), & \text{if } K_1 < \mathrm{len}(p) \leq K_2, \\ f(p) - \alpha_1(K_2 - K_1) - \alpha_2(\mathrm{len}(p) - K_2), & \text{if } K_2 < \mathrm{len}(p) \leq K_3, \\ \dots, & \\ -\infty, & \text{if } \mathrm{len}(p) > K_n, \end{cases} \tag{3.4.1}$$

where $f(p)$ is the original fitness score of program p, $\mathrm{len}(p)$ is the program length, and $\tilde{f}(p)$ is the penalised fitness score.

The penalty is conducted in a piece-wise manner, for a set of pre-defined thresholds $K_1, K_2, ..., K_n$, a list of incremental penalty factors are applied to corresponding length according to the thresholds. In this way, one can flexibly determine the penalty strength for different length ranges, and prevent the program length from exploding.

Alpha Selection from Validation

More attributes are added to the gplearn symbolic regressor classes, so that one could retrieve the fittest programs from each generation to inspect the evolution process. Also, the GP experiment framework now computes the validation fitness score and other selected metrics in-time during the experiment, allowing alpha selection to be guided by the relative performance on the training and validation data. The framework also tracks for each seed, whether the programs converge by their fitness scores. These modifications render the experiments more interpretable and improve alpha discovery efficiency.

3.5 Systematic Discovery of Alpha

Apart from the GP experiments, a systematic framework of alpha generation could also be constructed containing procedures from data acquirement, processing, GP, parameter fine-tuning and iterative refinement.

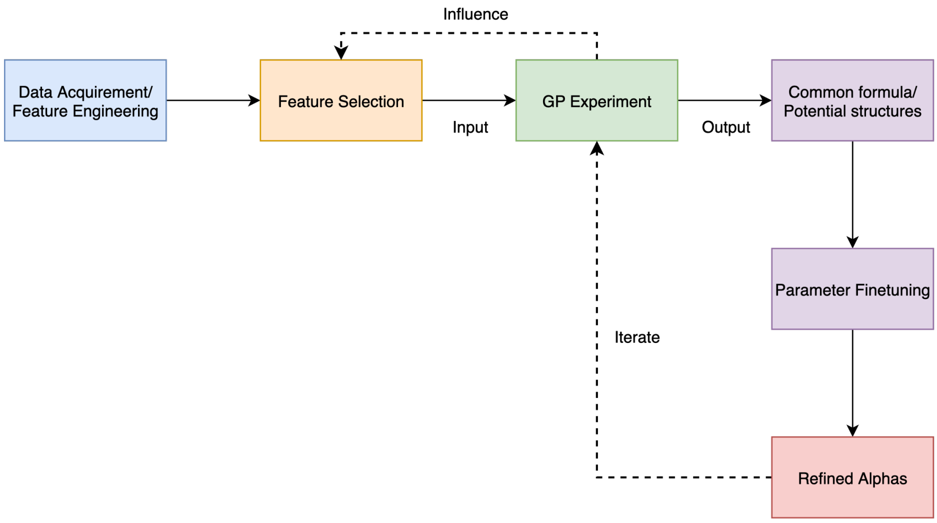

Figure 3.4: Systematic Discovery of Alpha

In the architecture illustrated in Figure 3.4, one can see the whole working pipeline, where the GP experiments generate potential alpha structures and inspirations, and these outputs are then fine-tuned (in-sample) and verified (out-of-sample). The reason for the fine-tune procedure is that the structures proposed by GP might be constrained by the setup of the functional pool, where one could manually inspect the formulas by simplifying the expression or tuning the parameters. A prominent example would be operator composition. A formula of $\text{lag1}(\text{lag1}(\text{lag1}(X)))$ could be composed as $\text{lag3}(X)$.

The quality of GP experiments could also backwardly influence the feature selection stage, and the refined alphas could also serve as warm-start foundations for the GP experiments. Holistically this pipeline formalises the process of alpha discovery, and delivers more robust alphas originated from Genetic Programming and data.

3.6 Multi-objective Fitness Function

Eventually, we propose a novel multi-objective fitness function applied in Genetic Programming, which provides new learning dynamics for alpha discovery. For a trading signal Q, the multi-objective fitness function is defined as

$$\phi(Q) = \lambda_1 \mathrm{SR}_Q - \lambda_2 \mathrm{DD}^*_Q + \lambda_3 \mathrm{PnL}^*_Q + \lambda_4 \beta, \tag{3.6.1}$$

where SR_Q is the annualised Sharpe ratio, DD^*_Q is the accumulation of large values in the drawdown series (larger than the 90% quantile q) normalised by the magnitude of PnL:

$$\mathrm{DD}^*_Q = \frac{1}{\lfloor \log_{10} |\mathrm{PnL}_T| \rfloor} \sum_{t=0}^{T} \mathrm{DD}_t \mathbf{1}_{\mathrm{DD}_t > q}; \tag{3.6.2}$$

$\mathrm{PnL}^*_Q = \log_{10} \mathrm{PnL}_T$ represents the final profit or loss (computed when $\mathrm{PnL}_T > 0$, penalised by $-\infty$ otherwise), and β is the (linear) regression coefficient of the PnL curve.

$\lambda_i, i = 1, 2, 3, 4$ are hyperparameters which could be tuned for different purposes. These hyperparameters can be selected by preliminary training experiments, or cross-validation.

The major terms used in the experiments below is Sharpe ratio and drawdown, since they are the most characteristic statistics of a good trading strategy: high and stable profits with small probabilities of losses. The drawdown statistic (Equation 3.6.2) is designed as such to capture both the magnitude and the frequency of the losses during trading, and it is normalised by the overall magnitude of profits to prevent the GP outputs results which are too extreme. Also, more terms could be added in GP for exploration, such as the transaction costs described in Equation 2.2.10.

The main purpose of this multi-objective fitness function is to obtain a more comprehensive statistic for the performance of alpha signal. Using Sharpe ratio along is useful as the fitness function in GP to a certain extent, but in more complicated scenarios, it is often not representative enough to award the best alpha and penalise the rest.

For example, a major subject in using GP for alpha discovery is finding the alpha which is stable but not over-fitting. It is likely that the GP optimises an excellent curve which completely overfits, or an alpha which has large gains in general, but also has large drawdown which is potentially unstable in the future. Therefore, it is of great importance to balance the profitability and stability of the signal, making the outputs of GP more likely to be valid results.

Following the arguments above, the multi-objective fitness function alleviates this issue by penalising large and persistent draw back periods (through summation of values exceeding a quantile) and awarding large final profits as well as steep upward-slopping curves. It helps to identify the formulas which are more robust and powerful.

Through our experiments in the next chapter, λ_1 is set as 1 as default for calibration purpose, and it is observed that $\lambda_2 = 0.0005$, $\lambda_3 = 0$, $\lambda_4 = 0$ are generally suitable for the GP experiments; $\lambda_3 = 0.1$, $\lambda_4 = 1 \times 10^{-5}$ is used in the alpha refinement stage displayed in Figure 3.4. For features that are more sensitive to noises (like futures prices), the penalty coefficients should be set even smaller, such as $\lambda_1 = 1.25$, $\lambda_2 = 0.00025$.

Chapter 4

Experiments and Results

4.1 Experiment Setup

From this section, the experiments conducted in this project will be elaborated. The experiment procedures closely follow the formulations in section 3.4 and section 3.5. The details of each section are as follows.

4.1.1 Configuration Setting

First of all, the configuration of the experiment is set to control the procedures. It consists of three categories: paths, data generator and experiment setting. Paths control where the data is loaded from and where the experiment outputs will be saved. Data generator controls which data file will be loaded for the experiment, which commodity future is investigated, training-validation-testing split ratio for the dataset, how the raw data is feature engineered and so on. In the experiments, we mainly focus on Brent Crude Front Month. The experiment setting determines the experiment name, date, and other hyperparameters listed in Table 4.1. These information is stored in yaml files, which could be read by the program.

Table 4.1: Configuration of GP experiment settings

Variable	**Explanation**	**Value**
seedAnount	Number of random seeds to draw	5
seedBound	Upper bound for seed values	100
populationSize	Size of the initial GP population	2500
generation	Number of generations of GP	10
initDepth	Range of initial tree depths	$[2, 3]$
tournamentSize	Tournament size for selection	50
stoppingCriteria	Fitness threshold for early stopping	10.0
pCrossover	Probability of crossover	0.5
pSubtreeMutation	Probability of subtree mutation	0.1
pHoistMutation	Probability of hoist mutation	0.1
pPointMutation	Probability of point mutation	0.2
functionSet	Function pool used for GP	1
metric	Fitness function used for GP	`multi`
multiCoef	Coefficients for multi-objective fitness function	$[1, 0.0005, 0, 0]$

4.1.2 Data Loading and Processing

After configurations are read, the IMF PortWatch and commodities futures data are loaded and organised according to the processing methods introduced in section 3.3. The processed maritime trade data is then aligned with the commodities futures data, with the time frame being from 2019-01-01 to 2025-02-21. The aligned data is of hour frequency, and the maritime trade data is filled with appropriate broadcasting. Finally, the data will be split into training, validation and testing according to the ratios (55%, 30%, 15%), and the selected features and targets for GP are organised into datasets which could be loaded into the gplearn framework.

4.1.3 GP Experiment

For each individual experiment, a number of GPs are run with a number of different random seeds. The GP is trained using only the training data, and for each generation, the fittest program will be recorded, and prediction along with computation of metric scores are conducted on the validation set. The potential alphas are selected based on the training performance and the relative rank of the validation metric score. For example, the GP results will be ranked by $\text{fitness}_{\text{val}}$, and while the training fitness score needs to be good, the validation score should not decay to a large extent. The top formulas following these criteria will be selected as potential alphas. Finally, these potential alphas are evaluated on the out-of-sample testing dataset, for testing their unbiased efficacy.

However, it is observed that for different data and feature engineering, the quality and quantity of potential alphas generated by GP are different. For some cases where the effective alphas are sparse in the space, it needs more experiment rounds, and to select potential alpha from the fittest programs in non-final (but stable) generations.

4.1.4 Alpha Refinement

As stated in Figure 3.4, after the GP procedure, a list of alphas are proposed. For the most promising formulas from the fitness scores and the formulas which often converge from different seeds, refinement will be made to further craft these alphas. It will further conduct feature selection and parameter tuning on the training and validation data (in-sample), and eventually they are evaluated using the testing data (out-of-sample). For instance, the subset of ports and weight coefficients will be tuned for chokepoint capacity data, and different principal components will be examined for IPCA data.

4.1.5 Final Evaluation and Visualisation

After the alphas are proposed and refined, the comprehensive out-of-sample analysis and evaluation will be made using the unseen testing data. The trading signal, cumulative PnL and other plots are generated, and the metrics in section 2.2.3 are computed, including Sharpe ratio, multi-objective fitness score, profit per trade, average holding period, and autocorrelation of the signal.

4.2 Alphas from Chokepoint Mean Capacity

The first and most crucial data investigated is the chokepoint trade estimate data, which shows promising results in the experiments. Two methods of feature engineering are attempted: momentum on (weighted) average and IPCA. Through our experiments, it is shown that chokepoint capacities are much more superior than ship quantities, and that the efficacy between whole capacity and individual ship category are not distinctive. Both data preparation methods could make the GP experiments generate potential alphas, some of which could be further fine-tuned. The in-sample results from GP and fine-tuning, and the out-of-sample evaluation are presented below in order.

4.2.1 In-sample Results

GP Experiments

For momentum on (weighted) average of chokepoint capacities, the features are mean capacity and momentum factors listed in Table 4.2. GPs are conducted using three fitness functions: Sharpe Ratio (without length penalty (LP)), Sharpe Ratio (with length penalty $(10, 15)$ (LP)), and multi-objective fitness function (with length penalty $(10, 15)$ (LP), and coefficients: $(1, 0.0007, 0, 0)$ as in section 3.6). Some training and validation scores as well as the fittest programs for each generation are displayed in Table 4.3. It is worth stating that since the multi-objective fitness function is penalised by the drawdown, it is sensible to have a negative fitness score with moderate Sharpe ratio and positive cumulative PnL when the market is volatile or unstable for some periods.

Table 4.2: Input features for GP on chokepoint average capacities

Feature name	**Variable name**	**Expression**
Mean capacity	X0	Capacity: c
Momentum Factor 1	X1	$c - \text{lag40}(c)$
Momentum Factor 2	X2	$c - \text{lag50}(c)$
Momentum Factor 3	X3	$c - \text{lag60}(c)$
Momentum Factor 4	X4	$c - \text{lag70}(c)$
Volatility	X5	$\text{rolling\_std24} * 60(\text{price})$

In these experiments, the potential alphas are selected using these criteria:

- Rank the relative performance of outputs by training and validation fitness scores;
- The training fitness score is within a desired range (for multi-objective, above 1);
- The validation fitness score does not deteriorate to a large extent (for multi-objective, above -5 for smaller training scores, above -11 for larger training scores);
- Choose the one which has the highest relative validation score if it converges;
- If neither of these are satisfied, no alpha is chosen.

For example, add(mean60(X1), lag70(X4)) and lag70(X4) will be selected from the two seed cases from Table 4.3, since they are both converged formula with good training and relatively good validation scores, and the validation scores do not deteriorate to a large extent. It should be stated that the validation score is relatively smaller than the training score due to the fact that there are different economic cycles within the data, so that the relative performance could be different. Due to the scope of IMF PortWatch data available, only one training-validation set is established. Nevertheless, all these potential alphas will be evaluated on the pure out-of-sample testing data for testing efficacy.

For different fitness functions, it is observed that all three types (Sharpe without length penalty, Sharpe with length penalty, multi-objective) could guide the program to converge. However, Sharpe without length penalty often leads to program length explosion

Table 4.3: Training and validation results for GP on chokepoint average capacities (multi-objective fitness function)

Gen	Best program	Train	Validation
Seed 27			
0	`lag70(X4)`	1.0888	-1.5024
1	`lag70(add(-0.733, X4))`	1.0888	-1.5024
2	`lag70(X4)`	1.0888	-1.5024
3	`lag70(X4)`	1.0888	-1.5024
4	`lag70(X4)`	1.0888	-1.5024
5	`lag70(X4)`	1.0888	-1.5024
6	`lag70(sub(X4, 0.822))`	1.0888	-1.5024
7	`lag70(X4)`	1.0888	-1.5024
8	`add(mean60(X1), lag70(X4))`	1.1716	0.9266
9	`add(mean60(X1), lag70(X4))`	1.1716	0.9266
Seed 9			
0	`lag70(sub(X4, 0.594))`	1.0888	-1.5024
1	`lag70(add(lag50(X5), mean1(X4)))`	1.1008	-1.6022
2	`lag70(add(lag50(X5), mean1(X4)))`	1.1008	-1.6022
3	`lag70(add(X4, X5))`	1.0900	-1.5023
4	`lag70(X4)`	1.0888	-1.5024
5	`lag70(X4)`	1.0888	-1.5024
6	`lag70(X4)`	1.0888	-1.5024
7	`lag70(X4)`	1.0888	-1.5024
8	`lag70(X4)`	1.0888	-1.5024
9	`lag70(X4)`	1.0888	-1.5024

Table 4.4: Training and validation results for GP on chokepoint average capacities (Sharpe-ratio with length penalty)

Gen	Best program	Train	Validation
Seed 38			
0	`sub(lag10(X4), X4)`	1.2091	-0.5785
1	`add(lag10(X4), lag70(X4))`	1.4346	0.4742
2	`add(lag1(lag50(X1)), sub(lag10(X4), X4))`	1.5253	0.0379
3	`add(lag10(X4), add(lag1(lag30(lag10(X2))), lag70(X4)))`	1.8732	0.4102
4	`add(lag10(X4), add(lag1(lag30(lag10(X2))), lag70(X4)))`	1.8732	0.4102
5	`add(add(lag10(X4), lag70(X4)), lag1(lag10(lag30(X2))))`	1.8732	0.4102
6	`add(lag1(lag10(lag30(X2))), add(lag10(X4), lag70(X4)))`	1.8732	0.4102
7	`add(lag1(X3), add(lag1(lag30(lag10(X2))), lag70(X4)))`	1.9833	0.5226
8	`add(lag1(X3), add(lag1(lag30(lag10(X2))), lag70(X4)))`	1.9833	0.5226
9	`add(lag1(X3), add(lag1(lag30(lag10(X2))), lag70(X4)))`	1.9833	0.5226
Seed 35			
0	`add(lag1(mean60(X3)), abs(mean20(X4)))`	1.1553	0.1159
1	`add(lag1(mean60(X3)), abs(lag10(abs(mean60(X2)))))`	1.3082	0.4271
2	`add(lag1(X3), lag50(X1))`	1.5756	0.8173
3	`add(lag1(X3), lag50(lag1(X1)))`	1.7015	0.8428
4	`lag1(add(X3, add(abs(mean20(X4)), lag10(lag30(X2)))))`	1.8104	0.5192
5	`lag1(add(X3, add(abs(mean20(X4)), lag50(lag1(X1)))))`	1.8116	0.8291
6	`lag1(add(X3, add(lag50(mean20(X4)), lag10(lag30(X2)))))`	1.8258	0.5222
7	`lag1(add(X3, add(abs(mean60(X4)), lag50(lag1(X1)))))`	1.8372	0.9078
8	`lag1(add(X3, add(abs(mean80(X4)), lag1(lag50(X1)))))`	1.8397	0.9840
9	`lag1(add(X3, add(abs(mean80(X4)), lag1(lag50(X1)))))`	1.8397	0.9840

(above 30), which frequently results in overfitting. Other two methods could find more interpretable alphas due to the length penalisation mechanism.

Alpha Fine-tuning

From different seeds, it is observed that structures like lag of momentum factor (such as lag70(X4)) and rolling mean of momentum factor (such as mean80(X4)) are frequently output in the converged formula in GP experiments and have good fitness scores. Hence, fine tuning is proposed to further strengthen the efficacy of these factors.

The procedure is conducted as follows: First of all, the hyperparameters of these alphas are optimised using the multi-objective function based on training and validation data (lag size, rolling mean window size and momentum factor parameters), then subsets of chokepoints are iterated to find the best configuration. After a certain subset of chokepoints is determined, the weights for each chokepoint in this subset is further optimised.

It is shown that using the whole in-sample data for optimisation is more stable than splitting the data sections for cross-validation. The coefficients of the multi-objective function are set as $(1, 0.0005, 1, 0)$ for the hyperparameter tuning, and are set as $(1, 0.00005, 1, 0.00001)$ for tuning chokepoints and weights. The drawdown term decreases while the slope coefficient is introduced, in order to find a strategy which is fully profitable.

After this process, a weighted average on a subset of chokepoints is selected as the input feature for this refined alpha. An example of the results is displayed below:

Table 4.5: Optimal setting for selected maritime chokepoints

Chokepoint #	Name	Weight
5	Malacca Strait	0.20
8	Gibraltar Strait	0.20
9	Dover Strait	0.20
13	Tsugaru Strait	0.30
14	Luzon Strait	0.10

4.2.2 Out-of-sample Results

In this section, the out-of-sample results for alphas on weighted average chokepoint capacities using the testing data are displayed, based on the formulas selected and refined from previous parts.

Testing Metrics of Alphas

The cumulative PnL curves of two selected alphas based on (weighted) average of chokepoint capacities are displayed in Figure 4.1. They both manifest great increasing trends for profits, and possess Sharpe ratio of 2.92 and 2.67 respectively. The actual trading signals of these alphas are plotted in Figure 4.2, and other metric scores are listed in Table 4.6 (The metrics without units behind have tick size as units).

One can see that Strategy 1 not only has an outstanding Sharpe Ratio (2.92), but also has reasonable profit per trade (293.68 \$) and average holding period (1.81 days). Although Strategy 2 has lower Sharpe Ratio and PnL, it has lower volatility and drawdown, making it a more conservative choice.

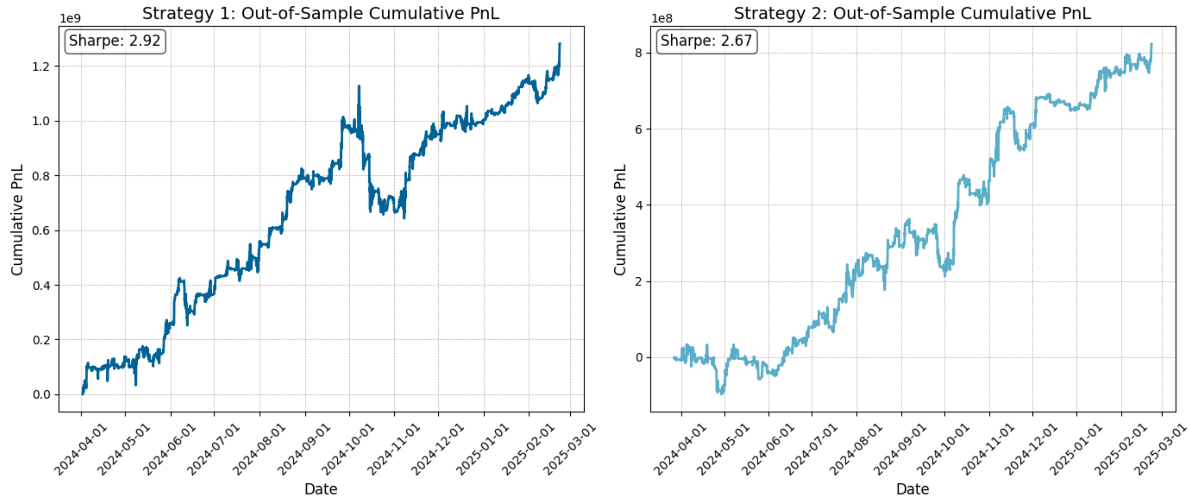

Figure 4.1: Out-of-sample cumulative PnL ((weighted) average of chokepoint capacities)

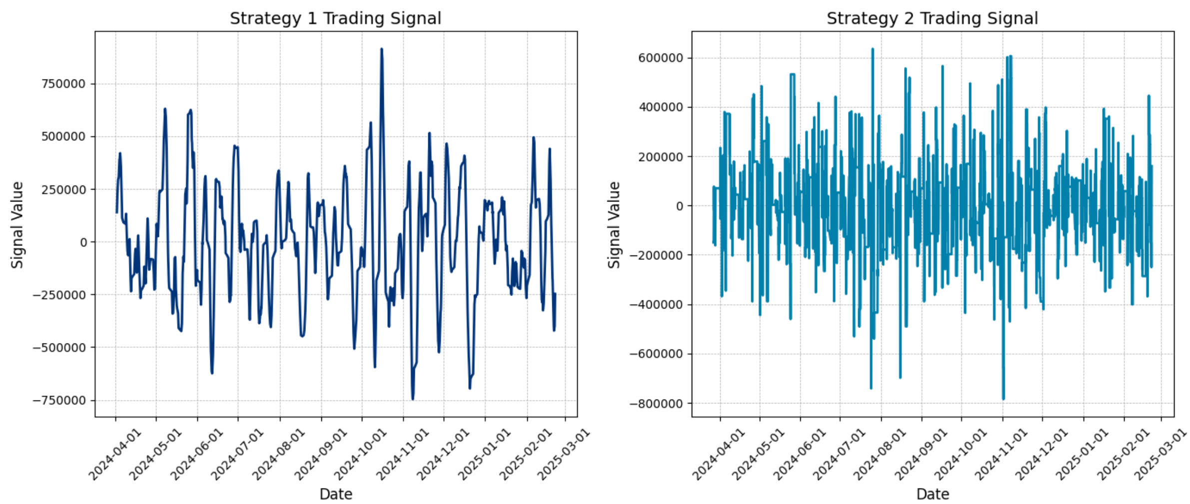

Figure 4.2: Signals of alphas ((weighted) average of chokepoint capacities)

Table 4.6: Key performance metrics for two strategies ((weighted) average of chokepoint capacities)

Strategy	Sharpe	Total PnL	Volatility	Max Drawdown	Turnover	Profit/Trade ($)	Avg Holding (days)	AC(1)
Strategy 1	2.9210	1.281e9	6.726e6	4.838e8	4.350e7	293.68	1.81	0.999
Strategy 2	2.6743	8.230e8	4.627e6	1.507e8	1.498e8	54.85	0.44	0.902

4.2.3 Comparison Between Fitness Functions

We also compare the GP results for different fitness functions (Table 4.7), to investigate how the alpha discovery will be affected by different fitness functions. It is shown that although both three methods could find profiting alphas in general, the score dynamics and proposed alphas are different in nature.

For Sharpe ratio without length penalty, the GP is likely to fit a formula with a large length, resulting in a much higher average length of 25. Even though some of these long alphas have moderate Sharpe ratio for testing (0.799), but it is hard to be interpreted, and many other cases lead to overfitting. When the length penalty mechanism is introduced in section 3.4.2, the GP prunes excessively long formulas, and the adjusted fitness scores guide the programs to discover shorter outputs (average length of 9.40 and 6.30). A major leap of testing scores is for the multi-objective fitness function, where the average Sharpe ratio for testing is increased to above 1, with the best one being 2.436. This indicates that the alphas selected using the multi-objective fitness functions have better performance than other fitness functions.

Table 4.7: Summary statistics for different fitness functions ((weighted) average of chokepoint capacities)

Method	Avg Length	Avg Training Score	Avg Testing Score	Avg Sharpe Ratio	Best Sharpe Ratio
Sharpe (Without LP)	25.00	2.172469	0.002094	0.002094	0.798712
Sharpe (With LP)	9.40	1.796359	-0.110804	-0.110804	1.031494
Multi-objective	6.30	1.124490	-2.683167	1.007765	2.436171

4.3 Alphas from Chokepoint Capacity (IPCA)

As stated above, the second operation scheme on chokepoint capacity data is IPCA, where the features are the data projected onto the principal components, which are obtained from IPCA on capacities for different 24 chokepoints (10 top principal components are used, Table 4.8). A series of GP experiments are made either on all the IPCA features, or a truncated version where only the features corresponding to the first few eigenvectors are selected. Adhering to similar procedures, alphas are selected based on the training and validation scores from GPs. Some of the selected alphas are further fine-tuned, and finally, the out-of-sample results are displayed.

Table 4.8: Input features for GP on chokepoint capacities (IPCA)

Feature name	**Variable name**	**Expression**
Principal Component Score 1	X0	$X_{\text{IPCA}}[:, 0]$
Principal Component Score 2	X1	$X_{\text{IPCA}}[:, 1]$
...	...	...
Principal Component Score 9	X8	$X_{\text{IPCA}}[:, 8]$
Principal Component Score 10	X9	$X_{\text{IPCA}}[:, 9]$

4.3.1 In-sample Results

GP Experiments

Since the multi-objective fitness function is proved to be useful for chokepoint capacity data, we mainly focus on this for result demonstrations, and make the other fitness metrics as benchmark. The GP experiments could output converged alphas, and some of the selected formula evolutions are displayed in Table 4.9 (Multi-objective) and Table 4.10 (Sharpe with LP). The GP outputs a variety of formulas, where there are still momentum based alphas on the principal components, which is consistent with our previous findings (seed 36), and other formula structures on single (seed 98) or multiple principal components (seed 11, 25).

Table 4.9: Training and validation results for GP on chokepoint capacities IPCA (multi-objective fitness function)

Gen	Best program	Train	Validation
	Seed 98		
0	`neg(mean40(lag70(X4)))`	1.9011	-35.3237
1	`lag70(mean60(neg(sign(X4))))`	2.1290	-59.1230
2	`lag70(mean60(neg(sign(X4))))`	2.1290	-59.1230
3	`neg(mean40(lag70(sign(X4))))`	2.3144	-3.5417
4	`neg(mean40(lag70(sign(X4))))`	2.3144	-3.5417
5	`neg(mean40(lag70(sign(X4))))`	2.3144	-3.5417
6	`neg(mean40(lag70(sign(X4))))`	2.3144	-3.5417
7	`neg(mean40(lag70(sign(X4))))`	2.3144	-3.5417
8	`neg(mean40(lag70(sign(X4))))`	2.3144	-3.5417
9	`neg(mean40(lag70(sign(X4))))`	2.3144	-3.5417
	Seed 36		
Gen	**Best program**	**Train**	**Validation**
0	`neg(lag50(lag50(X4)))`	1.5497	-18.4518
1	`lag50(sub(mean80(lag70(X6)), mean60(X4)))`	1.8006	-5.8254
2	`lag50(sub(mean80(lag70(X6)), mean60(X4)))`	1.8006	-5.8254
3	`lag50(sub(mean40(lag70(X6)), mean60(X4)))`	2.0299	-3.8607
4	`lag50(sub(mean40(lag70(X6)), mean60(X4)))`	2.0299	-3.8607
5	`lag50(sub(mean40(lag70(X6)), mean60(X4)))`	2.0299	-3.8607
6	`lag50(sub(mean40(lag70(X6)), mean60(X4)))`	2.0299	-3.8607
7	`lag50(sub(mean40(lag70(X6)), mean60(X4)))`	2.0299	-3.8607
8	`lag50(sub(mean40(lag70(X6)), mean60(X4)))`	2.0299	-3.8607
9	`lag50(sub(mean40(lag70(X6)), mean60(X4)))`	2.0299	-3.8607

Table 4.10: Training and validation results for GP on chokepoint capacities IPCA (Sharpe with LP)

Gen	Best program	Train	Validation
	Seed 25		
0	`mean40(lag30(sub(X13, X22)))`	2.1137	-0.7277
1	`lag70(sub(lag1(X13), mean40(X4)))`	2.3094	-0.4187
2	`lag70(sub(mean40(lag30(sub(X13, X22))), mean40(X4)))`	2.4476	0.3201
3	`lag70(sub(mean60(lag30(sub(X13, X22))), mean40(X4)))`	2.4786	0.1648
4	`add(lag70(sub(mean80(X13), mean40(X4))), mean40(lag30(sub(X13, X22))))`	2.5492	-0.5079
5	`add(lag70(sub(lag50(X13), mean40(X4))), mean40(lag30(sub(X13, X22))))`	2.5993	0.0833
6	`add(lag70(sub(lag50(X13), mean40(X4))), mean40(lag30(X13)))`	2.7616	0.2977
7	`add(lag70(sub(lag50(X13), mean40(X4))), mean20(lag30(X13)))`	2.7783	0.2965
8	`add(X20, add(X11, lag70(sub(lag50(X13), mean40(X4)))))`	2.7873	0.3155
9	`add(X20, add(X11, lag70(sub(lag50(X13), mean40(X4)))))`	2.7873	0.3155
	Seed 11		
Gen	**Best program**	**Train**	**Validation**
0	`lag70(lag30(sub(X22, X4)))`	1.9052	-0.6558
1	`mean80(mean80(sub(X13, X4)))`	2.1872	-0.2254
2	`add(mean60(mean1(mean60(X13))), lag50(mean80(sub(X13, X4))))`	2.2986	-0.3658
3	`add(mean40(lag30(X13)), lag50(mean80(sub(X13, X4))))`	2.4336	-0.4656
4	`add(mean60(mean40(X13)), lag70(mean40(sub(mean20(mean80(X13)), X4))))`	2.5557	-0.4530
5	`add(mean40(lag30(X13)), lag70(mean40(sub(lag30(X13), X4))))`	2.7450	-0.0056
6	`add(mean40(lag30(X13)), lag70(mean40(sub(lag30(X13), X4))))`	2.7450	-0.0056
7	`add(mean40(lag30(X13)), lag70(mean40(sub(lag30(X13), X4))))`	2.7450	-0.0056
8	`add(mean20(lag30(X13)), lag70(mean40(sub(lag30(X13), X4))))`	2.7523	-0.0056
9	`add(mean20(lag30(X13)), lag70(mean40(sub(lag30(X13), X4))))`	2.7523	-0.0056

Alpha Fine-tuning

After the GP experiments, alpha refinement is conducted on the most promising alphas assessed by the criteria above. The selected structures include the lagged difference between 2-3 rolling mean/lagged principal components, and the hyperparameters in these alphas are optimised using the training and validation data. An example is presented below:

$$\alpha_{\text{GP}} = \text{lag}_{10}\big(\text{lag}_{30}(|\text{lag}_{50}(X_1 - X_6)|)\big) + \text{mean}_{80}\big(-|X_0|\big)$$
$$\xrightarrow{\text{refinement}} \alpha_{\text{RE}} = \text{lag}_{8}\big(|\text{lag}_{82}(X_1 - X_6)|\big) + \text{mean}_{84}\big(-|X_0|\big).$$

where $\text{lag}_n(X)$ and $\text{mean}_n(X)$ are lag and rolling mean operations of window size n.

4.3.2 Out-of-sample Results

In this section, the out-of-sample results on the testing data are displayed, based on the alphas selected and refined from the previous part.

Testing Metrics of Alphas

From Figure 4.3 and Figure 4.4, the out-of-sample cumulative PnL and trading signal curves are plotted. The IPCA processed features of chokepoint capacities show even greater potentials for profitability, where the highest Sharpe ratio increased to 3.24 and the highest final PnL increased to around 8×10^9. Other performance metrics of these alphas are reported in Table 4.11.

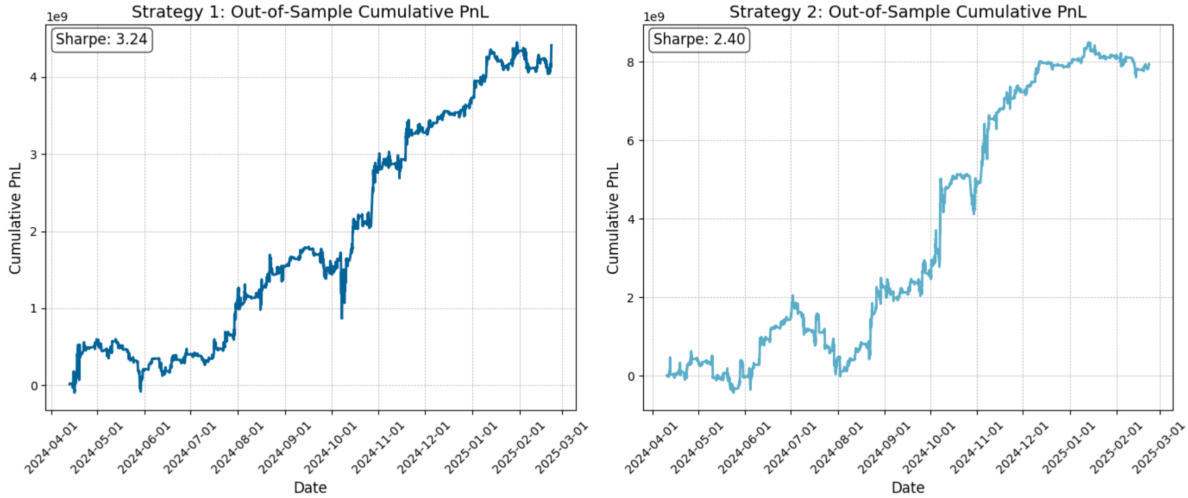

Figure 4.3: Out-of-sample cumulative PnL (chokepoint capacities IPCA)

Table 4.11: Key performance metrics for two trading strategies (chokepoint capacities IPCA)

Strategy	Sharpe	Total PnL	Volatility	Max Drawdown	Turnover	Profit/Trade ($)	Avg Holding (days)	AC(1)
Strategy 1	3.2397	4.409e9	2.357e7	9.295e8	1.359e8	320.75	1.62	0.999
Strategy 2	2.3969	7.951e9	4.510e7	2.066e9	4.621e8	171.98	0.34	0.962

From Table 4.11, one can see strategy 1 has higher Sharpe ratio, lower volatility, higher profit per trade and autocorrelation, while strategy 2 has even higher PnL, but with larger volatility, drawdown, lower profit per trade and autocorrelation. Hence, strategy 1 is more robust while strategy 2 is more risky but potentially more profitable.

Comparison Between Fitness Functions

The GP experiment results for different fitness functions are presented in Table 4.12. It is shown that the length explosion problem for Sharpe Ratio without length penalty is

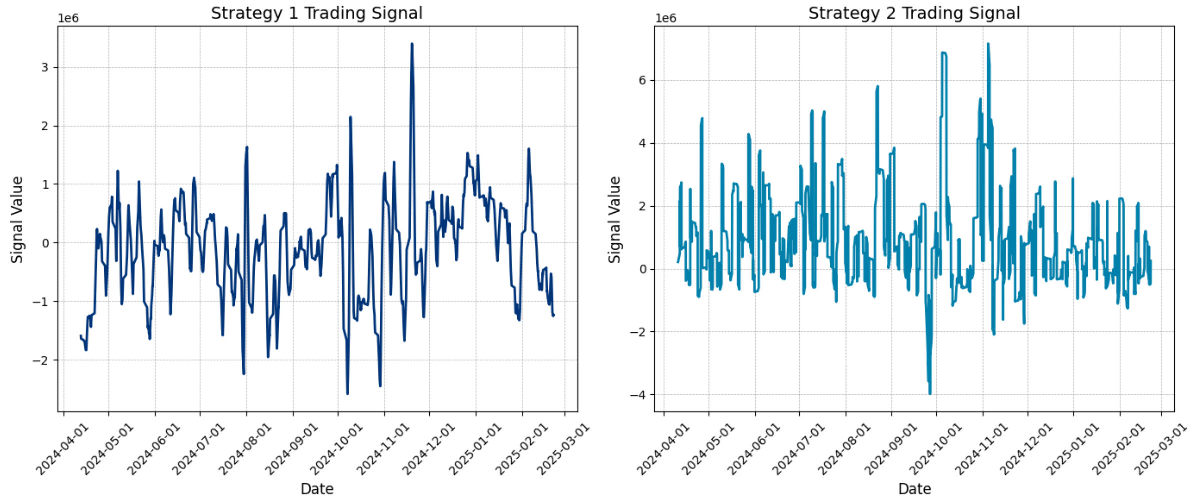

Figure 4.4: Signals of alphas (chokepoint capacities IPCA)

alleviated for these IPCA features, that Sharpe ratio methods could both propose moderate alphas with similar training and testing scores. However, the results of multi-objective fitness function still considerably exceed other methods, where the average and best Sharpe ratios are better than that of other fitness function groups.

Table 4.12: Summary statistics for different fitness functions (chokepoint capacities IPCA)

Method	Avg Length	Avg Training Score	Avg Testing Score	Avg Sharpe Ratio	Best Sharpe Ratio
Sharpe (Without LP)	8.40	2.798837	0.332630	0.332630	1.098517
Sharpe (With LP)	9.20	2.682134	0.258050	0.258050	0.715489
Multi-objective	8.60	2.085852	-0.207702	1.065904	2.203508

4.3.3 Large Scale Experiments

To fully verify the robustness of the experiment procedure, a large scale experiment is conducted using the best configurations sated above, with the multi-objective fitness function. 50 random seeds are chosen for GP experiments, and top 10 results are chosen based on the relative rank for validation score (while training score is satisfactory). This large scale experiment further tests the validity of the GP experiments, and increases one's confidence of taking inspirations or outputs from the GP results for refinement and production.

The results are displayed in Table 4.13. The average out-of-sample Sharpe ratio is around 2.05, indicating that the formulas selected in-sample are indeed valid for the unseen out-of-sample testing data. And it generally follows the pattern that programs with higher rank have higher fitness scores and Sharpe ratio for testing.

4.4 Alphas from Commodity Futures Data

We also tried to construct alphas solely from the commodity futures data. Based on some preliminary GP experiments on the training and validation data, alphas constructed from price mainly follow mean-reverting structures. Thus, the input features are formulated as follows in Table 4.14.

Table 4.13: Test performance of best GP programs (IPCA capacities, large scale)

Seed	**Rank**	**Fitness_test**	**Sharpe_test**
192	1	1.8764	2.2172
24	2	2.0588	2.3377
41	3	1.3963	1.8093
174	4	1.9810	2.4551
182	5	2.0823	2.7179
98	6	1.4143	1.7975
138	7	1.4143	1.7975
88	8	1.4143	1.7975
164	9	1.4143	1.7975
63	10	1.4143	1.7975

Table 4.14: Input features for GP on commodity futures

Feature name	**Variable name**	**Expression**
Commodity Futures	X0	Price: p
Mean-Reversion Factor 1	X1	rolling_mean10$(p) - p$
Mean-Reversion Factor 2	X2	rolling_mean20$(p) - p$
Mean-Reversion Factor 3	X3	rolling_mean40$(p) - p$
Mean-Reversion Factor 4	X4	rolling_mean60$(p) - p$
Mean-Reversion Factor 5	X5	rolling_mean80$(p) - p$
Volatility	X6	rolling_std24 $* 60(p)$

4.4.1 In-sample Results

GP Experiments

For GP experiments using the price features, the coefficients of the multi-objective fitness function are adjusted. Some preliminary runs show that it is more sensitive to over-fitting, and the drawdown term in the fitness function is not as stable as previous cases, since the price fluctuates with larger volatility than maritime trade factors. Therefore, we adjusted the coefficients as $(1.25, 0.00025, 0, 0)$. Some GP formula evolutions are displayed in Table 4.15 and Table 4.16.

Alpha Fine-tuning

The best performing alphas using the multi-objective fitness function are selected based on the training and validation score, and then are refined by parameter optimisation. The structures of these alphas are based on one lagged mean-reversion feature, and summation of two lagged mean-reversion features, and some are further applied with a sign operator. The coefficients for the multi-objective function is further set as $(1.25, 0.0001, 1, 0)$ for refinement. However, the refinement shows that the formula selected by GP is already having the highest scores for a broader parameter space, and thus the refined alpha is constructed by simplifying the formula proposed in the GP, such as:

$$\alpha_{\text{RE}} = \text{lag}_{70}(X_1) + \text{lag}_{22}(X_1).$$

where $\text{lag}_n(X)$ is the lag operation of window size n.

Table 4.15: Training and validation results for GP on commodity futures (multi-objective fitness function)

Gen	Best program	Train	Validation
Seed 36			
0	`lag70(X1)`	0.1187	-8.5661
1	`lag1(lag10(lag10(X1)))`	0.2185	-1.5328
2	`lag1(lag1(lag10(lag10(X1))))`	0.7426	0.3650
3	`lag1(lag1(lag1(lag10(lag10(X1)))))`	0.7483	0.3348
4	`add(lag70(X1), lag1(lag1(lag10(lag10(X1)))))`	0.8002	-0.3207
5	`add(lag70(X1), lag1(lag1(lag10(lag10(X1)))))`	0.8002	-0.3207
6	`add(lag70(X1), lag1(lag1(lag10(lag10(X1)))))`	0.8002	-0.3207
7	`add(lag70(X1), lag1(lag1(lag10(lag10(X1)))))`	0.8002	-0.3207
8	`add(lag70(X1), lag1(lag1(lag10(lag10(X1)))))`	0.8002	-0.3207
9	`add(lag70(X1), lag1(lag1(lag10(lag10(X1)))))`	0.8002	-0.3207
Seed 41			
Gen	**Best program**	**Train**	**Validation**
0	`lag10(lag1(lag10(X1)))`	0.2185	-1.5328
1	`lag10(lag1(lag10(X1)))`	0.2185	-1.5328
2	`lag10(lag1(lag1(lag10(X1))))`	0.7426	0.3650
3	`lag10(lag1(lag1(lag1(lag10(X1)))))`	0.7483	0.3348
4	`lag10(lag1(lag1(lag1(lag10(X1)))))`	0.7483	0.3348
5	`lag10(lag1(lag1(lag1(lag10(X1)))))`	0.7483	0.3348
6	`lag10(lag1(lag1(lag1(lag10(X1)))))`	0.7483	0.3348
7	`lag10(lag1(lag1(lag1(lag10(X1)))))`	0.7483	0.3348
8	`lag10(lag1(lag1(lag1(lag10(X1)))))`	0.7483	0.3348
9	`lag10(lag1(lag1(lag1(lag10(X1)))))`	0.7483	0.3348

Table 4.16: Training and validation results for GP on commodity futures (Sharpe with LP)

Gen	Best program	Train	Validation
Seed 49			
0	`lag70(X1)`	0.6629	-0.4184
1	`add(lag70(neg(mean60(X5))), add(X3, -0.175))`	0.7267	0.3064
2	`add(lag70(neg(mean60(X5))), add(X3, lag70(X1)))`	0.9051	0.1689
3	`add(lag70(neg(mean60(X5))), add(X2, lag70(X1)))`	0.9170	0.2262
4	`add(lag70(neg(lag1(mean80(X3)))), add(X2, lag70(X1)))`	0.9248	0.1628
5	`add(lag70(neg(mean60(X5))), add(X2, lag70(X2)))`	0.9270	0.1305
6	`add(lag70(neg(lag10(X5))), add(X2, lag70(X3)))`	0.9333	0.0665
7	`add(lag70(neg(lag10(X5))), add(X2, lag70(X2)))`	0.9604	0.1733
8	`add(lag70(neg(lag10(X5))), add(X2, lag70(X2)))`	0.9604	0.1733
9	`add(lag70(neg(lag10(X5))), add(X2, lag70(X2)))`	0.9604	0.1733
Seed 59			
Gen	**Best program**	**Train**	**Validation**
0	`sub(mean1(mean1(X1)), lag70(neg(X2)))`	0.6696	-0.2202
1	`lag1(add(X1, lag10(lag10(X2))))`	0.8599	0.1922
2	`lag1(add(X1, lag10(lag10(X2))))`	0.8599	0.1922
3	`add(lag1(add(X1, lag10(lag10(X2)))), lag70(X1))`	1.0520	-0.0100
4	`lag1(add(X1, lag1(add(lag70(X1), lag10(lag10(X2))))))`	1.0877	0.0538
5	`add(lag1(lag1(lag10(lag10(X2)))), lag70(X1))`	1.0934	0.1198
6	`lag1(add(X1, lag1(add(lag70(X1), lag10(lag10(X1))))))`	1.1054	0.0656
7	`add(lag1(lag1(lag10(lag10(X1)))), lag70(X1))`	1.1559	0.1609
8	`add(lag1(lag1(lag10(lag10(X2)))), mean40(neg(X2)))`	1.1851	0.4932
9	`add(lag1(lag1(lag10(lag10(X2)))), mean40(neg(X2)))`	1.1851	0.4932

4.4.2 Out-of-sample Results

After the potential alphas are selected and refined using the in-sample data, an out-of-sample analysis is conducted using the testing data.

Testing Metrics of Alphas

We choose the summation of two mean-reversion factors as benchmark here, to compare the unseen out-of-sample performance of different selected alphas. Figure 4.5 shows the cumulative PnL for the benchmark (left) and further applying a sign operator on it (right). The Sharpe ratios are more moderate compared with the ones based on chokepoint capacities in previous sections, but these strategies still generate relatively good cumulative PnL with Sharpe ratio round or above 1. The corresponding trading signals are displayed in Figure 4.6.

Figure 4.7 shows the cumulative PnL for the benchmark (left) and one mean-reversion factor (right), which is the most common output of the GP programs. The corresponding trading strategies are displayed in Figure 4.8. The one mean-reversion factor has a higher Sharpe ratio (1.50), lower volatility and drawdown, but with a lower final PnL compared to the two factor benchmark.

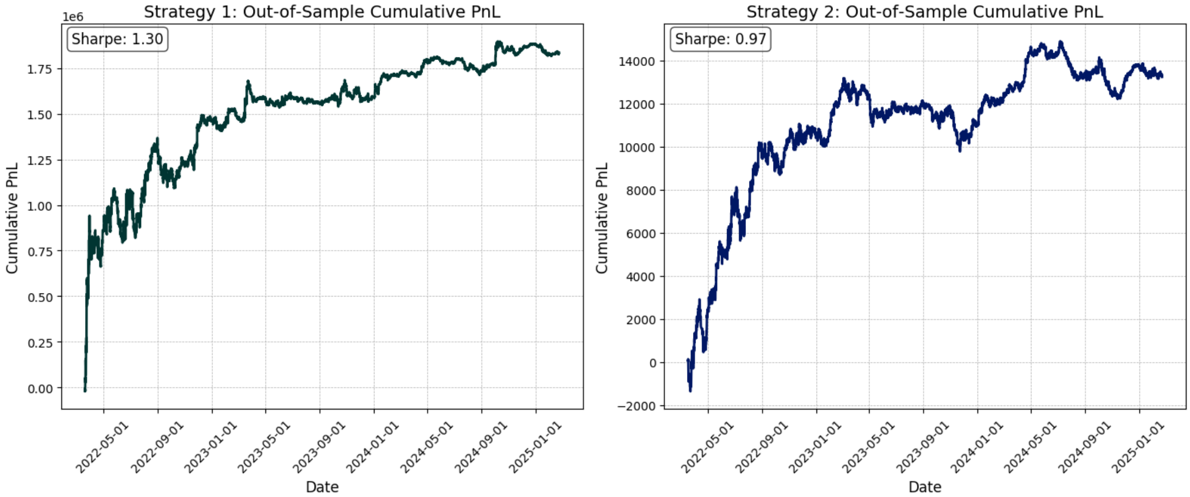

Figure 4.5: Out-of-sample cumulative PnL (commodity futures, benchmark and signed)

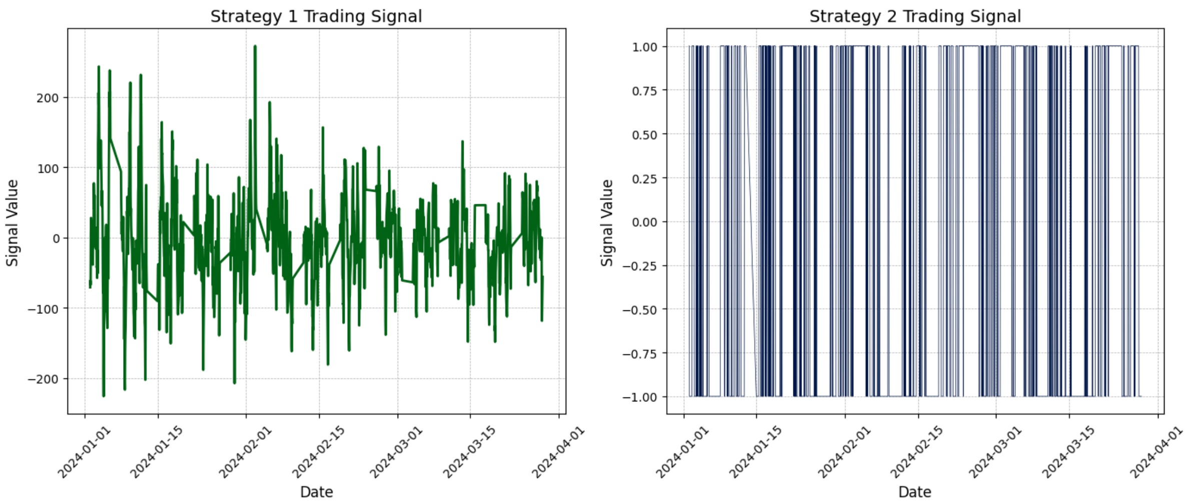

Figure 4.6: Signals of alphas (commodity futures, benchmark and signed)

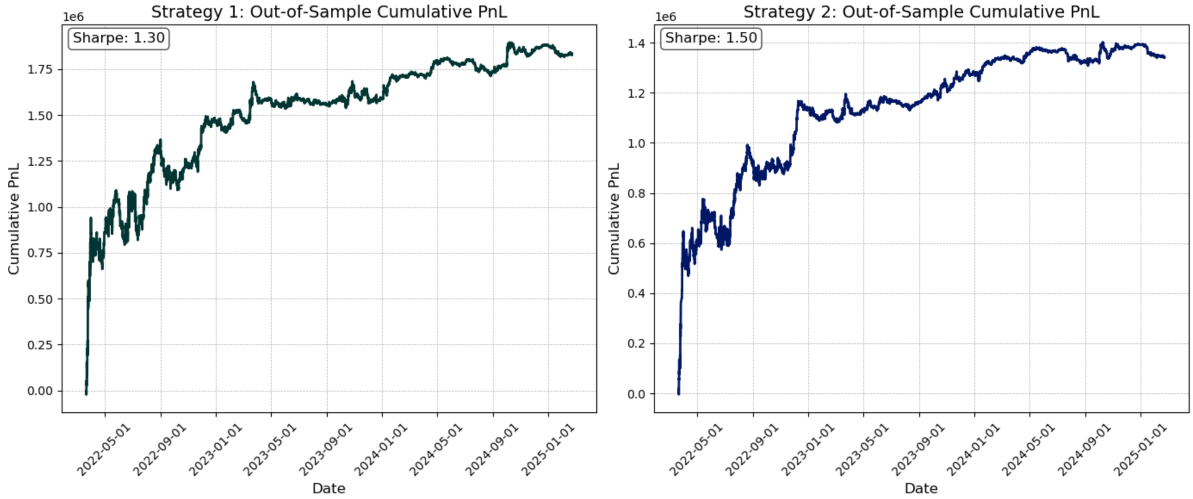

Figure 4.7: Out-of-sample cumulative PnL (commodity futures, benchmark and best)

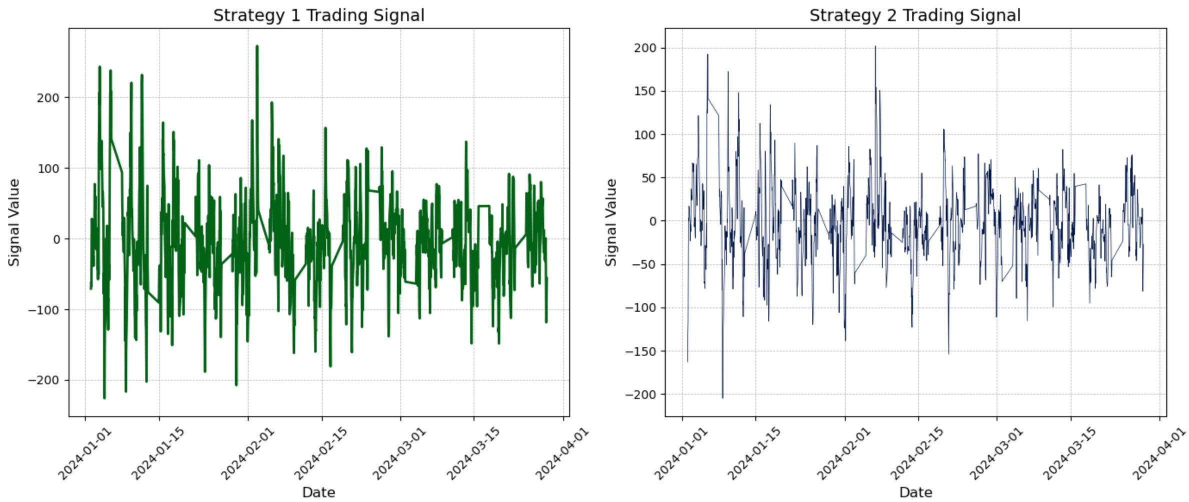

Figure 4.8: Signals of alphas (commodity futures, benchmark and best)

Table 4.17 presents the performance metrics of these three alphas. Since the second one applied the the sign operator, the portfolio positions are normalised and lead to smaller PnL as well as volatility. They have similar profit per trade, and the signed strategy has higher average holding days. However, due to higher oscillation (from Figure 4.6), it has lower auto-correlation than others. Other alphas still preserve an auto-correlation of 0.837 even though it is a mean-reversion strategy.

Table 4.17: Key performance metrics for trading strategies (commodity futures)

Strategy	Sharpe	Total PnL	Volatility	Max Drawdown	Turnover	Profit/Trade ($)	Avg Holding (days)	AC(1)
Benchmark	1.2969	1.8356e6	5648.4063	297026.4000	638621.3000	28.74	0.14	0.837
Signed	0.9653	13226.0000	40.9040	3406.0000	6076.0000	21.76	0.35	0.641
Best	1.4985	1.3421e6	3516.5784	202855.1000	432393.1000	31.03	0.15	0.837

4.4.3 Comparison Between Fitness Functions

From Table 4.18, one can see GP results for different fitness functions. For Sharpe without LP, the GP again experiences length explosion problem, where for one seed the best program length reached 31 without good testing statistics. The testing metrics for pure Shape based methods are not satisfactory (around 0), while the multi-objective fitness function successfully proposes potential alphas with an average Sharpe of 0.6103, and highest Sharpe of 1.2822.

Table 4.18: Summary statistics for different fitness functions (commodity futures)

Method	Avg Length	Avg Training Score	Avg Testing Score	Avg Sharpe Ratio	Best Sharpe Ratio
Sharpe (Without LP)	13.4	1.4725	-0.0394	-0.0394	0.5197
Sharpe (With LP)	6.6	1.0475	0.03258	0.03258	0.5653
Multi-objective	8.20	0.9366	-4.3611	0.6103	1.2822

4.4.4 Large Scale Experiments

Again, large scale experiments are conducted to vindicate the validity of the results from GP. For 50 different random seeds, top 10 outputs are chosen based on the validation ranks. It is observed that all top alphas are single factor mean-reversion alpha, and the results after fine-tuning is presented in Table 4.19. It confirms that the most stable alphas composed solely from futures prices are of this mean-reverting structure.

Table 4.19: Test performance of best GP programs (futures, large scale)

Seed	Program	Rank	Fitness_test	Sharpe_test
2, 41, 190, 174, 155, ...	`lag70((X1))`	1	1.7511	1.5014

4.5 Alphas from Port Activity Data

Finally, we exploit the port activity data for alpha generation, adhering to the experiment procedures above. The feature engineering is conducted on the IPCA projections using the clustered port activity data (Table 4.20), as described in section 3.3.3. The GP experiments are first discussed, then fine-tuning and out-of-sample testing metrics are reported.

Table 4.20: Input features for GP on port activities (IPCA)

Feature name	Variable name	Expression
Principal Component Score 1	X0	$X_{\text{IPCA}}^{cluster}[:, 0]$
Principal Component Score 2	X1	$X_{\text{IPCA}}^{cluster}[:, 1]$
...	...	...
Principal Component Score 19	X18	$X_{\text{IPCA}}^{cluster}[:, 18]$
Principal Component Score 20	X19	$X_{\text{IPCA}}^{cluster}[:, 19]$

Table 4.21: Training and validation results for GP on port exports (multi-objective fitness function)

Gen	Best program	Train	Validation
Seed 9			
0	`add(mul(abs(X4), neg(X2)), neg(X6))`	1.6182	-1.5765
1	`add(neg(X2), lag60(add(lag60(X0), X2)))`	2.3916	-2.7199
2	`sub(lag60(add(lag60(X0), X2)), lag1(X2))`	2.7361	-2.3698
3	`lag1(add(neg(X2), lag60(add(lag60(X0), X2))))`	2.9120	-2.5308
4	`sub(sub(lag60(lag60(X0)), lag1(X8)), add(X2, X3))`	2.9319	-1.1109
5	`lag1(sub(sub(lag60(add(lag60(X0), X2)), lag1(X8)), lag1(X2)))`	2.9698	-3.3400
6	`lag1(sub(sub(lag60(lag60(X0)), lag1(X8)), add(X2, X3)))`	3.3574	-0.9391
7	`lag1(sub(sub(lag60(lag60(X0)), lag1(X8)), add(X2, X3)))`	3.3574	-0.9391
8	`lag1(sub(sub(lag60(lag60(X0)), mean10(X8)), add(X2, X3)))`	3.3591	-0.8063
9	`lag1(sub(sub(lag60(lag60(X0)), lag10(X8)), add(X2, X3)))`	3.5264	-0.7025
Seed 8			
Gen	**Best program**	**Train**	**Validation**
0	`mean10(add(mul(X8, -0.799), div(X4, X3)))`	1.3227	-249.1235
1	`sign(mean50(sub(0.055, X2)))`	1.4030	-190.9394
2	`lag1(add(sub(X5, X2), lag1(neg(X2))))`	1.9060	-2.1389
3	`lag1(add(sub(neg(mean10(X8)), X2), lag1(neg(X2))))`	2.2743	-2.1187
4	`lag1(add(neg(X8), add(neg(X2), lag60(lag60(X0)))))`	3.0575	-1.0216
5	`sub(lag1(sub(add(neg(X8), lag60(lag60(X0))), X2)), X4)`	3.2259	-0.8930
6	`sub(lag1(add(sub(neg(X8), X2), lag60(lag60(X0)))), X4)`	3.2259	-0.8930
7	`sub(sub(lag1(add(neg(X2), lag60(lag60(X0)))), X8), X4)`	3.2600	-1.0143
8	`sub(sub(lag1(add(neg(X2), lag60(lag60(X0)))), X8), X4)`	3.2600	-1.0143
9	`sub(sub(lag1(add(neg(X2), lag60(lag60(X0)))), X8), X4)`	3.2600	-1.0143

4.5.1 In-sample Results

GP Experiments

A range of GP experiments are conducted for the clustered IPCA data of port activity. For the multi-objective fitness function, the coefficients are set as $(1, 0.0005, 0, 0)$. It is shown that the alphas generated from port activity data have much more diverse mathematical structures (an example is displayed in Table 4.21). However, valid alphas are harder to be obtained, even if the high-dimensional data is clustered and prominent principal components are selected. Also, export features tend to form better alphas than import features.

The potential alphas are selected based on the relative ranks of the validation fitness scores, given that the training fitness scores are in a good range, and alpha refinements are attempted on these potential alphas. It is observed that the alphas proposed by the GP are already optimised with respect to their parameters.

Testing Metrics of Alphas

The out-of-sample metrics of two selected alphas from the unseen testing data are presented in this section, where the cumulative PnL (Figure 4.9) and signals (Figure 4.10) are displayed. One can see they still have good Sharpe ratios and relatively good PnL curve, where the second strategy has a Sharpe ratio of 2.17.

Table 4.22 demonstrates some performance metrics of these alphas. Strategy 2 has a higher Sharpe ratio, PnL as well as volatility. Also, it has lower relative drawdown and higher profit per trade, making it a more robust strategy compared to Strategy 1.

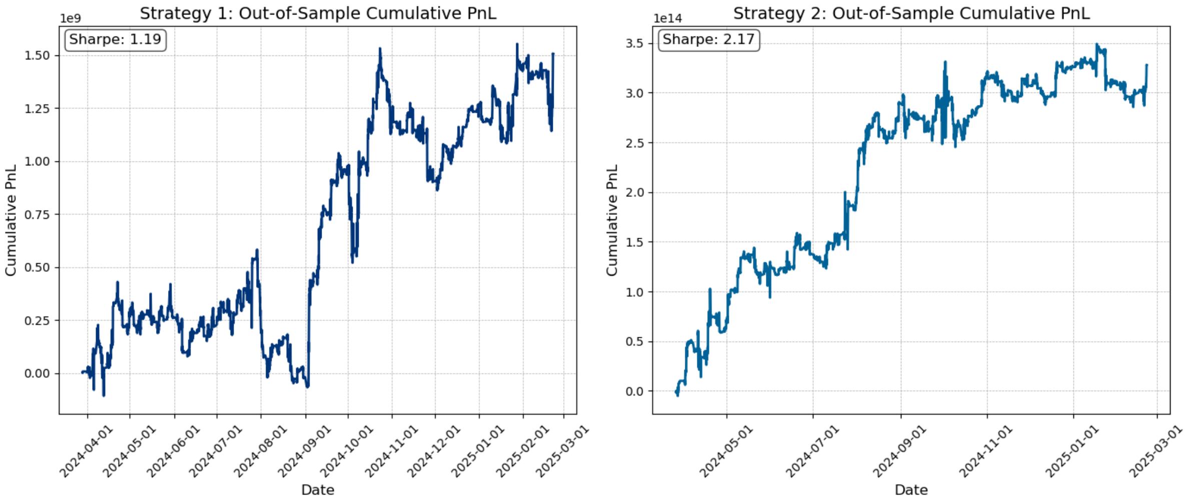

Figure 4.9: Out-of-sample cumulative PnL (clustered export IPCA)

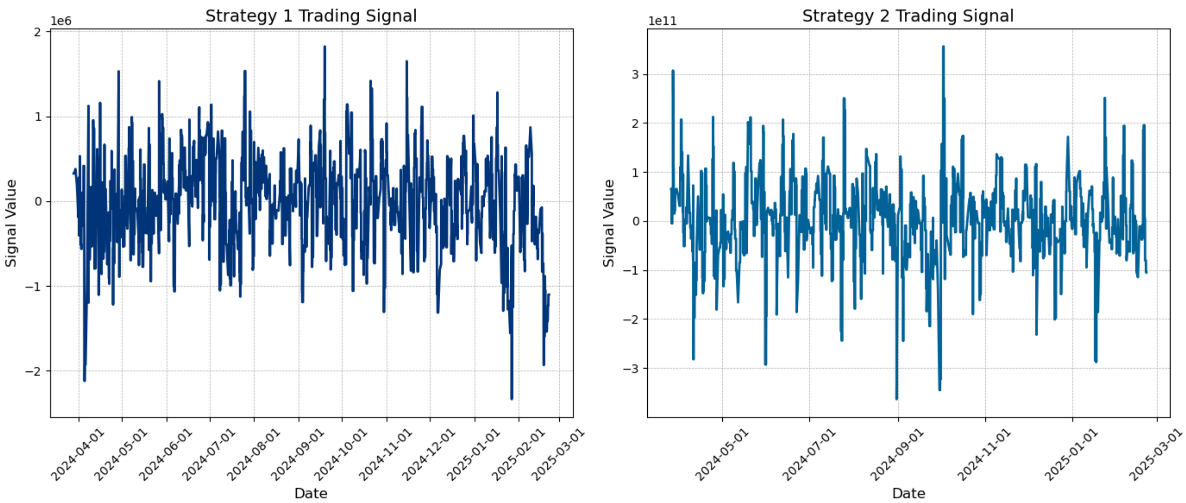

Figure 4.10: Signals of alphas (clustered export IPCA)

Table 4.22: Key performance metrics for trading strategies (clustered export IPCA)

Strategy	Sharpe	Total PnL	Volatility	Max Drawdown	Turnover	Profit/Trade ($)	Avg Holding (days)	AC(1)
Strategy 1	1.1850	1.5030e9	14760529	6.6989e8	342786949	43.87	0.61	0.934
Strategy 2	2.1732	3.2752e14	2.4724e12	8.6005e13	3.9969e13	81.86	0.78	0.939

4.6 Transaction Cost Analysis

To further ensure the validity of the alphas generated in the experiments, a transaction cost analysis will be conducted in this section for verifying the economic and production value of these strategies. The alphas investigated are the ones selected and reported in previous experiments.

In Table 4.23, the transaction costs related metric scores are displayed, including profit per trade, net Sharpe ratio (adjusted by the transaction cost), the ratio between total transaction cost and final profit, and the ratio between profit per trade and unit transaction cost are computed on the out-of-sample testing data. The transaction cost is assumed to be 1 tick per side per trade.

Table 4.23: Cost-adjusted performance metrics for commodity futures strategies

Strategy	Profit per trade	Net Sharpe	Cost/Profit ratio	Edge/Cost ratio
Weighted average 1	293.68	2.83	0.03	14.68
Weighted average 2	54.85	2.22	0.18	2.74
IPCA capacities 1	320.75	3.15	0.03	16.04
IPCA capacities 2	171.98	2.23	0.06	8.60
Price reversion 1	28.74	0.56	0.35	1.44
Price reversion 2	31.03	0.71	0.32	1.55
IPCA port 1	43.87	0.85	0.23	2.09
IPCA port 2	81.86	1.93	0.12	4.09

All selected strategies possess reasonable profit per trade for production, where IPCA capacities alphas and the first weighted average capacity alpha have the best scores (171.98 - 320.75). For these good performing ones, the net Sharpe ratios are still high, and the total transaction costs only contribute to a small proportion of the final PnL (3% to 6%). The port activity based alphas are moderate among them, with some reduction of Sharpe and 10% to 20% loss to final PnL. However, the price reversion alphas suffer the most from the transaction costs, since they are the most high frequency approach which needs to adjust positions more frequently. Nevertheless, by the assumptions it is still tradable, but more sensitive to adverse scenarios.

Table 4.24: Training and validation results for GP on IPCA capacities with transaction costs (multi-objective fitness function)

Seed 95			
Gen	**Best program**	**Train**	**Validation**
0	`mean20(abs(lag70(X1)))`	-0.1961	-304.4262
1	`lag70(neg(mean40(X4)))`	1.3054	-33.6086
2	`lag70(neg(mean40(X4)))`	1.3054	-33.6086
3	`mean20(mean20(sub(mean80(mean80(lag70(mean80(X8)))), lag70(X4))))`	1.5736	-8.5599
4	`mean20(mean20(sub(mean80(mean80(lag70(mean80(X8)))), lag70(X4))))`	1.5736	-8.5599
5	`mean20(mean20(sub(lag70(mean80(lag70(mean80(X8)))), lag70(X4))))`	1.5791	-8.0355
6	`mean20(mean20(sub(mean80(mean1(lag70(X6))), lag70(X4))))`	1.6398	-3.7084
7	`mean20(mean20(sub(mean80(mean1(lag70(X6))), lag70(X4))))`	1.6398	-3.7084
8	`mean20(mean20(sub(mean80(mean1(lag70(X6))), lag70(X4))))`	1.6398	-3.7084
9	`mean20(mean20(sub(mean80(mean1(lag70(X6))), lag70(X4))))`	1.6398	-3.7084

Attempts are also made to include transaction costs into the multi-objective fitness function, but experiments show that the quality of alphas discovered by the GP will not increase for this addition. There are some similar results (Table 4.24), while other seeds might output worse alphas. The reasons may come from the fact that very good alphas (non high frequency) are already robust to transaction costs (high profit per trade and more stable holding periods), and the relative high-frequency but profitable alphas will be pruned in the GP process. Also, it contributes to even faster overfitting.

Chapter 5

Discussion

From this chapter, a detailed discussion on the results presented in Chapter 4 will be made, along with an evaluation of the methodologies of this thesis.

5.1 GP Experiments

Overall, the GP experiments could generate valid potential alphas from commodities futures data and IMF PortWatch data. Different data could induce GP to discover alphas with different mathematical structures. Four broad types of alphas are investigated. Firstly, mean chokepoint capacities are used as input features for GP. This data reflects the active trading volumes at the crucial chokepoints around the world. It is shown that the mean capacities exhibit momentum properties as trading factors, such as the lagged momentum factor and rolling mean of the momentum factor. These selected alphas could be further fine-tuned via the multi-objective function (in-sample), and a number of alphas have out-of-sample Sharpe ratios above 2.5 (e.g. 2.92, 2.67, Table 4.6).

Secondly, IPCA is applied to chokepoint capacities, and GP gives more diverse alpha structures, from transformed single data on principal components, to alpha momentum factors on IPCA features. After in-sample selection and fine-tuning, the out-of-sample results are even better than the first type, where the highest Sharpe Ratio is above 3 (3.24, Table 4.11).

These GP results vindicate the existence of momentum effects on chokepoint capacity data, and show that IPCA is instrumental for obtaining better insights. IPCA could guide the alphas to focus on the most essential directions of the data, which are more likely to have an explicit influence on price. Also, they are robust to transaction costs since they are mid-frequency strategies (holding periods more than a day) which have good profit per trade (203.68 for weighted average, 320.75 for IPCA).

From another perspective, efforts are made to discover alphas from the port activity data, which measures the import and export volumes of trades for ports globally. The mathematical structures to form alphas based on this data are more complex, and dimension reduction has to be applied to gather useful information in the vast dataset (more than 1700 dimensions). Concomitantly, it is shown that finding valid alphas are much more difficult than chokepoint capacities, not only since the explained variance is low due to the high dimension of the data, but also the probability that activities of some particular ports would reflect the price movement on an international exchange is relatively low. Therefore, one needs to select fewer top performing alphas in-sample to ensure the alpha is valid out-of-sample, and the risk of overfitting is larger.

Nevertheless, some trading strategies are developed using port activities, and achieving a relatively good out-of-sample Sharpe ratios of 2.17 and 1.19 (Table 4.22). These alphas have more complicated mathematical relationships involving multiple (3 to 4) IPCA features, with higher frequencies (average holding period less than a day). Also, the cumulative PnL curve is more volatile (Figure 4.9) than the ones before.

Finally, GPs are conducted solely on prices for alpha generation. From the large scale experiments (section 4.4.4), most good performing seeds converge to single (double) factor mean-reversion strategies. The reason may comes from the fact that the frequency of price data is hour, so that the GP tends to find shorter time span strategies, where reversion structures come into the picture. From Table 4.17, these alphas have the highest frequencies (average holding periods around 0.14-0.35) and the lowest profit per trade (21.76-31.03), due to their reversion nature to exploit the short-term price fluctuations. They have moderate out-of-sample Sharpe ratios of 0.97-1.50, and are more sensitive to transaction costs.

From the experiments, it can be argued that GP results rely on the quality of the input features to a large extent. When the data is fundamentally related to price movements, this GP framework is robust and are able to generate valid outputs. However, if the input features are unable to form very strong trading signals in the first place, no matter how one tunes them via GP, the probability of obtaining a group of valid alphas is low. In those cases, one needs to select fewer top performing alphas in-sample, or find more insightful alternative data. This is also the reason that after data exploration, this thesis focuses on oil data since it heavily relies on maritime trade. Some attempts are made on natural gas as well, but since there is a substantial amount of transportation conducted by pipelines [50], the results are not as satisfactory as oil.

5.2 Feature Engineering Analysis

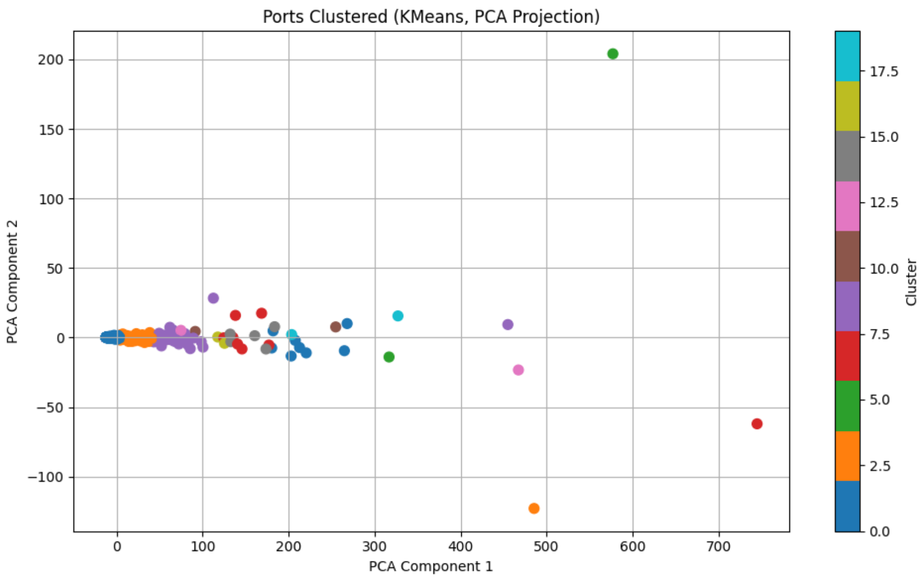

Figure 5.1: Visualisation of clustered export IPCA data

A range of feature engineering methods are utilised in this thesis to provide good input features for Genetic Programming. It is observed that setting semi-processed structures as inputs for GPs are useful for converging to valid alpha structures, such as momentum (chokepoint capacities), mean-reversion (prices) and other common types of factors. In many cases, if one already has legitimate ideas about the market, the consequent factors could better guide GP than the raw data.

Furthermore, dimension reduction such as PCA is a common approach for exploring data in commodity markets [51] [52] [53]. We applied a time-aware (causal) variant of PCA, IPCA, to obtain a more robust and numerically accurate representation of the data on top principal components (section 3.3.2). For 24 chokepoints, experiments show that IPCA could obtain more insightful factors on capacities.

However, for very high dimension such as the port activity data, further summary operations need to be conducted for compressing the large amount of information. Hence, a clustering algorithm is first applied, followed by IPCA on the clustered data. The port data could be clustered to a certain extent based on the characteristics of their trade flow via principal components, as displayed in Figure 5.1. Although mining alphas from this substantial data is relatively difficult (sparse search space and prone to overfitting), one can still find valid ones with care (section 4.5).

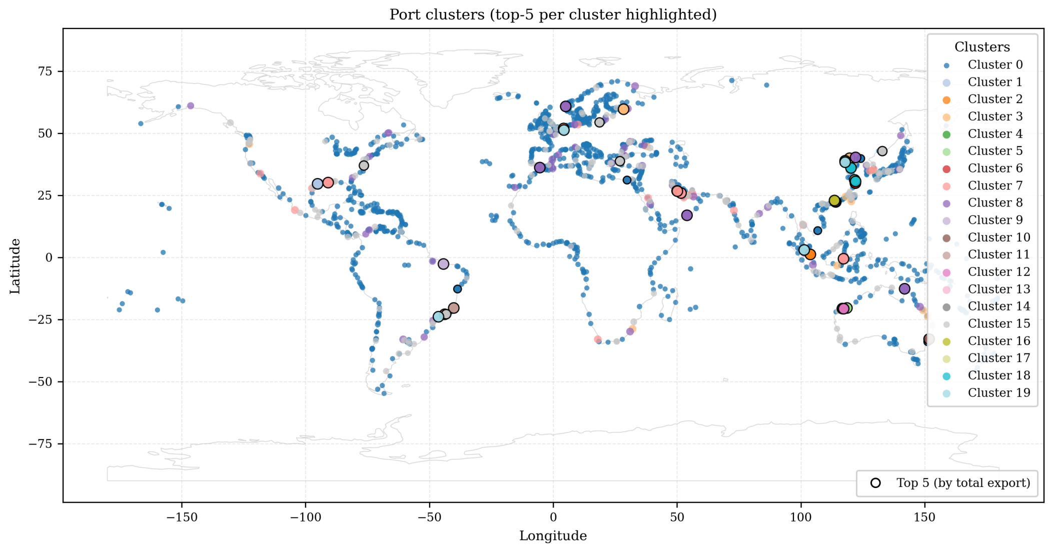

Figure 5.2: Visualisation of clustered ports on export

For the phenomenon that export port activities are more instrumental than import activities for alpha generation, one can see from Figure 5.2 and Appendix A.1 that the export ports are more related to the sources of commodity production (such as the Middle East and Australia), whereas import major ports are mostly in China. In this sense, it could be inferred that price movements are more likely to be predicted by the supplier side than the consumer side for Brent oil.

5.3 Multi-objective Fitness Function and Alpha Selection

In section 3.6, we proposed a novel multi-objective fitness function in GP, which balances the capability of making profits (Sharpe ratio) as well as strategy stability (drawdown statistics). More terms could be utilised in fine-tuning, including PnL and regression slope coefficient. It tries to contribute to finding more robust mathematical expressions in GP, and further boosting performance for fine-tuning.

The results from GP experiments (Table 4.7, Table 4.12 and Table 4.18) show that this multi-objective fitness function could alleviate the length explosion and overfitting problem in GP to a large extent, and it has better out-of-sample performance metric scores. For the cases where alphas are relatively easy to locate (such as chokepoint capacities), the Sharpe ratio based methods could discover valid alphas (but with a lower efficiency), while for more sensitive data which is easy for overfitting and harder to locate alphas, the Sharpe based methods lost efficacy to a certain extent, while the multi-objective method still takes effects (such as commodity futures, average out-of-sample Sharpe ratios for Sharpe (with LP) and Multi-objective (with LP) are 0.57 and 1.28 respectively).

Therefore, it can be deduced that by encouraging alphas which maximise the profits and control drawdown, GP could output results which are more likely to be valid out-of-sample, and it is particularly useful in scenarios where the data is easy to be overfit. Also, it could facilitate alpha refinement which further exploits the potentials of the selected alphas.

One also attempts to use other statistics for constructing the multi-objective fitness function for GP, such as transaction costs, volatility or adding final PnL, but experiments show that either the optimisation dynamics are similar, or the GP will be encouraged to output some ill-posed strategies which are not desired. For example, final PnL could encourage GP to output alphas which multiply (or scale up) the features, which may boost PnL but increases risks at a large scale. Adding transaction costs may output similar formulas for small penalty coefficients, and will miss some profitable opportunities. Hence, it is more sensible to find strategies first, and then verify by inspecting transaction implications.

As for alpha selection, we used the relative rank for validation fitness scores in GPs for selecting the potential valid alphas, as well as examining the quality of training fitness scores. Large scale experiments confirm the efficacy of this approach (section 4.3.3 and section 4.4.4), that for setting a large number of random seeds and selecting the top 20%, the alphas selected show good out-of-sample metrics (average Sharpe ratio 2.05 and 1.50).

5.4 Economic Interpretations and Production Feasibility

Finally, we discuss economic interpretations of some alphas selected, and the feasibility for applying them into production. From Figure 3.2, one can already see a lagged cross correlation between the chokepoint capacities and oil prices. The alpha generation procedures also indicate the presence of momentum effects of chokepoint capacities on price, either by weighted average or IPCA. The economic reasons could arise from the fact that chokepoint trade volumes (capacities) reflect the active supply-demand dynamics of commodities, and indicate future price movements. For instance, an uprising momentum (trend) for transport at some chokepoints signals a larger than average demand cycle, and the price would take this momentum as well to keep rising (an explicit example is the revival of industrial activities after the first COVID peak).

For port activities, as discussed above, strategies could be developed by inspecting the IPCA features of trade flows of certain ports around the world. A certain set of top ports for export could serve as valid factors for trading, since their activities are closely linked with commodity supply. Also, all alphas from IMF PortWatch data convey the idea that

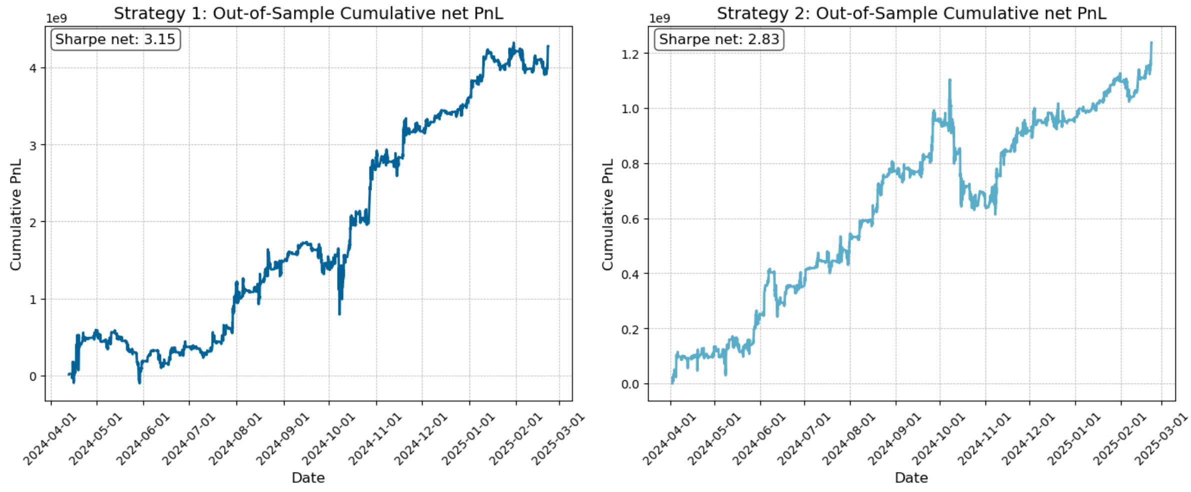

Figure 5.3: Out-of-sample cumulative PnL (net with transaction costs)

choices of subjects matter. IPCA or weighted average on a chosen subset could give better economic insights than the global summary statistics, that the trading activities at certain chokepoints and ports have a more significant impact on commodities prices. There are immature alphas which could capture port specific and event-based opportunities, such as the substantial increment in price at the beginning of 2022.

We also analysed the production feasibility by computing out-of-sample testing statistics and inspecting transaction cost implications (section 4.6). The alphas using chokepoint capacities data have the highest quality, with large profit per trade, robust net Sharpe ratio (Figure 5.3), and high Edge/Cost ratio. Therefore, they are robust with transaction costs added, and are feasible for production needs. The port activities alphas are in the middle range, with moderate profit per trade and shorter holding periods. In this way, some are still robust to transaction costs, such as IPCA port 2 in Table 4.23. The price reversion alphas, however, are more sensitive to transaction costs due to their high frequency and the nature of exploiting short-term mispricings. Therefore, it needs favorable trading conditions (for example, low market impact and trading fees) to make it suitable for production.

Chapter 6

Conclusion

To conclude, this thesis investigates alpha generation in commodity markets using maritime trade data, proposing a systematic framework which consists of alpha formulation and selection by Genetic Programming, alpha fine-tuning and out-of-sample evaluation. It elaborates the mathematical and economical pricing theories for commodity futures, and summaries traditional and emerging methods for alpha research in commodities. Also, it explores a novel dataset, IMF PortWatch, including port trade activities, chokepoint trade estimates and disruptions, and experiments a new multi-objective fitness function in the Genetic Programming procedure.

The alpha generation framework constructed in this thesis could discover a range of valid alphas from IMF PortWatch data as well as commodity futures data. Good structures include momentum alphas on IPCA and weighted average of chokepoint capacities, reversion alphas on futures prices, as well as alphas using port activities data. Some top performing alphas achieve out-of-sample testing annualised Sharpe ratios above 2.5 (highest above 3), with good profit per trade and autocorrelation statistics. The multi-objective fitness function facilitates Genetic Programming by improving consistency of profits and reducing overfitting, compared to other fitness functions.

For future research directions, research could be made to develop alphas using other commodities such as metal and agricultural products. More types of mathematical operators could be used for Genetic Programming on commodity futures prices, such as more complicated time series or cross sector operators. The proposed framework of alpha generation could be formalised into more structured computer program for large scale research and production of alphas.

All in all, this thesis provides insights in developing alphas using maritime trade data, and proposes new methods for generating alphas with Genetic Programming. It is the very hope of this thesis that this work could be helpful not only for investors, but also for a wider range of stakeholders for establishing a more robust, prosperous and sustainable market for commodities, leaving a legacy for future generations.

Appendix A

Data Specifications

A.1 Code

The code of this thesis is stored in a private GitHub repository, please email zhengyang.li21@imperial.ac.uk for access of the repository. The Genetic Programming utilities are adapted from the gplearn package [49]. Since some experiments require advance computing resources to complete, the code displayed are demos of the core results, and the large scale experiment results are summarised as tables (available for display in Jupyter notebooks).

A.2 IMF PortWatch Data

Table A.1: Port activity dataset: columns, types, and meanings

Column	Dtype	Meaning / Example (for classification fields)
date	object	Reporting date-time (UTC) for the record. *Example:* `2019-02-20 00:00:00+00:00`.
year	int64	Calendar year extracted from `date`. *Example:* `2019`.
month	int64	Calendar month (1–12) extracted from `date`. *Example:* `2`.
day	int64	Day of month (1–31) extracted from `date`. *Example:* `20`.
portid	object	PortWatch unique port identifier. *Example:* `port1325`.
portname	object	Official port name. *Example:* `Tsuruga`.
country	object	Country of the port. *Example:* `Japan`.
ISO3	object	ISO 3166-1 alpha-3 country code. *Example:* `JPN`.
portcalls_container	int64	Number of container-vessel calls at the port on the given day.
portcalls_dry_bulk	int64	Number of dry-bulk vessel calls that day.
portcalls_general_cargo	int64	Number of general-cargo vessel calls that day.
portcalls_roro	int64	Number of roll-on/roll-off (RoRo) vessel calls that day.
portcalls_tanker	int64	Number of tanker vessel calls that day.
portcalls_cargo	int64	Number of cargo-vessel calls (aggregate across cargo types).
portcalls	int64	Total vessel calls at the port that day (all included types).
import_container	float64	Estimated imported cargo via container vessels (daily).
import_dry_bulk	float64	Estimated imported cargo via dry-bulk vessels (daily).
import_general_cargo	float64	Estimated imported cargo via general-cargo vessels (daily).
import_roro	float64	Estimated imported cargo via RoRo vessels (daily).
import_tanker	float64	Estimated imported cargo via tanker vessels (daily).

Continued on next page

Column	Dtype	Meaning / Example (for classification fields)
import_cargo	float64	Total imported cargo across cargo vessel types (daily).
import	float64	Overall imports for the port-day (all relevant vessel categories).
export_container	float64	Estimated exported cargo via container vessels (daily).
export_dry_bulk	float64	Estimated exported cargo via dry-bulk vessels (daily).
export_general_cargo	float64	Estimated exported cargo via general-cargo vessels (daily).
export_roro	float64	Estimated exported cargo via RoRo vessels (daily).
export_tanker	float64	Estimated exported cargo via tanker vessels (daily).
export_cargo	float64	Total exported cargo across cargo vessel types (daily).
export	float64	Overall exports for the port-day (all relevant vessel categories).
ObjectId	int64	Row identifier / surrogate key for the record.

Table A.2: Chokepoint trade dataset: columns, types, and meanings

Column	Dtype	Meaning / Example (for classification fields)
date	object	Reporting date-time (UTC) for the chokepoint-day record. *Example:* `2019-04-11 00:00:00+00:00`.
year	int64	Calendar year extracted from `date`. *Example:* `2019`.
month	int64	Calendar month (1–12) extracted from `date`. *Example:* `4`.
day	int64	Day of month (1–31) extracted from `date`. *Example:* `11`.
portid	object	Chokepoint identifier. *Example:* `chokepoint1`.
portname	object	Chokepoint name. *Example:* `Suez Canal` (or `Dover Strait`).
n_container	int64	Number of container-vessel transits through the chokepoint on the given day.
n_dry_bulk	int64	Number of dry-bulk vessel transits that day.
n_general_cargo	int64	Number of general-cargo vessel transits that day.
n_roro	int64	Number of roll-on/roll-off (RoRo) vessel transits that day.
n_tanker	int64	Number of tanker vessel transits that day.
n_cargo	int64	Total cargo-vessel transits (aggregate across cargo categories) that day.
n_total	int64	Total vessel transits across all included types that day (cargo plus other categories, if any).
capacity_container	float64	Aggregated carrying capacity of container vessels transiting that day (sum across ships).
capacity_dry_bulk	float64	Aggregated carrying capacity of dry-bulk vessels transiting that day.
capacity_general_cargo	float64	Aggregated carrying capacity of general-cargo vessels transiting that day.
capacity_roro	float64	Aggregated carrying capacity of RoRo vessels transiting that day.
capacity_tanker	float64	Aggregated carrying capacity of tanker vessels transiting that day.
capacity_cargo	float64	Total carrying capacity across cargo vessel categories (daily).
capacity	float64	Overall carrying capacity across all vessel types for the chokepoint-day.
ObjectId	int64	Row identifier / surrogate key for the record.

Table A.3: Chokepoint information: selected columns, types, and meanings

Column	Dtype	Meaning / Example
portid	object	Unique chokepoint identifier. *Example:* `chokepoint1`.
portname	object	Chokepoint name. *Example:* `Suez Canal`.
lat	float64	Latitude (decimal degrees, North positive). *Example:* `30.593346`.
lon	float64	Longitude (decimal degrees, East positive). *Example:* `32.436882`.
industry_top1	object	Most prevalent traded industry group associated with the chokepoint. *Example:* `Mineral Products`.
industry_top2	object	Second most prevalent industry group. *Example:* `Vegetable Products`.
industry_top3	object	Third most prevalent industry group. *Example:* `Chemical & Allied Industries`.

Table A.4: Chokepoint information

portid	portname	lat	Lon	industry_top1	industry_top2	industry_top3
1	Suez Canal	30.593	32.437	Mineral Products	Vegetable Products	Chemical & Allied Industries
2	Panama Canal	9.121	-79.767	Mineral Products	Vegetable Products	Chemical & Allied Industries
3	Bosporus Strait	41.169	29.092	Mineral Products	Vegetable Products	Chemical & Allied Industries
4	Bab el-Mandeb Strait	12.789	43.35	Mineral Products	Chemical & Allied Industries	Vegetable Products
5	Malacca Strait	1.517	102.665	Mineral Products	Chemical & Allied Industries	Vegetable Products
6	Strait of Hormuz	26.297	56.86	Mineral Products	Chemical & Allied Industries	Vegetable Products
7	Cape of Good Hope	-34.927	20.883	Mineral Products	Vegetable Products	Prepared Foodstuffs & Beverages
8	Gibraltar Strait	35.942	-5.755	Mineral Products	Vegetable Products	Chemical & Allied Industries
9	Dover Strait	51.03	1.506	Mineral Products	Chemical & Allied Industries	Vegetable Products
10	Oresund Strait	55.508	12.851	Mineral Products	Wood & Wood Products	Chemical & Allied Industries
11	Taiwan Strait	24.724	119.831	Mineral Products	Vegetable Products	Chemical & Allied Industries
12	Korea Strait	34.131	129.209	Mineral Products	Vegetable Products	Metals
13	Tsugaru Strait	41.328	140.353	Mineral Products	Vegetable Products	Metals
14	Luzon Strait	20.489	121.352	Mineral Products	Vegetable Products	Wood & Wood Products
15	Lombok Strait	-8.419	115.801	Mineral Products	Vegetable Products	Chemical & Allied Industries
16	Ombai Strait	-8.399	125.091	Mineral Products	Chemical & Allied Industries	Prepared Foodstuffs & Beverages

Continued on next page

portid	portname	lat	Lon	industry_top1	industry_top2	industry_top3
17	Bohai Strait	38.373	120.9	Mineral Products	Metals	Vegetable Products
18	Torres Strait	-9.863	142.248	Mineral Products	Vegetable Products	Chemical & Allied Industries
19	Sunda Strait	-5.967	105.775	Mineral Products	Vegetable Products	Prepared Foodstuffs & Beverages
20	Makassar Strait	0.352	119.257	Mineral Products	Vegetable Products	Prepared Foodstuffs & Beverages
21	Magellan Strait	-52.64	-69.595	Mineral Products	Vegetable Products	Prepared Foodstuffs & Beverages
22	Yucatan Channel	21.815	-85.647	Mineral Products	Vegetable Products	Chemical & Allied Industries
23	Windward Passage	19.986	-73.698	Mineral Products	Vegetable Products	Wood & Wood Products
24	Mona Passage	18.449	-67.711	Mineral Products	Vegetable Products	Chemical & Allied Industries

Table A.5: Disruption events dataset: columns, types, and meanings

Column	Dtype	Meaning / Example
eventid	int64	Unique event identifier. *Example:* `1000552`.
eventtype	object	Event type code (e.g., `TC`=Tropical Cyclone, `EQ`=Earthquake, `FL`=Flood).
eventname	object	Short code name of the event. *Example:* `KENNETH-19`.
htmlname	object	Human-readable event title for UI. *Example:* `Tropical Cyclone KENNETH-19`.
htmldescription	object	HTML-ready description of the event, locations, and dates.
alertlevel	object	Alert level category. *Example:* `RED`.
country	object	Country or countries affected (semicolon-separated). *Example:* `Mozambique; Comoros`.
fromdate	object	Event start date/time (string). *Example:* `4/23/2019 12:00:00 AM`.
year	int64	Calendar year associated with the event. *Example:* `2019`.
todate	object	Event end date/time (string). *Example:* `4/25/2019 6:00:00 PM`.
severitytext	object	Textual severity description (if present).
lat	float64	Latitude of reference point (decimal degrees). *Example:* `-19.6`.
long	float64	Longitude of reference point (decimal degrees). *Example:* `34.8`.
editdate	object	Last edit timestamp (string). *Example:* `9/10/2023 1:47:19 AM`.
affectedports	object	Semicolon-separated port IDs affected by the event. *Example:* `port137; port784`.
n_affectedports	int64	Count of affected ports. *Example:* `3`.
affectedpopulation	object	Narrative estimate of affected population. *Example:* `1.9 million in Category 1 or higher`.
pageid	object	Page/document identifier (GUID-like).

Continued on next page

Column	Dtype	Meaning / Example
ObjectId	int64	Row identifier / surrogate key.
Shape__Area	float64	Geometry area of the event polygon (dataset units).
Shape__Length	float64	Geometry perimeter/length of the event polygon (dataset units).

Table A.6: Top-5 ports by cluster ranked by total import capacity

Cluster	Top-5 ports
0	Dar Es Salaam, Yokkaichi, Baltimore, Dublin, Geelong
1	Lanqiao Port, Los Angeles-Long Beach, Antwerp, Port Klang, Jebel Ali
2	Tianjin Xin Gang
3	Cartagena, Lumut, Saigon, Bremerhaven, Fos
4	Guangzhou
5	Ningbo
6	Tangshan (Jingtang), Dongjiakou, Zhanjiang, New York-New Jersey, Gwangyang (Kwangyang)
7	Singapore
8	Qingdao
9	Zhoushan, Bayuquan, Manzanillo, Houston, Ulsan
10	Dalian
11	Yangshan (Shangai)
12	Rotterdam
13	Zhouliwang
14	Xiamen
15	Taicang (Suzhou), Jiangyin, Nantong, Qiwei
16	Hamburg
17	Bohai Bay
18	Shougang Jingtang
19	Shanghai

Table A.7: Top-5 ports by cluster ranked by total export capacity

Cluster	Top-5 ports
0	El Dekheila, Port Botany, Cat Lai (Saigon New Port), Dandong, Madre De Deus
1	Houston
2	Singapore
3	Rotterdam, Shougang Jingtang, Hong Kong, Ust-Luga, Qinhuangdao
4	Juaymah
5	Port Hedland
6	Ningbo
7	Samarinda, Ras Laffan, Porto de Itaguai, South Louisiana, Ras Tanura
8	Mongstad, Algeciras, Bayuquan, Weipa, Salalah
9	Itaqui
10	Dampier, Newcastle
11	Tubarao
12	Port Walcott
13	Shekou (Shenzhen)
14	Tianjin Xin Gang
15	Rio De Janeiro, Newport News, Nemrut Bay, Nakhodka, Gdansk
16	Guangzhou
17	Shanghai
18	Qingdao, Yangshan (Shangai)
19	Antwerp, Bohai Bay, Port Klang, Santos

A.3 Supplementary Plots

In this section, the supplementary plots of this thesis are displayed.

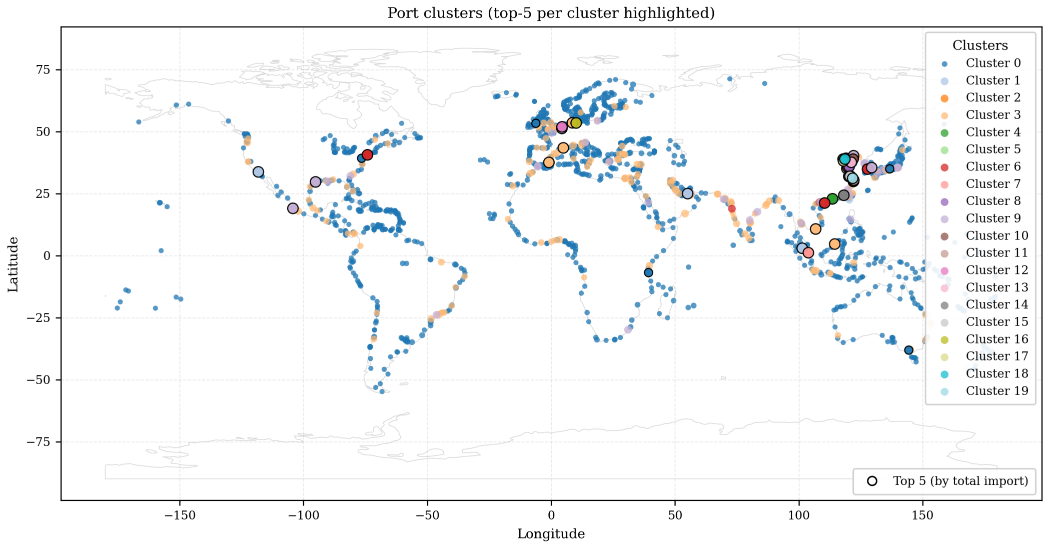

Figure A.1: Visualisation of clustered ports on import

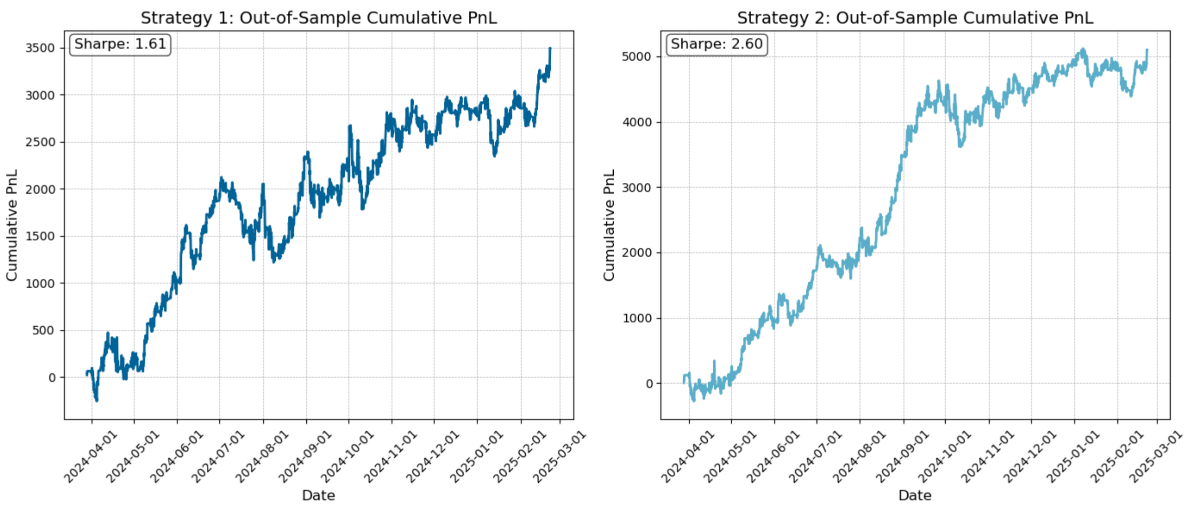

Figure A.2: Out-of-sample cumulative PnL (weighted) average of chokepoint capacities, sign)

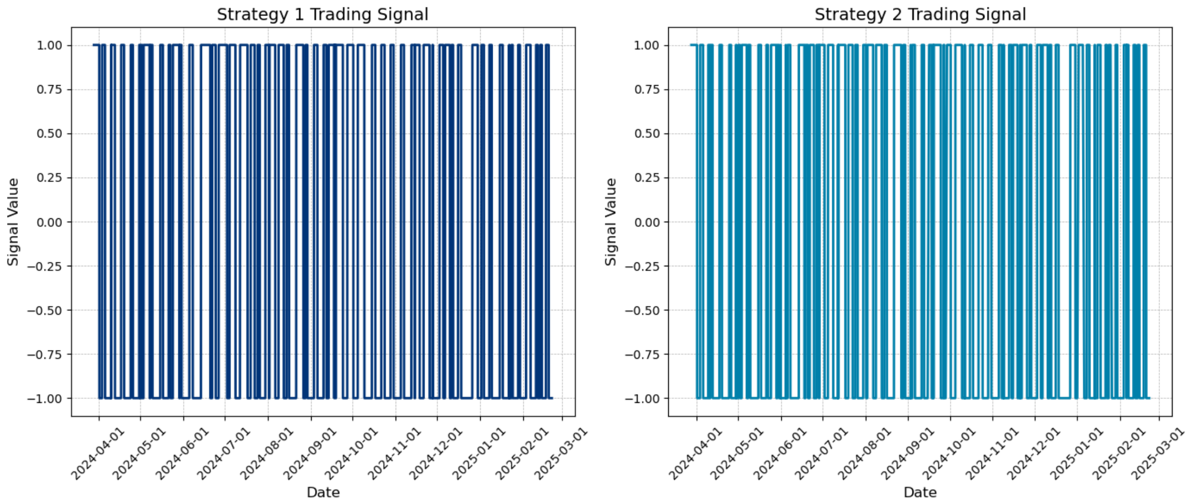

Figure A.3: Signals of alphas (weighted) average of chokepoint capacities, sign)

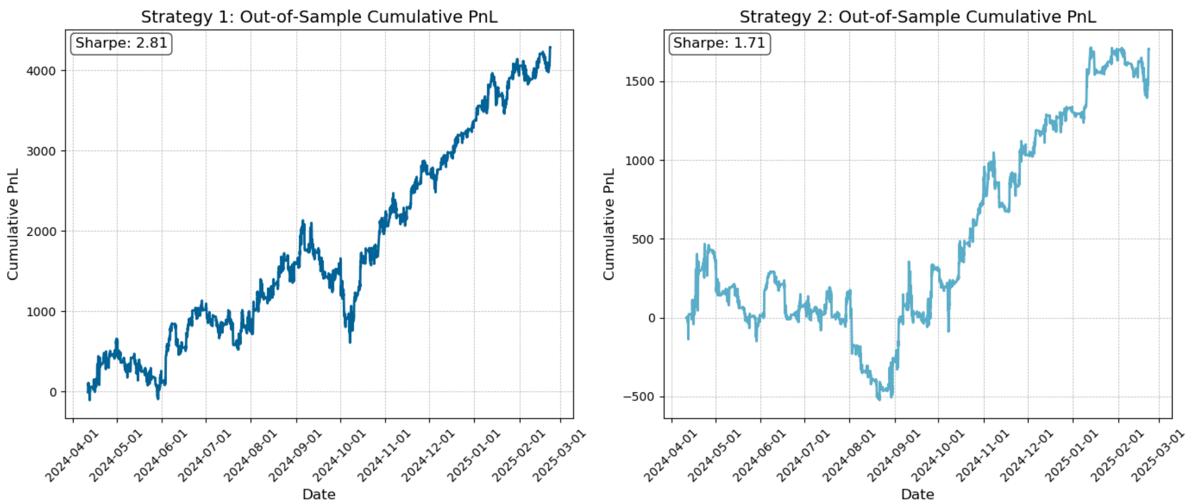

Figure A.4: Out-of-sample cumulative PnL (chokepoint capacities IPCA, sign)

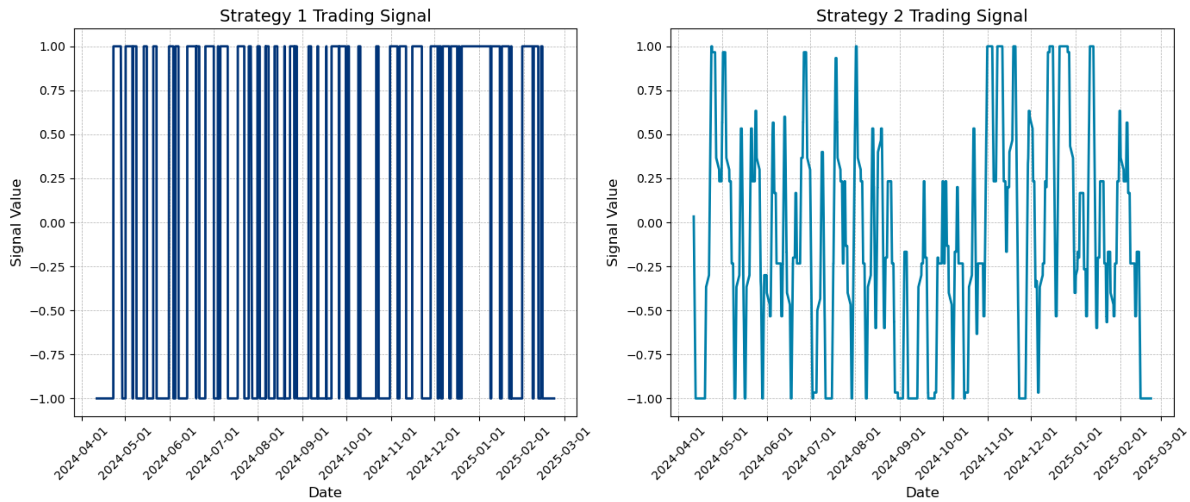

Figure A.5: Signals of alphas (chokepoint capacities IPCA, sign)

Bibliography

[1] Georgios Rallis. *On commodity trading strategies: momentum, term structure, maturity, indexation*. PhD thesis, City, University of London, 2010.

[2] CME Group Education. Introduction of standardized futures in the u.s. Online educational article, 2025. CBOT introduced standardized futures contracts in 1865.

[3] J.P. Morgan Research. Oil price forecast, 2025. Discusses the implications of inflation and supply on oil prices.

[4] Morgan Stanley Investment Management. 2025 commodity outlook: Three areas to watch, February 2025. Discusses trends in inflation, the U.S. dollar, and physical market supply for assessing the 2025 commodity investment environment.

[5] Viktor Manahov. The rise of the machines in commodities markets: new evidence obtained using strongly typed genetic programming. *Annals of Operations Research*, 260(1):321–352, 2018.

[6] Tereza Malátková and Mikuláš Gangur. Analysis of the use of artificial intelligence and machine learning in commodity markets'. In *Proceedings of the 9th International Scientific Conference Business Trends*, page 1113134, 2024.

[7] Weizhe Ren, Yichen Qin, and Yang Li. Alpha mining and enhancing via warm start genetic programming for quantitative investment. *arXiv preprint arXiv:2412.00896*, 2024.

[8] Hao Shi, Weili Song, Xinting Zhang, Jiahe Shi, Cuicui Luo, Xiang Ao, Hamid Arian, and Luis Angel Seco. Alphaforge: A framework to mine and dynamically combine formulaic alpha factors. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 12524–12532, 2025.

[9] Feng Xu, Yan Yin, Xinyu Zhang, Tianyuan Liu, Shengyi Jiang, and Zongzhang Zhang. Alpha2: Discovering logical formulaic alphas using deep reinforcement learning. *CoRR*, 2024.

[10] Yang Liu, Guofu Zhou, and Yingzi Zhu. Maximizing the sharpe ratio: A genetic programming approach. *SSRN Electronic Journal. https://doi. org/10.2139/ssrn*, 3726609, 2020.

[11] International Monetary Fund PortWatch. Portwatch: Monitoring trade disruptions from space, 2025. Open platform providing real-time indicators and simulations of maritime trade disruptions.

[12] Frederick LA Grauer and Robert H Litzenberger. The pricing of commodity futures contracts, nominal bonds and other risky assets under commodity price uncertainty. *Journal of Finance*, pages 69–83, 1979.

[13] Gary B Gorton, Fumio Hayashi, and K Geert Rouwenhorst. The fundamentals of commodity futures returns. *Review of Finance*, 17(1):35–105, 2013.

[14] Lester G Telser. Returns to speculators: reply. *Journal of Political Economy*, 68(4):404–415, 1960.

[15] Adam Zaremba, Mateusz Mikutowski, Jan Jakub Szczygielski, and Andreas Karathanasopoulos. The alpha momentum effect in commodity markets. *Energy Economics*, 93:104421, 2021.

[16] Zura Kakushadze. 101 formulaic alphas. *Wilmott*, 2016(84):72–81, 2016.

[17] Joëlle Miffre and Georgios Rallis. Momentum strategies in commodity futures markets. *Journal of Banking & Finance*, 31(6):1863–1886, 2007.

[18] Paresh Kumar Narayan, Huson Ali Ahmed, and Seema Narayan. Do momentum-based trading strategies work in the commodity futures markets? *Journal of Futures Markets*, 35(9):868–891, 2015.

[19] Jangkoo Kang and Kyung Yoon Kwon. Momentum in international commodity futures markets. *Journal of Futures Markets*, 37(8):803–835, 2017.

[20] Jesse Blocher, Ricky Cooper, and Marat Molyboga. Performance (and) persistence in commodity funds. Technical report, working paper, 2015.

[21] Thorben Lubnau and Neda Todorova. Trading on mean-reversion in energy futures markets. *Energy Economics*, 51:312–319, 2015.

[22] Marco Avellaneda and Jeong-Hyun Lee. Statistical arbitrage in the us equities market. *Quantitative Finance*, 10(7):761–782, 2010.

[23] Jeffrey Gropp. Mean reversion of industry stock returns in the us, 1926–1998. *Journal of Empirical Finance*, 11(4):537–551, 2004.

[24] Dingjiang Huang, Junlong Zhou, Bin Li, Steven CH Hoi, and Shuigeng Zhou. Robust median reversion strategy for online portfolio selection. *IEEE Transactions on Knowledge and Data Engineering*, 28(9):2480–2493, 2016.

[25] Vladimir V. Piterbarg. Topics in derivatives pricing, 2025. Lecture Notes, MSc Mathematics and Finance, Imperial College London.

[26] Mauricio Barahona. Methods for data science, 2024. Lecture Notes, Mathematics, Imperial College London.

[27] LightGBM Developers. Lightgbm 4.6.0.99 documentation. `https://lightgbm.readthedocs.io/en/stable/`, 2025. Accessed: 2025-08-20.

[28] Lukas Gonon. Deep learning, 2025. Lecture Notes, MSc Mathematics and Finance, Imperial College London.

[29] Sreelekshmy Selvin, R Vinayakumar, EA Gopalakrishnan, Vijay Krishna Menon, and KP Soman. Stock price prediction using lstm, rnn and cnn-sliding window model. In *2017 international conference on advances in computing, communications and informatics (icacci)*, pages 1643–1647. IEEE, 2017.

[30] Paul Alexander Bilokon. Introduction to reinforcement learning. Lecture notes, February 2024. Presented at Imperial College London, Thalesians Ltd.

[31] Jiayi Du, Muyang Jin, Petter N Kolm, Gordon Ritter, Yixuan Wang, and Bofei Zhang. Deep reinforcement learning for option replication and hedging. *The Journal of Financial Data Science*, 2(4):44–57, 2020.

[32] Alexander Fleiss, Amrith Kumaar, Adam Rida, Junsup Shin, Xinying Lai, Vivian Fang, Jialiang Chen, and Ang Li. Deep reinforcement learning & feature extraction for constructing alpha generating equity portfolios. *Available at SSRN 3958478*, 2021.

[33] Wenhang Bao and Xiao-yang Liu. Multi-agent deep reinforcement learning for liquidation strategy analysis. *arXiv preprint arXiv:1906.11046*, 2019.

[34] Johannes Muhle-Karbe. *Capital Asset Pricing Model*. Portfolio Management. 2021, Imperial College London, 2021. [lecture].

[35] Jay Cao, Jacky Chen, John Hull, and Zissis Poulos. Deep hedging of derivatives using reinforcement learning. *arXiv preprint arXiv:2103.16409*, 2021.

[36] John H Holland. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.

[37] John R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, MA, 1992.

[38] Trevor Stephens. Introduction to gp — gplearn 0.4.2 documentation. `https://gplearn.readthedocs.io/en/stable/intro.html`, 2022. Accessed: 2025-08-20; Version 0.4.2.

[39] Wolfgang Banzhaf, Frank D Francone, Robert E Keller, and Peter Nordin. *Genetic programming: an introduction: on the automatic evolution of computer programs and its applications*. Morgan Kaufmann Publishers Inc., 1998.

[40] Stefan Sette and Luc Boullart. Genetic programming: principles and applications. *Engineering applications of artificial intelligence*, 14(6):727–736, 2001.

[41] International Monetary Fund PortWatch. Data & methodology. `https://portwatch.imf.org/pages/data-and-methodology`. Accessed: 2025-08-21.

[42] International Monetary Fund PortWatch. Daily port activity data and trade estimates. `https://portwatch.imf.org/datasets/959214444157458aad969389b3ebe1a0/about`. Accessed: 2025-08-21.

[43] International Monetary Fund PortWatch. Daily chokepoint transit calls and trade volume estimates. `https://portwatch.imf.org/datasets/42132aa4e2fc4d41bdaf9a445f688931/about`. Accessed: 2025-08-21.

[44] International Monetary Fund PortWatch. Disruptions. `https://portwatch.imf.org/datasets/d9b37bf4b2104c85aebdcc0c1d8a2ab7/about`. Accessed: 2025-08-21.

[45] Dr. Paul Bilokon. Proprietary data source of sensibly preprocessed continuation commodity futures prices. Provided by Dr. Bilokon, 2025.

[46] Thomas Cass. Statistical methods in finance, 2025. Lecture Notes, MSc Mathematics and Finance, Imperial College London.

[47] Takeshi Ogita and Kensuke Aishima. Iterative refinement for symmetric eigenvalue decomposition. *Japan Journal of Industrial and Applied Mathematics*, 35(3):1007–1035, 2018.

[48] Paul Bilokon and David Finkelstein. Iterated and exponentially weighted moving principal component analysis. *arXiv preprint arXiv:2108.13072*, 2021.

[49] Trevor Stephens. gplearn: Genetic programming in python, with a scikit-learn inspired api. `https://github.com/trevorstephens/gplearn`, 2022. Version 0.4.2 (released 2022-05-03). GitHub repository.

[50] MET Group. Natural gas transportation. `https://group.met.com/en/media/energy-insight/natural-gas-transportation/`, 2020. Aaccessed: 2025-08-29.

[51] Thalia Chantziara and George Skiadopoulos. Can the dynamics of the term structure of petroleum futures be forecasted? evidence from major markets. *Energy Economics*, 30(3):962–985, 2008.

[52] Chao Liang, Feng Ma, Ziyang Li, and Yan Li. Which types of commodity price information are more useful for predicting us stock market volatility? *Economic Modelling*, 93:642–650, 2020.

[53] Christian Gross. Examining the common dynamics of commodity futures prices. In *Center for Quantitative Economics (CQE)*. University of Muenster, 2017.