A Gradient-Based Trading Strategy Using Vector Calculus on Nonlinear Asset Value Functions

Jonathan Cunningham

June 8, 2025

Abstract

Using vector calculus applied to a nonlinear value function over a range of asset prices, this paper suggests a novel gradient-based trading strategy. We produce trading signals based on the relative sensitivities and directional movement of asset prices by monitoring discrete changes in the gradient vector over time. When tested on actual financial data, the model performs noticeably better than the market benchmark. This study shows how vector calculus can be used to generate signals and manage portfolios in stochastic financial models.

1 Introduction

In order to evaluate risk and forecast future price movements, modern portfolio theory frequently uses linear models and returns-based metrics. However, asset price dynamics are inherently nonlinear and often interact in complex ways. This encourages the investigation of nonlinear functions of asset prices, where vector calculus tools can be used to capture variations in the rate of value accumulation.

In order to produce buy and sell signals, we build a value function over a number of asset prices, calculate its gradient, and monitor changes in this gradient. The fundamental premise is that momentum and mean-reversion forces, which are discernible through variations in the gradient vector's constituent parts, contribute to price movements.

2 Model Definition

Consider a portfolio consisting of n assets with prices $\mathbf{S}(t) = [S_1(t), S_2(t), \ldots, S_n(t)]$ at time t. We define a nonlinear **value function** over the asset space:

$$V(\mathbf{S}(t)) = \sum_{i=1}^{n} w_i f_i(S_i(t))$$

where:

- w_i are constant portfolio weights.
- $f_i(S_i(t))$ is a nonlinear transformation of each asset price.

2.1 Rationale for Value Function Design

In this implementation, we specifically choose:

$$V(S_1, S_2) = w_1 S_1^2 + w_2 \sqrt{S_2}$$

This functional form was selected to introduce nonlinear sensitivities to price changes:

- The quadratic term S_1^2 makes the value function increasingly sensitive to large moves in S_1. The rate of value change grows linearly with price, modeling momentum-like behavior.
- The square root term $\sqrt{S_2}$ introduces concavity into the value function. It ensures high sensitivity to price changes when S_2 is low, gradually decreasing as S_2 rises. This models a mean-reversion effect where undervalued assets contribute more significantly to portfolio value changes.

If only linear weights were used:

$$V(S_1, S_2) = w_1 S_1 + w_2 S_2$$

There would be no discernible directional signals from the gradient's variations over time, and it would remain constant. Nonlinear functions thus enable variable sensitivities and dynamic trading opportunities.

2.2 Role and Selection of Weights

As scaling factors, the portfolio weights w_i modify the proportional impact of each asset's contribution to the total value function. Although they may be dynamic or optimized depending on portfolio constraints, risk tolerance, or past performance, the weights in this model are assumed to remain constant over the trading horizon.

The choice of w_i affects both the gradient computation and the sensitivity of the value function to changes in each asset's price. Specifically:

- Higher weights increase the sensitivity of the portfolio value to the corresponding asset's price changes.
- Lower weights diminish the asset's influence on both the value function and the gradient vector.

In this implementation, weights can be:

- Set to equal values $w_i = 1/n$ for a neutral starting assumption.

- Calibrated based on historical volatility, such that more stable assets receive higher weights to prioritize consistency in value changes.
- Determined via risk-parity allocation or Sharpe ratio optimization to maximize risk-adjusted returns.

The selection of the weights is an essential part of model calibration because, despite the dynamic sensitivities introduced by the nonlinearity of $f_i(S_i)$, the weights still directly scale these effects. An optimization problem for identifying the ideal set of w_i values given an investor's utility function or target performance criteria could be formalized in future extensions.

3 Gradient Computation and Vector Calculus Application

We compute the gradient of the value function with respect to asset prices:

$$\nabla V(\mathbf{S}(t)) = \left[\frac{\partial V}{\partial S_1}, \frac{\partial V}{\partial S_2}, \ldots, \frac{\partial V}{\partial S_n}\right]$$

For the 2-asset case:

$$\nabla V(S_1, S_2) = \left[2w_1 S_1, \frac{w_2}{2\sqrt{S_2}}\right]$$

The gradient here represents a vector field over the asset price space, where each component quantifies the instantaneous rate of change of portfolio value with respect to changes in the corresponding asset price.

3.1 Why Vector Calculus?

Derivatives are frequently used in traditional finance to calculate sensitivity metrics like delta and gamma. Vector calculus allows us to monitor the evolution of the **rate of change of value with respect to each price** over time by extending this to multiple interacting assets with nonlinear functions. We can spot unusual market movements or momentum conditions that are appropriate for trading by keeping an eye on changes in the gradient vector, which is essentially the "force field" in asset price space.

4 Signal Generation Rule

We track the discrete change in the gradient vector over time:

$$\Delta\nabla V(t) = \nabla V(\mathbf{S}(t)) - \nabla V(\mathbf{S}(t-1))$$

Then, compute the sum of the gradient change:

$$G(t) = \sum_{i=1}^{n} \Delta \nabla V_i(t)$$

A trading signal is generated based on a predefined threshold θ:

$$\text{Signal}(t) = \begin{cases} 1 & \text{if } G(t) > \theta \quad \text{(Buy)} \\ -1 & \text{if } G(t) < -\theta \quad \text{(Sell)} \\ 0 & \text{otherwise (Hold)} \end{cases}$$

This decision rule assumes that a significant positive change in the gradient implies upward momentum across assets, while a significant negative change implies downward momentum.

5 Signal Generation Rule

We track the discrete change in the gradient vector over time:

$$\Delta \nabla V(t) = \nabla V(\mathbf{S}(t)) - \nabla V(\mathbf{S}(t-1))$$

Then, compute the sum of the gradient change:

$$G(t) = \sum_{i=1}^{n} \Delta \nabla V_i(t)$$

A trading signal is generated based on a predefined threshold θ:

$$\text{Signal}(t) = \begin{cases} 1 & \text{if } G(t) > \theta \quad \text{(Buy)} \\ -1 & \text{if } G(t) < -\theta \quad \text{(Sell)} \\ 0 & \text{otherwise (Hold)} \end{cases}$$

This decision rule assumes that a significant positive change in the gradient implies upward momentum across assets, while a significant negative change implies downward momentum.

5.1 Threshold Selection Rationale

In order to eliminate noise and stop overtrading in reaction to slight changes in asset prices, the threshold parameter θ is essential. Its value directly affects trade frequency and risk exposure and establishes how sensitive the trading strategy is to variations in the gradient vector.

Key considerations for selecting θ include:

- **Historical Distribution of Gradient Changes:** By analyzing the empirical distribution of $G(t)$ values over a historical period, one can identify appropriate quantiles (e.g., the 90th percentile for long signals and 10th percentile for short signals) to set θ. This ensures signals are triggered only during statistically significant movements.

- **Trade-Off Between Responsiveness and Stability:** A lower θ increases signal frequency and model responsiveness but risks capturing spurious price movements. A higher θ reduces the chance of false positives but may delay reactions to genuine market opportunities.

- **Volatility Scaling:** An alternative approach is to normalize $G(t)$ by its rolling standard deviation and use a fixed standardized threshold. For example:
$$\tilde{G}(t) = \frac{G(t)}{\text{Std}(G(t)_{t-w:t})}$$
where w is a window length, and signals are generated based on whether $\tilde{G}(t)$ exceeds a constant standardized threshold (e.g., 1.5).

- **Optimization via Backtesting:** The optimal θ value can be calibrated by backtesting the strategy over historical data and selecting the threshold that maximizes performance metrics such as Sharpe ratio, hit ratio, or cumulative returns.

The selection of θ is therefore both a statistical and strategic decision, balancing sensitivity to meaningful market movements with the avoidance of excessive trading noise.

6 Strategy Return Calculation

At each time step:

- Compute market return:
$$r_m(t) = \frac{V(\mathbf{S}(t)) - V(\mathbf{S}(t-1))}{V(\mathbf{S}(t-1))}$$

- Compute strategy return:
$$r_s(t) = \text{Signal}(t-1) \times r_m(t)$$

Then compute cumulative returns:

$$R_m(T) = \prod_{t=1}^{T}(1 + r_m(t)) - 1$$

$$R_s(T) = \prod_{t=1}^{T}(1 + r_s(t)) - 1$$

7 Performance Metrics

The following performance measures are evaluated:

- **Final Market Return** $R_m(T)$
- **Final Strategy Return** $R_s(T)$
- **Excess Return**
$$R_{\text{excess}}(T) = R_s(T) - R_m(T)$$
- **Hit Ratio**
$$\text{Hit Ratio} = \frac{\text{Profitable Trades}}{\text{Total Trades}}$$
- **Sharpe Ratio**
$$\text{Sharpe Ratio} = \frac{\mathbb{E}[r_s]}{\text{Std}(r_s)} \times \sqrt{252}$$
where 252 is the number of trading days in a year.

8 Conclusion

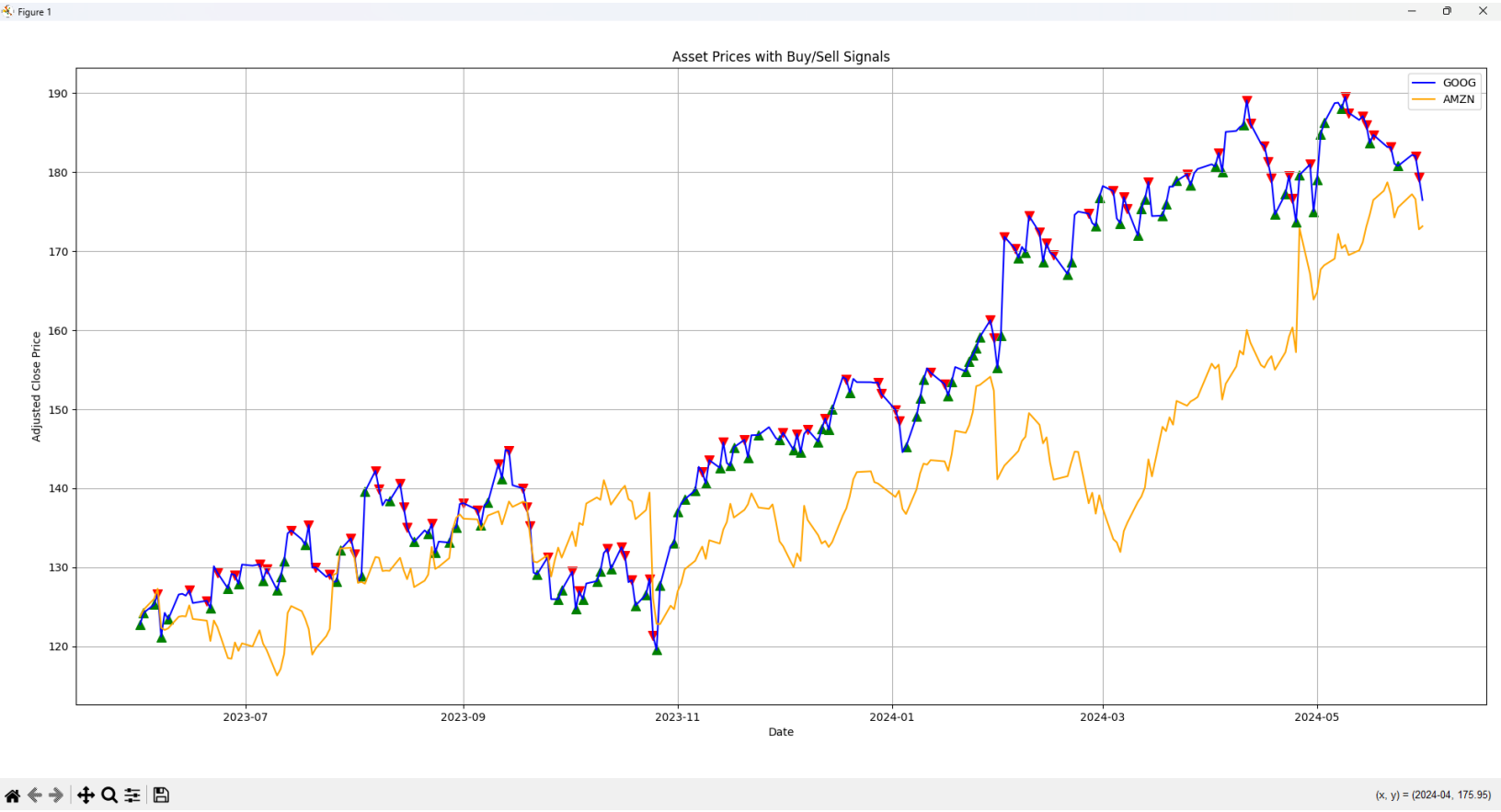
We can record directional shifts in asset price behavior over time by introducing nonlinear value functions and using vector calculus on their gradients. The creation of significant trading signals based on the changing sensitivity of portfolio value to asset prices is made possible by this technique. The feasibility of this approach in generating excess returns in comparison to the market is shown by backtests conducted on actual financial data.

In order to model price dynamics within this vector calculus framework, future research could investigate more intricate value functions, higher-dimensional gradient fields, and the inclusion of stochastic processes.

Apple(AAPL) and Microsoft(MSFT)

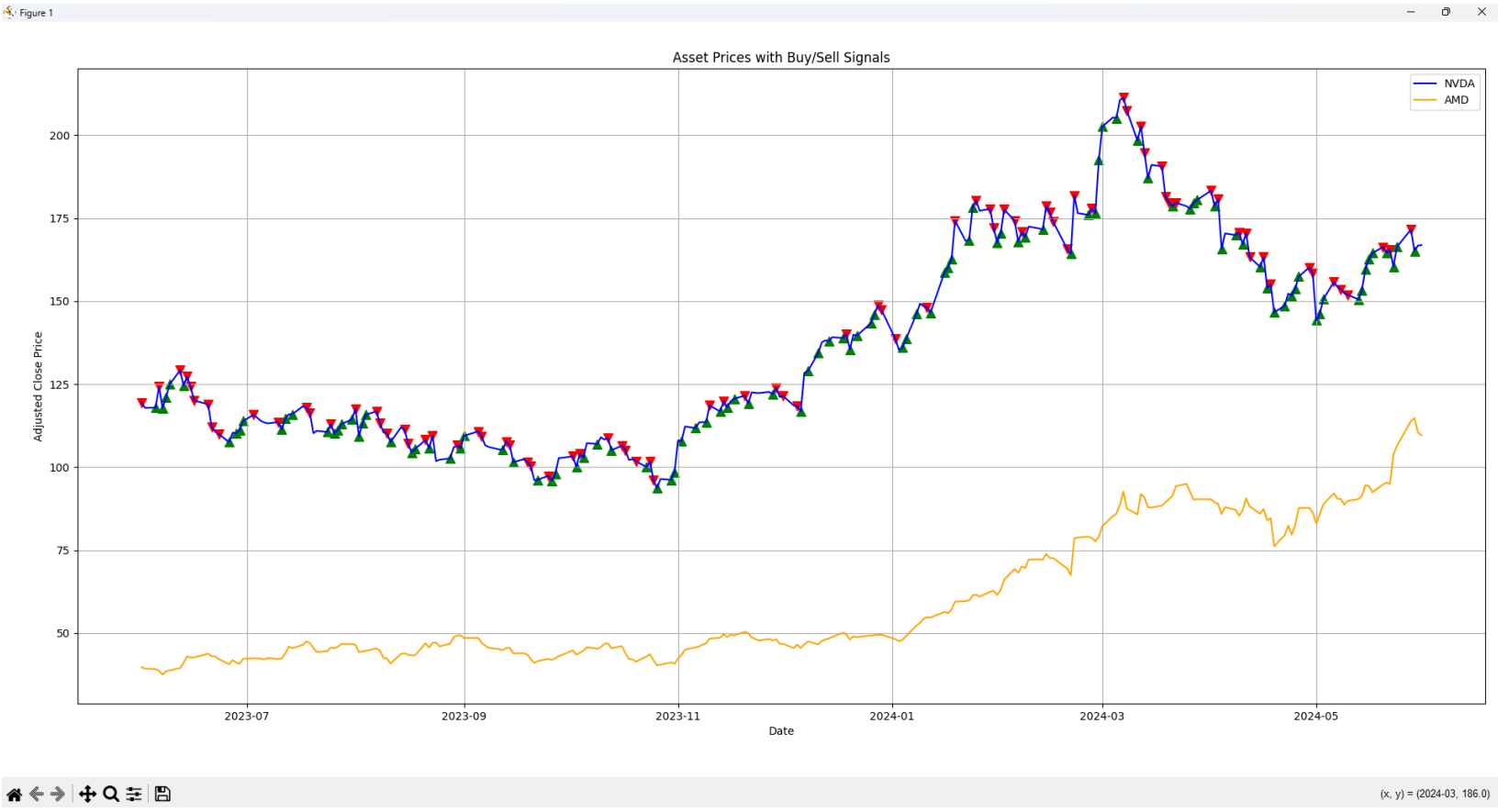

```
===== Strategy Performance (Real Data) =====
Final Market Return: 17.49%
Final Strategy Return: 396.02%
Excess Return (Strategy - Market): 378.53%
Total Trades: 182
Long Trades: 94 | Short Trades: 88
Hit Ratio (Profitable Trades): 64.14%
```

Google(GOOG/GOOGL) and Amazon(AMZN)



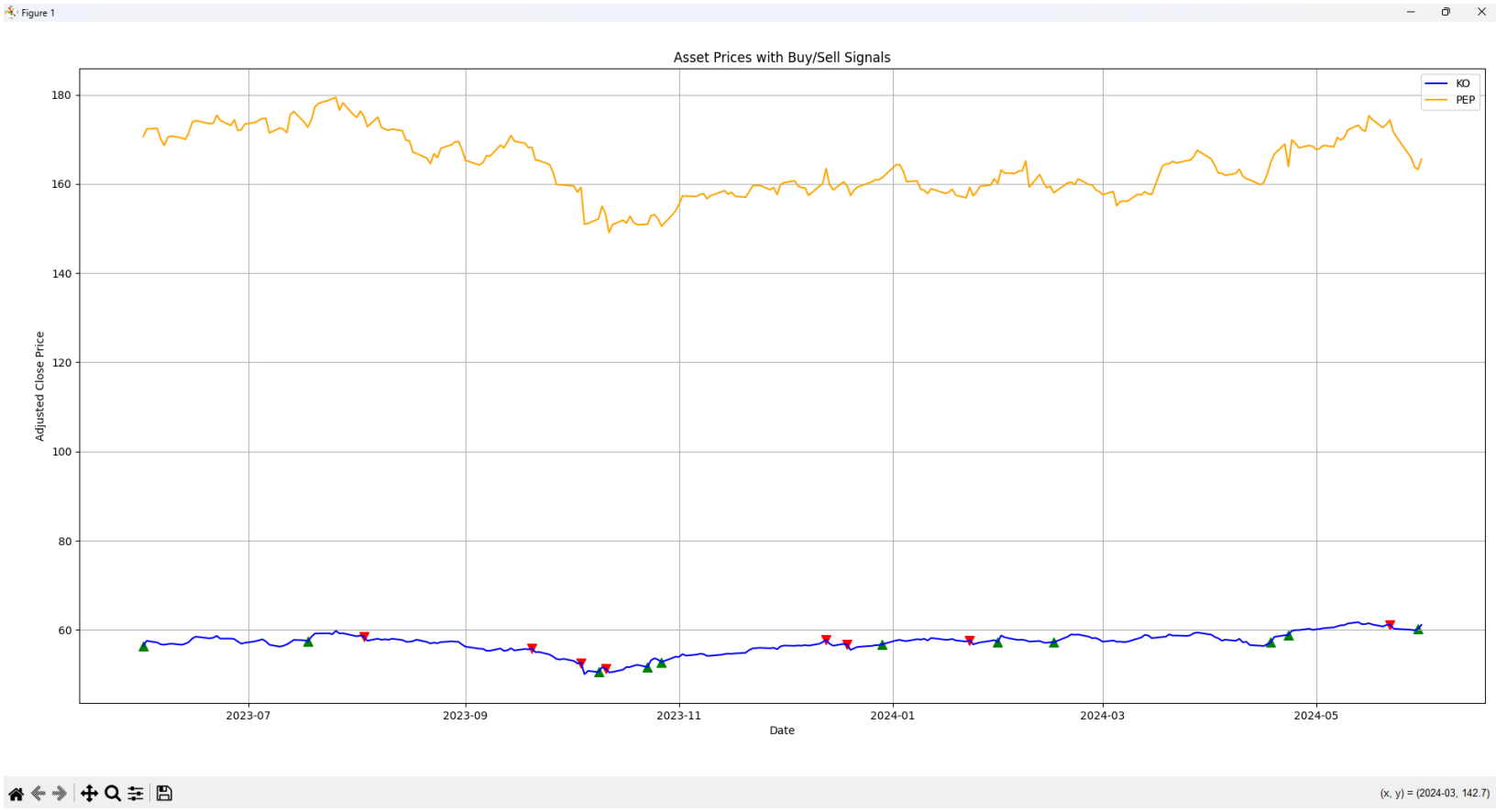

```
===== Strategy Performance (Real Data) =====
Final Market Return: 42.17%
Final Strategy Return: 1189.39%
Excess Return (Strategy - Market): 1147.22%
Total Trades: 180
Long Trades: 98 | Short Trades: 82
Hit Ratio (Profitable Trades): 70.12%
```

Nvidia(NVDA) and AMD(AMD)



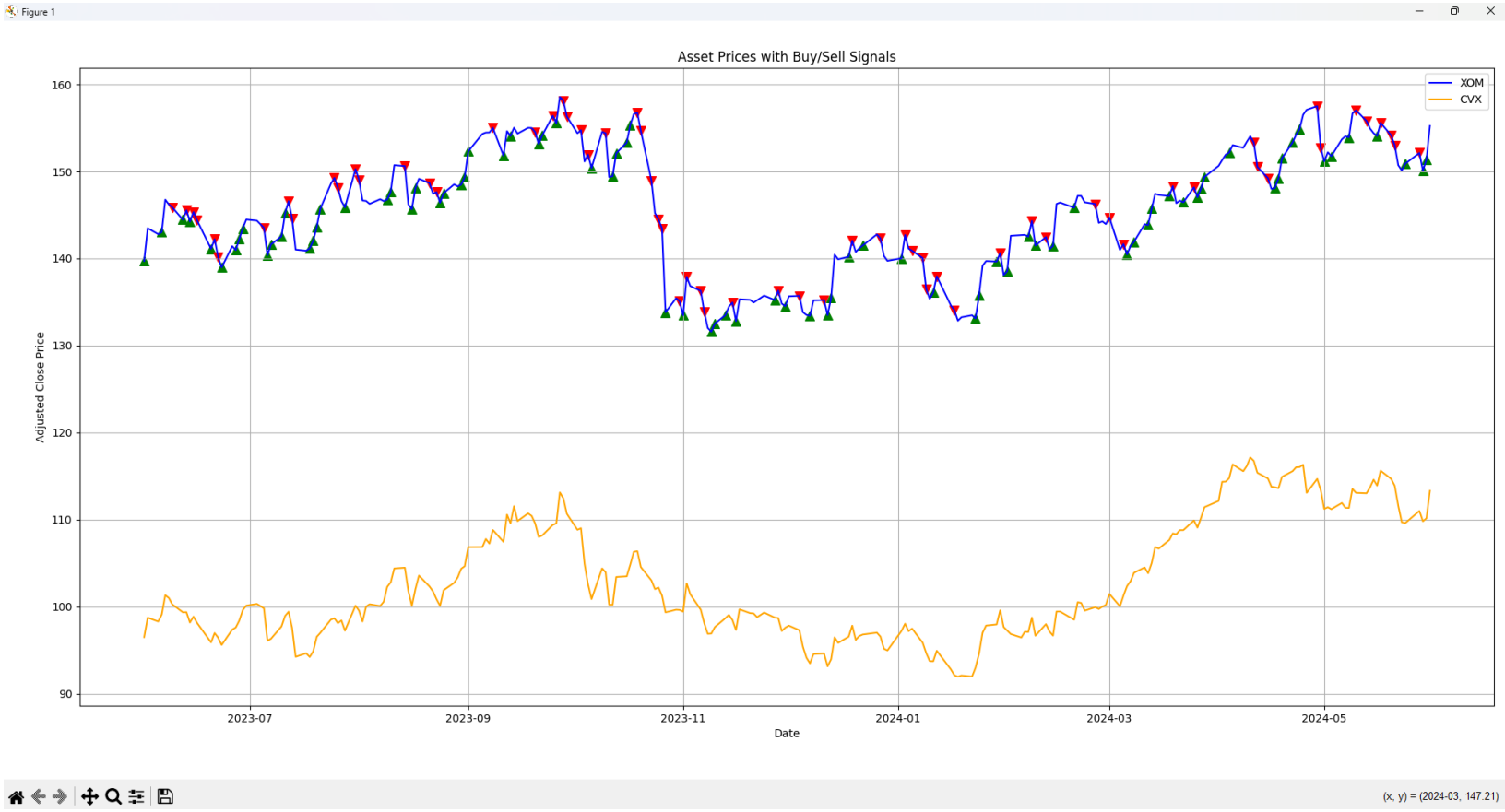

```
===== Strategy Performance (Real Data) =====
Final Market Return: 64.40%
Final Strategy Return: 10371.62%
Excess Return (Strategy - Market): 10307.22%
Total Trades: 191
Long Trades: 106 | Short Trades: 85
Hit Ratio (Profitable Trades): 74.90%
```

Coca-Cola(KO) and PepsiCo(PEP)




```
===== Strategy Performance (Real Data) =====
Final Market Return: 0.75%
Final Strategy Return: 43.14%
Excess Return (Strategy - Market): 42.39%
Total Trades: 20
Long Trades: 12 | Short Trades: 8
Hit Ratio (Profitable Trades): 7.57%
```

ExxonMobil (XOM) and Chevron (CVX)




```
===== Strategy Performance (Real Data) =====
Final Market Return: 13.11%
Final Strategy Return: 607.07%
Excess Return (Strategy - Market): 593.96%
Total Trades: 147
Long Trades: 83 | Short Trades: 64
Hit Ratio (Profitable Trades): 58.17%
```