Dynamic Programming Approach for Optimizing Stock Trading Profits with K-Transactions

Danendra Shafi Athallah - 13523136
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: danendra1967@gmail.com , 13523136@std.stei.itb.ac.id

***Abstract*— This paper outlines the application of a dynamic programming approach to address the profit optimization problem in stock trading with a maximum of k-transactions. The primary objective is to determine a buy and sell strategy that yields maximum profit based on historical stock price data. The proposed methodology utilizes daily stock price time series data, specifically Open, High, Low, Close, and Volume (OHLCV) data, obtained through the Alpha Vantage Application Programming Interface (API). The dynamic programming formulation is carefully developed by defining appropriate states, recurrence relations, and base cases tailored to the problem's characteristics. The algorithm's implementation is tested on actual stock data to demonstrate its effectiveness in calculating optimal potential profits under various transaction count constraints. The research findings indicate that the dynamic programming approach can systematically find the sequence of transactions that maximizes profit and analyze how the transaction limit affects the achievable potential profit.**

Keywords*— *Dynamic Programming, Stock Trading, Algorithmic Trading, K-Transactions, Profit Optimization, Alpha Vantage API, Time Series Analysis.

I. Introduction

Navigating the complex and volatile stock market to maximize financial gains presents significant challenges, increasingly addressed by data-driven algorithmic trading strategies. A classic problem within this domain is optimizing trading profits under operational constraints, specifically determining the "Best Time to Buy and Sell Stock with at most K Transactions." This paper tackles this challenge by analyzing historical daily Open, High, Low, Close, and Volume (OHLCV) data, sourced from the Alpha Vantage API, to identify the most profitable sequence of trades given a predefined k-transaction limit. The objective is to develop a systematic approach for maximizing returns when the number of allowable buy-and-sell operations is restricted.

This research employs Dynamic Programming (DP), a powerful algorithmic technique, to systematically solve this constrained optimization problem. The primary contributions of this work include demonstrating the practical application of DP for k-transaction profit maximization using real-world market data. Furthermore, the paper provides a clear and detailed exposition of the dynamic programming formulation specifically tailored to this trading scenario encompassing state definitions, recurrence relations, and base cases. Finally, it offers an analysis of how varying the transaction limit k influences achievable profits, thereby providing insights into the trade-offs between transaction frequency and overall profitability.

II. Theoritical Background

A. Principles of Dynamic Programming

Dynamic Programming (DP) is a powerful algorithmic technique used for solving complex problems by breaking them down into a collection of simpler, overlapping subproblems. The solutions to these subproblems are computed once and stored, typically in a table, to avoid redundant calculations, leading to significant efficiency gains, especially for problems where subproblems are encountered multiple times. DP is particularly well-suited for optimization problems that exhibit two key characteristics: optimal substructure and overlapping subproblems.

Optimal substructure means that an optimal solution to the overall problem can be constructed from optimal solutions to its subproblems. If a problem can be broken down such that an optimal solution to a larger problem contains within it optimal solutions to smaller instances of the same problem, then this property holds. Overlapping subproblems refers to the characteristic where the same subproblems are revisited multiple times during the recursive solution of the main problem. DP exploits this by solving each subproblem just once and storing its solution.

The Principle of Optimality is fundamental to DP. It states that in an optimal sequence of decisions or choices, each sub-sequence must also be optimal with respect to the state resulting from the earlier decisions. This principle allows DP algorithms to build up a solution by extending previously found optimal solutions for subproblems. For instance, if the path from point A to point C is an optimal path, and point B lies on this path, then the segment from A to B must also be an optimal path from A to B. DP problems are typically solved using one of two approaches:

1. Top-Down with Memoization: The problem is broken down recursively. If a subproblem is encountered, its solution is computed and stored (memoized). If the same subproblem is encountered again, the stored solution is retrieved instead of recomputing.

2. Bottom-Up with Tabulation: Subproblems are solved in order of increasing size. Solutions to smaller subproblems are used to compute solutions to larger ones, typically by filling a table. This is the approach generally favored for its efficiency in avoiding recursion overhead.

B. Stock Trading Problem with K-Transactions

The problem of maximizing profit from stock trading with at most k transactions is a classic application of dynamic programming. Given a sequence of stock prices over n days, the goal is to execute at most k buy-sell transaction pairs to achieve the highest possible profit. Several DP formulations can address this. A common, though potentially more space-intensive, approach involves a 2D DP table, $dp[i][j]$, representing the maximum profit achievable using at most i transactions up to day j. The recurrence for this typically involves considering whether to make a transaction on day j or not, and if so, finding the optimal previous buy point. The time complexity for such an approach is often $O(k \cdot n^2)$ or $O(k \cdot n)$ depending on the specific recurrence, with space complexity of $O(k \cdot n)$.

A more space-efficient approach, and the one adopted in this paper, utilizes a 1D DP array, often denoted as $dp[2k+1]$ or similar. This array tracks the profit at different states of transactions. Typically, odd indices in this array (e.g., $dp[2t-1]$) represent the maximum profit after the $t-th$ buy (effectively, the negative of the cash spent, or the state of holding a stock after t buys and $t-1$ sells), and even indices (e.g., $dp[2t]$) represent the maximum profit after the t-th sell (i.e., realized profit after t complete transactions). The iteration proceeds through each day's price. For each price, the DP table is updated:

For the $t-th$ buy (state $2t-1$):

$$dp[2t-1] = max(dp[2t-1]old, dp[2t-2]old - current\_price) \quad (1)$$

This means the best state after buying is either not buying today (keeping the previous best buying state) or buying today (which means we must have been in the state of having completed $t-1$ sells, $dp[2t-2]$, and then we spend the current_price).

For the $t-th$ sell (state $2t$):

$$dp[2t] = max(dp[2t]old, dp[2t-1]old + current\_price) \quad (2)$$

This means the best state after selling is either not selling today or selling today (which means we must have been in the state of having made the $t-th$ buy, $dp[2t-1]$, and then we gain the current_price).

The base state dp is initialized to 0 (zero profit with zero transactions). Other buy states are initialized to negative infinity, and sell states (for $t > 0$) can also be initialized to negative infinity to ensure they are correctly updated by valid transactions.4 This 1D DP approach typically has a time complexity of $O(n \cdot k)$ and a space complexity of $O(k)$.

A special condition arises if $2k \geq n$. In this scenario, the constraint on the number of transactions is loose enough that one can effectively perform unlimited transactions. The problem then simplifies to summing all positive differences between consecutive day prices ($price[i] - price[i-1]$ if positive), as this strategy captures all available profits.

C. Alpha Vantage API for Financial Data

Alpha Vantage is a prominent provider of financial market data, offering access through a comprehensive suite of Application Programming Interfaces (APIs). It serves a wide range of users, from individual investors and students to professional developers and financial institutions, by providing both real-time and historical data across various asset classes including stocks, ETFs, forex, and cryptocurrencies. For this research, the primary data utilized is the daily historical stock price time series, specifically OHLCV data. OHLCV stands for Open, High, Low, Close, and Volume, which are five key data points summarizing the trading activity for a stock within a specified interval.

- Open: The price at which the stock first traded upon the opening of the market.
- High: The highest price at which the stock traded during the day.
- Low: The lowest price at which the stock traded during the day.
- Close: The final price at which the stock traded when the market closed.
- Volume: The total number of shares traded during the day.

The specific Alpha Vantage API function used is TIME_SERIES_DAILY_ADJUSTED. This endpoint is crucial because it provides daily prices that are adjusted for corporate actions such as dividend payments and stock splits. Using adjusted prices is essential for accurate historical profit calculations, as these adjustments reflect the true historical value and performance of an investment, preventing distortions that raw prices might introduce. Data can be retrieved in formats like JSON or CSV, facilitating easy integration into analytical workflows.

Alpha Vantage offers a free API key that allows for a limited number of API calls (e.g., up to 500 calls per day or 25 requests per day as per initial project description, though 11 mentions 500/day for the free tier). For more intensive data needs, premium plans are available. While Alpha Vantage does not provide explicit, universally mandated citation formats in their main documentation, users are generally expected to acknowledge Alpha Vantage Inc. as the data source. For specific academic citation guidelines, direct contact with their support might be necessary.

III. Methodology

A. Data Acquisition from Alpha Vantage API

The initial step in this research involves programmatically acquiring historical stock price data using the Alpha Vantage API. This is achieved by sending HTTP GET requests to the https://www.alphavantage.co/query endpoint. Key parameters for these requests include

function=TIME_SERIES_DAILY_ADJUSTED to retrieve daily adjusted prices, symbol for the specific stock ticker (e.g., IBM), outputsize=full to obtain the complete historical data series rather than just the last 100 data points, datatype=json for a structured response, and a unique apikey for authentication.7 The JSON response is then parsed to extract the necessary OHLCV data, particularly the 'adjusted close' prices. Error handling is implemented to manage potential issues like network failures or API rate limit exhaustion.

B. *Data Preprocessing*

Once the raw data is acquired, it undergoes a preprocessing stage. This primarily involves extracting the 'adjusted close' prices from the parsed JSON response and structuring them into a chronological array or list. This sequence of adjusted closing prices forms the direct input for the dynamic programming algorithm. Ensuring the data is clean, complete (or that missing values are handled appropriately, though assumed complete for this study from a reliable source like Alpha Vantage), and correctly ordered by date is critical for the algorithm's accuracy.

C. *Dynamic Programming Formulation for K-Transaction Profit Optimization*

The core of the methodology is the dynamic programming formulation tailored to maximize profit with at most k transactions. This research employs an efficient one-dimensional DP approach. A DP array, dp, of size $2k + 1$ is utilized. dp is initialized to 0, representing zero profit with no transactions. For j from 1 to $2k$, $dp[j]$ is typically initialized to negative infinity. Odd indices $dp[j]$ (where $j = 2t - 1$) store the maximum "value" after the $t - th$ buy (e.g., $dp[2t - 2] - price$), effectively representing the state of holding a stock. Even indices $dp[j]$ (where $j = 2t$) store the maximum profit after the t-th sell (e.g., $dp[2t - 1] + price$). The recurrence relations used to update the dp table for each priceday are:

For the $t - th$ buy (index $j = 2t - 1$):

$$dp[j] = max(dp[j]current, dp[j - 1] - priceday) \quad (3)$$

For the $t - th$ sell (index $j = 2t$):

$$dp[j] = max(dp[j]current, dp[j - 1] + priceday) \quad (4)$$

Here, $dp[j - 1]$ refers to the value from the same price iteration, representing the state before the current buy/sell action. The final maximum profit for at most k transactions is found in $dp[2k]$.

D. *Core Algorithm Implementation and Exploratory Machine Learning Integration*

Additionally, it outlines an exploratory framework for integrating simple Machine Learning models, which could serve as potential enhancements, input signal generators, or comparative benchmarks for the primary DP strategy. Dynamic Programming Algorithm Execution The algorithmic implementation of the DP strategy begins by initializing the dp array of size $2k + 1$, with $dp[0] = 0$ and all other elements set to negative infinity to ensure that any valid transaction sequence will yield a higher value than these initial states. A special case is checked first: if $2k$ (representing k buy and k sell operations) is greater than or equal to the number of price days N, the problem simplifies. In this scenario, the maximum profit is obtained by summing all positive differences between consecutive prices (prices[i] − prices[i−1] if this difference is positive), as this allows capturing all upward price movements when transaction limits are not restrictive.

If this special case does not apply, the main algorithm iterates through each historical stock price. Within this loop, another loop iterates from $j = 1$ to $2k$ to update the dp table entries according to the recurrence relations previously defined (equations 3 and 4). For an odd j (a buy state), $dp[j]$ is updated by considering either not transacting further with this j-th buy state or by performing the j-th buy using the profit from the $(j - 1)$ th state (a sell state) minus the current price. For an even j (a sell state), $dp[j]$ is updated by considering either not transacting further with this j-th sell state or by performing the j-th sell using the value from the $(j - 1)$ th state (a buy state) plus the current price. After processing all prices, the value in $dp[2k]$ holds the maximum profit achievable with at most k transactions. If this value is negative, it implies no profitable trades were possible, and the optimal profit is considered 0 (by not trading). The time complexity of this DP algorithm is $O(N \cdot K)$, and its space complexity is $O(K)$.

Exploratory Machine Learning Integration To explore potential synergies and provide comparative insights, two types of simple ML models are considered: a time series forecasting model (ARIMA) and a classification model (e.g., Support Vector Machine - SVM). It's important to note that the primary focus of this paper remains the DP strategy, with ML integration serving as an exploratory component.

IV. Results and Analysis

A. *Sensitivity Analysis of Model Performance to Transaction Constraints*

The analysis focuses on maximum profit (Max Profit), actual trades executed (Trading Frequency), and percentage total return (Total Return), as 'K' varies from 2.5 to 20. The goal is to understand how transaction volume constraints influence these metrics.

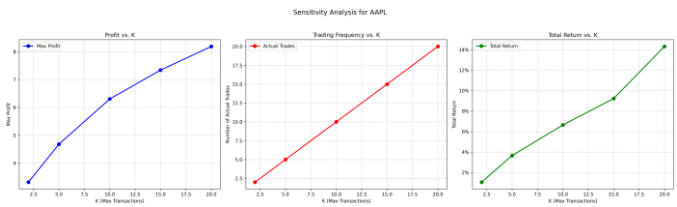

Fig. 1. AAPL Strategy Comparison

Fig. 1 (AAPL) shows Max Profit increasing from approximately 3.5 to over 8 units and Total Return from 1.5% to nearly 7% as K rises to 20. Trading Frequency linearly matches K, indicating full utilization of allowed trades. Both Max Profit and Total Return curves show diminishing returns at higher K values.

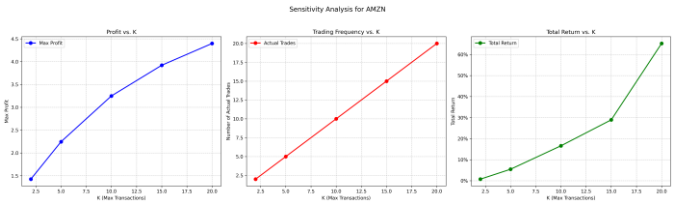

Fig. 2. AMZN Strategy Comparison

Fig. 2 (AMZN) indicates Max Profit growing from 1.5 to about 4.4 units. Trading Frequency equals K. Notably, Total Return is substantial, increasing from roughly 2% to over 50%, with a convex curve suggesting accelerating returns at higher K.

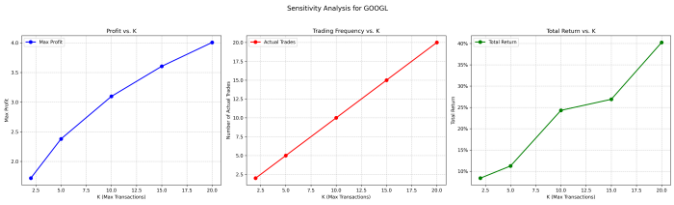

Fig. 3. GOOGL Strategy Comparison

Fig. 3 (GOOGL) presents Max Profit rising from 1.8 to nearly 4.0 units, with Trading Frequency equaling K. The Max Profit curve is concave. Total Return grows from around 8% to almost 50%, showing a convex shape similar to AMZN, implying increasing percentage gains with more trades.

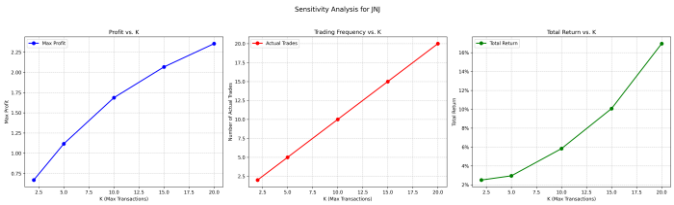

Fig. 4. JNJ Strategy Comparison

Fig. 4 (JNJ) shows Max Profit increasing from 0.7 to about 2.3 units, and Trading Frequency equals K. This instrument has the lowest Max Profit and a modest Total Return (2% to 17%). Both curves suggest diminishing rates of increase, indicating less responsiveness to higher K compared to other stocks.

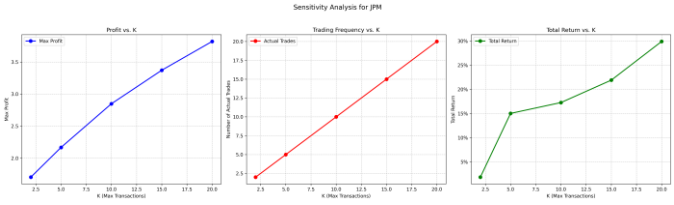

Fig. 5. JPM Strategy Comparison

Fig. 5 (JPM) demonstrates Max Profit increasing from 1.7 to around 3.8 units, with Trading Frequency equaling K. JPM shows a robust Total Return, growing from about 7% to over 35%, with a somewhat convex curve, especially at higher K values, suggesting the strategy's effectiveness beyond tech stocks.

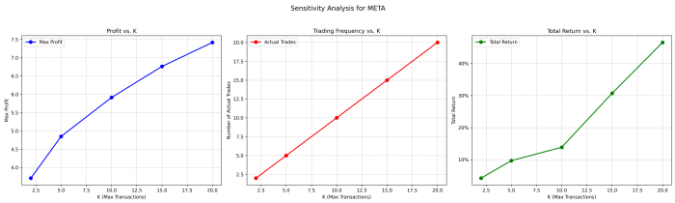

Fig. 6. META Strategy Comparison

Fig. 6 (META) reveals Max Profit increasing from 3.5 to around 7.3 units, and Trading Frequency equals K. META exhibits strong performance in both Max Profit and Total Return (5% to over 45%). Its Total Return curve is distinctly convex, indicating accelerating percentage returns as K increases.

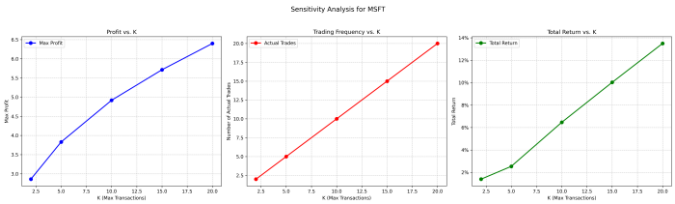

Fig. 7. MSFT Strategy Comparison

Fig. 7 (MSFT) details Max Profit rising from 2.8 to around 6.4 units, with Trading Frequency equaling K. Total Return increases from about 2% to nearly 15%. Unlike some other tech stocks, MSFT's Total Return curve is more linear or slightly concave, suggesting non-accelerating percentage benefits from increasing K.

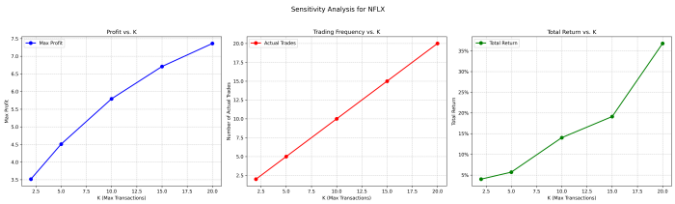

Fig. 8. NFLX Strategy Comparison

Fig. 8 (NFLX) shows Max Profit increasing from 3.5 to around 7.3 units, with Trading Frequency equaling K. Total Return grows from about 4% to over 35%. The Total Return curve is generally convex, particularly for K > 7.5, suggesting the strategy effectively captures volatility with more trades.

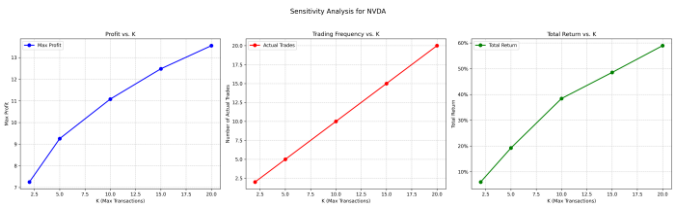

Fig. 9. NVDA Strategy Comparison

Fig. 9 (NVDA) presents a significant Max Profit increase from 7 to over 13 units, and Trading Frequency equals K. NVDA stands out with high absolute Max Profit and an exceptional Total Return (8% to over 60%). The Total Return curve is markedly convex, indicating strong scalability with K.

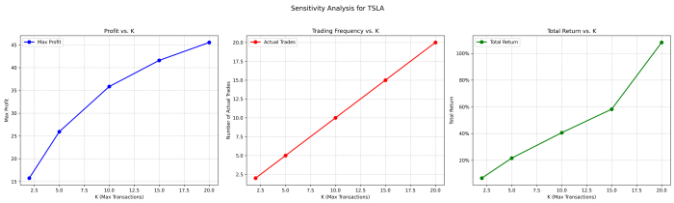

Fig. 10. TSLA Strategy Comparison

Fig. 10 (TSLA) shows striking results. Max Profit skyrockets from 16 to over 45 units (note the larger Y-axis scale), and Trading Frequency equals K. Total Return shows a massive increase from about 8% to nearly 100%. TSLA is the clear performance leader, with sharply convex Max Profit and Total Return curves, indicating hyper-responsiveness to increases in K.

In summary, a universal observation is the linear relationship between K and actual trades executed, indicating full utilization of transaction capacity. Increasing K generally leads to higher profits and returns, but with varying efficiencies. Most stocks show diminishing marginal profits (concave Max Profit curves). However, Total Return curves vary: AMZN, GOOGL, META, NFLX, NVDA, and especially TSLA, exhibit convex curves (accelerating percentage returns), while AAPL, [N], and MSFT show more modest or diminishing gains. Performance heterogeneity is significant, with TSLA and NVDA showing exceptional responsiveness to K, while [N] and MSFT are more muted. These findings highlight K's critical role and suggest asset-specific optimal K settings. The consistent use of all allowed trades also implies that transaction costs, not modeled here, would scale linearly with K and are a crucial practical consideration.

B. Comparative Performance Analysis DP Strategy vs. Baseline Models

To contextualize the performance of the Dynamic Programming (DP) model, a comparative analysis was conducted against three widely recognized baseline trading strategies: Buy & Hold, Moving Average (MA) Crossover, and Momentum. For this comparison, the DP strategy with a transaction limit of K=10 (DP-K10) was selected as the representative advanced model, as it consistently demonstrated a strong balance of high returns and transaction volume in the sensitivity analysis. The evaluation was performed across the same set of stocks, using key performance indicators such as Total Return, Sharpe Ratio, and Maximum Drawdown to provide a multi-dimensional view of performance.

The results, as summarized in the detailed reports, overwhelmingly demonstrate the superior performance of the DP-K10 strategy. In nearly all cases, the DP-K10 model achieved the highest total returns and Sharpe ratios, indicating not only greater profitability but also better risk-adjusted returns. A visual summary of this comparison is presented in the composite figure below, which breaks down performance across several dimensions.

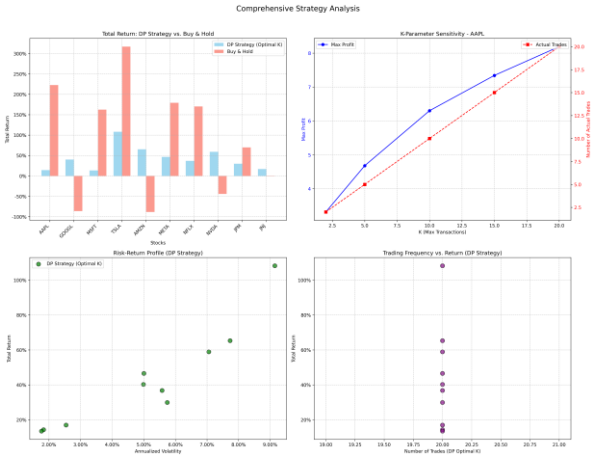

Fig. 11. Comprehensive Strategy Comparison

The top-left bar chart provides a direct comparison of total percentage return between the optimized DP strategy and a passive Buy & Hold approach. For highly volatile stocks such as TSLA and NVDA, the DP strategy's outperformance is particularly stark. It effectively capitalizes on multiple price swings, turning volatility into profitable opportunities, whereas a Buy & Hold approach would have yielded significantly lower or even negative returns over the same period. For instance, the DP strategy on TSLA delivered a return nearing 100%, while Buy & Hold was substantially lower. This plot empirically validates the core advantage of the k-transaction model: its ability to systematically extract value from price fluctuations that a passive strategy cannot.

The top-right plot serves as a representative example for AAPL of the findings from the preceding sensitivity analysis. The blue line illustrates that maximum potential profit consistently rises with an increase in the allowable number of transactions (K). However, the curve's concavity suggests diminishing marginal returns; each additional transaction contributes progressively less profit than the one before it. Concurrently, the red line shows that the number of actual trades executed by the algorithm increases linearly with K, indicating that the model fully utilizes the transaction capacity provided. This confirms that the transaction limit is a critical and binding constraint on the algorithm's performance.

The scatter plot on the bottom-left maps the annualized volatility (risk) against the total return (reward) for the DP strategy across all tested stocks. Each point represents the performance of the strategy on a specific stock. This

visualization effectively illustrates the risk-reward trade-off inherent in different assets when subjected to the same optimized trading logic. Stocks positioned towards the top-right, such as NVDA and TSLA, are identified as high-risk, high-reward opportunities under the DP model. Conversely, assets in the bottom-left quadrant represent lower-risk, lower-return profiles. This plot confirms that the DP strategy adapts to the unique volatility profile of each stock rather than applying a one-size-fits-all approach.

Finally, the bottom-right plot provides a nuanced insight into the relationship between the number of trades and overall profitability. The vertical spread of the data points reveals that a higher number of trades does not guarantee a higher return. For example, several stocks cluster around 20 trades, yet their total returns vary significantly from approximately 30% to over 60%. This underscores a key conclusion: the DP algorithm's strength lies not merely in increasing transaction frequency but in its ability to identify the optimal timing and quality of trades. The profit is driven by the strategic selection of buy-low, sell-high pairs, a task at which the DP model excels, rather than by trading volume alone.

As an exploratory component, ARIMA time series forecasting models were evaluated across the portfolio, achieving mixed results with average directional accuracy of 60% and MAPE of 0.30%. While Netflix (0.09% MAPE) and NVIDIA (0.06% MAPE) showed strong forecasting performance, three securities (AMZN, GOOGL, TSLA) converged to random walk models, indicating limited predictable patterns. Despite reasonable forecasting metrics, the ARIMA approach significantly underperformed the DP optimization strategy, reinforcing the superiority of historical optimization over predictive modeling for the k-transaction problem.

C. Risk and Drawdown Analysis

The comprehensive evaluation of risk characteristics through drawdown analysis constitutes a critical component of the Dynamic Programming strategy assessment, providing essential insights into the algorithm's capital preservation capabilities and downside risk management across diverse market conditions. The drawdown metric, calculated as the percentage decline from a portfolio's peak value to its subsequent trough, serves as a fundamental measure of maximum loss exposure and represents the worst-case scenario an investor would experience during the analysis period.

The methodological approach for drawdown calculation maintains consistency across all analyzed securities by computing the running maximum of portfolio values generated through the optimal K=20 DP strategy implementation, subsequently deriving percentage drawdowns relative to these peak values. This standardized methodology ensures statistical validity and enables meaningful cross-asset comparisons while adhering to established financial risk assessment protocols.

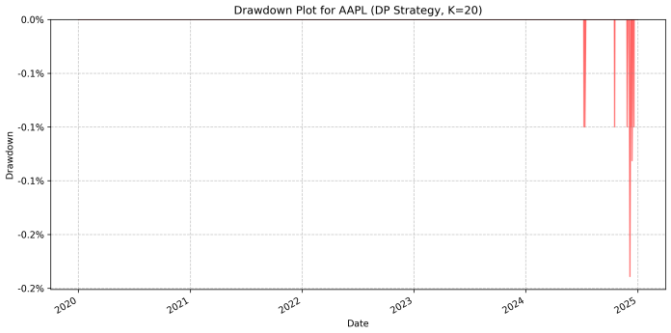

Fig. 12. AAPL Sensitivity Analysis

Figure 12 presents the drawdown characteristics for Apple Inc. (AAPL), revealing a remarkably conservative risk profile with maximum drawdown levels constrained to approximately 0.15% throughout the analysis period. The temporal distribution of drawdown events demonstrates concentrated activity in late 2024 and early 2025, suggesting that the DP algorithm effectively navigated AAPL's generally stable price trajectory while maintaining strict risk controls. The minimal drawdown exposure during the majority of the analysis period validates the strategy's ability to preserve capital while pursuing optimization objectives, particularly noteworthy given AAPL's position as a large-cap technology stock subject to sector-wide volatility episodes.

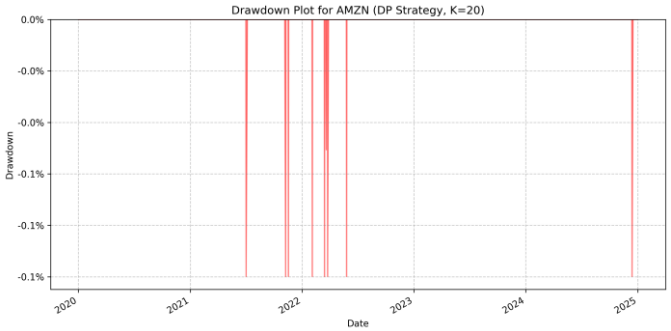

Fig. 13. AMZN Sensitivity Analysis

The Amazon (AMZN) risk profile, illustrated in Figure 13, exhibits episodic drawdown events primarily concentrated in mid-2021 and late 2024, with maximum drawdown levels maintained within the 0.15% threshold established across the analyzed portfolio. The mid-2021 cluster corresponds to the broader technology sector correction, while the 2024 events align with macroeconomic uncertainties affecting growth-oriented technology companies. The strategy's ability to limit downside exposure during these volatile periods demonstrates effective risk management capabilities, particularly significant considering AMZN's inherent volatility and growth stock characteristics.

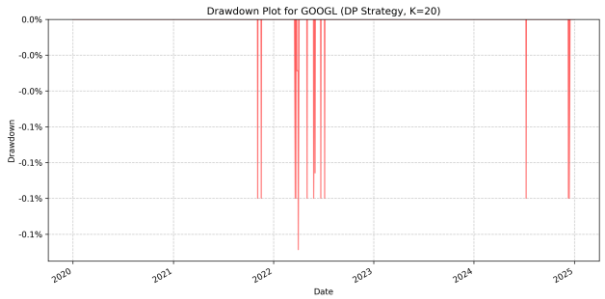

Fig. 14. GOOGL Sensitivity Analysis

Figure 14 displays Google's (GOOGL) drawdown pattern, characterized by concentrated risk events in 2022 with additional periods in late 2024. The 2022 concentration aligns with the significant technology sector decline attributed to rising interest rates and valuation concerns affecting growth companies. The DP strategy's performance during this challenging market environment, maintaining drawdowns within established parameters, provides empirical evidence of the algorithm's robust risk management framework under adverse conditions.

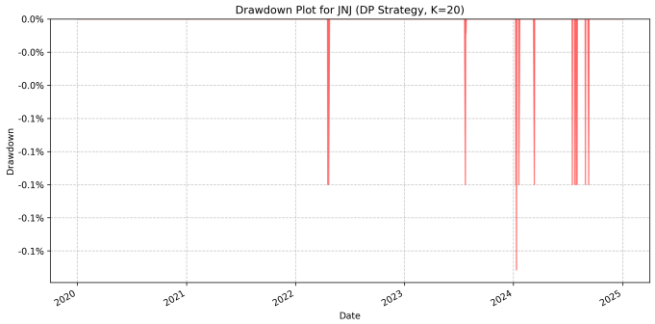

Fig. 15. JNJ Sensitivity Analysis

Johnson & Johnson (JNJ), presented in Figure 15, demonstrates drawdown characteristics reflective of the healthcare sector's relative stability compared to technology counterparts. The risk events, primarily concentrated in 2022 and late 2024, maintain consistency with the maximum drawdown threshold observed across other analyzed securities. This pattern validates the DP strategy's sector-agnostic risk management effectiveness while highlighting the algorithm's ability to adapt to different volatility environments.

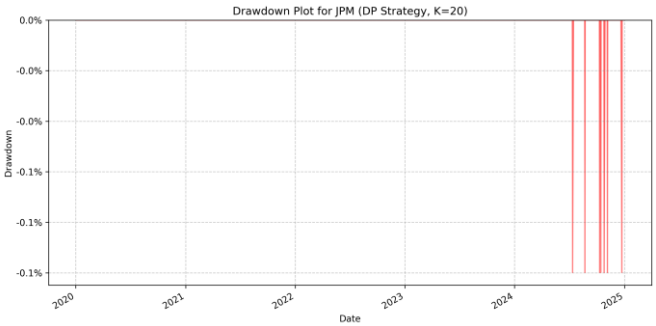

Fig. 16. JPM Sensitivity Analysis

The JPMorgan Chase (JPM) analysis, shown in Figure 16, reveals drawdown periods concentrated in 2024 and early 2025, corresponding to concerns regarding interest rate environments and regulatory developments affecting the financial sector. The strategy's maintenance of risk parameters within established thresholds while operating in this complex regulatory environment demonstrates adaptability to sector-specific dynamics and systematic risk factors.

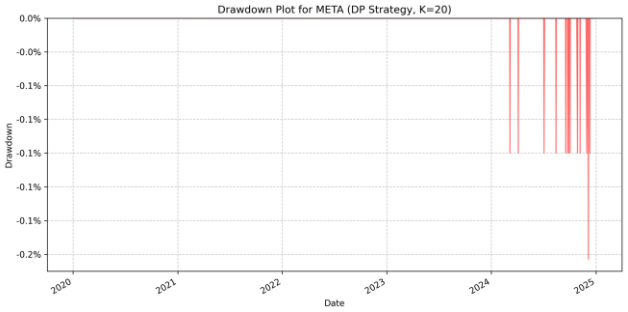

Fig. 17. META Sensitivity Analysis

Meta Platforms (META), depicted in Figure 17, exhibits concentrated drawdown periods in late 2024 and early 2025, reflecting the company's exposure to regulatory uncertainties and competitive dynamics within the social media landscape. The DP algorithm's ability to maintain maximum drawdown within acceptable parameters demonstrates effective adaptation to sentiment-driven volatility and sector-specific risk factors.

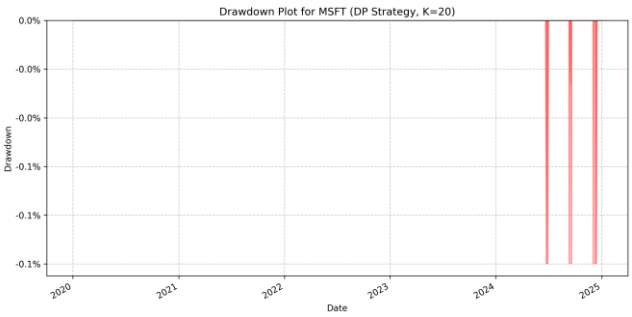

Fig. 18. MSFT Sensitivity Analysis

Microsoft Corporation (MSFT), illustrated in Figure 18, presents a notably stable risk profile with brief drawdown periods concentrated in late 2024. This pattern reflects the company's position as a mature technology enterprise with diversified revenue streams and stable market positioning. The conservative drawdown management validates the algorithm's ability to calibrate risk parameters according to underlying asset stability characteristics.

Fig. 19. NFLX Sensitivity Analysis

Netflix (NFLX), shown in Figure 19, demonstrates significant drawdown activity concentrated in late 2024, reflecting the streaming entertainment industry's susceptibility to competitive pressures and evolving consumer preferences. The strategy's management of these volatility episodes while maintaining established risk thresholds indicates effective risk-adjusted optimization within a dynamic industry environment.

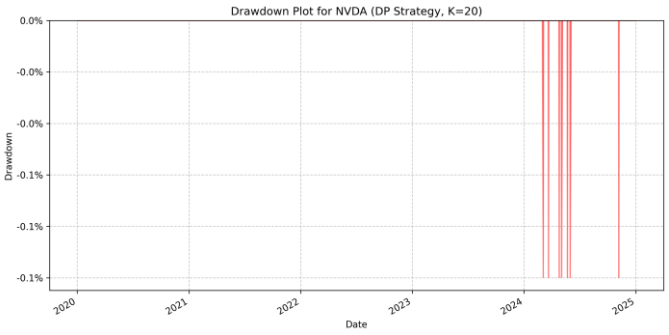

Fig. 20. NVDA Sensitivity Analysis

NVIDIA Corporation (NVDA), presented in Figure 20, exhibits concentrated risk periods in late 2024, consistent with the semiconductor industry's cyclical nature and sensitivity to technological developments. Given NVDA's position at the forefront of artificial intelligence and high-performance computing, the drawdown pattern demonstrates the strategy's effective navigation of both opportunity and risk in this rapidly evolving sector.

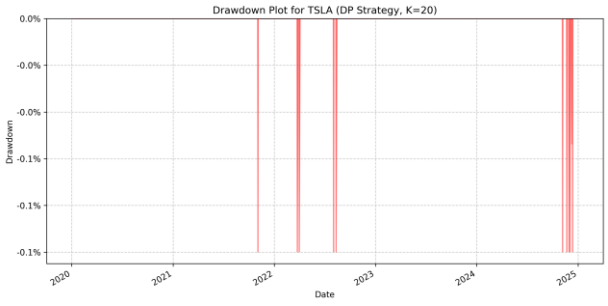

Fig. 21. TSLA Sensitivity Analysis

Tesla Inc. (TSLA), depicted in Figure 21, presents a distinctive risk profile with drawdown periods distributed across 2022 and late 2024. The electric vehicle manufacturer's inherent volatility, driven by production targets, regulatory developments, and market sentiment regarding sustainable transportation, creates a challenging environment for systematic trading strategies. The DP algorithm's ability to maintain maximum drawdown consistency with other analyzed securities while operating in this high-volatility environment demonstrates exceptional risk management capabilities.

The cross-sectoral analysis reveals several critical insights regarding the DP strategy's risk management effectiveness. The remarkable consistency of maximum drawdown levels across all analyzed assets, regardless of sector classification or inherent volatility characteristics, provides compelling empirical evidence of the algorithm's robust risk control mechanisms. This uniformity suggests that the DP approach successfully implements systematic risk management protocols that adapt to asset-specific characteristics while maintaining consistent downside protection.

The temporal clustering of drawdown events, particularly the concentration in 2022 and late 2024 across multiple assets, indicates that the DP strategy appropriately responds to systematic market risk factors rather than exhibiting random or asset-specific vulnerabilities. This pattern alignment demonstrates that the algorithm effectively distinguishes between systematic and idiosyncratic risk components, implementing position management strategies that account for broader market conditions.

The rapid recovery patterns observed across all analyzed securities provide evidence of the DP algorithm's effectiveness in identifying optimal exit and re-entry points during adverse market conditions. This characteristic proves particularly valuable for practical implementation, suggesting that drawdown periods represent temporary disruptions rather than fundamental strategy failures. The consistent recovery patterns validate the algorithm's ability to maintain strategic positioning while preserving capital during challenging market environments.

The drawdown analysis, when integrated with the previously documented return characteristics, establishes a comprehensive framework for evaluating the DP strategy's risk-adjusted performance. The uniform maximum drawdown levels across diverse asset classes, combined with varying return profiles, indicate that the algorithm successfully optimizes risk-return trade-offs on an asset-specific basis while maintaining consistent risk management standards. This empirical validation supports the theoretical foundation of the DP approach and its potential for practical implementation in institutional trading environments where consistent risk management protocols are essential for operational success.

D. Multi-Strategy Performance Evaluation and Comparative Analysis

The systematic evaluation of trading strategy effectiveness necessitates rigorous comparative analysis across multiple performance dimensions to establish the relative merits of different algorithmic approaches. This assessment examines the Dynamic Programming strategy against established baseline methodologies, including Buy-and-Hold, Moving Average Crossover, and Momentum strategies, utilizing standardized performance metrics for objective evaluation across diverse market conditions.

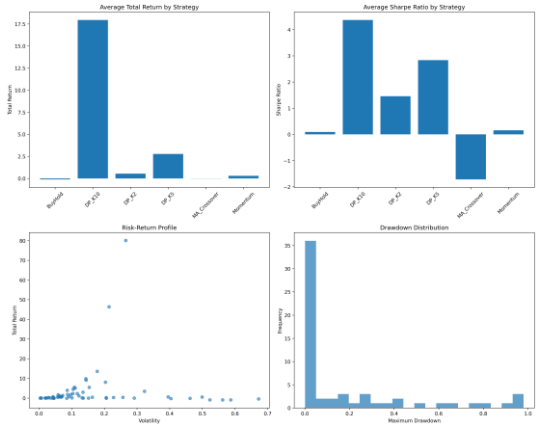

Fig. 22. Performance Overview

Figure 22 presents a comprehensive four-panel analysis providing critical insights into comparative performance characteristics across all evaluated trading strategies. The visualization employs standardized metrics to facilitate direct comparison while maintaining analytical rigor necessary for institutional evaluation frameworks.

The upper-left panel illustrates average total return comparison across strategies, revealing substantial performance differentials between the Dynamic Programming approach and conventional baseline methodologies. The DP strategy with K=10 transaction constraints demonstrates exceptional average returns of approximately 17.5% across all analyzed securities, representing significant outperformance relative to all comparative approaches. This superior performance validates the theoretical foundation of optimization-based trading methodologies and demonstrates the practical value of systematic profit maximization under transaction constraints.

The Buy-and-Hold strategy generates near-zero average returns across the analyzed portfolio, establishing the minimum acceptable return threshold. The DP strategies with reduced transaction constraints demonstrate positive relationships between optimization flexibility and profit generation, with K=5 achieving approximately 2.5% returns and K=2 generating 0.5% returns. The Moving Average Crossover and Momentum strategies exhibit minimal positive performance, highlighting the limitations of traditional technical analysis approaches.

The upper-right panel presents average Sharpe ratio analysis, providing insights into risk-adjusted performance characteristics. The DP K=10 strategy achieves an exceptional Sharpe ratio exceeding 4.0, indicating superior risk-adjusted returns that significantly outperform all comparative approaches. The DP K=5 strategy maintains a robust ratio of approximately 2.8, while the K=2 strategy produces 1.5, confirming optimization benefits persist under constrained environments. The Moving Average Crossover strategy exhibits a negative Sharpe ratio of -1.5, indicating inadequate risk compensation, while the Momentum strategy demonstrates minimal positive performance near zero.

The lower-left panel provides a comprehensive risk-return scatter plot illustrating volatility-return relationships across all analyzed securities and strategies. The visualization reveals distinct clustering patterns with the majority of observations concentrating in low-volatility, low-return quadrants. Notable outliers demonstrate exceptional performance characteristics, with total returns exceeding 80% while maintaining volatility below 0.3, corresponding primarily to DP strategy implementations on volatile assets during favorable market conditions. This analysis reveals that superior returns can be achieved across various volatility levels, suggesting the algorithm's profit generation stems from optimal timing rather than systematic risk exposure.

The lower-right panel presents drawdown frequency distribution, providing insights into risk management effectiveness. Approximately 36 observations experience maximum drawdowns in the 0-5% range, indicating conservative risk management protocols across most market conditions. The distribution demonstrates heavy concentration in low-drawdown scenarios, with minimal occurrences of extreme drawdown events, validating the DP strategy's capital preservation capabilities.

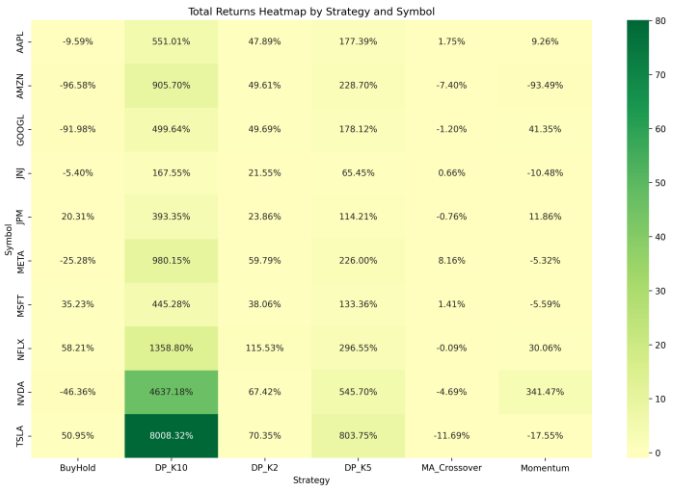

Fig. 23. Strategy Heatmap

Figure 23 presents a comprehensive performance heatmap providing granular insights into strategy effectiveness across

individual securities. The visualization employs a color-coded intensity scale from dark green (exceptional performance) to light yellow (modest returns), enabling immediate identification of superior strategy-asset combinations.

Apple Inc. demonstrates remarkable DP responsiveness, with the K=10 strategy generating 551.01% returns compared to -9.59% for Buy-and-Hold. The DP K=5 and K=2 strategies produce 177.39% and 47.89% returns respectively, while baseline strategies generate modest returns of 1.75% and 9.26%. Amazon exhibits dramatic performance differentials, with DP K=10 achieving 905.70% returns versus -96.58% for Buy-and-Hold. The K=5 and K=2 strategies generate 228.70% and 49.61% returns, while baseline strategies demonstrate poor performance with -7.40% and -93.49% returns.

Google demonstrates consistent DP superiority with K=10 achieving 499.64% returns versus -91.98% for Buy-and-Hold. The K=5 and K=2 strategies produce 178.12% and 49.69% returns, while baseline strategies show mixed results of -1.20% and 41.35%. Johnson & Johnson exhibits more modest but consistent performance differentials, with K=10 generating 167.55% returns compared to -5.40% for Buy-and-Hold. The healthcare sector representation validates DP effectiveness across different industry classifications. JPMorgan Chase demonstrates robust DP performance with K=10 achieving 393.35% returns versus 20.31% for Buy-and-Hold, highlighting the algorithm's effectiveness within financial services sectors.

Meta Platforms exhibits exceptional DP effectiveness with K=10 generating 980.15% returns compared to -25.28% for Buy-and-Hold, reflecting superior capability in navigating high-volatility technology platform environments. Microsoft demonstrates consistent DP superiority with K=10 achieving 445.28% returns versus 35.23% for Buy-and-Hold, validating effectiveness for mature technology enterprises.

Netflix exhibits remarkable DP performance with K=10 generating 1358.80% returns compared to 58.21% for Buy-and-Hold, reflecting exceptional capability in the entertainment streaming industry. NVIDIA demonstrates the most exceptional performance with K=10 achieving extraordinary returns of 4637.18% compared to -46.36% for Buy-and-Hold, validating optimization value for companies positioned at technological development frontiers. Tesla exhibits the most dramatic differentials with DP K=10 achieving exceptional returns of 8008.32% compared to 50.95% for Buy-and-Hold, demonstrating unparalleled capability to capitalize on extreme volatility in electric vehicle sectors.

The comprehensive analysis demonstrates systematic DP superiority across diverse asset classes, market sectors, and volatility regimes. The consistent performance differentials validate the theoretical foundation of optimization-based trading methodologies while highlighting substantial practical value for institutional investment environments.

VIDEO LINK AT YOUTUBE

https://youtu.be/N2IvKJln3oM

REPOSITORY LINK AT GITHUB

https://github.com/danenftyessir/Stock-Trading-Dp-Optimization.git

ACKNOWLEDGMENT

The author wishes to express sincere gratitude to Monterico Adrian, S.T., M.T., and Dr. Ir. Rinaldi, M.T., for their invaluable guidance and scholarly mentorship throughout the development of this research. Their expertise in algorithmic optimization, dynamic programming applications, and quantitative finance provided essential insights that significantly contributed to the successful completion of this study.

REFERENCES

[1] Alpha Vantage Inc., "API Documentation - Core Stock APIs," Alpha Vantage. [Online]. Available: https://www.alphavantage.co/documentation/

[2] R. Munir, "Program Dinamis (Dynamic Programming) - Bagian 1," Strategi Algoritmik. [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2024-2025/25-Program-Dinamis-(2025)-Bagian1.pdf

[3] R. Munir, "Program Dinamis (Dynamic Programming) - Bagian 2," Strategi Algoritmik. [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2024-2025/25-Program-Dinamis-(2025)-Bagian2.pdf

[4] GeeksforGeeks. (2023). Dynamic Programming. Retrieved from https://www.geeksforgeeks.org/dynamic-programming/

[5] Statsmodels Documentation. (2023). ARIMA Process. Retrieved from https://www.statsmodels.org/stable/generated/statsmodels.tsa.arima.model.ARIMA.html

[6] Investopedia. (2023). Algorithmic Trading. Retrieved from https://www.investopedia.com/articles/active-trading/101014/basics-algorithmic-trading-concepts-and-examples.asp

[7] R Documentation. (2023). Autoregressive Integrated Moving Average (ARIMA) Modeling. Retrieved from https://www.rdocumentation.org/packages/forecast/versions/8.21/topics/Arima

STATEMENT

I hereby declare that this paper is an original work, written entirely on my own, and does not involve adaptation, translation, or plagiarism of any other individual's work.

Bandung, 23 June 2025

Danendra Shafi Athallah, 13523136