An Algorithmic Framework for Deriving Optimal Asset Allocation Paths in Personalized Retirement Planning using Dynamic Programming

Muhammad Edo Raduputu Aprima - 13523096
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail: muhammadedo017@gmail.com , 13523096@std.stei.itb.ac.id

***Abstract*— Effective long-term financial planning is a critical yet challenging task for individuals. This paper addresses the limitations of traditional static asset allocation rules by developing a framework to generate personalized, time-adaptive investment strategies. Leveraging the principles of Dynamic Programming, specifically backward induction, a computational engine was implemented to solve the lifecycle savings problem as a finite-horizon optimization task. Experimental results demonstrate that the framework successfully derives non-linear asset allocation glide paths that are highly sensitive to an investor's goals and time horizon. The model reveals complex, rational strategies, such as aggressive de-risking when a target is on track and calculated risk-taking when falling behind. This work validates the application of Dynamic Programming as a robust method for creating sophisticated, personalized financial advisory tools and provides a strong foundation for future development in algorithmic financial technology.**

Keywords—Dynamic Programming, Asset Allocation, Retirement Planning, Algorithmic Framework, Glide Path.

I. Introduction

In the modern financial landscape, the responsibility for securing long-term financial stability, particularly for retirement, has increasingly shifted from institutions to individuals. The decline of traditional defined-benefit pension plans, coupled with increased life expectancy, has created a critical need for effective personal investment strategies. This paper addresses the challenge that many individuals face: relying on simplistic, static rules-of-thumb which often fail to account for personalized goals and the dynamic nature of a multi-decade investment horizon. The core problem this study tackles is the inadequacy of a one-size-fits-all strategy for a problem that is inherently personal and time-sensitive.

The primary objective of this work is to design, implement, and analyze an algorithmic framework that provides a solution to the aforementioned problem. This paper is presented as a technical report detailing the application of an algorithm strategy, with its findings supported by experimental results from a purpose-built program. The key contribution is not merely the application of a known algorithm, but the development of a complete framework that translates an individual's personal financial profile into a mathematically optimal, dynamic investment policy, often referred to as a "glide path". Through this, the work aims to bridge the gap between theoretical algorithmic concepts and practical, high-impact financial applications, communicating these findings to a wider audience.

To ensure the problem remains computationally tractable and the analysis focused, the scope of this study is defined with several limitations. The investment model considers two primary asset classes (an aggressive and a conservative asset), assumes constant long-term market parameters, and does not factor in complex variables such as taxes or transaction costs. The framework exclusively models the wealth accumulation phase of retirement planning. The remainder of this paper is organized to detail this work systematically.

II. Theoritical Foundation

A. Dynamic Programming (DP)

Dynamic Programming is a mathematical optimization method used for solving complex problems by breaking them down into a collection of simpler, overlapping subproblems.

For problems with a finite time horizon, such as the retirement planning model in this paper, a specific DP technique called Backward Induction is employed. The process begins at the final time step, T, where the outcome is known. It then iterates backward in time, step by step (t = T-1, T-2, ..., 0). At each step, it solves for the optimal action by using the pre-computed optimal values from the subsequent step, t+1. This paper's processing engine is a direct implementation of this backward induction method to find the optimal policy.

B. Modern Portfolio Theory (MPT) and Asset allocation

Modern Portfolio Theory, pioneered by Nobel laureate Harry Markowitz, is a foundational investment theory that provides a mathematical framework for assembling a portfolio of assets [1]. The core idea of MPT is that an investor can construct a portfolio that maximizes the expected return for a given level of risk. This is achieved through careful asset allocation and diversification. The key concepts include:

- Return: The expected gain or loss on an investment over a period.
- Risk: Typically measured as the volatility or standard deviation of an asset's returns.
- The Risk-Return Trade-off: A fundamental principle stating that higher potential returns are associated with higher risk.
- Diversification: The practice of spreading investments among various financial instruments to reduce overall risk.

The model in this paper navigates the risk-return trade-off by creating a portfolio composed of two distinct asset classes: a high-risk, high-return "aggressive asset" (representing stocks) and a low-risk, low-return "conservative asset" (representing bonds). The algorithm's task at each time step is to find the optimal allocation percentage between these two assets.

C. Lifecycle Investing and Glide Paths

Lifecycle Investing is a financial theory that posits an individual's asset allocation should change systematically over the course of their life [2]. The central principle is that an investor's ability and willingness to take on risk decreases as they approach retirement.

- Young Investors: With a long time horizon, they can afford to take on more risk (a higher allocation to stocks) to maximize long-term capital growth, as they have more time to recover from market downturns.
- Older Investors: As they near retirement, their priority shifts from growth to capital preservation. They should therefore reduce their allocation to risky assets to protect their accumulated wealth.

This pre-planned, gradual reduction in portfolio risk over time is known as a "Glide Path." It is a common strategy implemented in financial products like Target-Date Funds.

While traditional glide paths are often based on static, predetermined rules (e.g., the "100 minus age" rule), the key contribution of this paper is to utilize Dynamic Programming to *derive* a personalized and mathematically optimal glide path. The generated path is not based on a general rule, but is instead tailored to the investor's specific age, financial goals, and savings capacity

III. Problem Modelling

The problem is structured as a finite-horizon dynamic optimization task, which is well-suited for a Dynamic Programming approach. To manage the components of the solution, a three-part framework was designed.

A. Framework Architecture

The implemented solution is architected as a modular framework consisting of three main components:

1) Input Module

This component is responsible for capturing the investor's unique financial profile. It gathers essential parameters such as current age, desired retirement age, initial wealth, monthly investment capacity, and the final wealth target. This ensures the subsequent optimization is personalized to the user's specific circumstances and goals.

2) Processing Engine

This is the core of the framework, containing the Dynamic Programming engine. It takes the parameters from the Input Module and executes the backward induction algorithm to solve for the optimal investment policy across the entire time horizon.

3) Output Module

This component receives the raw policy data from the Processing Engine and transforms it into human-interpretable visualizations. It generates the glide path charts and stochastic wealth projections, translating the complex numerical output into actionable financial insights.

B. Dynamic Programming Model

The lifecycle investment problem is formally modeled as a finite-horizon dynamic optimization task, which is a requirement for this technical report that applies an algorithm strategy. The model is defined by the following key components, each described in detail below.

- State Space (S) The State Space defines all possible situations an investor can be in at any given time. A state, denoted as *s_t*, must encapsulate all information necessary to make a future decision without needing to know the history of how that state was reached. In this model, a state is sufficiently described by a tuple containing the current time and the investor's accumulated wealth. Thus, the state is formally represented as *s_t = (t, W_t),* where t represents the current year within the total investment horizon (from year 0 to the final year T), and *W_t* represents the total monetary value of the portfolio at the start of that year.
- Action Space (A) The Action Space consists of all possible decisions that can be made at any given state. For this financial planning problem, the action *alpha_t* corresponds to the core investment decision made in year t. It is defined as the fraction, or percentage, of the total portfolio to be allocated to the high-risk, high-return aggressive asset. The value of *alpha*_t is continuous between 0 and 1, inclusive. Consequently, the remaining portion of the portfolio, calculated as *(1 - alpha_t),* is automatically allocated to the low-risk, low-return conservative asset. The set of all possible *alpha_t* values forms the action space available to the decision-making policy.
- State Transition Equation The State Transition Equation describes the system's dynamics, dictating how the current state evolves into a future state based on the action taken. This function calculates the wealth at the beginning of the next year, W_{t+1}, based on the wealth from the current year, W_t. The model assumes that the total annual contribution, C, is added to the principal at the start of the year. This new, larger

principal is then invested for one year according to the chosen allocation alpha_t. The resulting portfolio grows based on the weighted average of the expected annual returns of the aggressive asset (r_s) and the conservative asset (r_c). This entire process defines how the wealth component of the state transitions from one year to the next.

- Objective and Terminal Utility Function Finally, the Objective Function defines the ultimate goal of the optimization process, which is guided by a Terminal Utility Function. The overall objective is to find an optimal policy, which is a complete plan of actions for all possible states, that maximizes the expected utility of the final wealth at the retirement year T. To achieve this, a utility function, U(W_T), is assigned to the terminal wealth states. This function is designed to reflect the user's goal. It assigns a value equal to the final wealth, W_T, if the user's wealth target is met or exceeded. However, to create a strong incentive for the algorithm to succeed, a significant penalty is applied if the final wealth is below the target. This penalty is proportional to the size of the deficit, ensuring that the algorithm actively avoids strategies that are projected to fail.

IV. Implementation

The program was developed as a command-line tool to prioritize algorithmic logic and efficiency over user interface complexity.

A. Technology Stack

The framework was implemented entirely in Python, chosen for its robust ecosystem of scientific computing libraries. The following key libraries were instrumental in the development:

- NumPy: This library was fundamental for all numerical operations. Its high-performance *ndarray* objects were used to implement the core data structures of the model, including the state and action grids, as well as the multi-dimensional *value_table* and *policy_table*. Its vectorization capabilities were crucial for efficient computation.
- Matplotlib: All static visualizations, including the glide path charts and wealth projections, were generated using the Matplotlib library. It provided fine-grained control to produce publication-quality graphs that were saved as image files for inclusion in this report.

B. Implementation of the Dynamic Programming Engine

The core of the program is the Dynamic Programming (DP) engine, which implements the backward induction algorithm to solve the lifecycle investment problem.

First, the continuous state space (investor's wealth) and action space (asset allocation percentage) were discretized into finite grids using NumPy's *linspace* function. This step is essential to transform the problem into a computationally tractable format. The main data structures, the *policy_table* and *value_table*, were initialized as two-dimensional NumPy arrays, with dimensions corresponding to the discretized wealth levels and time steps (years).

The engine's primary logic resides in a nested loop structure that iterates backward in time, from the final year T-1 down to the initial year t=0. For each time step and for each discrete wealth level in the grid, the engine evaluates every possible allocation action. A key challenge in this step is that a given action may result in a future wealth state that falls between the discrete points on the wealth grid. This was resolved through linear interpolation, implemented efficiently using NumPy's *interp* function. This function estimates the value of the future state by interpolating from the already-computed values in the *value_table* of the subsequent time step (t+1). The action that maximizes this interpolated future value is then stored as the optimal policy for the current state.

C. Data Generation and Simulation Modules

Once the backward induction loop completes and the *policy_table* is fully populated, two subsequent processes are executed to generate the final results for analysis.

1) Deterministic Path Extraction

To visualize a single, representative glide path, a forward simulation is performed. Starting with the user's initial wealth, this module iteratively determines the optimal allocation for the current state by interpolating from the policy_table, calculates the resulting wealth in the next year using the mean expected returns, and repeats the process for the entire time horizon. This generates the primary glide path and deterministic wealth projection.

2) Stochastic Monte Carlo Simulation

To provide a more realistic view of potential outcomes and risk, a Monte Carlo module was implemented. This module takes the optimal policy derived from the DP engine and runs hundreds of simulations. In each simulation, the annual asset returns are not fixed but are drawn randomly from a normal distribution defined by the assumed mean return and volatility. The aggregate results of these simulations are used to generate the probabilistic wealth projection, including the median outcome and the confidence interval cone

V. Result and Analysis

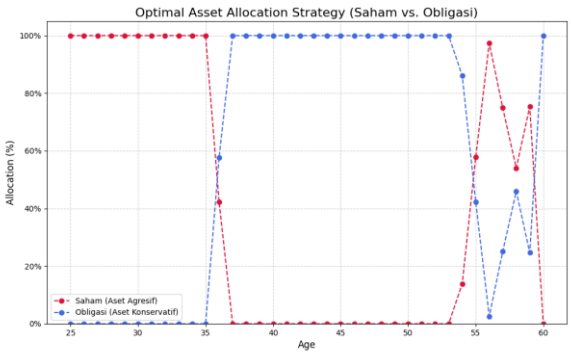
The primary goal is to analyze the optimal asset allocation strategies derived from the Dynamic Programming model and to understand the model’s behavior user various user-defined scenarios. This paper, presented as a technical report, validates the application of this algorithm through these quantitative experiments.

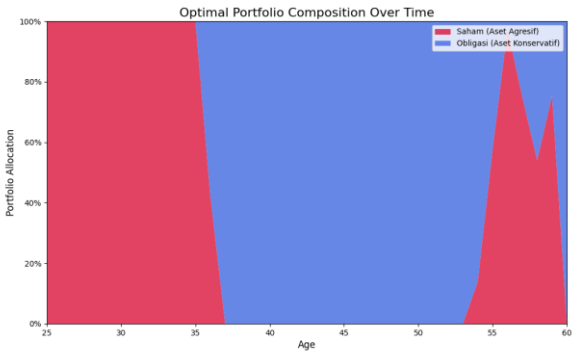
A. Base Case Scenario Analysis

To establish a baseline, a primary scenario was simulated using a typical profile for a young investor.

```
current_age: 25
retirement_age: 60
initial_wealth: 50000000
target_wealth: 5000000000
annual_contribution: 36000000
time_horizon: 35
aggressive_asset: {'return': 0.12, 'volatility': 0.18}
conservative_asset: {'return': 0.06, 'volatility': 0.05}
n_simulations: 1000
```

The principal output from the framework is the optimal asset allocation glide path, which dictates the percentage of the portfolio to be allocated to the aggressive assets (stocks) each year. This path is illustrated in Fig. 2 and Fig. 3.



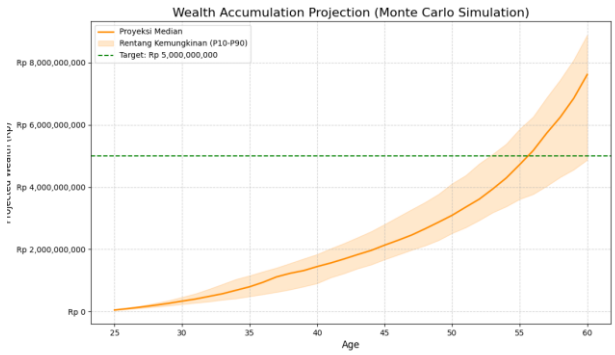


The resulting glide path is non-linear and reveals several distinct strategic phases. The initial phase, from age 25 to 35, is characterized by a 100% allocation to the aggressive asset. This strategy is driven by the long time horizon, where the model prioritizes maximizing expected returns to accelerate wealth accumulation, understanding that there is ample time to recover from potential short-term market volatility.

A significant strategic shift occurs around age 36, where the allocation drops sharply to 0%. This de-risking event indicates that the model projects the accumulated wealth is now on a sufficient trajectory to meet the final target through continued contributions and low-risk growth alone. The model's priority shifts from growth maximization to capital preservation, entering a "cruise control" phase that lasts until approximately age 53.

Perhaps the most counter-intuitive yet insightful behavior is the re-emergence of high-risk allocation in the final years before retirement. This is the model's rational response to a new projection that the safe, low-growth path is no longer sufficient to meet the specific wealth target. Faced with a choice between the certainty of falling short and a small probability of success by taking on high risk, the utility-maximizing algorithm chooses the latter. This "end-game" volatility underscores the model's dynamic and state-aware nature.

To understand the potential outcome of following this policy, the framework performs a Monte Carlo simulation (1000 trials). The results, illustrating the range of possible final wealth, are visualized in Fig. 4.



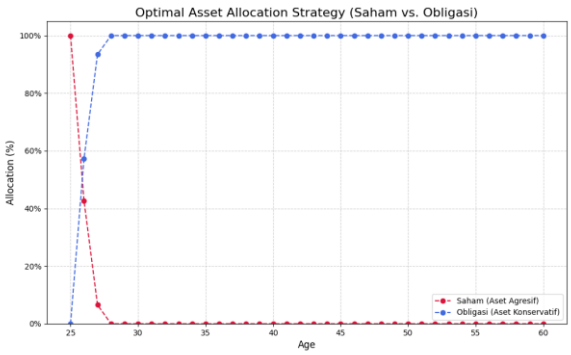
The median projected outcome (the 50th percentile) successfully meets the retirement target. The shaded "cone of uncertainty" represents the range between the 10th and 90th percentile outcomes, illustrating the inherent market risks associated with the strategy.

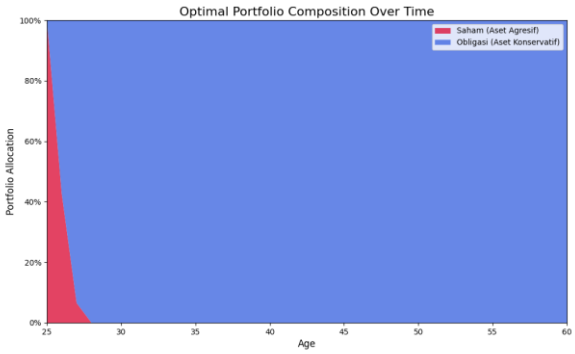
B. Comparative analysis of Scenarios

To test the framework's adaptability and robustness, several other scenarios were simulated by altering key input parameters. The result demonstrated the model's ability ti generate unique strategies tailored to different financial situations.

1) The Low-Stress Scenario

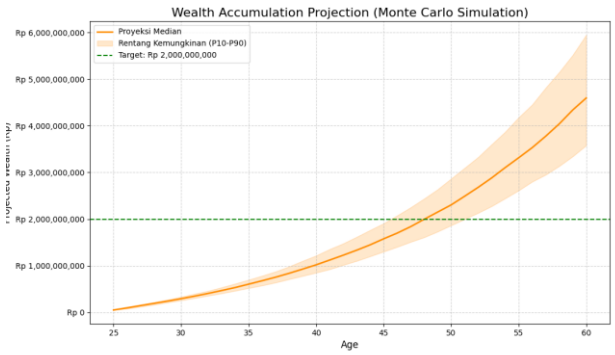
In the first test case, the retirement target was lowered significantly to IDR 2,000,000,000 to model a scenario where the financial goal much less ambitious and easier to achieve.





The strategy derived for this scenario exhibits a different pattern from the base case. The model recommends a 100% aggressive allocation for only the first year. A rapid de-risking process immediately follows, with the allocation to aggressive assets dropping to near zero by age 28. For the remainder of the 32-year investment horizon, the strategy remains entirely conservative (100% in bonds).

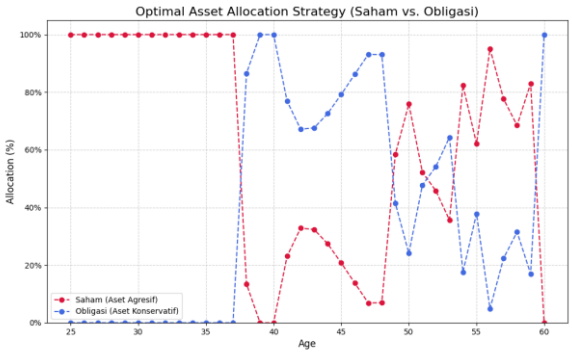
This result strongly confirms the initial hypothesis. The volatile "end-game" behavior observed in the base case is completely absent. The framework correctly identifies that the combination of initial capital, steady contributions, and a few years of early growth is more than sufficient to meet the modest target. Therefore, its logical priority shifts from growth maximization to aggressive capital preservation to eliminate market risk as quickly as possible.

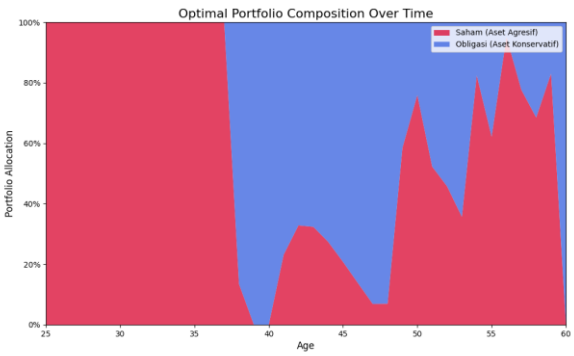


As illustrated, the median wealth projection (orange line) not only meets but vastly exceeds the IDR 2 billion target, ending closer to IDR 4.7 billion. Critically, even the 10th percentile outcome (the bottom edge of the shaded area), representing a relatively poor sequence of market returns, comfortably surpasses the target line by mid-life. This demonstrates the model's rationality: it correctly avoids taking on unnecessary risk when the stated goal is already well secured.

2) The High-Ambition Scenario

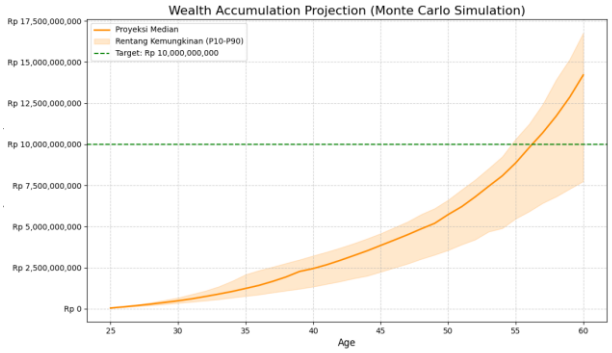
To further explore the model's decision-making under pressure, a third scenario was designed with a highly ambitious financial goal. The retirement target was increased to IDR 10,000,000,000, with a slightly higher monthly contribution of IDR 5,000,000 to reflect a more aggressive savings plan. The resulting optimal strategy reveals a complex, multi-phased approach that differs significantly from prior cases.





The initial strategy, from age 25 to 37, is a prolonged period of 100% aggressive allocation. This extended growth phase is the model's logical response to the daunting target. However, a surprising de-risking event occurs around age 38, where the model temporarily shifts to a zero-risk portfolio. This suggests that the combination of a long, high-growth period and increased contributions led the model to momentarily project that it was on a viable path.

Unlike the "Low-Stress Scenario," this conservative phase is short-lived. From age 40 onwards, the framework determines that the safe path is insufficient and begins to gradually re-introduce risk, creating a volatile period of calculated allocations. This culminates in an early onset of the "end-game" high-risk behavior starting around age 49. The severe fluctuations in this final decade indicate the model is constantly re-evaluating its slim chances of success, taking on significant risk as a necessity to bridge the large remaining gap to the 10 billion target.



While the median projection (the 50th percentile) successfully surpasses the ambitious target, the cone of uncertainty reveals a crucial detail: the 10th percentile outcome (the lower bound of the shaded area) finishes just below the IDR 10 billion target line. This slim margin for error is precisely why the algorithm maintains a high-risk stance for the majority of the investment horizon. It calculates that a conservative strategy would lead to a near-certain failure to meet the goal, thus making a volatile, high-risk path the mathematically optimal choice. This scenario effectively showcases the framework's ability to craft complex, dynamic strategies when faced with challenging, high-stakes objectives.

VI. Conclusion

This paper has successfully designed, implemented, and analyzed an algorithmic framework for generating personalized, multi-stage investment strategies for retirement planning. Addressing the limitations of static, one-size-fits-all financial advice, this work framed the lifecycle savings problem as a finite-horizon dynamic optimization task. By applying the principles of Dynamic Programming, specifically the backward induction method, the developed engine is capable of computing a mathematically optimal asset allocation policy tailored to an individual's unique financial profile, goals, and time horizon.

The experimental results demonstrate the framework's effectiveness and adaptability. The model successfully derived non-linear, intuitive, and sometimes counter-intuitive asset allocation glide paths. The analysis of various scenarios confirmed that the generated strategies are highly sensitive to input parameters; the model rationally advises more aggressive strategies for ambitious goals or shorter time horizons, and prioritizes capital preservation when goals are more easily attainable. A key insight is the model's "end-game" behavior, where it rationally takes on significant risk to avoid a certain failure to meet a target, highlighting its utility not just as a prescriptive tool, but as a powerful diagnostic system for financial plans. The primary contribution of this work is the validation that a DP-based framework can serve as a robust and sophisticated engine for personalized financial technology applications.

https://github.com/poetoeee/Stima_Paper_13523096

VII. Suggestion

While the current framework provides a robust proof-of-concept, several avenues exist for future enhancement to increase its realism and scope. A primary extension would involve expanding the model from two asset classes to multiple categories such as international stocks, real estate investment trusts (REITs), or commodities, although this would require managing a significantly more complex, multi-dimensional action space. Further sophistication could be achieved by incorporating stochastic market models where asset returns and volatilities are time-varying, moving beyond the current assumption of constant long-term averages. To improve its practical applicability, the framework could also be modified to include real-world financial frictions, such as the impact of capital gains taxes and transaction costs on net returns. Finally, a substantial future development would involve modeling the decumulation (withdrawal) phase of retirement, using a similar DP approach to optimize spending strategies and ensure the longevity of the accumulated wealth.

Acknowledgment

All praise to Allah Subhanahu wa Ta'ala, for His blessing, which have guided me to complete this paper succesfully. I would also like to thank to lecturer for IF2211 Algorithm Strategy, Dr. Ir. Rinaldi Munir, M.T. for the motivation, knowledge, and guidance throughout the course. I hope that this paper will provide valuable insight and benefits to its readers.

References

[1] H. Markowitz, "Portfolio Selection," The Journal of Finance, vol. 7, no. 1, pp. 77-91, Mar. 1952.

[2] F. Modigliani and R. Brumberg, "Utility Analysis and the Consumption Function: An Interpretation of Cross-Section Data," in Post-Keynesian Economics, K. Kurihara, Ed. New Brunswick, NJ: Rutgers University Press, 1954, pp. 388-436.

[3] R. Bellman, Dynamic Programming. Princeton, NJ: Princeton University Press, 1957.

Statement

I hereby declare that this paper I have written is my own work, not a paraphrase or translation of someone else's paper, and not plagiarism.

Bandung, 24 Juni 2025

Muhammad Edo Raduputu Aprima
13523096