**📈 Financial Forecasting – Recursive Algorithm Analysis**

This document provides the theoretical explanation and performance analysis of a recursive approach for financial forecasting, as part of the assignment requirements.

**1. 🧠 Understanding Recursive Algorithms**

**What is Recursion?**

**Recursion** is a technique in which a function calls itself to solve smaller instances of the same problem until a defined base condition is met. It is widely used in problems involving repetition or that can be divided into simpler sub-problems.

**Why Use Recursion in Financial Forecasting?**

Financial forecasting often involves **compound calculations**, where each future value is dependent on the previous value and a constant **growth rate**. This recursive relationship makes recursion a natural fit for such problems.

**Recurrence Formula:**

FV(n)=FV(n−1)×(1+r)FV(n) = FV(n - 1) \times (1 + r)FV(n)=FV(n−1)×(1+r)

Where:

* **FV(n)** = Future value after n years
* **r** = Annual growth rate
* **FV(0)** = Initial investment or base value

**2. ⏱️ Time and Space Complexity Analysis**

In a basic recursive implementation of forecasting:

* **Time Complexity:** O(n)  
  Each year involves one recursive call.
* **Space Complexity:** O(n)  
  Due to the recursive function call stack (one for each year).

This means both time and memory usage grow **linearly** with the number of years being forecasted.

**3. ⚙️ Optimization Strategies**

To improve performance, especially when dealing with large inputs, the following optimizations are recommended:

**a. Iterative Approach**

An **iterative solution** uses a loop to perform the same computation without recursion. It reduces the overhead of function calls and uses **constant memory**.

Suitable for large-scale forecasting where recursion might lead to stack overflow.

**b. Memoization**

If the recursive function is called multiple times with the same inputs (e.g., in simulations or batch forecasts), **memoization** can cache results of previous computations. This significantly reduces redundant calculations and improves efficiency.

**✅ Summary**

* Recursion provides a simple and readable way to model future value calculations.
* For small datasets or conceptual clarity, recursion works well.
* For larger datasets or production-level tools, it is better to use **iteration or memoization** to enhance performance and scalability.