# Projet De Mathématiques Financières

Erkin Tunc Boya | Rania Seddouki | Nasrallah Layada

Groupe 13

Mars - Avril 2025

## Table des matières

# 1 Introduction

## 1.1 Contexte du projet

Dans le cadre de ce projet de mathématiques financières, nous nous intéressons à l'optimisation d'une stratégie d'investissement sur une période de temps donnée. Un investisseur dispose d'un capital initial qu'il peut placer de deux manières :

— à un taux d'intérêt de base constant appliqué à chaque période individuelle ;

— via des produits financiers spécifiques offrant des rendements plus intéressants, mais sur des intervalles plus longs.

Ces options sont limitées dans le temps et ne sont pas toujours disponibles simultanément. Cela soulève naturellement la question du meilleur enchaînement possible de placements pour maximiser le capital final.

## 1.2 Objectif et approche adoptée

L'objectif principal est de déterminer une politique d'investissement optimale permettant de maximiser le capital final à l'issue de la période d'étude.

Pour résoudre ce problème, nous utilisons :

— une **modélisation mathématique** du problème sous forme d'un graphe orienté représentant les différentes possibilités de placements ;

— une **approche algorithmique** basée sur la programmation dynamique, évitant l'énumération exhaustive de toutes les possibilités, qui serait inefficace dès que l'horizon d'investissement devient important.

Dans la suite du rapport, nous allons présenter le modèle, illustrer son fonctionnement à travers un exemple concret, détailler les étapes algorithmiques et conclure par une analyse des résultats obtenus.

# 2 Modélisation mathématique

## 2.1 Hypothèses et notations

L'horizon d'étude est une période discrète notée $T = [0, n]$, divisée en $n$ unités de temps. Un investisseur peut placer son capital de deux manières :

— **Placement de base** : À chaque période $[t, t+1]$, un taux d'intérêt fixe $\tau_0$ est appliqué. Il est disponible en continu sur toute la période $T$.

— **Placements spécifiques** : On dispose de $m$ opportunités de placement $P_k = (d_k, f_k, \tau_k)$ avec :

  — $d_k$ : date de début du placement $k$ ;

  — $f_k$ : date de fin du placement $k$ ;

  — $\tau_k$ : taux d'intérêt applicable sur l'intervalle $[d_k, f_k]$.

Chaque placement $P_k$ ne peut être utilisé qu'une seule fois et uniquement sur l'intervalle $[d_k, f_k]$.

## 2.2 Représentation par un graphe orienté

Pour modéliser ce problème, nous utilisons un graphe orienté noté $D = (V, A)$ :

— L'ensemble des **sommets** $V$ correspond aux dates $t \in \{0, 1, ..., n\}$.

— L'ensemble des **arcs** $A$ contient deux types de transitions :

— Les arcs de base $(t, t+1)$ avec un coefficient $c(t, t+1) = 1 + \tau_0$ ;
— Les arcs spécifiques $(d_k, f_k)$ correspondant aux produits d'investissement, avec $c(d_k, f_k) = 1 + \tau_k$.

Chaque chemin dans ce graphe représente une stratégie d'investissement possible.

### 2.3 Chemins et stratégie de placement

Un **chemin** $P$ du sommet 0 au sommet $n$ dans le graphe $D$ représente une séquence de décisions d'investissement. À chaque étape, le capital est multiplié par le coefficient de l'arc emprunté.

On définit :

$$C(P) = \prod_{a \in P} c(a)$$

comme le **coefficient multiplicatif total** associé à un chemin $P$.

L'objectif est de déterminer le chemin $P^*$ tel que :

$$C(P^*) = \max_{P \in \mathcal{P}(0,n)} C(P)$$

où $\mathcal{P}(0, n)$ est l'ensemble des chemins allant de 0 à $n$ dans le graphe.

Pour cela, on introduit une fonction de programmation dynamique :

$$\text{Coef}(t) = \max_{P \in \mathcal{P}(0,t)} C(P)$$

avec l'initialisation $\text{Coef}(0) = 1$. Cette fonction permet de calculer de manière séquentielle la valeur maximale atteignable à chaque date $t$.

### 2.4 Cardinalité des chemins dans le graphe

Le nombre de chemins possibles de 0 à $n$ dépend de la structure du graphe $D$.

**Cas général (avec seulement les arcs $(t, t+1)$)** Si l'on ne considère que les placements de base (un seul arc $(t, t+1)$ entre deux dates consécutives), alors il n'existe qu'un seul chemin possible de 0 à $n$.

**Cas complet (tous les arcs $(t', t)$ possibles pour $t' < t$)** Si le graphe contient tous les arcs $(t', t)$ avec $0 \leq t' < t \leq n$, alors le nombre total de chemins distincts de 0 à $n$ est :

$$|\mathcal{P}(0, n)| = 2^{n-1}$$

**Exemple : $n = 4$** Dans ce cas, on a :

$$|\mathcal{P}(0, 4)| = 2^{4-1} = 8$$

Ce résultat s'explique par le fait que, à chaque étape intermédiaire $t \in \{1, 2, ..., n-1\}$, on a le choix : soit on continue avec un arc $(t, t+1)$, soit on saute directement à une date ultérieure (grâce à un arc $(t, t+k)$). Cela génère $2^{n-1}$ combinaisons possibles de décisions.

**Remarque** Si le graphe est partiellement connecté (par exemple, uniquement certains arcs $(d_k, f_k)$ sont disponibles), alors le nombre de chemins doit être calculé au cas par cas en fonction des données fournies.

# 3 Étude d'un exemple concret

## 3.1 Données du problème

Considérons une période d'étude de $n = 7$ unités de temps, avec un taux d'intérêt de base $\tau_0 = 0.009$ (0.9% par période). L'investisseur dispose de cinq produits d'investissement spécifiques, caractérisés comme suit :

| Produit $k$ | Date début $d_k$ | Date fin $f_k$ | Taux $\tau_k$ (%) |
|---|---|---|---|
| 1 | 0 | 2 | 1.9 |
| 2 | 1 | 3 | 2.0 |
| 3 | 2 | 5 | 3.0 |
| 4 | 3 | 6 | 3.0 |
| 5 | 4 | 7 | 2.8 |

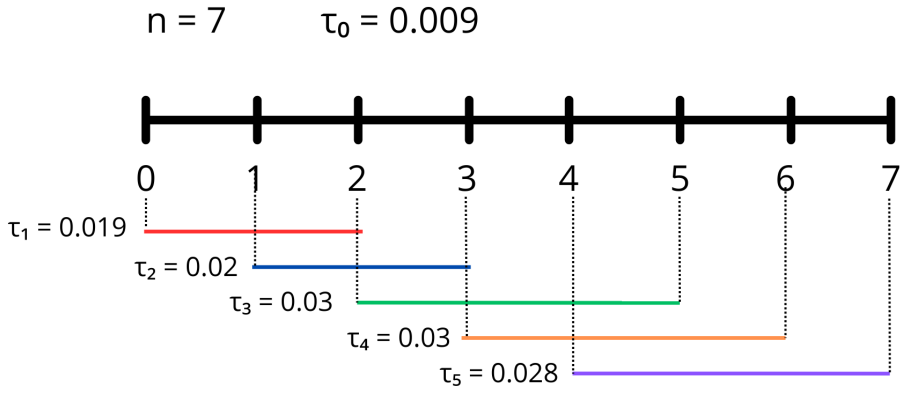


FIGURE 1 – Table d'orienté $D$ représentant les arcs $(t, t+1)$ et $(d_k, f_k)$ pour l'exemple avec $n = 7$.

## 3.2 Construction du graphe

On construit un graphe orienté $D = (V, A)$ tel que :

— Les sommets $V = \{0, 1, ..., 7\}$ représentent les dates.

— Les arcs $(t, t+1)$ avec un taux $1 + \tau_0 = 1.009$ représentent les placements de base.

— Les arcs spécifiques $(d_k, f_k)$ avec coefficient $1 + \tau_k$ représentent les produits d'investissement disponibles.

Le graphe contient donc :

— Arcs de base : (0,1), (1,2), ..., (6,7)

— Arcs spécifiques : (0,2), (1,3), (2,5), (3,6), (4,7)

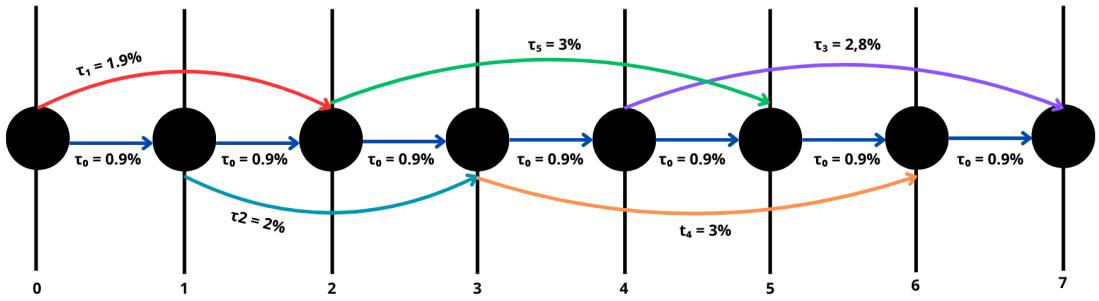


FIGURE 2 – Graphe orienté $D$ représentant les arcs $(t, t+1)$ et $(d_k, f_k)$ pour l'exemple avec $n = 7$.

### 3.2.1 Énumération des chemins de P(0, 7).

Voici une liste des chemins possibles pour aller de la date 0 à la date 7, en ayant utilisé les arcs disponibles, y compris ceux par défaut ("t->t+1") :

— Chemin 1 : $(0,1) \to (1,2) \to (2,3) \to (3,4) \to (4,5) \to (5,6) \to (6,7)$
$\Rightarrow C_1 \approx 1.0647$

— Chemin 2 : $(0,1) \to (1,2) \to (2,3) \to (3,4) \to (4,7)$
$\Rightarrow C_2 \approx 1.0655$

— Chemin 3 : $(0,1) \to (1,2) \to (2,3) \to (3,6) \to (6,7)$
$\Rightarrow C_3 \approx 1.0676$

— Chemin 4 : $(0,1) \to (1,2) \to (2,5) \to (5,6) \to (6,7)$
$\Rightarrow C_4 \approx 1.0676$

— Chemin 5 : $(0,1) \to (1,3) \to (3,4) \to (4,5) \to (5,6) \to (6,7)$
$\Rightarrow C_5 \approx 1.0667$

— Chemin 6 : $(0,1) \to (1,3) \to (3,4) \to (4,7)$
$\Rightarrow C_6 \approx 1.0675$

— Chemin 7 : $(0,1) \to (1,3) \to (3,6) \to (6,7)$
$\Rightarrow \mathbf{C_7 \approx 1.0696}$ **(chemin optimal)**

— Chemin 8 : $(0,2) \to (2,3) \to (3,4) \to (4,5) \to (5,6) \to (6,7)$
$\Rightarrow C_8 \approx 1.0657$

— Chemin 9 : $(0,2) \to (2,3) \to (3,4) \to (4,7)$
$\Rightarrow C_9 \approx 1.0665$

— Chemin 10 : $(0,2) \to (2,3) \to (3,6) \to (6,7)$
$\Rightarrow C_{10} \approx 1.0685$

— Chemin 11 : $(0,2) \to (2,5) \to (5,6) \to (6,7)$
$\Rightarrow C_{11} \approx 1.0685$

Pour trouver ces routes L'algorithmique de *enumerate_paths* qui situe en `utils.py` est utilisé.

### 3.3 Cardinalité des chemins pour $n = 7$

Dans cet exemple, nous avons $n = 7$. Si l'on supposait que le graphe contient tous les arcs possibles $(t', t)$ avec $0 \leq t' < t \leq 7$, alors selon la formule théorique vue précédemment, le nombre total de chemins distincts de 0 à 7 serait :

$$|\mathcal{P}(0,7)| = 2^{7-1} = 64$$

Cependant, dans notre cas particulier, seuls certains arcs $(d_k, f_k)$ sont disponibles, en plus des arcs de base $(t, t+1)$. Cela signifie que le graphe est partiellement connecté, et donc le nombre réel de chemins est inférieur à 64.

Une énumération explicite ou une exploration du graphe permettrait de déterminer ce nombre avec précision. Cela justifie encore une fois l'intérêt d'une approche algorithmique dynamique plutôt que l'énumération brute, qui devient rapidement coûteuse.

### 3.4 Conclusion sur l'exemple

Le chemin $(0,2) \to (2,5) \to (5,6) \to (6,7)$ représente une stratégie optimale dans cet exemple. Il combine deux produits à taux élevé (1.9%, 3.0%) et termine avec des placements de base. Il permet de maximiser le coefficient multiplicatif total du capital, atteignant environ 1.069 — supérieur à toutes les stratégies fondées uniquement sur le taux de base.

Cet exemple valide la nécessité d'utiliser une approche dynamique pour détecter efficacement la meilleure combinaison.

## 4 Recherche de la solution optimale

### 4.1 Pourquoi une approche dynamique ?

Comme le nombre de chemins possibles de 0 à $n$ peut croître de manière exponentielle, une approche par énumération brute devient rapidement inefficace. Il est donc essentiel d'utiliser une stratégie algorithmique plus performante. Nous adoptons une **programmation dynamique**, qui permet de calculer, de manière itérative, la meilleure valeur atteignable à chaque date $t$ en évitant les répétitions inutiles.

### 4.2 Définition de la fonction $Coef(t)$

Pour chaque date $t \in \{0, 1, ..., n\}$, on définit $Coef(t)$ comme le **coefficient multiplicatif maximal du capital** que l'on peut obtenir à la date $t$.

Les deux premières valeurs sont connues par hypothèse :

$$Coef(0) = 1, \quad Coef(1) = Coef(0) \cdot (1 + \tau_0)$$

Ensuite, pour chaque $t \geq 2$, la valeur de $Coef(t)$ est calculée récursivement selon la formule suivante :

$$Coef(t) = \max \left[ Coef(t-1) \cdot (1 + \tau_0), \max_{k \in N^-(t)} (Coef(d_k) \cdot (1 + \tau_k)) \right]$$

où :

— $\tau_0$ est le taux d'intérêt de base par période,

— $\tau_k$ est le taux associé au placement spécifique $P_k$,

— $d_k$ est la date de début du placement $P_k$,

— $f_k$ est la date de fin du placement $P_k$,

— $N^-(t)$ est l'ensemble des indices $k$ tels que $f_k = t$ :

$$N^-(t) = \{k \in \{1, 2, ..., m\} \mid f_k = t\}$$

### 4.3 Interprétation de la formule

À chaque instant $t$, deux choix s'offrent à l'investisseur :

— continuer avec un placement de base : capital multiplié par $(1 + \tau_0)$,

— profiter d'un produit spécifique $P_k$ terminé à la date $t$ : capital multiplié par $(1 + \tau_k)$ à partir de la date $d_k$.

La meilleure option est retenue et permet de construire progressivement la valeur optimale $Coef(t)$.

### 4.4 Propriété du graphe

Le graphe $D$ étant orienté et sans cycle (acyclique), on peut traiter les dates dans l'ordre croissant ($t = 0 \to n$), ce qui garantit que chaque $Coef(t)$ peut être construit à partir des valeurs précédentes. Cela assure l'existence d'une solution optimale et rend l'algorithme efficace.

### 4.5 Complexité temporelle

Grâce à l'approche dynamique, la complexité temporelle de l'algorithme est de l'ordre de $\mathcal{O}(n \cdot m)$, où :

— $n$ est l'horizon temporel (nombre total de périodes),

— $m$ est le nombre de produits financiers spécifiques disponibles.

À chaque étape $t$, on ne considère que les $P_k$ tels que $f_k = t$ ; ainsi, le coût est linéaire par rapport à $m$. Cela représente une amélioration significative par rapport à l'énumération exhaustive de tous les chemins, dont la complexité est exponentielle en $n$.

**Remarque sur le sens du calcul** L'algorithme de programmation dynamique progresse dans le sens chronologique : les valeurs $Coef(t)$ sont calculées séquentiellement pour $t = 1$ jusqu'à $t = n$.

Cela est dû au fait que chaque $Coef(t)$ dépend uniquement des valeurs antérieures :

$$Coef(t-1) \quad \text{et} \quad Coef(d_k) \quad \text{pour tout } k \text{ tel que } f_k = t$$

Par conséquent, il est essentiel de connaître toutes les valeurs précédentes avant de pouvoir calculer $Coef(t)$.

En revanche, une fois toutes les valeurs $Coef(t)$ connues, l'extraction du **meilleur chemin d'investissement** (i.e. la politique optimale) peut se faire en sens inverse, en retraçant les décisions à partir de la date finale $t = n$ jusqu'à $t = 0$.

# 5 Implémentation algorithmique

## 5.1 Lecture des données

L'algorithme commence par lire les données du problème à partir d'un fichier Excel fourni par l'utilisateur.

Le fichier contient :

— le nombre total de périodes $n$ ;

— le taux d'intérêt de base $\tau_0$ ;

— une liste de $m$ produits financiers spécifiques sous forme de triplets $(\tau_k, d_k, f_k)$.

Ces données sont extraites et stockées dans trois objets Python :

— un entier $n$ : horizon d'étude (nombre d'unités de temps) ;

— un flottant $\tau_0$ : taux d'intérêt applicable à chaque intervalle de base $[t, t+1]$ ;

— une liste `placements` contenant les produits $P_k$ sous forme de tuples : `(tauk, dk, fk)`.

La fonction responsable de cette opération est définie comme suit :

**Algorithm 1** Lecture des données depuis un fichier Excel

```
function LECTURE_DONNEES(fichier)
    Lire n et τ0 depuis la première ligne
    Initialiser une liste vide placements
    for chaque ligne suivante du fichier do
        Lire (τk, dk, fk)
        Ajouter (tauk, dk, fk) à placements
    end for
    return (n, τ0, placements)
end function
```

Cette fonction s'appuie sur la bibliothèque `openpyxl` pour la lecture du fichier Excel et effectue les conversions nécessaires pour assurer la compatibilité avec les étapes suivantes de l'algorithme.

Un extrait du fichier `exemple.xlsx` est présenté ci-dessous :

| $\tau_k$ | $d_k$ | $f_k$ |
|---|---|---|
| 0.019 | 0 | 2 |
| 0.020 | 1 | 3 |
| 0.030 | 2 | 5 |
| 0.030 | 3 | 6 |
| 0.028 | 4 | 7 |

TABLE 1 – Extrait du fichier `exemple.xlsx` contenant les produits $P_k$

## 5.2 Calcul du coefficient optimal `Coef(t)`

La fonction `optimiz_coef` calcule, pour chaque instant $t$, le coefficient multiplicatif maximal `Coef(t)` que l'on peut atteindre, en choisissant soit de continuer avec le taux d'intérêt de base, soit d'utiliser un placement spécifique qui se termine à $t$. En plus, elle enregistre le chemin optimal en stockant dans une liste `chemin[t]` la date précédente qui a permis d'atteindre cette valeur.

**Entrées :**

— $t$ : date actuelle.

— `coef` : liste des valeurs déjà calculées pour `Coef(0)` jusqu'à `Coef(t-1)`.

— $\tau_0$ : taux d'intérêt de base, constant.

— `placements` : liste des placements spécifiques $(\tau_k, d_k, f_k)$.

— `chemin` : tableau où l'on enregistre, pour chaque $t$, la date précédente optimale.

**Sortie :**

— La valeur optimale `Coef(t)`.

**Algorithm 2** Calcul optimal du coefficient `Coef(t)`

```
1:  function OPTIMIZ_COEF(t, coef, τ0, placements, chemin)
2:      best ← coef[t − 1] · (1 + τ0)                          ▷ Option de base
3:      chemin[t] ← t − 1                                      ▷ On suppose qu'on vient de t − 1
4:      for all (τk, dk, fk) dans placements do
5:          if fk = t then                                     ▷ Placement se terminant à t
6:              val ← coef[dk] · (1 + τk)
7:              if val > best then
8:                  best ← val
9:                  chemin[t] ← dk                             ▷ On vient de dk via ce placement
10:             end if
11:         end if
12:     end for
13:     return best
14: end function
```

### 5.3 Reconstruction du chemin optimal

Après avoir calculé les valeurs optimales `Coef(t)` pour chaque $t$, on souhaite reconstruire le chemin qui a permis d'atteindre la valeur finale maximale. Pour cela, on utilise le tableau `chemin`, dans lequel chaque case `chemin[t]` indique de quelle date provient la meilleure décision menant à $t$.

La fonction `reconstruct_path` suit ce chemin en partant de $t = n$ et en remontant étape par étape jusqu'à $t = 0$, en collectant les couples de transitions $(\texttt{chemin}[t], t)$ qui forment le chemin optimal.

**Entrées :**

— `chemin` : liste des antécédents optimaux pour chaque $t$.

— $n$ : instant final.

**Sortie :**

— `path` : liste de transitions optimales sous forme de couples $(t_{\text{précédent}}, t_{\text{actuel}})$.

**Algorithm 3** Reconstruction du chemin optimal

```
function RECONSTRUCT_PATH(chemin, n)
    t ← n
    path ← liste vide
    while t > 0 do
        prev ← chemin[t]
        Ajouter le couple (prev, t) à path
        t ← prev
    end while
    Inverser la liste path                          ▷ Pour aller de 0 vers n
    return path
end function
```

## 5.4 Programme principal

Le programme principal permet d'exécuter l'ensemble des étapes de résolution :

— Lecture des données à partir d'un fichier Excel (`lecture_donnees`).

— Initialisation des structures de calcul (`coef`, `chemin`).

— Calcul des coefficients optimaux $Coef(t)$ par programmation dynamique.

— Reconstruction du chemin optimal à partir du tableau `chemin`.

— Affichage du résultat final : valeurs des $Coef(t)$, chemin optimal, et coefficient final.

**Algorithm 4** Programme principal d'optimisation

```
function MAIN
    Lire les données : n, τ0, placements            ▷ Depuis un fichier Excel
    Initialiser les tableaux coef et chemin
    coef[0] ← 1
    coef[1] ← coef[0] · (1 + τ0)
    chemin[1] ← 0
    for t de 2 à n do
        coef[t] ← optimiz_coef(t, coef, τ0, placements, chemin)
    end for
    path ← reconstruct_path(chemin, n)
    Afficher tous les Coef(t) pour t = 0 à n
    Afficher le chemin optimal path
    Afficher Coef(n) comme capital final
end function
```

L'exécution du programme sur différentes instances fournit les résultats suivants.

## 5.5 Exemple 1 — Données issues du fichier `exemple.xlsx`

### 5.5.1 Fichier utilisé

Les données de cette instance sont extraites du fichier :

`data-folder/data-exemple/exemple.xlsx`

### 5.5.2 Valeurs optimales du capital

Voici les coefficients multiplicatifs optimaux $Coef(t)$ calculés pour chaque date $t$ :

| $t$ | $Coef(t)$ |
|---|---|
| 0 | 1.00000 |
| 1 | 1.00900 |
| 2 | 1.01900 |
| 3 | 1.02918 |
| 4 | 1.03844 |
| 5 | 1.04957 |
| 6 | 1.06006 |
| 7 | 1.06960 |

### 5.5.3 Chemin d'investissement optimal

Le chemin optimal permettant d'atteindre le capital final maximal est le suivant :

$$\texttt{0} \rightarrow \texttt{1} \rightarrow \texttt{3} \rightarrow \texttt{6} \rightarrow \texttt{7}$$

Ce chemin correspond à la stratégie suivante :

— placement de base entre $t = 0$ et $t = 1$ ;
— produit spécifique $P_2$ utilisé de $t = 1$ à $t = 3$ ;
— produit spécifique $P_4$ utilisé de $t = 3$ à $t = 6$ ;
— placement de base entre $t = 6$ et $t = 7$.

### 5.5.4 Capital final obtenu

Le coefficient multiplicatif total à la fin de la période est :

$$\boxed{Coef(7) = 1.06960}$$

Ce résultat valide la performance de l'algorithme dans l'identification d'une stratégie d'investissement optimale.

## 5.6 Exemple 2 — Données issues d'un fichier personnalisé

### 5.6.1 Fichier utilisé

Les données de cette instance proviennent d'un fichier plus long contenant 129 périodes et un ensemble élargi de produits spécifiques. (Nom du fichier non précisé ici — à ajouter si nécessaire.)

### 5.6.2 Valeurs optimales du capital

Voici un extrait des valeurs optimales calculées pour `Coef(t)` :

| $t$ | $Coef(t)$ | $t$ | $Coef(t)$ | $t$ | $Coef(t)$ |
|---|---|---|---|---|---|
| 0 | 1.00000 | 20 | 1.23242 | 40 | 1.54233 |
| 60 | 1.92632 | 80 | 2.36578 | 100 | 2.94091 |
| 110 | 3.34756 | 120 | 3.78792 | 128 | 4.22391 |

### 5.6.3 Chemin d'investissement optimal

Le chemin optimal trouvé est relativement long. En voici les principales transitions clés (extrait simplifié) :

| Étapes | Transition |
|---|---|
| $0 \to 4$ | Placement spécifique |
| $14 \to 19$ | Placement spécifique |
| $34 \to 54$ | Placement spécifique |
| $61 \to 75$ | Placement spécifique |
| $94 \to 102$ | Placement spécifique |
| $102 \to 107$ | Placement spécifique |
| $109 \to 118$ | Placement spécifique |
| $124 \to 126$ | Placement spécifique |
| $126 \to 128$ | Placement final |

(Toutes les autres transitions intermédiaires sont des placements de base successifs.)

### 5.6.4 Capital final obtenu

Le coefficient final à la date $t = 128$ est :

$$\boxed{Coef(128) = 4.22391}$$

Ce résultat montre que la stratégie optimale sélectionnée par l'algorithme permet de quadrupler le capital initial, en combinant plusieurs placements spécifiques à long terme avec des placements de base réguliers.

# 6 Conclusion

### Résumé du Rapport

Dans ce projet, nous avons étudié un problème d'optimisation de stratégie d'investissement sur un horizon discret donné. En modélisant les opportunités de placement sous forme d'un graphe orienté acyclique, nous avons pu appliquer une approche efficace de programmation dynamique permettant de déterminer, à chaque instant $t$, le coefficient multiplicatif optimal du capital.

L'algorithme proposé évite l'énumération exhaustive de tous les chemins possibles (dont le nombre croît exponentiellement) et garantit une solution optimale en temps raisonnable. Cette approche est à la fois théoriquement fondée, simple à mettre en œuvre, et très efficace en pratique.

# Annexes

### Installation du module `openpyxl`

Pour exécuter correctement le programme Python et lire les données depuis un fichier Excel (.xlsx), il est nécessaire d'installer la bibliothèque `openpyxl`.

— Tapez dans le terminal : `pip install openpyxl`