Devoir 3 : Goal-Based Wealth Management

Nasser-Harorld-Kisswendsiida Diallo; Léo Reuther; Malika Saloucou

4 décembre 2023

1 Introduction

Dans le contexte d'optimisation de portefeuille, plusieurs techniques de résolution ont été présentées au fil des années, telles que la programmation dynamique (DP), l'apprentissage par renforcement (RL), la programmation linéaire et quadratique, etc. Dans ce rapport, nous allons aborder plusieurs aspects liés à la gestion dynamique de portefeuille basée sur des objectifs à l'aide de l'apprentissage par renforcement .Nous allons nous inspiré des recherches de S.R. Das et S. Varma dans leur article intitulé "Dynamic Goals-Based Wealth Management using Reinforcement Learning" publié dans le Journal of Investment Management en 2020. Nous nous concentrerons sur le Q-learning et le SARSA-learning afin de répondre à la question du choix optimal de la politique. Pour ce faire nous allons en premier lieu expliquer nos méthode de résolution, ensuite nous allons tester ces deux méthodes en comparant le hors échantillon et le "in-sample" et pour finir nous allons faire varier quelques parametres afin d'approfondir nos recherches.

2 Optimisation par objectif ou GBWM

Le modèle traditionnel pour les problèmes d'optimisation cherchait à trouver le meilleur couple risque-rendement maximisant la fonction d'utilité de l'investisseur (Markowitz). Cependant, le GBWM, un modèle plus récent, visant à maximiser la probabilité que la valeur du portefeuille $W(T)$ atteigne ou dépasse un certain niveau de richesse H sur un horizon temporel T. L'objectif n'est pas de maximiser les rendements, mais de se concentrer sur la réalisation d'objectifs financiers spécifiques. Le risque est la probabilité que l'investisseur n'atteigne pas son objectif à la fin de la période T. Le GBWM cherche à atteindre l'objectif avec le moins de risque possible, en maximisant la fonction objectif suivante :

$$\max_{w(t),\, t<T} \Pr[W(T) \geq H]$$

Ainsi, chaque année, en commençant par l'année 0, nous sélectionnerons un portefeuille $w(t) = (\mu, \sigma)_t$ qui représente les actions $A(t)$ ($w(t) \equiv A(t) \equiv (\mu, \sigma)_t$) d'une certaine moyenne μ et d'un certain écart-type σ, parmi un ensemble

de portefeuilles acceptables afin d'augmenter la chance d'atteindre l'objectif au temps T. Dans le cadre de notre problème, nous choisirons nos portefeuilles uniquement parmi ceux qui sont sur la frontière efficiente (selon la théorie moderne du portefeuille de Markowitz), qui consiste à choisir le portefeuille qui minimise la variance pour chaque moyenne donnée. Il est également à noter que ce problème sera résolu comme une chaîne de Markov ou un processus de décision de Markov (MDP).

- Chaîne de Markov : Il s'agit d'un modèle mathématique décrivant un système qui passe d'un état à un autre parmi un ensemble fini ou dénombrable d'états possibles. La particularité d'une chaîne de Markov est que la probabilité de transition vers l'état suivant dépend uniquement de l'état actuel et non de l'historique des états précédents, ce qu'on appelle la propriété de "sans mémoire" ou "propriété de Markov". Ces chaînes peuvent être discrètes ou continues. Pour faciliter l'utilisation des algorithmes d'apprentissage par renforcement (RL), nous considérerons, dans le cadre de ce problème, un horizon de temps fini et discret, ainsi que l'espace des états et des actions, qui seront également discrets.

3 Problèmes d'optimisation et méthodes de résolution

Pour notre étude , l'horizon temporel sera $t = 1, 2, \ldots, T+1$ où $T = 10$. Nous considérons également que l'évolution du niveau de richesse $W(t)$ à $W(t+h)$ pour $h = 1$ suit un mouvement brownien géométrique (GBM) :

$$W(t+h) = W(t) \exp\left[\mu(t) - \frac{1}{2}\sigma^2(t)h + \sigma(t)\sqrt{h} \cdot Z\right]$$

où la variable Z suit une distribution normale standard : $Z \sim \mathcal{N}(0,1)$. Les algorithmes d'apprentissage par renforcement (RL) présentés seront utilisés pour trouver la politique optimale, c'est-à-dire, pour un certain niveau de richesse W au temps t, quelle action (portefeuille) choisir afin de maximiser notre fonction objectif. L'espace des états finis, représenté par $W(t)$, est constitué de $10T+1 = 101$ points de richesse, avec une probabilité de transition entre les points de richesse :

$$\Pr\left[W_j(t) \,|\, W_i(0)\right] = \text{Prob}(Z = B) = \phi(B)$$

avec

$$B = \frac{\ln\left(\frac{W_j(t)}{W_i(0)}\right) - \left(\mu - 0.5\sigma^2\right)t}{\sigma\sqrt{t}}$$

où $\phi(\cdot)$ est la fonction de densité de la loi normale.

Pour faciliter la résolution, au lieu de résoudre directement la fonction objectif mentionnée précédemment, nous maximiserons la fonction $Q(W, a)$, qui est la valeur espérée que l'on peut obtenir jusqu'à la fin de l'horizon temporel

T à partir d'un certain W en utilisant l'action a pour la période qui commence, puis une politique optimale par la suite. Nous résoudrons donc :

$$Q(W_i(t), a_t) = \sum_{j=0}^{imax} \max_{a_{t+1} \in A_{t+1}} Q(W_j(t+1), a_{t+1}) \Pr\left[W_j(t+1) \,|\, W_i(t), a_t\right]$$

Veuillez noter également que dans le cadre de ce problème, la récompense $R = 1$ sera uniquement obtenue au temps $T + 1$ si $W(T) \geq H$, et que le facteur d'actualisation $\gamma = 1$. Pour entraîner notre modèle, nous considérerons $K = 15$ portefeuilles, $W = 100$ comme richesse initiale, $H = 200$ comme objectif, et ferons varier le niveau d'exploration epsilon entre 0.1 et 0.4.

Avant de présenter le SARSA-learning et le Q-learning, il est important de mentionner que ces algorithmes se basent sur les principes du Temporal Difference (TD) learning, un algorithme qui apprend directement à partir de l'expérience sans modèle de l'environnement, et ajuste ses estimations en partie sur la base de nouvelles estimations. Il suit l'itération suivante :

$$Q(s, a) \leftarrow Q(s, a) + \alpha \left[R + \gamma Q(s', a') - Q(s, a)\right]$$

où s' est le prochain état, a' la prochaine action.

3.1 Q-Learning

Le Q-Learning est la version hors politique du TD-learning c'est a dire qu"elle apprend la politique optimal indépendammment de la politique suivie par l'agent pour explorer l'environnement.Il utilise la fonction d'évaluation Q pour estimer la valeur de chaque état-action. L'objectif du Q-Learning est de trouver une politique qui maximise les récompenses cumulatives dans l'environnement.Dans notre problème la fonction Q depends de l'espace des états w $Q(W, t, a)$,le temps t et de l'action prise a. La formule utilisé pour choisir la nouvelle action a^n a est :

$$a_n = \underset{a \in A}{\operatorname{argmax}}\, Q_{n-1}(s_{n-1}, a)$$

L'iteration dans le Q learning est la suivante :

$$Q(s, a) \leftarrow Q(s, a) + \alpha \left[R + \gamma \max_{a'} Q(s', a') - Q(s, a)\right]$$

Dans notre code Matlab, nous avons d'abord initialisé notre matrice $Q(W, t, a)$ à zéro. Cette matrice 3D a des dimensions correspondant au nombre d'états, à $T + 1$ (le nombre de périodes temporelles), et au nombre de portefeuilles.

Ensuite, à partir de $t = 0$, nous réalisons un nombre d'itérations défini par `epoch` à chaque instant t pour entraîner notre modèle.

À chaque itération, nous sélectionnons l'action a_k qui maximise $Q(W(t), t, a)$ pour $1 \leq k \leq K$, en tenant compte du niveau d'exploration. Après avoir observé l'état suivant $W(t+1)$ au temps $t+1$, nous choisissons l'action a' qui maximise

$$Q(W(t+1), t+1, a_k), \quad 1 \leq k \leq K.$$

Nous déterminons ensuite la nouvelle valeur de Q.

Étant donné que les récompenses ne sont disponibles qu'au temps $T+1$, tant que $t < T$, la valeur de Q devient simplement la partie

$$\max_{Q}(W(t+1), t+1, a').$$

Nous intégrons ensuite la nouvelle valeur de Q en appliquant une opération de lissage de cette façon :

$$Q(W(t), t, a) = \alpha \cdot Q(W(t+1), t+1, a') + (1-\alpha) \cdot Q(W(t), t, a).$$

Au temps $T+1$, notre algorithme suit la logique suivante :

$$Q(W(T+1), T+1, a) \leftarrow Q(W(T+1), T+1, a) + \alpha \cdot [1 - Q(W(T+1), T+1, a)] \quad \text{si } W(T+1) \geq H, \tag{1}$$

$$Q(W(T+1), T+1, a) \leftarrow Q(W(T+1), T+1, a) + \alpha \cdot [0 - Q(W(T+1), T+1, a)] \quad \text{si } W(T+1) < H. \tag{2}$$

Cette approche permet d'ajuster la valeur de Q en fonction de la réalisation effective des gains ou pertes du portefeuille, comparativement à un seuil de performance H.

3.2 SARSA-Learning

Le SARSA (State-Action-Reward-State-Action) est la version en politique du TD-learning, c'est-à-dire qu'il apprend la valeur des actions selon la politique actuellement suivie par l'agent, y compris les décisions d'exploration. Alors que le Q-Learning met à jour la fonction Q en utilisant la valeur maximale de Q pour l'état suivant, le SARSA met à jour la fonction Q en utilisant la valeur de Q pour l'action suivante, choisie selon la politique courante. L'itération dans l'apprentissage SARSA est la suivante :

$$Q(s, a) \leftarrow Q(s, a) + \alpha \left[R + \gamma Q(s', a') - Q(s, a) \right].$$

Dans l'approche SARSA, à partir de $t = 0$ à chaque itération, nous choisissons une action a_k en suivant une politique d'exploration pour l'état actuel $W(t)$ et le temps t. Après avoir exécuté cette action et observé l'état suivant $W(t+1)$, nous choisissons la prochaine action a' en suivant la même politique pour l'état $W(t+1)$ et le temps $t+1$.

Dans SARSA, la mise à jour de la valeur Q se fait en utilisant la paire état-action suivante, et en considérant que la récompense est nulle pour tous les temps sauf à $T+1$:

$$Q(W(t), t, a) = Q(W(t), t, a) + \alpha[R(t+1) + Q(W(t+1), t+1, a') - Q(W(t), t, a)],$$

où $R(T+1)$=1 est la récompense obtenue au temps final $T+1$.

La mise à jour de Q au temps $T+1$ tient compte de l'objectif final :

$$Q(W(T+1),T+1,a) \leftarrow Q(W(T+1),T+1,a) + \alpha[1 - Q(W(T+1),T+1,a)] \quad \text{si } W(T+1) \geq H, \tag{3}$$

$$Q(W(T+1),T+1,a) \leftarrow Q(W(T+1),T+1,a) + \alpha[0 - Q(W(T+1),T+1,a)] \quad \text{si } W(T+1) < H. \tag{4}$$

Cela permet à l'algorithme SARSA d'ajuster la valeur de Q en fonction des actions prises et des états successifs observés, en se concentrant sur la réalisation de l'objectif au temps final $T+1$.

3.3 Test hors politique

Le but du test hors échantillon est de vérifier si nos politiques optimales génèrent toujours des taux de succès similaires lorsqu'elles sont confrontées à des données inédites. Les politiques optimales que nous générons sont basées sur un certain ensemble de rendements, et seront donc optimisées pour les données à partir desquelles elles ont été générées. Cependant, il n'est pas toujours évident que la politique optimale restera optimale une fois qu'elle est sortie de l'ensemble d'entraînement.

Pour le test hors échantillon, nous suivons la même structure que dans les cas de Q-Learning et SARSA. Nous procédons selon les étapes suivantes :

1. Générer les mêmes états de richesse, W, que dans l'échantillon.
2. Définir la même richesse initiale que dans l'échantillon.
3. En utilisant la politique optimale que nous avons dérivée pendant l'échantillon, nous sélectionnons le portefeuille optimal en fonction de notre richesse et de l'étape temporelle actuelle.
4. Simuler le rendement de ce portefeuille :

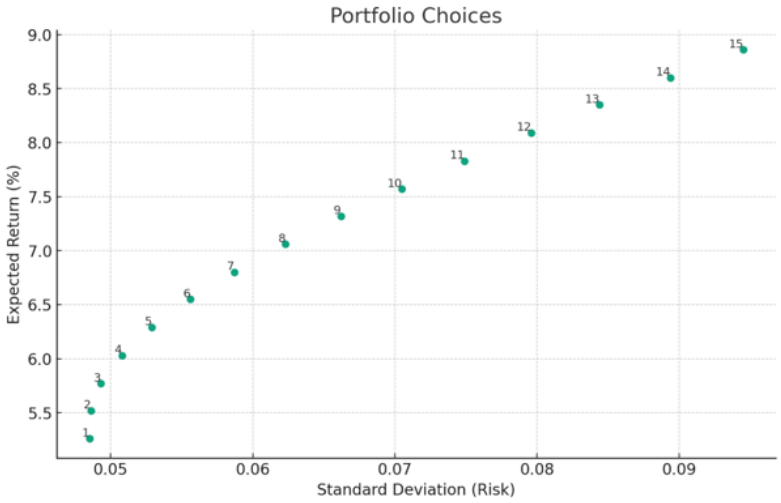
   $$R = \mu + \sigma \times \epsilon$$

   où
   — μ : rendement attendu du portefeuille
   — σ : écart-type du portefeuille
   — ϵ : variable aléatoire suivant une distribution normale standard
5. Mettre à jour la richesse :

   $$W_{\text{new}} = W_{\text{current}} \times e^{R}$$

6. Mettre à jour l'étape temporelle.
7. Si toutes les étapes temporelles ne sont pas complétées, répéter à partir de l'étape 3.
8. Si toutes les étapes temporelles sont complétées, noter si nous avons atteint notre objectif de richesse.
9. Répéter le processus 100 000 fois, la moyenne des fois où nous atteignons notre objectif au cours de ces simulations est notre taux de succès.

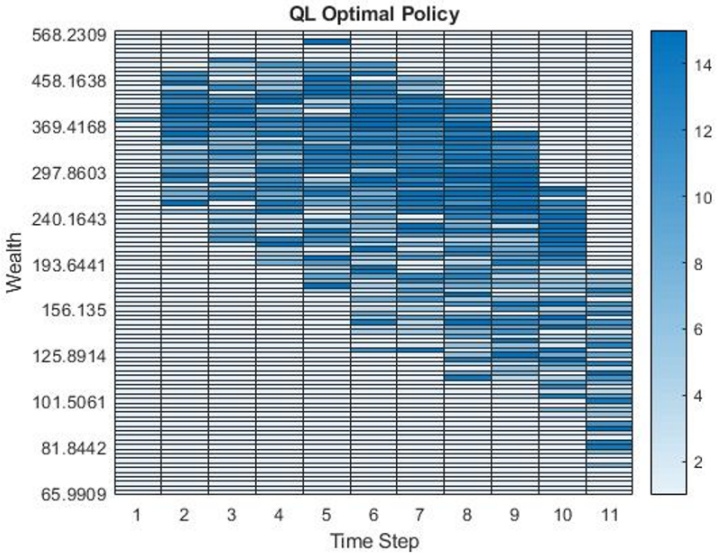
3.4 Les choix de portefeuilles disponibles



4 Résultats

4.1 Q-Learning

En exécutant notre algorithme Q-Learning pour $50,000$ epochs et un α de 0.1, nous obtenons un taux de succès de 66.12% et la politique optimale suivante :



Avec 15 choix de portefeuille offrant des rendements attendus et des écarts-types croissants, la politique optimale dérivée de l'apprentissage par renforce-

ment Q-Learning reflète cela, favorisant les portefeuilles avec des rendements attendus plus élevés dans les premières étapes, lorsqu'on commence avec un niveau de richesse élevé. Cette stratégie tire efficacement parti du potentiel de rendements plus élevés, augmentant la probabilité de maintenir la richesse bien au-dessus de l'objectif à long terme. À mesure que l'on approche T et que les niveaux de richesse dépassent déjà la cible, la politique devient naturellement plus conservatrice. Ce changement vise à atténuer le risque de ne pas atteindre l'objectif en raison de la volatilité du marché à court terme.

L'essence de cette approche réside dans son contraste avec les objectifs d'investissement traditionnels ; dans la GBWM (gestion de richesse basée sur des objectifs), atteindre l'objectif est primordial, et une richesse excédentaire au-delà de la cible n'offre aucune récompense supplémentaire.

Lorsqu'on commence avec des niveaux de richesse plus faibles, la politique de Q-Learning recommande initialement des portefeuilles moins risqués, adoptant une approche 'lente mais sûre'. Cependant, à mesure que le temps avance et si la richesse reste en dessous du niveau souhaité, la politique commence à favoriser des portefeuilles plus risqués. Ce changement est logique ; avec un temps limité restant et une richesse toujours inférieure à l'objectif, les stratégies conservatrices peuvent s'avérer insuffisantes. Dans de tels scénarios, prendre des risques plus élevés devient une stratégie nécessaire pour maximiser les chances d'atteindre l'objectif, surtout puisque la préoccupation principale n'est pas la préservation de la richesse mais l'atteinte de l'objectif.

4.1.1 Q-Learning In Sample Replication

Table 1 – MODEL OUTPUTS Q-LEARNING IN SAMPLE

Epsilon	No. of Epochs	$V(W(0), t = 0)$	$V(W(0), t = 0)$ REPLICATED	Delta
0.10	50000	0.65	0.60	-0.05
0.10	100000	0.65	0.62	-0.03
0.20	50000	0.69	0.63	-0.06
0.20	100000	0.71	0.65	-0.06
0.25	100000	0.71	0.69	-0.02
0.30	50000	0.72	0.66	-0.06
0.30	100000	0.72	0.70	-0.02
0.40	50000	0.73	0.74	0.01
0.40	100000	0.77	0.73	-0.04
0.40	200000	0.75	0.74	-0.01
0.40	500000	0.71	0.73	0.02

Nous sommes capables d'obtenir des résultats similaires à ceux obtenus dans notre article, démontrant ainsi que notre algorithme Q-Learning fonctionne comme prévu. Sans connaissance du « seed » et du langage de programmation utilisés par les auteurs, nous ne pouvons pas reproduire exactement leurs résultats, mais le delta proche de 0 pour toutes les combinaisons d'epochs et

d'epsilons suggère qu'en l'absence de hasard, les deux modèles fonctionnent de la même manière et parviennent à atteindre l'objectif dans la majorité des cas pour ces paramètres. De plus, nous remarquons que des epsilons et epochs plus élevés tendent légèrement à surperformer leurs homologues inférieurs.

4.1.2 Q-Learning Out Of Sample Replication

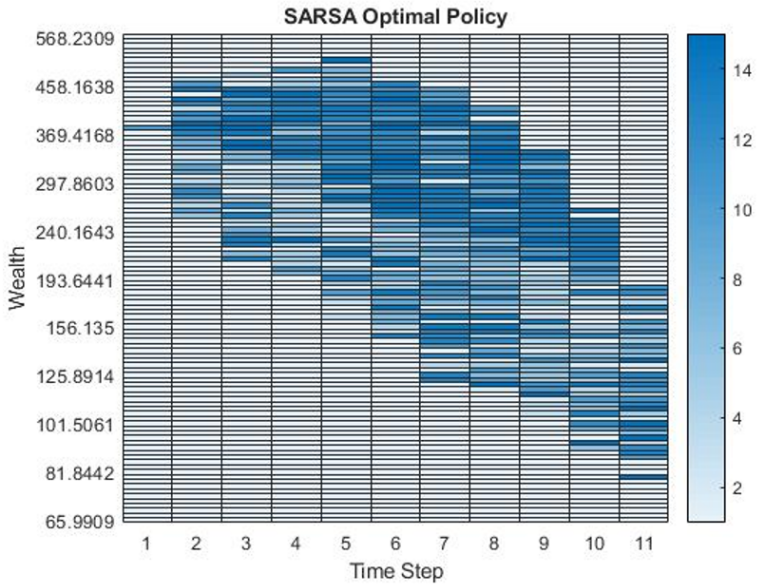
TABLE 2 – MODEL OUTPUTS Q-LEARNING OUT OF SAMPLE

Epsilon	No. of epochs	$V(W(0), t = 0)$	OOS $V(W(0), t = 0)$	Delta
0.10	50000	0.60	0.65	0.05
0.10	100000	0.62	0.63	0.01
0.20	50000	0.63	0.59	-0.04
0.20	100000	0.65	0.65	-0.01
0.25	100000	0.69	0.63	-0.06
0.30	50000	0.66	0.64	-0.02
0.30	100000	0.70	0.64	-0.06
0.40	50000	0.74	0.65	-0.09
0.40	100000	0.73	0.65	-0.08
0.40	200000	0.74	0.65	-0.08
0.40	500000	0.73	0.65	-0.08

La performance hors échantillon est légèrement inférieure à la performance sur l'échantillon dans la plupart des cas. Cela est prévisible car la politique optimale est testée sur des données inédites. Cependant, la performance hors échantillon ne s'écarte pas significativement de celle sur l'échantillon, ce qui suggère que les politiques dérivées sont suffisamment stables sur des données non observées.

4.2 SARSA-Learning

En exécutant notre algorithme SARSA pour 50000 epochs et un alpha de 0.1, nous obtenons un taux de succès de 40.11% ce qui est significativement plus bas que le Q-Learning avec les même paramètres, et la politique optimale suivante :



Nous remarquons que la politique optimale SARSA pour les paramètres donnés suit la même logique que celle pour Q-Learning, bien qu'il y ait de légères différences. Ces légères différences sont à prévoir, dans les deux cas la politique optimale n'est pas complètement lisse et, dans les deux cas, certaines zones ne sont pas explorées, en particulier les coins où nous avons une richesse élevée dans les dernières étapes temporelles et vice versa.

4.2.1 SARSA-Learning In Sample Replication

Table 3 – MODEL OUTPUTS SARSA IN SAMPLE

Epsilon	No. of epochs	$V(W(0), t = 0)$
0.10	50000	0.39
0.10	100000	0.50
0.20	50000	0.39
0.20	100000	0.50
0.25	100000	0.52
0.30	50000	0.40
0.30	100000	0.51
0.40	50000	0.38
0.40	100000	0.51
0.40	200000	0.54
0.40	500000	0.58

Le taux de succès sur l'échantillon de SARSA est significativement plus bas que le taux de succès de Q-Learning lorsqu'on les compare avec le même epsi-

lon et le même nombre d'epochs. Ceci est particulièrement vrai pour les petits epsilons.Le taux de performance inférieur observé dans SARSA par rapport à Q-learning peut s'expliquer par la nature de la politique choisie dans chaque algorithme. Comme mentionner plus haut le SARSA suit une approche en politique , où les décisions sont prises en fonction de la politique actuellement en cours d'exécution, y compris les choix d'exploration. En conséquence, la politique choisie par SARSA n'est pas nécessairement la plus optimale à chaque étape, car elle dépend des actions actuelles et potentiellement exploratoires de l'agent.Alors qu'en revanche le Q-learning avec son approche hors politique , la décision pour l'état suivant s' est basée sur le choix de l'action,a' qui maximise la valeur de la fonction Q pour cet état. Cela signifie que Q-learning évalue les actions non pas sur la base de la politique actuellement suivie, mais en cherchant la meilleure action possible dans un état donné. Cette maximisation assure que Q-learning est toujours orienté vers la recherche de la politique optimale, indépendamment de la politique suivie durant le processus d'apprentissage.

4.2.2 SARSA-Learning Out Of Sample Replication

Table 4 – MODEL OUTPUTS SARSA OUT OF SAMPLE

Epsilon	No. of epochs	$V(W(0), t = 0)$	OOS $V(W(0), t = 0)$	Delta
0.1	50000	0.39	0.64	0.25
0.1	100000	0.50	0.61	0.12
0.2	50000	0.39	0.63	0.24
0.2	100000	0.50	0.63	0.14
0.25	100000	0.52	0.61	0.09
0.3	50000	0.40	0.63	0.23
0.3	100000	0.51	0.66	0.15
0.4	50000	0.38	0.60	0.22
0.4	100000	0.51	0.63	0.12
0.4	200000	0.54	0.63	0.09
0.4	500000	0.58	0.65	0.08

De manière surprenante, le taux de succès hors échantillon est significativement plus élevé que celui sur l'échantillon dans tous les cas et ressemble étroitement au taux de succès de l'algorithme Q-Learning. C'est possible que cela peut s'expliquer par le fait que l'algorithme SARSA possède une bonne capacité de généralisation, c'est-à-dire qu'il a appris une politique qui peut s'adapter facilement à de nouveaux environnements. If faudrait effectuer des analyses plus approfondies pour valider ceci.

4.2.3 Variation des niveaux des parametres

Nous avons ensuite fait varier les différents niveaux d'exploration et d'apprentissage dans le hors échantillon du Q-Learning et du SARSA. On observe

une amélioration du niveau de succès avec un certain niveau de baisse mais aussi de hausse du alpha (0.20 et 0.07) et l'augmentation des itérations. Cependant on remarque une stabilisation lorsque le niveau d'exploration ne varie plus. Le nombre d'itération aussi semble être relié positivement au niveau du succès car avec les mêmes paramètres on observe une hausse du succès avec un plus grand nombre d'itérations pour la plupart.

TABLE 5 – MODEL OUTPUTS Q-Learning OUT OF SAMPLE different parameter

Epsilon	No. of Epochs	Alpha	V(W(0),t-0)
0.10	50000	0.15	0.62
0.10	100000	0.15	0.68
0.20	50000	0.20	0.71
0.20	100000	0.20	0.68
0.25	100000	0.0050	0.40
0.30	50000	0.07	0.63
0.30	100000	0.07	0.67
0.40	50000	0.050	0.61
0.40	100000	0.050	0.70
0.40	200000	0.050	0.70
0.40	500000	0.050	0.70

TABLE 6 – MODEL OUTPUTS SARSA-Learning OUT OF SAMPLE different parameter

Epsilon	No. of Epochs	Alpha	V(W(0),t-0)
0.10	50000	0.15	0.44
0.10	100000	0.15	0.51
0.20	50000	0.20	0.48
0.20	100000	0.20	0.53
0.25	100000	0.0050	0.04
0.30	50000	0.07	0.30
0.30	100000	0.07	0.46
0.40	50000	0.050	0.20
0.40	100000	0.050	0.40
0.40	200000	0.050	0.50
0.40	500000	0.050	0.58

5 Conclusion

En conclusion, le Q-learning et le SARSA sont tous des algorithmes d'apprentissage par renforcement populaires qui ont leurs propres avantages et limitations. Le GBWM offre une bonne précision des estimations de la valeur, le

Q-learning est efficace pour les environnements simples, tandis que le SARSA est adapté aux politiques stochastiques. Le choix de l'algorithme dépend du problème spécifique et des contraintes de l'environnement. Dans nos expériences, le Q-Learning a donné les meilleurs résultats globaux. Cependant, il est important de noter que les performances peuvent varier en fonction des paramètres, de l'environnement et de l'ensemble de données utilisé.)

6 Références

Sanjiv R. Das et Subir Varma, "Dynamic Goals-Based Wealth Management Using Reinforcement Learning", Journal Of Investment Management, Vol. 18, No. 2, (2020), pp. 1–20