Deep Hedging

Équipe Recherche & Développement
i-Fihn Consulting

1 INTRODUCTION

Un des enjeux majeurs de l'activité d'une banque d'investissement ou de toute entité financière est l'évaluation et la couverture de portefeuille de produits dérivés. La finance quantitative telle que nous la connaissons i.e. l'utilisation de modèle stochastique a été l'approche la plus populaire ces 30 dernières années pour répondre à ces problématiques. Malheureusement, ces méthodes reposent généralement sur des hypothèses qui ne sont pas vérifiées dans un marché réel.

Le développement des techniques de *"machine learning"* ces derniers années ont montré la transversalité de ces méthodes et les perspectives d'évolution paraissent sans limites. Les chercheurs et équipes R&D des banques d'investissement commencent seulement à réaliser le potentiel de ces méthodes et à les utiliser comme alternatives efficientes aux problématiques des banques. C'est dans cette optique que Hans Buehler, Lukas Gonon, Josef Teichmann et Ben Wood (JP Morgan et ETH Zürich) ont présenté une approche révolutionnaire (cf. article [1]) de la gestion de la couverture de portefeuille de produits dérivés à partir de réseaux de neurones profonds et reinforcement learning (quelques autres articles autour du domaine [3**?** , 4]).

2 L'ALGORITHME

Soit S la dynamique d'un ensemble de sous-jacents. Le problème de couverture optimale d'un portefeuille Z peut s'écrire sous la forme d'un problème d'optimisation sous contraintes:

$$\sup_{\alpha \in \mathcal{H}} \mathbb{E}\left[\varphi\left(-Z + (\alpha \cdot S)_T + C_T(\alpha)\right)\right], \tag{1}$$

où α est une stratégie de trading et $\mathcal{H}$ est l'ensemble des stratégies de trading admissibles, $C_T(\alpha)$ le coût total de la stratégie α jusqu'à la maturité $T \in \mathbb{R}^+$, φ la fonction d'utilité i.e. typiquement $\varphi(x) = \exp(-\lambda x)$ où λ est le paramètre d'aversion au risque.

On utilise une architecture de réseau de neurones profonds récurrents (RNN) comme approximateur de la stratégie de trading α en tout instant (*outputs*). Les *inputs* de l'algorithme d'apprentissage sont les prix des instruments de couverture disponibles dans le marché et les cashflows futurs des produits dérivés du portefeuille à couvrir obtenus à partir de simulation.

L'architecture LSTM (long-short term memory, type de RNN) permet de capturer les effets de *path-dependency* sous contraintes de coûts de transaction et permet de garder en mémoire les décisions de stratégies de couverture passées.

Le réseau de neurones contient $L = 2$ couches cachées paramétré comme suit:

$$\mathcal{NN}_\sigma^L(x) = \theta_L \circ \Phi_{L-1} \circ \cdots \circ \Phi_1, \tag{2}$$

avec $\Phi_l = \sigma \circ \theta_l$, $\forall l = 1, \ldots, L-1$ où θ_l sont des fonctions affines données comme $W_l(x) = A^l x + b^l$ et les fonctions d'activation choisies en $\sigma(x) = \max(x, 0)$.

3 ANALYSES

Avantages.

- Model-free:
  Pas de dépendance aux modèles sous-jacents, ce qui permet d'obtenir une architecture générique et décomplexifie la librairie de pricing.
- Réduction du coût de la couverture:
  L'algorithme permet d'obtenir la meilleure sur-réplication de portefeuille de couverture.
- Couverture instantanée:
  Lorsque le réseau de neurone est calibré, l'évaluation de la couverture du portefeuille sous-jacent est instantanée.
- Prise en compte des contraintes globales:
  Les contraintes imposées peuvent être les coûts de transaction, la liquidité des instruments de couverture, les limitations de risques, l'état du marché, l'historique de la couverture du portefeuille, les bid/ask spreads, les impacts de marché etc...
- Couverture de portefeuille de produits dérivés complexes:
  La méthode permet de résoudre des problèmes de couvertures pour l'entiereté d'un portefeuille de produits dérivés, ce qui était inaccesible auparavant.

Inconvénients.

- Calibration du réseau de neurones:
  Un changement de régime de marché peut nécessité une recalibration des réseaux de neurones sous-jacents (ex: crise sanitaire de la covid19). Le processus de calibration est peut être long i.e. plusieurs heures de calculs.
- Le calcul d'erreurs est une moyenne sur un ensemble global d'un échantillon test:
  Des erreurs peuvent être beaucoup plus grosses localement
- Données historiques insuffisantes:
  L'entraînement d'un réseau de neurones requiert une masse de données importantes, ce qui peut être uniquement comblé par l'ajout de données simulées (problème de paradigme puisque les données sont simulées à partir de modèles qui ne représentent pas la réalité - hypothèses etc...) et engendre donc un biais prédictif.
- Conformité à la régulation/validation:
  Il est difficile, voire impossible d'expliquer le comportement d'un réseau de neurones ce qui pose un problème quant à la validation d'une telle méthode.

Recherche en cours, i-Fihn Consulting,
2020.

4 RÉSULTATS NUMÉRIQUES

Nous avons procédé à quelques applications numériques sur le cas simple d'un portefeuille comportant un Call option. Les paramètres sont:

- $S_0 = 100$
- $K = 100$
- $\sigma = 20\%$
- $r = 0\%$
- $T = 1$ an

Le réseau de neurones apprend à partir de 250,000 trajectoires, voici ce que nous obtenons sur le deep hedge price (figure 1) et le delta (figure 2):

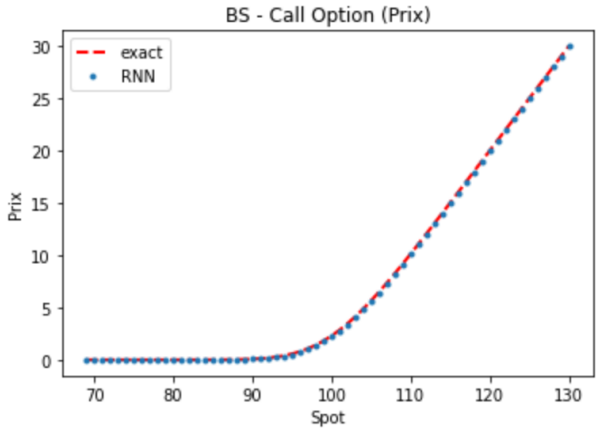

Figure 1: Comparaison des prix obtenus par le modèle BS et par réseau de neurones

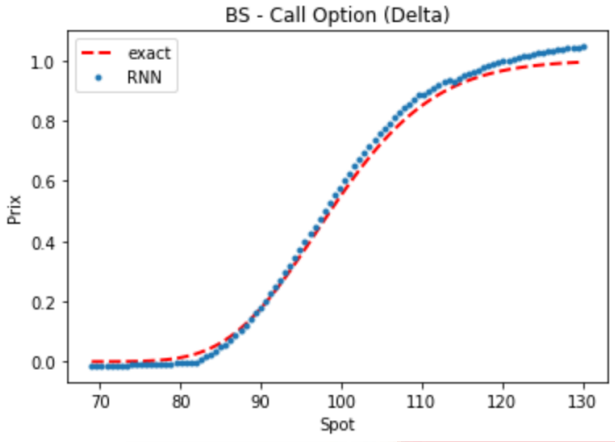

Figure 2: Comparaison des delta obtenus par le modèle BS et par réseau de neurones

Notons, que le réseau de neurone a plus de difficultés à approximer la fonction du delta. Une explication simple est qu'un bon approximateur d'une fonction n'implique pas qu'il soit capable de bien approximer les dérivées partielles de cette fonction, voire l'article de Brian Huge et Antoine Savine [2]. Un moyen de remédier à ce problème serait d'inclure les processus tangents[1] des trajectoires comme inputs pendant la phase d'entraînement du réseau de neurones.

5 PROCHAINS CHALLENGES

Au sein de la R&D d'i-Fihn Consulting, les projets futurs autour du Deep Hedging sont:

- Tester des portefeuilles de produits dérivés exotiques complexes (autocall etc...)
- Génération des simulations à partir de modèles à volatilité locale et/ou stochastique
- Inclure les techniques de différentiation automatique (cf. article [2]) pour rajouter les processus tangents des trajectoires comme inputs de la méthode

Les questions que nous sommes actuellement en train d'explorer:

- Comment déduire la fréquence de rebalancement de la couverture du portefeuille ?
- Est-ce que le même réseau de neurones peut être utiliser pour n'importe quel produit dérivé (exotique ou vanille)?
- À partir de quand une calibration du réseau de neurone est inférieure à N évaluations (à déterminer) de produits dérivés?

REFERENCES

[1] Hans Buehler, Lukas Gonon, Josef Teichmann, and Ben Wood. 2018. Deep Hedging. https://arxiv.org/abs/1802.03042
[2] Brian Huge and Antoine Savine. 2020. Differential Machine Learning. https://arxiv.org/abs/2005.02347
[3] Gordon Ritter and Peter N Kolm. 2019. Dynamic Replication and Hedging: A Reinforcement Learning Approach. *Journal of Financial Data Science* 1 (2019).
[4] Magnus Wiese, Lianjun Bai, Ben Wood, and Hans Buehler. 2019. Deep Hedging: Learning to Simulate Equity Option Markets. https://arxiv.org/pdf/1911.01700

[1] Les processus tangents sont les sensibilités des trajectoires à des paramètres. Par exemple le processus tangent d'une trajectoire par rapport à la variation du prix du sous-jacent à l'état initial est donné par:

$$\frac{\partial S_t}{\partial S_0} \tag{3}$$

Ces processus contiennent beaucoup d'informations et permettraient d'avoir une meilleure approximation des dérivées. L'idée vient de la méthode pathwise qui stipule que:

$$\frac{\partial}{\partial x}\mathbb{E}\left[(S_T - K)^+\right] = \mathbb{E}\left[\frac{\partial}{\partial S_T}(S_T - K)^+ \frac{\partial S_T}{\partial x}\right] \tag{4}$$

Ici, on voit que dans l'expression d'un grecque d'un Call option, le processus tangent apparaît dans le calcul.