Rapport de Projet : Stratégie de Trading Pairs sur Binance

Nicolas Agraffel & Louison Poilpot
M2 FINANCE DATA – Université de rennes

Amélioration et Optimisation d'une Stratégie de Trading Algorithmique

Introduction

Le trading algorithmique est devenu un outil incontournable dans les marchés financiers modernes, notamment dans le domaine des cryptomonnaies où la volatilité et la liquidité offrent des opportunités uniques. Ce projet vise à développer, tester et optimiser une stratégie de trading pairs entre deux actifs cryptographiques, Ethereum (ETH) et SafePal (SFP), en utilisant l'API Binance. L'objectif est de créer une stratégie robuste, basée sur des indicateurs statistiques, capable de générer des rendements attractifs tout en minimisant les risques.

Ce rapport présente une analyse approfondie de la stratégie, depuis la récupération des données jusqu'à l'optimisation des paramètres, en passant par la simulation et l'évaluation des performances. Des améliorations significatives ont été apportées pour renforcer la méthodologie, les résultats et les conclusions.

Contexte et Objectifs

Contexte

Le trading pairs, ou "statistical arbitrage", est une stratégie populaire qui consiste à exploiter les écarts de prix entre deux actifs corrélés. L'idée est de profiter des divergences temporaires pour générer des profits tout en limitant l'exposition au risque de marché.

Dans ce projet, nous nous concentrons sur les paires ETHUSDT et SFPUSDT, en utilisant le z-score du spread (écart de prix) comme indicateur principal pour prendre des décisions de trading.

Objectifs

1. Développer une stratégie de trading pairs basée sur le z-score du spread entre ETH et SFP.

2. Simuler et optimiser la stratégie sur des données historiques pour évaluer sa performance.

3. Tentative d'implémentation de la stratégie en temps réel sur Binance Testnet.

4. Comparer les performances de la stratégie avec une approche Buy & Hold traditionnelle.

Architecture du Code

Le code est structuré en quatre modules principaux :

1. Récupération des Données

- Utilisation de l'API Binance pour obtenir les données historiques de prix (OHLC) pour ETHUSDT et SFPUSDT a la minute sur environ un an. Pour faciliter l'import des données qui prend du temps, un CSV est fourni avec le code.

- Calcul du spread (différence entre les prix de clôture) et normalisation via le z-score sur une fenêtre glissante de 60 minutes.

2. Simulation de la Stratégie

- La fonction « Stratégie » utilise des seuils dynamiques pour les entrées et sorties, ainsi qu'un stop-loss de -3 %.

- Les paramètres clés (seuil de z-score, décalage d'entrée, frais de transaction) sont ajustables pour optimiser la performance.

3. Optimisation et Analyse

- La stratégie est testée sur des fenêtres glissantes de 3 jours pour déterminer le seuil optimal. Une fois le seuil trouvé, il est utilisé pour les 3 prochains jours.

- Les performances sont évaluées via des métriques telles que le rendement total et le ratio de Sharpe. Comme précédemment, un CSV avec les résultats est fourni pour éviter l'attente de lancement de la stratégie (environ 20 minutes).

4. Implémentation en Temps Réel

- Une section du code est dédiée à l'exécution en temps réel sur Binance Testnet, avec des ordres passés automatiquement en fonction des signaux générés par le z-score. Il faut y rentrer son API personnelle. C'est un exemple de code, l'implémentation n'ayant pas été réussi.

Méthodologie

1. Calcul du Spread et Z-Score

Le spread est calculé comme la différence entre les prix de clôture de ETH et SFP :

Spread = Close_ETH − Close_SFP

Le z-score est normalisé sur une fenêtre glissante de 60 minutes :

z-score = (spread − spread_mean) / (spread_std)

Où :

- spread_mean est la moyenne mobile du spread
- spread_std est l'écart-type mobile du spread.

2. Stratégie de Trading

Entrée :

Short sur ETH et long sur SFP si z-score > (threshold + entry_offset)
z- score > (threshold+entry_offset)

Long sur ETH et short sur SFP si z-score < − (threshold + entry_offset)
z-score < −(threshold+entry_offset).

Sortie :

La position est fermée lorsque le z-score revient dans une plage normale ou lorsque le stop-loss de -3 % est atteint.

Frais : Des frais de transaction de 0,05 % sont appliqués à chaque changement de position. Ces frais sont calculés par approximation des frais moyen de Binance.

3. Optimisation des seuils

Le seuil optimal est déterminé en maximisant le ratio de Sharpe et en ayant des rendements positifs sur des fenêtres glissantes de 3 jours. La stratégie est ensuite utilisée sur les 3 jours suivants pour évaluer sa performance. Le but étant de rentre le code dynamique aux environnements de marché très volatile que sont les cryptomonnaies. Si aucun seuil n'est trouvé sur les 3jours, alors un décalage de un jour est effectué, pour tenter de calculer, et ainsi de suite.

4. Back Testing
La stratégie ne faisant pas de prévision, il n'y a pas de backtest avec une différente base de données à effectuer. Le seuil est juste ajusté dynamiquement en fonction de ce qui aurait été optimal les jours précédents.

Résultats et Analyse

1. Performance de la Stratégie

- La stratégie génère des rendements variables selon les périodes, avec des ratios de Sharpe allant jusqu'à 0,026, et des performances par trade jusqu'à 2,8%.

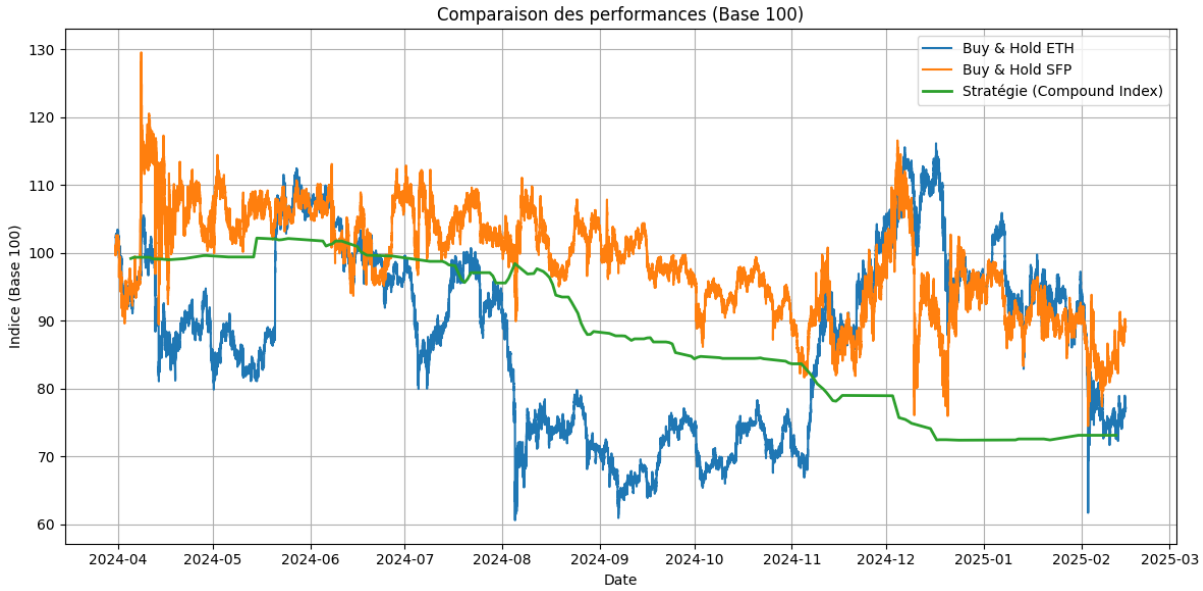
- Comparaison avec Buy & Hold :

- La stratégie est en ligne avec le Buy & Hold ETH sur la période, mais avec des périodes de sur ou sous-performance. De manière générale, on voit que les baisses sont atténuées, mais que les périodes de hausses ne profitent pas à la stratégie.

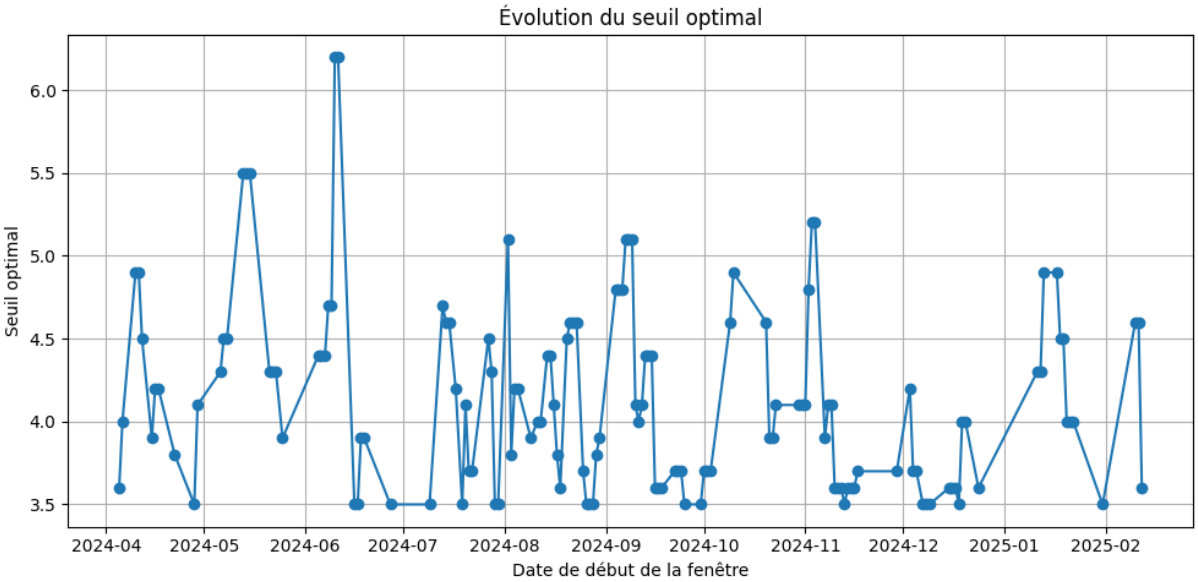
- Exemple : Sur la période du 2024-05-15 au 2024-05-18, le rendement était de 2,8%.

2. Visualisations

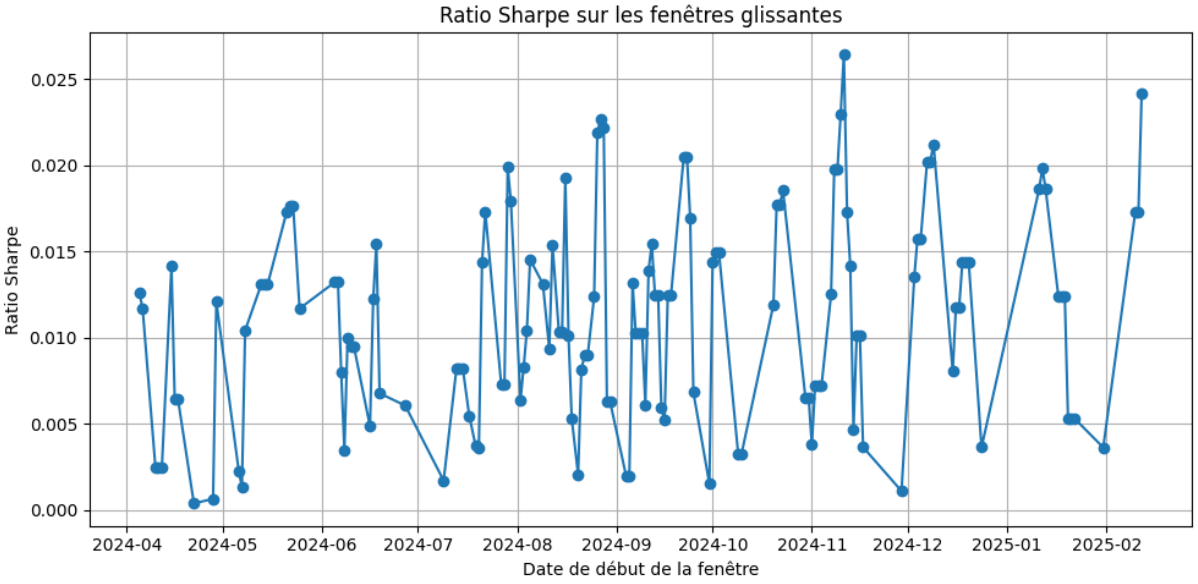
<u>Graphique 1 : Comparaison des performances entre la stratégie et le Buy & Hold pour ETH et SFP.</u>



<u>Graphique 2 : Évolution du seuil optimal au fil du temps.</u>



Graphique 3 : Ratio de Sharpe par fenêtre glissante.



Limites et Améliorations

Limites

- Sensibilité à la volatilité et aux changements de corrélation entre ETH et SFP.

- Une fenêtre de seuil à tester pouvant être améliorer, mais qui prenait trop de temps de calcul pour nos ordinateurs.

- Optimisation sur des fenêtres courtes (3 jours) pouvant manquer de robustesse. La encore, une fenêtre plus large aurait été potentiellement plus robuste, mais cela demandait un temps de calcul trop important.

Améliorations Proposées

- Tester sur d'autres paires : Explorer des paires plus corrélées pour réduire le risque de divergence.

- Ajouter des filtres : Intégrer des indicateurs supplémentaires (volume, RSI) pour éviter les faux signaux.

- Gestion dynamique du risque : Ajuster la taille des positions en fonction de la volatilité.

- Backtesting étendu : Tester la stratégie sur des périodes plus longues et sur différents marchés.

Conclusion

Ce projet démontre la faisabilité d'une stratégie de trading pairs basée sur le z-score du spread entre ETH et SFP. Bien que la stratégie montre des signes de performance, des ajustements sont nécessaires pour améliorer sa robustesse et sa rentabilité à long terme. L'implémentation en temps réel sur Binance Testnet offre une opportunité de valider la stratégie dans des conditions réelles avant de la déployer sur le marché live.

Les résultats obtenus sont encourageants, mais ils soulignent l'importance d'une optimisation continue et d'une gestion rigoureuse du risque.

Recommandations

1. Backtesting Étendu : Tester la stratégie sur des périodes plus longues et sur différentes paires de cryptomonnaies.

2. Optimisation des Paramètres : Explorer des seuils et des fenêtres de calcul du z-score plus adaptés aux caractéristiques du marché.

3. Gestion du Risque : Implémenter des mécanismes de gestion du risque plus sophistiqués pour limiter les pertes lors de périodes de forte volatilité.

4. Automatisation Complète : Déployer la stratégie sur un serveur cloud pour une exécution continue et surveillée.

Annexes

- Fichiers de Données : dataset_crypto_projet.csv, df_results.csv.

- Bibliothèques Utilisées : binance.client, pandas, numpy, matplotlib, yfinance.
- Code Source : Disponible sur https://github.com/agrnicolas/CryptoTrading