Mots clés :

* Mean reversion
* Multithreading ( dask )
* Backtesting

Objectif :

* Backtester en python une strategie de mean reversion en multithreading
* Optimiser une strategie de lean reversion en environnement HPC

Plan :

* Stratégie mean reversion
  + Qu’est-ce que c’est

La stratégie mean reversion est repose sur le postulat que les cours d’un actif financier évoluent autour de la moyenne, ajustant constamment leur trajectoire autour de celle-ci.

* + Définition de la stratégie

Définir les indicateurs observés puis présenter les configurations attendues pour lancer un signal d’achat ou de vente

* + Indicateurs
    - RSI
    - Moyennes mobiles
    - Bande de Bollinger
    - Volumes
    - ATX
  + Cas de figure
* Configuration de l’environnement HPC
  + Installation de Dask
  + Développement de fonctions adaptées
* Préparation des données
* Backtest et simulation
* Optimisation des paramètres

Article : ( from chatgpt )

**Optimisation et Simulation de la Stratégie de Mean Reversion en HPC**

Dans cet article, nous explorons la stratégie de **mean reversion** appliquée aux marchés financiers, et son implémentation en **Python** dans un environnement de calcul haute performance (**HPC**) à l'aide de **Dask**. Nous démontrons également comment backtester et optimiser cette stratégie pour maximiser ses performances.

**1. Stratégie Mean Reversion**

**Qu’est-ce que c’est ?**

La stratégie **mean reversion** repose sur l’idée que les prix des actifs financiers fluctuent autour d’une moyenne à long terme. Lorsqu’un prix s’éloigne trop de cette moyenne, la stratégie suppose un retour probable vers la moyenne, offrant des opportunités d'achat (survente) ou de vente (surachat).

**Définition de la stratégie**

Pour implémenter une stratégie de mean reversion :

1. **Définir la moyenne de référence** : Cela peut être une moyenne mobile (SMA/EMA) ou une valeur statistique calculée.
2. **Déterminer les signaux d'achat ou de vente** :
   * **Achat** : Lorsque le prix est significativement en dessous de la moyenne (sous-évalué).
   * **Vente** : Lorsque le prix est significativement au-dessus de la moyenne (sur-évalué).
3. **Ajouter des filtres** pour renforcer les signaux, comme le **RSI**, les **Bandes de Bollinger**, ou les volumes.

**Indicateurs couramment utilisés**

* **RSI (Relative Strength Index)** : Mesure la vitesse et l’amplitude des mouvements de prix pour détecter des niveaux de surachat/survente.
* **Moyennes mobiles** : Les SMA et EMA sont essentielles pour définir la moyenne de référence.
* **Bandes de Bollinger** : Identifient les écarts significatifs autour de la moyenne.
* **Volumes** : Utilisés pour confirmer la validité des signaux.
* **ATX (Average True Range)** : Indique la volatilité du marché.

**2. Configuration de l’Environnement HPC**

**Installation de Dask**

**Dask** est une bibliothèque Python conçue pour le calcul distribué. Elle permet de traiter efficacement de grandes quantités de données en exploitant le multi-threading ou le multi-processing.

1. **Installation via pip** :
2. pip install dask
3. **Création d’un client Dask** :
4. from dask.distributed import Client
5. client = Client(n\_workers=4, threads\_per\_worker=2)

**Développement de fonctions adaptées**

Dask facilite l'exécution parallèle de tâches répétitives. Par exemple, calculer la moyenne mobile ou le RSI sur plusieurs actifs simultanément.

**3. Préparation des Données**

1. **Acquisition** : Collecter des données de prix historiques via des API telles que **Yahoo Finance**, **Alpha Vantage**, ou **Quandl**.
2. **Nettoyage** :
   * Supprimer les valeurs manquantes.
   * Standardiser les formats (dates, fuseaux horaires, etc.).
3. **Organisation pour le calcul parallèle** :
   * Convertir les données en **Dask DataFrame** pour les partitions :
   * import dask.dataframe as dd
   * data = dd.from\_pandas(prices\_df, npartitions=10)

**4. Backtest et Simulation**

**Développement d’un outil de backtesting**

Voici un exemple de fonction de backtesting simple pour une stratégie de mean reversion :

def backtest(prices, short\_window=20, long\_window=50):

short\_ma = prices.rolling(short\_window).mean()

long\_ma = prices.rolling(long\_window).mean()

signals = (short\_ma > long\_ma).astype(int)

returns = prices.pct\_change() \* signals.shift(1)

return returns.sum() # Rendement total

**Exécution parallèle du backtesting**

Tester plusieurs configurations de paramètres (fenêtres de moyenne mobile) en parallèle avec Dask :

from dask import delayed, compute

configs = [(10, 50), (20, 100), (30, 150)]

tasks = [delayed(backtest)(prices, sw, lw) for sw, lw in configs]

results = compute(\*tasks)

print("Résultats :", results)

**5. Optimisation des Paramètres**

**Recherche de paramètres optimaux**

1. **Approche** : Effectuer une recherche sur grille (**Grid Search**) ou une recherche aléatoire pour trouver les meilleures combinaisons de paramètres (fenêtres, seuils, etc.).
2. **Optimisation avec Dask** :
   * Tester des milliers de combinaisons en parallèle.
   * Exemple d’optimisation :
   * def objective(params):
   * short\_window, long\_window = params
   * return -backtest(prices, short\_window, long\_window) # Maximiser les gains
   * params\_list = [(10, 50), (20, 100), (30, 150)]
   * tasks = [delayed(objective)(p) for p in params\_list]
   * best\_result = compute(\*tasks)
   * print("Meilleur résultat :", min(best\_result))

**Visualisation des résultats**

Pour mieux interpréter les résultats des backtests et des optimisations, utilise des outils comme **Matplotlib** ou **Seaborn** pour créer des graphiques de performance :

import matplotlib.pyplot as plt

plt.plot(prices.index, prices, label="Prix")

plt.legend()

plt.show()

**Conclusion**

L'utilisation de **Dask** pour implémenter une stratégie de mean reversion dans un environnement HPC permet de backtester et d'optimiser efficacement la stratégie. Cette approche offre une scalabilité importante et des résultats plus rapides, essentiels dans des environnements financiers exigeants.

En combinant des outils comme Python, Dask, et des indicateurs techniques robustes, cette méthode constitue une solution puissante pour développer des stratégies de trading quantitatives.