Olivier Braibant
Florian Cebeci

PROJET OPTIMISATION PORTFOLIO

Abstract

La théorie moderne du portefeuille développé par Markowitz (frontière efficiente) est un modèle d'allocations d'actifs visant à optimiser le couple rendement / risque d'un portefeuille boursier. Cette théorie permet de sélectionner des actifs dans une optique de diversification. Pour lui, le concept de diversification du portefeuille est à la base des choix d'investissements mais ce n'est pas suffisant, il faut optimiser cette diversification. Les actifs doivent être sélectionnés de manière globale en tenant compte des différentes corrélations entre leurs variations. L'objectif étant de réduire le risque au maximum pour un niveau de rendement donné. C'est le principe de la frontière efficiente.

Ce projet a pour but de d'utiliser les concepts abordés par cette théorie en proposant de déterminer la composition optimale d'un portefeuille d'actions.

Dans ce projet, nous proposons également d'utiliser la puissance des outils de calcul numériques tel que l'optimisation sous contraintes, la simulation de Monte Carlo et la simulation Random Walk afin d'obtenir des résultats statistiques plus robustes sur la composition optimum d'un portefeuille.

1. Rapide Rappel sur la Théorie Moderne du portefeuille

La théorie moderne du portefeuille, développée par Harry Markowitz dans les années 1950, définit le processus de sélection de titres pour créer le portefeuille le plus efficient possible, c'est à dire qui possède la rentabilité maximum pour un niveau de risque minimum.
Le concept dediversification est à la base de la théorie. En effet, Markowitz pense que les différents titres composant un portefeuille ne peuvent être sélectionnés individuellement et doivent au contraire être choisi selon la corrélation de leurs variations à celles du reste des actifs du portefeuille.
Ce mode de sélection permet de minimiser le risque pour un niveau de rendement choisi.
Analysons les différents composants du modèle :

Makowitz présuppose que les investisseurs sont rationnels et averses au risque et que le marché est efficient. Ainsi, les seuls éléments à prendre en compte sont le risque et le rendement des titres, car les investisseurs achèteront toujours l'actif qui présente un rendement optimal par rapport à son niveau de risque. Aucun investisseur purement rationnel n'achèterait en effet un actif A plus risqué qu'un actif B mais offrant un rendement inférieur.

Dans le modèle, le rendement d'un portefeuille consistera en la somme des rendements des actifs qui le composent, pondérés par leur poids.

Soit :

$$\mathrm{E}(R_p) = \sum_i w_i \mathrm{E}(R_i)$$

Dans notre projet, nous avons utilisé le concept de log return pour calculer le rendement E(Ri) de chaque actif.

Celui-ci se définit comme suit :

$$ln\left(\frac{P_t}{P_0}\right) = \ln(1+r) = R$$

où Pt et Po sont les prix de l'actif au temps t et t-1 et R le log return ainsi calculé.

Le risque est défini par la volatilité du portefeuille qui correspond à son écart-type :

$$\sigma_p = \sqrt{\sigma_p^2}$$

avec σ_p^2 la variance que l'on calcule de la manière suivante, pour un portefeuille composé de deux actifs :

$$\sigma_p^2 = \sum_{i=1}^{n}\sum_{j=1}^{n} w_i w_j \sigma_{ij} = \sum_{i=1}^{n}\sum_{j=1}^{n} w_i w_j \sigma_i \sigma_j \rho_{ij}$$

avec :

σ_{ij} : covariance entre les deux actifs que l'on peut exprimer en $\sigma_{ij} = \sigma_i \sigma_j \rho_{ij}$

σ_i^2 : la variance de l'actif et ρ_{ij} la corrélation entre les deux actifs.

Pour n actifs, les variances et covariances sont repris dans une matrice que l'on appelle matrice de Variance-CoVariance.

Avec ces éléments en main, on peut tester différentes combinaisons d'actifs avec des pondérations diverses pour calculer le risque et la rentabilité espérée d'un portefeuille. La diversification par la sélection d'actifs plus ou moins corrélés permettra d'optimiser cette relation rendement/volatilité.

2. Objectif de notre programme

Notre programme réalise les objectifs suivants :

- Acquisition des données sur les actifs : le programme construit un dataframe contenant toutes les informations de prix des actifs
- Calcul des rendements pour chaque actif et de la matrice de Variance/Covariance
- Calcul du portefeuille de rendement maximum (sur base du ratio de Sharp)
- Visualisation de la frontière d'efficacité (scatter plot)
- Calcule du portefeuille de risque minimum. 2 méthodes sont proposées :
    - simulation de Monte Carlo
    - optimisation sous contrainte

- Simulation des prix des actifs (type Random Walk) sur une période de 252 jours et calcul des 2 portefeuilles optimum qui s'en suivent.

3. Aperçu de du programme

Le programme se compose de 9 fonctions :

```
def calc_stock_data(stocks, start_date, end_date):

    """
    crée un dataframe contenant les prix des actions (close price)

    :param stocks : liste (of strings) contenant les actions du
portefeuille
    :param start_date : date de début
    :param end_date : date de fin

    :return returns : array contenant la moyenne des retuns journaliers des
actions
    :return matrix_cov : array contenant la matrice var/cov des actions
    :return stock_price : dataframe contenant les prix des actions
    """
```

```
def perf_portfolio(poids, returns, cov_matrix):

    """
    Calcul la performance annuelle et le risque d'un portefeuille en tenant
compte de la pondération individuelle
    des actions

    :param poids: array/DF contenant les pondérations du portefeuille
    :param returns: array/DF contenant le return journalier moyen des
actions
    :param cov_matrix: array/DF contenant la matrice var/cov des returns
journaliers
```

```
    :return return_port: un float égale au return annuel du portefeuille
    :return risk_port: un float égale au risque du return annuel du
portefeuille
    """
```

```
def poids_random(stocks):
    """
    Crée un np.array avec des pondérations aléatoires standardisées (somme
= 1)
    :param stocks: liste (string) contenant les actions du portefeuille

    :return poids: array contenant les pondérations aléatoires
standardisées
    """"
def portfolio_mc_simulation(stocks, start, end, nb_sim):
    """
    simulation de Monte Carlo: réaliser une simulation Monte-Carlo afin de
sonder l'espace retour-volatilité pour
    des portefeuilles composés d'actions.

    - La fonction identifie le portefeuille ayant le ratio de Sharp maximum
dans l'espace risque/return
    - La fonction identifie le portfeuille auant le risque minimum dans
l'espace risque/return
    - La fonction dessine un scatter plot de chaque simulation dans
l'espace risque/return

    :param stocks : liste (of strings) contenant les actions du
portefeuille
    :param start : date de début (utilise le module datetime as dt)
    param end_date : date de fin (utilise le module datetime as dt)
    :param nb_sim : nombre de simulations

    :return NONE
    """
```

```
def variance_port(poids, moy_return, cov_matrix):
    """
    Cette fonction est utilisée uniquement dans le cadre de la fonction
d'optimisation 'minimum_variance'
```

```
    :param poids: array/DF contenant les pondérations du portefeuille
    :param moy_return: array/DF contenant le return journalier moyen des
actions
    :param cov_matrix: array/DF contenant la matrice var/cov des returns
journaliers

    :return: renvoie le risque du portfeuille (retun de la fonction
perfportfoilio[1])
    """
```

```
def sharp_ratio_opp(poids, returns, cov_matrix, ss_risque=0):
    """
    Calcul du Sharp ratio (négatif ! afin d'utiliser la fonction
scipy.optimize.minimize de la librairie SciPy)
    d'un portefeuille pour une pondération donnée.

    :param poids: np.array contenant la pondération des actions
    :param return: array contenant la moyenne

    :return -sharp_ratio contenant la valeur negative du sharp ratio
    """
```

```
def max_sharp_ratio(returns, cov_matrix, ss_risque=0,
constraints_set=(0,1)):
    """
    Cette fonction calcule le portefeuille avec le ratio de Sharp maximum
en utilisant un algorythme de minimisation sous
    contraintes de la librairie SciPY.

    :param returns: array/DF contenant le return journalier moyen des
actions
    :param cov_matrix: array/DF contenant la matrice var/cov des returns
journaliers
    :param ss_risque: taux sans risque du marché (fixé à zero)
    :param constraints_set: les bornes pour les variables d'optimisation

    :return: none
    """
```

```
def minimum_variance(returns, cov_matrix, constraints_set=(0,1)):
    """
    Cette fonction calcule le portefeuille avec le risque minimum en
utilisant un algorythme de minimisation sous
    contraintes de la librairie SciPY.

    :param returns: array/DF contenant le return journalier moyen des
actions
    :param cov_matrix: array/DF contenant la matrice var/cov des returns
journaliers
    :param constraints_set: les bornes pour les variables d'optimisation

    :return: none
    """
```

```
def random_walk(stocks, start_date, end_date, nb_sim, nb_walk):
    """
    Cette fonction va exécuter un Random Walk sur les prix des actions pour
une période de 252 jours en partant du
    dernier prix connu de chaque action. Elle va ensuite exécuter une
simulation de Monte Carlo sur base des prix
    simulés pour chaque action.
    Sur base de cette dernière simulation, la fonction identifie:

        - le portefeuille ayant le ratio de Sharp maximum dans l'espace
risque/return
        - le portfeuille auant le risque minimum dans l'espace
risque/return

    La fonction dessine enfin un scatter plot de chaque simulation dans
l'espace risque/return

    :param stocks: liste (of strings) contenant les actions du portefeuille
    :param start_date: date de début
    :param end_date:
    :param nb_sim: nombre de simulation de Monte Carlo pour le portefeuille
    :param nb_walk: nombre de random walk pour chaque action
    :return:
    """
```

4. Résultats

Nous nous donnons la liste des actifs suivants :

- Microsoft (MSFT)
- IBM (IBM)
- Facebook (META)
- Google (GOOG)
- Visa (V)
- Johnson & Johnson (JNJ)
- Procter & Gamble (PG)

Voici les résultats obtenus :

Portefeuille avec ratio de Sharp Max (SR) :

1) Méthode d'optimisation sous contraintes (fonction max_sharp_ratio)

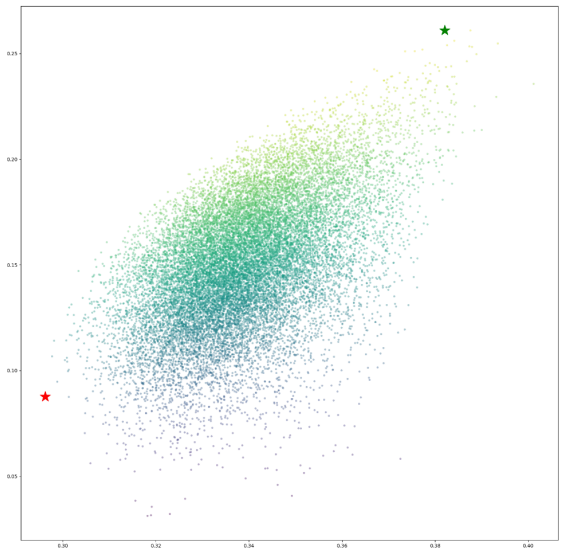
Optimisation réalisée. Sharp Ratio optimisé = 0.73

POIDS :

GOOG 6.941555e-02
IBM 0.000000e+00
JNJ 1.745565e-14
META 1.038666e-14
MSFT 9.993058e+01
PG 0.000000e+00
V 0.000000e+00

Return Portfolio Optimisation SR 32.34 %

2) Méthode avec Simulation Monte Carlo



Voici le portefeuille à ratio de Sharpe maximum:

returns 0.260940

risque 0.382042

sharpe_ratio 0.683013

MSFT poids 0.368815

IBM poids 0.000411

META poids 0.016174

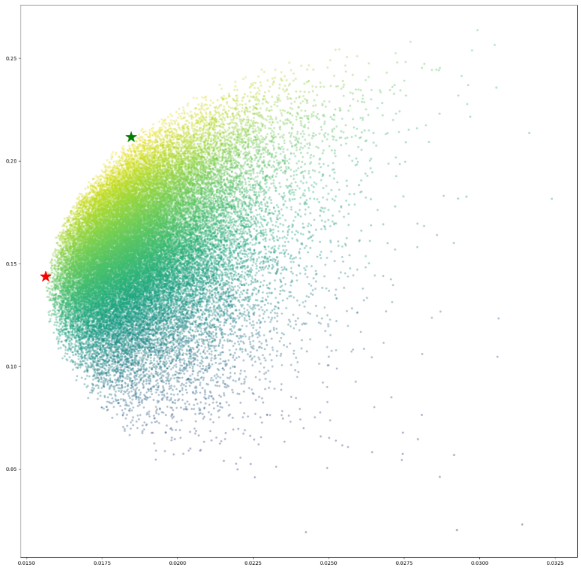
GOOG poids 0.164342

V poids 0.364781

JNJ poids 0.062547

PG poids 0.022930

3) Méthode Random Walk + Simulation Monte Carlo



Voici le portefeuille à ratio de Sharpe maximum:

returns	0.211651
risque	0.018463

sharpe_ratio	11.463545
MSFT poids	0.198824
IBM poids	0.001567
META poids	0.157640
GOOG poids	0.199458
V poids	0.157631
JNJ poids	0.072022
PG poids	0.212857

Portefeuille avec risque minimum :

1) Méthode d'optimisation sous contraintes (fonction minimum variance)

Optimisation réalisée. Risque du portefeuille: 0.2919904023523956

Poids des actions pour Risque Minimum:

GOOG 8.540728e+00

IBM 0.000000e+00

JNJ 5.792210e+01

META 6.220584e+00

MSFT 3.005951e-15

PG 2.731659e+01

V 0.000000e+00

Return du portefeuille de Risque Minimum 10.59 %
Sharp Ratio du portefeuille de Risque Minimum: 0.36

2) Méthode avec Simulation Monte Carlo

Voici le portefeuille à risque minimum:
returns 0.107037
risque 0.295090
sharpe_ratio 0.362726

MSFT poids 0.103488
IBM poids 0.026675
META poids 0.458185
GOOG poids 0.034589
V poids 0.008640
JNJ poids 0.321943
PG poids 0.046480

3) Méthode Random Walk + Simulation Monte Carlo

Voici le portefeuille à risque minimum:
returns 0.143690
risque 0.015638
sharpe_ratio 9.188730

MSFT poids	0.075446
IBM poids	0.116610
META poids	0.110353
GOOG poids	0.150845
V poids	0.157320
JNJ poids	0.195499
PG poids	0.193928

<u>References (manque de temps)</u>

Nous donnerons la liste des références lors de notre présentation