Rapport projet optimisation de portfolio

Malo Perrien, Armand Morin, Amaury Revel

CentraleSupélec – Projet ST7

Minimisation des risques sur les marchés Optimisation de portefeuille

On travaille avec un portefeuille qui est un vecteur de positions, avec une variable de cash stockée dans une autre variable.

Notre fonction objectif est la suivante :

$$F(P_t) = P_t \Sigma P_t - \alpha < \mu, P_t > + \gamma TC(P_t)$$

$$\gamma TC(P_t) = \gamma \sum_i \lambda |P - t(i) - P_{t-1}(i)|$$

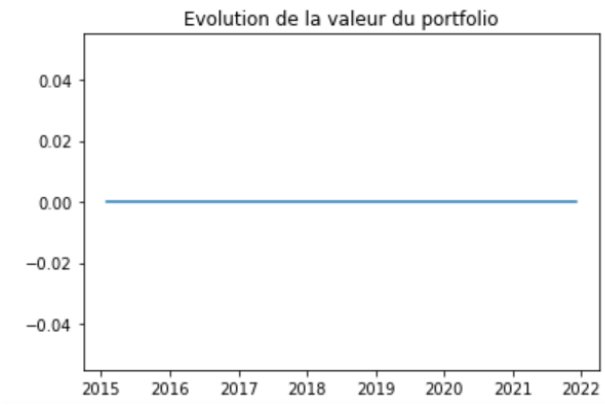
Où α et γ sont des hyperparamètres qu'on fait varier. On appelle λ le coût unitaire de transaction pour l'achat ou la vente d'actifs.

On y rajoute des contraintes :

- Le cash (cash(t+1) = pf_value(t) + cash(t) – TC(t+1) – pf_value(t+1)) doit toujours être supérieur ou égal à une certaine constante. On l'avait pris égale à 0 au début du projet, mais finalement on a enlevé cette contrainte
- Les gains prévus doivent être supérieurs aux coûts de transaction (ou n'importe quelle autre valeur) : peu utile, très vite éliminée
- Pour chaque actif, on ne peut pas faire changer la position de +/- dPmax (en nombre d'actions) : ça a permis d'avoir des courbes plus lisses, mais on a changé pour la condition suivante
- Pour chaque actif, on ne peut en changer plus de dPmax% de la valeur déjà investie : mieux
- Enfin, on doit avoir abs(pf_value(t+1) - pf_value(t))/pf_value(t) <= dNmax

Calcul des expected returns :

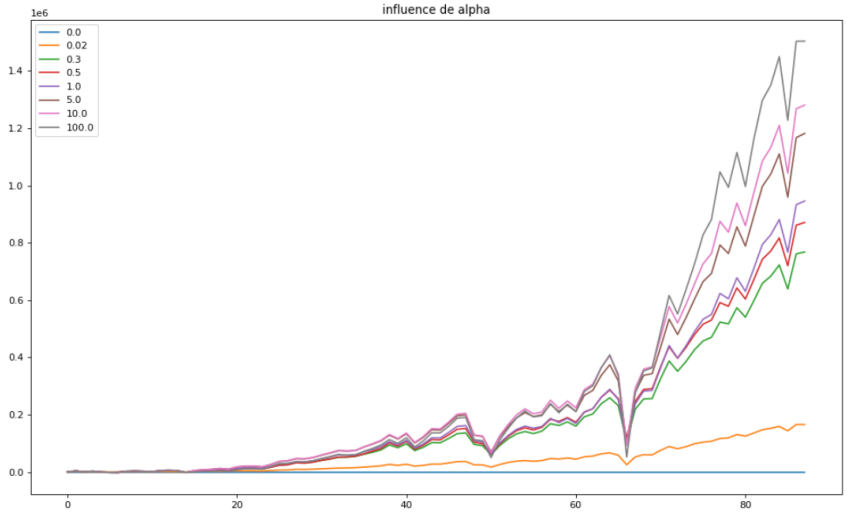
On a longtemps travaillé avec une standard moving average, sur les x dernières données accessibles (et on a pris x=20), puis on a essayé exponential moving average, qui a semblé donner de meilleurs résultats, donc c'est la méthode qu'on a gardée jusqu'à obtenir à la toute fin un résultat satisfaisant avec l'utilisation de multiples signaux dans pycaret.



Alpha = gamma = 0, aucune contrainte

Pour minimiser le risque : on ne fait rien.

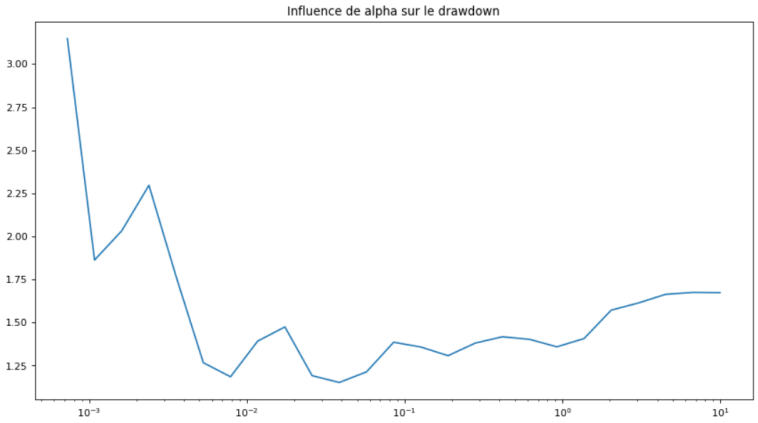
1) Influence de alpha : l'importance accordée aux returns



On se place dans un cas usuel avec lambda=0.01, gamma=1, c'est à dire on prend en compte les coûts de transaction. On n'a pas de contraintes.

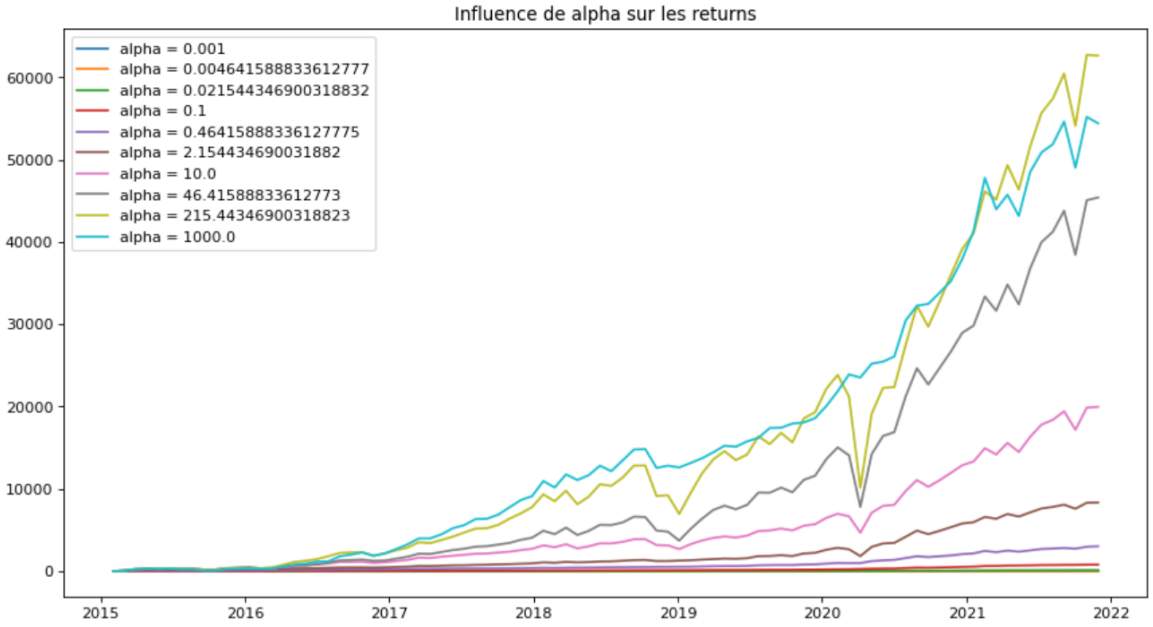
On remarque que plus on accorde de l'importance au return (i.e. plus alpha augmente), plus le portfolio semble rapporter de l'argent. On remarque également avec ces profils qu'augmenter alpha augmente le risque : les montées et chutes sont plus abruptes lorsque alpha est grand. C'est très apparent pour la chute autour de la 65ème période.

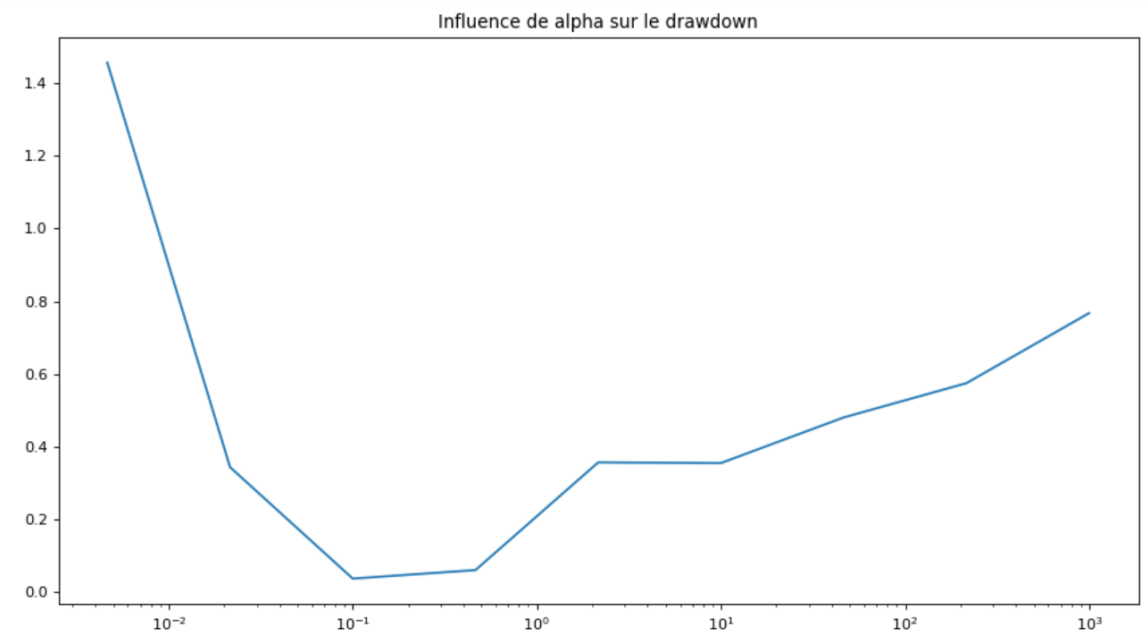
On peut observer l'évolution du drawdown en fonction de alpha (toujours avec lambda=0,01 et sans contraintes). Voici les valeurs de drawdown obtenues dans cette configuration avec un alpha balayant la plage $[10^{-4}, 10]$.



On observe un minimum pour des valeurs de alpha autour de 0,03. Si alpha est trop faible, la valeur du portfolio est proche de 0 et devient plus sensible aux fluctuations. Si alpha est trop grand, la stratégie d'allocation du portfolio est déterminée par les expected returns : le risque augmente et les fluctuations aussi.

1bis) Influence de alpha sous différentes contraintes : long-only, cash investi limité à 10k + i*500 (i = la i-ème période à laquelle s'effectue le rebalancement càd, le i-ème mois), et dPmax = 100%

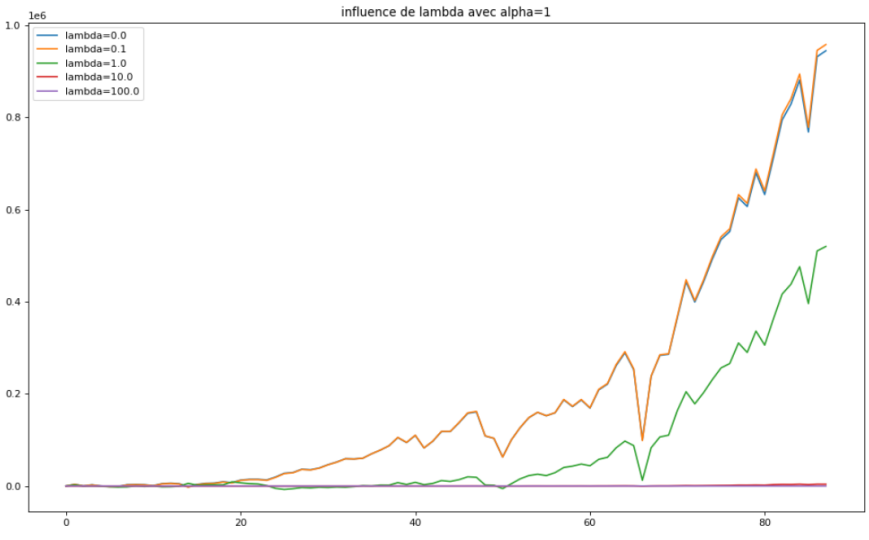




On retrouve la même tendance qu'en 1 :

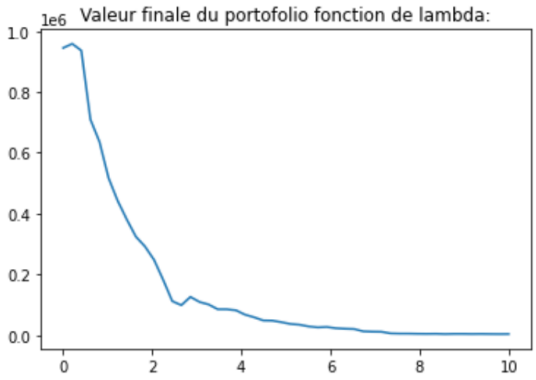
2) Influence de lambda, le prix unitaire des couts de transactions

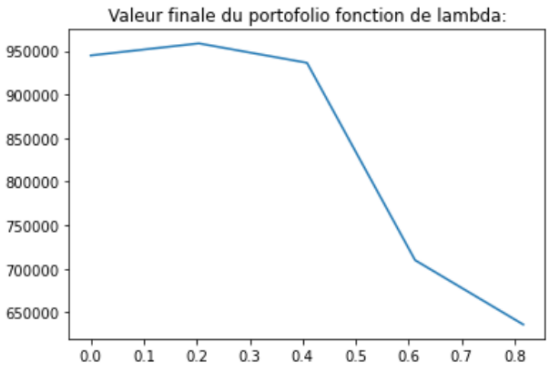
On se place dans un cas usuel avec alpha=1, c'est à dire on prend en compte les returns. Il n'y a pas de contraintes.



Le comportement est celui attendu : lorsque lambda (le cout de transaction) devient trop grand, il ne se passe plus rien. De plus les portfolios sont plus performants lorsque lambda diminue.

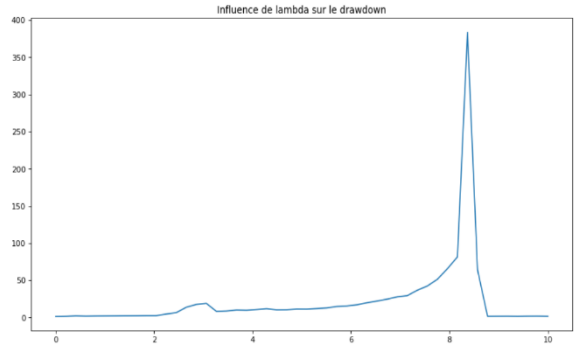
On peut de s'intéresser à la valeur finale du portofolio en fonction de lambda, on obtient le profil suivant :





On remarque un plateau pour les valeurs faibles de lambda, puis une chute brutale dès que les coûts de transaction dépassent 0,5. Avant cette valeur, le portfolio ne semble pas affecté par l'introduction des coûts de transaction. On pourrait régler avec un coefficient l'importance accordé aux coûts de transactions (gamma) si on souhaite décaler cette valeur de seuil de 0,5.

On peut s'intéresser à l'évolution du drawdown en fonction de lambda (toujours avec alpha=gamma=1 et sans contraintes).



Le drawdown est relativement stable avec lambda pour les petites valeurs. Le pic observé pour lambda=8 correspond au fait que la valeur du portfolio devient très proche de 0 et donc fluctue très fortement en pourcentage. Pour des valeurs de lambda supérieurs à 9, il ne se passe plus rien, le drawdown est nul.

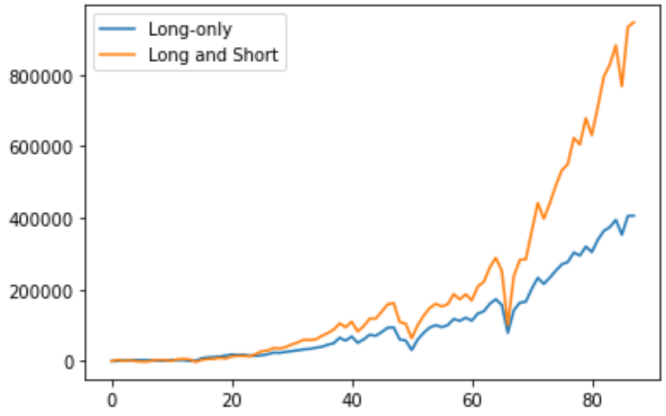
3) Contraintes

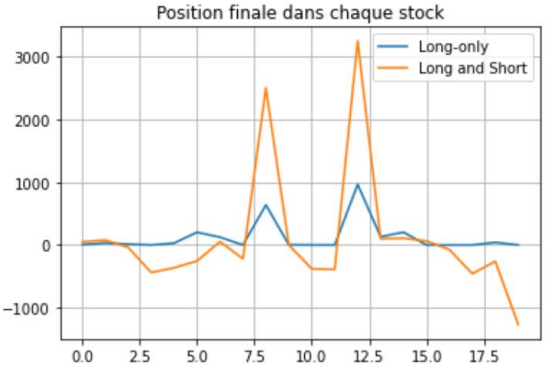
3.1 Long-Only

On peut imaginer un certain nombre de contraintes en fonction de nos objectifs.

On pourrait vouloir travailler en long-only dans l'idée de limiter le risque et le drawdown.

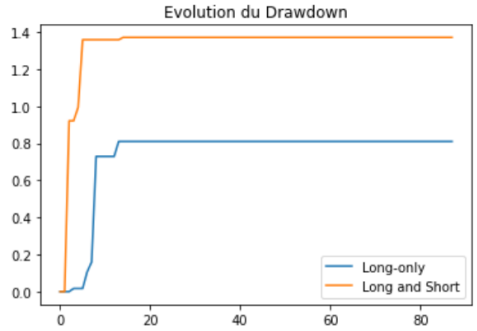
Voici la comparaison des résultats des deux portfolios : avec les paramètres alpha=1, et lambda=0,01 et sans autres contraintes





On observe que l'algorithme vérifie bien la contrainte de long-only. Le premier avantage du portfolio long-only est que cela limite le nombre de positions qu'on va avoir, et donc le risque. Il semblerait aussi que ce portfolio soit moins gourmand en investissement, on aurait pu s'attendre à l'inverse.

A l'inverse l'avantage du short est de pouvoir faire plus si le cash qu'on possède est limité.

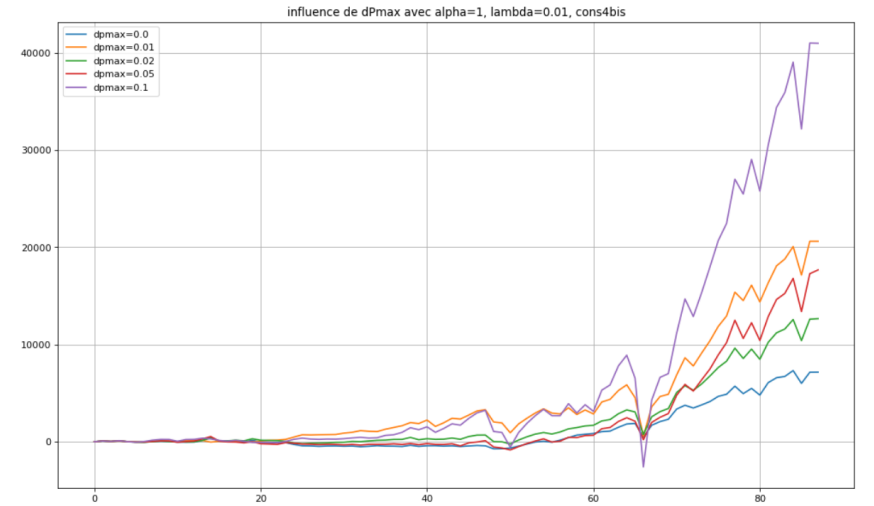


Dans ce cas, on a bien un drawdown plus faible pour le long-only. A noter que la valeur du drawdown est fixé par les débuts du portfolio qui connait de grandes fluctuations comme on démarre de 0.

3.2 dPmax –Changements de positions limités

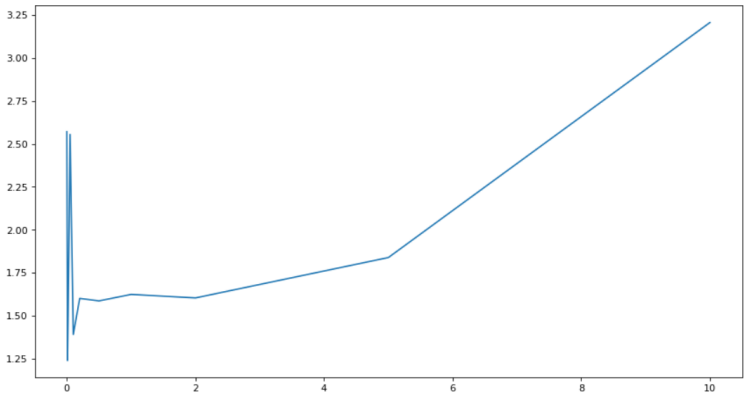
```
def cons4bis(x,pos,dPmax):
    # pour chaque actif, on ne peut en changer (en notionnel) plus de dPmax% de la valeur déjà investie dans l'actif
    return min([dPmax*abs(pos[i]) - abs(x[i]-pos[i]) for i in range (nb_actifs)])
```

L’objectif est de lisser d’avantage l’évolution de la valeur de notre porfolio en limitant la variation de la position sous le pourcentage dPmax de la valeur de la position.



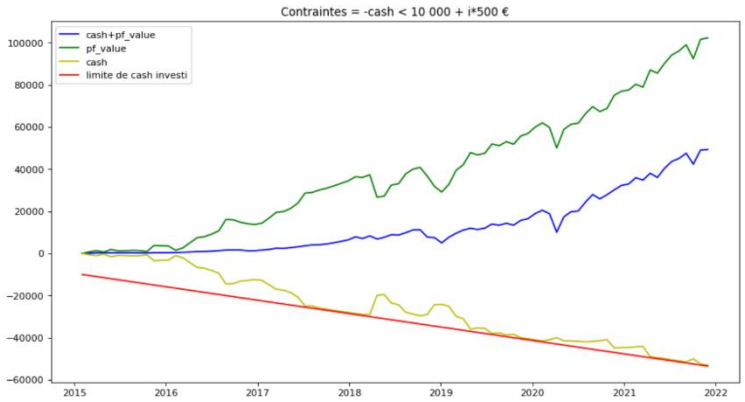
Déjà visuellement, on voit clairement que les variations augment lorsque dPmax augmente.

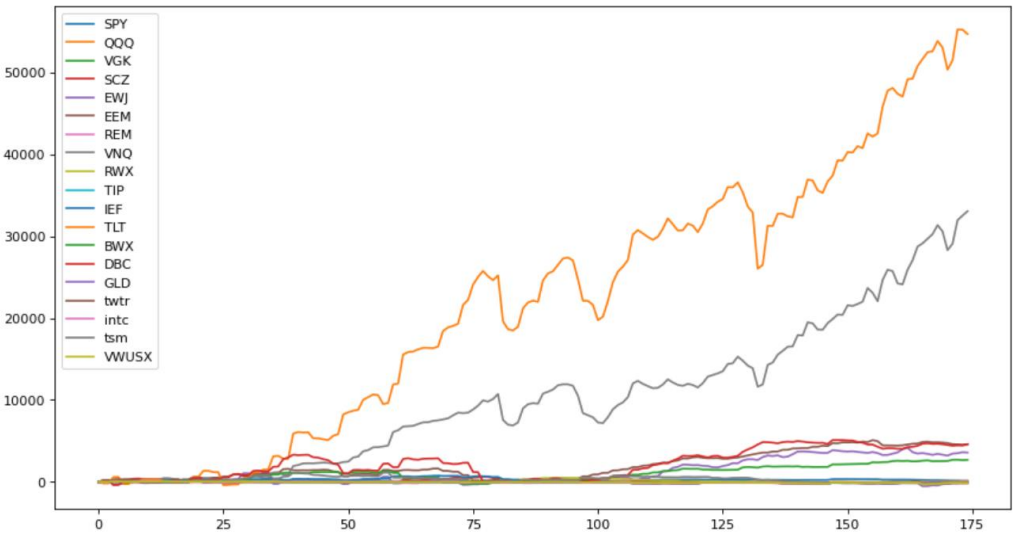
Maintenant le drawdown en fonction de dPmax nous donne :



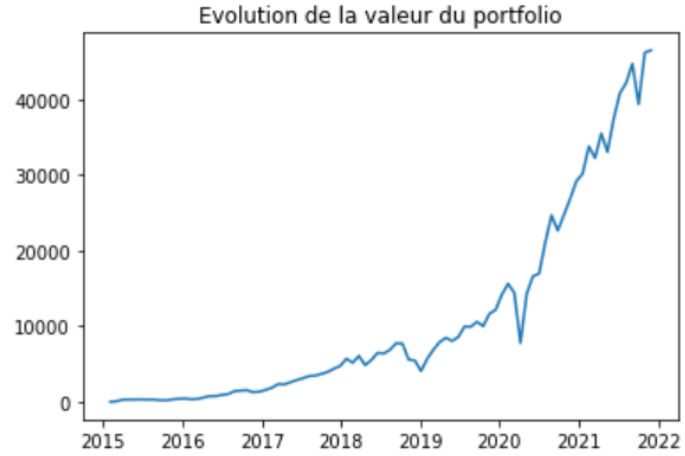
On observe qu’un dPmax trop proche de 0 est problématique. D’ailleurs le programme ne semble pas en mesure de tenir la contrainte. De plus, limiter les variations du nombre d’action a bien l’affet de diminuer la valeur du drawdown, cependant on perd dans la quantité d’argent gagné.

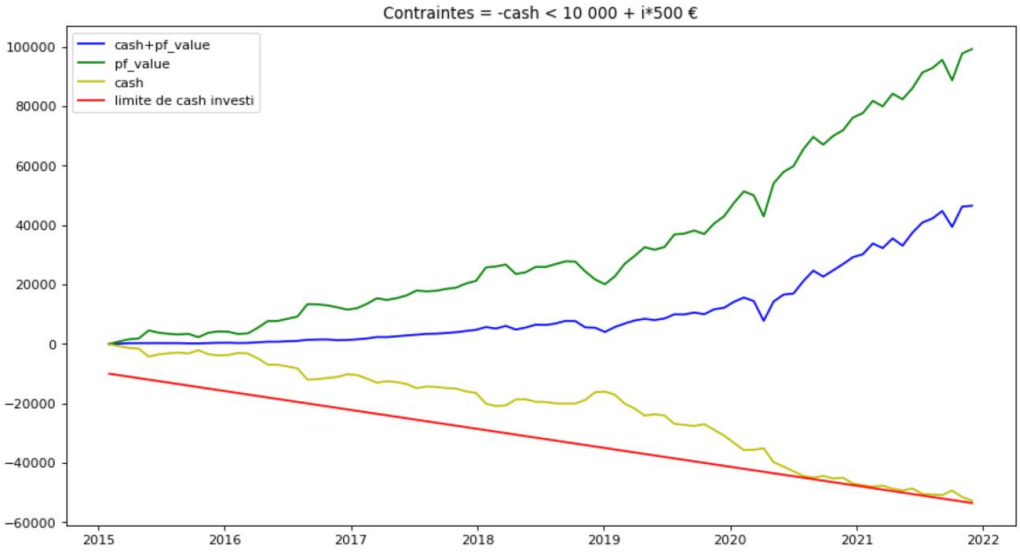
Avec matrice de covariance (long only)

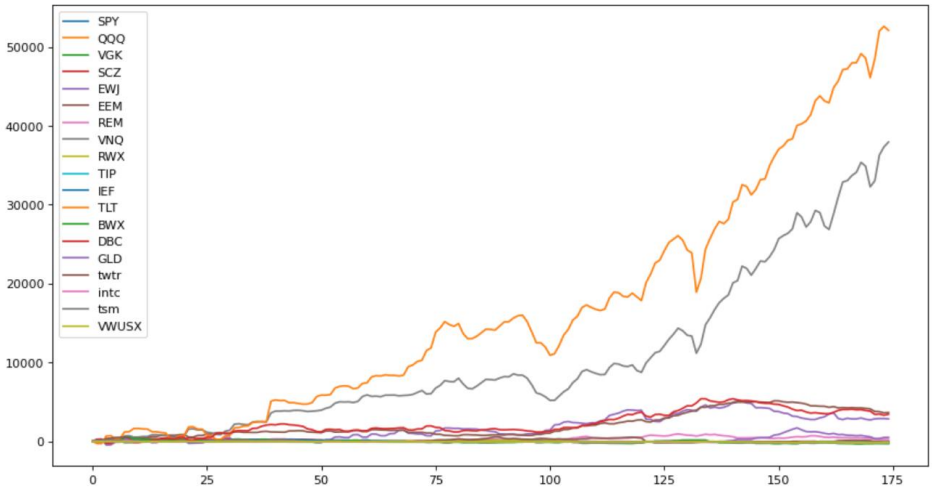




Avec matrice de corrélation (long only)





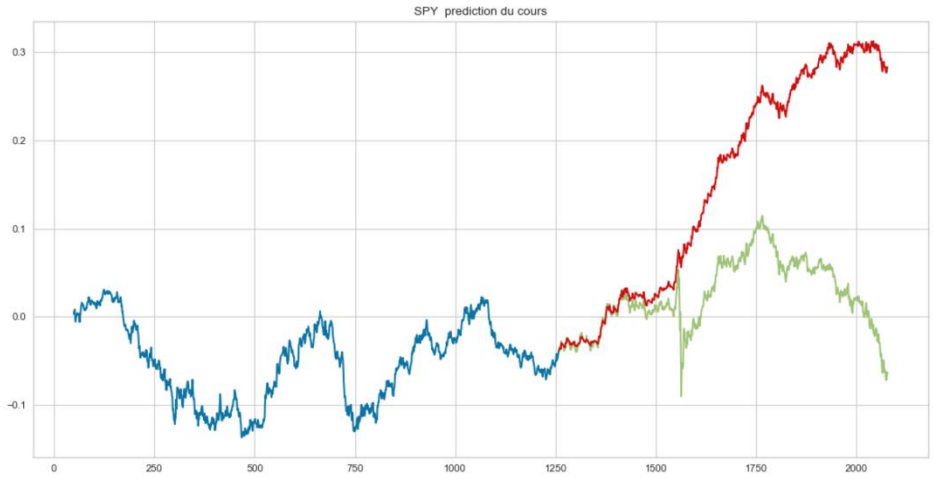


```
drawdown
```

```
0.5018666338248637
```

Prédiction :

On utilise pycaret pour faire une prédiction sur les expected returns. On remarque que sur une courte période au début des données de test la prédiction est bonne après quelques ajustements. Ensuite elle dévie. Mais on utilise cette prédiction que sur un mois (on la refait tous les mois) donc ce n'est pas un problème. La prédiction utilise beaucoup de signaux (générés par des bibliothèques de python) RSI, MACD, SMA, EMA, BB, Volume, Low, High, Mom, StochasticRSI, ForceIndex, etc...



On n'a pas eu le temps de tester en direct l'efficacité des prédictions dans les performances du portefeuille par manque de temps. On n'a utilisé simplement les moyennes mobiles.

Zoom sur la performance :

Backtesting par pyfolio:

Start date	2015-03-31
End date	2021-11-30
Total months	4

	Backtest
Annual return	30.206%
Cumulative returns	9.426%
Annual volatility	4.132%
Sharpe ratio	6.41
Calmar ratio	53.11
Stability	0.92
Max drawdown	-0.569%
Omega ratio	4.41
Sortino ratio	20.42
Skew	2.79
Kurtosis	12.32
Tail ratio	1.98
Daily value at risk	-0.415%