Deep learning for efficient frontier calculation in finance. *

Xavier Warin †

February 15, 2022

Abstract

We propose deep neural network algorithms to calculate the efficient frontier in the Mean-Variance and Mean-CVaR portfolio optimization problems. Starting with the Mean-Variance portfolio optimization problem in the Black Scholes framework, we first compare the analytical solution to the computed one and show the efficiency of the methodology in high dimension. Adding additional constraints, we compare different formulations and, getting the same frontier, show that the no short-selling and no borrowing problem can be solved with all formulations. Then a new projected feedforward network is shown to be able to deal with some local and global constraints on the weights of the portfolio while outperforming classical penalization methods. We extend our numerical results to assets following the Heston model and show that the results obtained in the Black Scholes method still hold. At last we give numerical results for the more difficult Mean-CVaR optimization problem getting realistic results but only for algorithms with a global resolution of the problem.

Key words: Deep neural networks, finance, Mean-Variance, Mean-CVaR, efficient frontier.

1 Introduction

Portfolio selection in order to achieve a given expected return given an accepted risk has been a subject of research for more than 60 years. The first problem studied was the one period Mean-Variance problem in Markowitz 1952, Markowitz 1959. An analytic solution has been proposed first in Merton 1972 for a positive covariance matrix and when short selling is allowed. It is only in Li and Ng 2000 that the multi-period case has been solved by a reformulation of the problem as a linear quadratic (LQ) one. The solution for the continuous case when short selling is allowed in a complete Black-Scholes market has been proposed in Zhou and Li 2000 still based on the LQ reformulation. The case without short selling has been solved two years after in Li, Zhou, and Lim 2002. Notice that, in this case, a solution is provided if borrowing is allowed so that the investment in the bond is not constrained. Then an extension with randomization in coefficients is proposed in Lim 2004, the risk on the correlations is studied in Chiu and Wong 2014, Ismail and Pham 2019. Very recently results are obtained in Jaber, Miller, and Pham 2020 supposing that the volatility is rough Gatheral, Jaisson, and Rosenbaum 2018 and follows an affine and quadratic Volterra model: the Mean-Variance problem is solved in some case as the explicit solution of Riccati backward stochastic differential equations.
All these theoretical results are interesting but of limited use by practitioner as borrowing is generally not used and other operational constraints are added: investors first tend to limit

*This work is supported by FiME, Laboratoire de Finance des Marchés de l'Energie
†EDF R&D & FiME xavier.warin at edf.fr

the rebalancing of the portfolio (only achieved at discrete dates) to limit transaction cost by imposing constraints on the variation of the investment weights in the assets composing the portfolio. Second, they generally impose strategic views on the weights that are only allowed to stay into given limits. In this case, numerical methods are necessary. Using conventional PDE methods Wang and Forsyth 2010 have solved many constrained problems in the case of a single risky asset following a Black-Scholes dynamic. The same methodology in the case of an asset with jumps has been used in Dang and Forsyth 2014. This kind of approach can only be used at most with two or three assets and the resolution of a realistic portfolio selection problem is out of reach. In order to tackle the multi-dimensional problem with constraints, Cong and Oosterlee 2016 have proposed two algorithms based on the LQ formulation: the first being based on pure forward simulations is suboptimal with constraints, while the second using a backward recursion is based on regressions such that only rather low dimensional cases can be solved.
The use of the variance to evaluate the risk has been questioned by both practitioners and researchers as it both penalizes gain and loss. Numerous downside risk measures penalizing losses or low gains have been proposed in the literature. Among them, Lower Partial Moments (LPM) which have been proposed more than forty years ago in Fishburn 1977 rely on two parameters : the gamma γ parameter named the "Benchmark" parameter is set by the investor and the second one q represents the risk attitude of the investor. This risk model embeds a lot of classical models. For example the case $q = 0$ corresponds to the safety rule of Roy 1952, the case $q = 1$ correspond to the expected regret of Dembo and Rosen 1999, $q = 2$ corresponds to the semi-deviation below the Benchmark parameter or the semi-variance if the Benchmark parameter is set to the expected wealth.
Another classical risk measure related is the CVaR introduced in Rockafellar, Uryasev, et al. 2000 corresponding the expected loss below the VaR measure. As shown in Rockafellar, Uryasev, et al. 2000, CVaR calculation can be parametrized as the minimum over a parameter α of a function value linked to a LPM risk measure with parameter $q = 1$. Using this formulation, Gao et al. 2017 studied the Mean-CVaR continuous case giving semi-analytical solutions to the Mean-CVaR problem when the wealth is bounded. Indeed the boundedness of the wealth or the control as studied in Miller and Yang 2017 is necessary as the general case is not well posed for downside risk measures as shown in Jin, Yan, and Zhou 2005: using this kind of risk measure, and without constraints, the investors tend to gamble more and more if the market is in bad shape, and investments in the most risky assets tend toward infinity. Therefore, the Mean-CVaR problem has to be solved with constraints or in discrete-time and numerical methods are necessary to optimize portfolio. A classical approach consists in using the auxiliary formulation proposed by Rockafellar, Uryasev, et al. 2000 and in using a gradient descend method on α as proposed in Miller and Yang 2017. However, this procedure is very time consuming and only can be solved in low dimension.
In order to optimize portfolio with general Mean-Risk measure, Neural Networks appears to be an interesting choice. Neural networks are known to be able to approximate function in high dimension. Very recently, neural networks have been used in risk management first in Buehler et al. 2019. Some cases with constraints on the hedging products and some downside risk measures have been studied in Fécamp, Mikael, and Warin 2019. In both cases, results reported are promising. Cases of Asset Liability Management have been reported very recently in Krabichler and Teichmann 2020.
In this article, we show that neural networks are able to calculate very realistic efficient frontier in the Mean-Variance case and the Mean-CVar case.
The main findings developed are the following ones:

- **Neural networks are able to solve continuous Mean Variance problems accurately in low and high dimension for the control**: using different formulations, we first solve the continuous Mean-Variance problem without constraints in the Black-Scholes model with a direct formulation and the LQ formulation. We introduce for each formu-

lation two methods to approximate the efficient frontier : the first one approximates the frontier point by point while the second permits to evaluate the global frontier in a single calculation. In all the cases, the frontier is correctly approximated in dimension 4 and 20 by comparison to the analytical solution.
Then we used the global and point by point local method and adapt the network to solve the problem when no short selling and borrowing are allowed. Although we do not have any analytical solution, all frontiers calculated are very similar indicating that they are correctly evaluated.

- **Neural networks are able to deal with operational constraints**: in order to calculate the frontier when global and local constraints are added to the resolution we propose different formulations based on different penalization methods. We show that the global constraints are hard to satisfy by penalization and we introduce a new projected feed-forward network which is able to deal with the fact that the weights in the portfolio are such that all of them are positive, that their sum is equal to one and that each weights are between given bounds. Numerical methods still indicate that the border is correctly calculated.

- **Neural networks are able to deal with the Mean Variance problem with a state in high dimension**: the Black-Scholes case is interesting but is special as the state of the problem only involve the global wealth: then as the number of assets increases, only the dimension of the control increases. In a section we use the Heston model Heston 1993 and show that the frontiers are still correctly calculated with or without constraints. As the state of the problem depends on the wealth and the variance of the assets, we have shown that we are able to solve a high dimension problem for both the state and the control.

- **Neural networks can solve Mean-CVaR problems accurately**: we solve some more difficult Mean-CVaR problems and show that on some cases, solutions are sometimes trapped in local minima that can be far from the optimum. In this case, multiple calculations can be achieved to get back the correct frontier using a global formulation while the point by point formulation always gives oscillations.

We also show that, depending on the problem, one formulation may be preferred:

- For the Mean-Variance problem, a point by point approximation of the frontier is the best choice for the Black-Scholes model, while the global formulation is the best choice for the Heston model.

- Whereas for the Mean-CVaR, the global approaches are the only tested approaches to always obtain good results in the high dimension case (or what seems to be good results because no reference is available).

The article is organized as follows: In the first section, we briefly recall what a neural network is, the gradient descent methodology used, the different choices of some hyper parameters, and how the convergence of the optimization is checked. In the second section, we solve the continuous Mean-Variance problem. Some constraints are added in the third section. Section four and five focus respectively on the use of the Heston model in the Mean-Variance setting and the Mean-CVaR setting with a Black-Scholes model.
In the whole sequel, we note $(\Omega, \mathcal{F}, \mathbb{P}, \mathcal{F}_t)$ a filtered probability space. For each set $\mathcal{A}$ in $\mathbb{R}^d$, we note $\mathcal{L}^2_{\mathcal{F}_t}(0, T, \mathcal{A})$ the set of the $\mathcal{F}_t$ adapted square integrable processes with values in $\mathcal{A}$. At last we suppose that the risk free rate is 0 which is equivalent to discount all asset values with a rate r, so that we consider all risky assets with an adapted trend equal to the surplus of trend with respect to the non-risky asset.

Notations: In the sequel, for an element x of $\mathbb{R}^d$, $x^+ = \max(x, 0)$ applied component by component, for $(x, y) \in \mathbb{R}^2$, $x \vee y$ stands for $\max(x, y)$ and $x \wedge y$ stands for $\min(x, y)$.

2 Neural networks as function approximators

Deep neural networks are designed to approximate large class of functions. They rely on the composition of simple functions, and appear to provide an efficient way to handle high-dimensional approximation problems, by finding the "optimal" parameters by stochastic gradient descent methods. We here use a basic type of network dubbed feedforward networks. We fix the input dimension $d_0 = d$ that will represent the dimension of the state variable x, the output dimension d_1 (here $d_1 = 1$ for approximating the real-valued solution to the PDE, or $d_1 = d$ for approximating the vector-valued gradient function), the global number $L + 1 \in \mathbb{N} \setminus \{1, 2\}$ of layers with m_ℓ, $\ell = 0, \ldots, L$, the number of neurons (units or nodes) on each layer: the first layer is the input layer with $m_0 = d$, the last layer is the output layer with $m_L = d_1$, and the $L - 1$ layers between are called hidden layers, where we choose for simplicity the same dimension $m_\ell = m$, $\ell = 1, \ldots, L - 1$. A feedforward neural network is a function from $\mathbb{R}^d$ to $\mathbb{R}^{d_1}$ defined as the composition

$$x \in \mathbb{R}^d \longmapsto A_L \circ \varrho \circ A_{L-1} \circ \ldots \circ \varrho \circ A_1(x) \in \mathbb{R}^{d_1}. \tag{1}$$

Here A_ℓ, $\ell = 1, \ldots, L$ are affine transformations: A_1 maps from $\mathbb{R}^d$ to $\mathbb{R}^m$, $A_2, \ldots, A_{L-1}$ map from $\mathbb{R}^m$ to $\mathbb{R}^m$, and A_L maps from $\mathbb{R}^m$ to $\mathbb{R}^{d_1}$, represented by

$$A_\ell(x) = \mathcal{W}_\ell x + \beta_\ell, \tag{2}$$

for a matrix $\mathcal{W}_\ell$ called weight, and a vector β_ℓ called bias term, $\varrho : \mathbb{R} \to \mathbb{R}$ is a nonlinear function, called activation function, and applied component-wise on the outputs of A_ℓ, i.e., $\varrho(x_1, \ldots, x_m) = (\varrho(x_1), \ldots, \varrho(x_m))$. Standard examples of activation functions are the sigmoid, the ReLU, the ELU, tanh. Generally, for stochastic control problems, the number of layers is kept low (between 2 and 4) in order to avoid the problem of vanishing gradient and the number of neurons depends on d but is kept generally between 10 and 100 (Warin 2021,Chan-Wai-Nam, Mikael, and Warin 2019, Fécamp, Mikael, and Warin 2019, Han, Jentzen, and Weinan 2018).

All these matrices $\mathcal{W}_\ell$ and vectors β_ℓ, $\ell = 1, \ldots, L$, are the parameters of the neural network, and can be identified with an element $\theta \in \mathbb{R}^{\kappa_{L,m}}$, where $\kappa_{L,m} = \sum_{\ell=0}^{L-1} m_\ell(1 + m_{\ell+1}) = d(1 + m) + m(1 + m)(L - 2) + m(1 + d_1)$ is the number of parameters, where we fix d_0, d_1, L, and m. The fundamental result of Hornick et al. Hornik, Stinchcombe, and White 1989 justifies the use of neural networks as function approximators by proving that the set of all feedforward networks letting m vary is dense in $L^2(\nu)$ for any finite measure ν on $\mathbb{R}^d$, whenever ϱ is continuous and non-constant.

In the whole article, we use three hidden layers and a number of neurons equal to $10 + d$ such that the dimensional space for the parameters of the neural network is $\hat{\kappa} = \kappa_{4,10+d}$. Gradient descent is implemented in Tensorflow Martin Abadi et al. 2015 using ADAM optimized Kingma and Ba 2014. The learning rate used by Adam is taken linearly decreasing with gradient iterations from a given initial value to a final one. The activation function used is the tanh function which has been widely used for the resolution of non linear PDEs Huré, Pham, and Warin 2020 Pham, Warin, and Germain 2021 Germain, Pham, and Warin 2020. Compared to ReLU activation functions it gives slightly better results and, as it is bounded, gradient descent algorithms diverge less easily for high learning rates. In the sequel the batch size is generally chosen between 100 and 300. Other different parameters such as the learning rates and the number of gradient iterations are chosen such that the solution obtained does not vary a lot with iterations : to check that the number of iterations is sufficient, every 100 gradient iterations, an accurate estimation of the objective function is achieved with a high number of samples permitting to check that the objective function value is stabilized.

3 Mean-Variance efficient frontier in a continuous setting

In this section we suppose that the asset prices follow a Black-Scholes model:

$$\frac{dS_t}{S_t} = \mu dt + \sigma dW_t \tag{3}$$

with S_t with values $\mathbb{R}^d$ with components $S_{t,j}, \quad j = 1, \dots, d$, μ with values in $\mathbb{R}^d$, $\sigma \in \{\text{diag}(v), v \in \mathbb{R}^d_{>0}\}$ the set of diagonal matrices with strictly positive values, $W_t = (\hat{W}^i_t)_{i=1,d}$ where the $\hat{W}^i$ are $\mathcal{F}_t$ adapted Brownian motions correlated with a correlation matrix ρ.
In the continuous setting, we note $\xi = (\xi_t)_{t>0}$ the investment strategy with values in $\mathbb{R}^d$ until maturity T, with a component i corresponding to the fraction of wealth invested in asset i. We suppose ξ_t is in $\mathcal{L}^2_{\mathcal{F}_t}(0, T, \mathbb{R}^d)$. The portfolio value at date T verifies:

$$X^\xi_T = X_0 + \int_0^T \xi_t X^\xi_t . \frac{dS_t}{S_t} = X_0 + \int_0^T X^\xi_t \xi_t . (\mu dt + \sigma dW_t) \tag{4}$$

The Mean-Variance problem consist in finding strategies ξ adapted to the available information that minimize:

$$(J_1(\xi), J_2(\xi)) := (-\mathbb{E}[X^\xi_T], \mathbb{E}[(X^\xi_T - \mathbb{E}[X^\xi_T])^2]). \tag{5}$$

An admissible strategy ξ^* is said to be efficient if there is no other strategy ψ such that

$$J_1(\psi) \leq J_1(\xi^*), \quad J_2(\psi) \leq J_2(\xi^*)$$

and at least one of the two previous inequalities is strict.
Then $(J_1(\xi^*), J_2(\xi^*))$ is an efficient point and the set of all efficient points defines the efficient frontier. By convexity (see Zhou and Li 2000), this Pareto frontier can be calculated by minimizing the function defined as the weighted average of the two criteria:

$$J_1(\xi) + \beta J_2(\xi) \tag{6}$$

where the parameter $\beta > 0$ defines a point of the efficient frontier.
In the continuous setting, the solution of the Mean-Variance problem is known explicitly Zhou and Li 2000,Ismail and Pham 2019 and the optimal investment strategy $\alpha_t = \xi^*_t X^{\xi^*}_t$ where ξ^* minimizes (6) and $X^{\xi^*}_t$ is the optimal portfolio associated is given by:

$$\alpha(X^{\xi^*}_t) = -(\sigma\rho\sigma)^{-1}\mu\Big[X^{\xi^*}_t - X_0 - \frac{e^{RT}}{2\beta}\Big], \quad 0 \leq t \leq T \tag{7}$$

where $R = \mu.((\sigma\rho\sigma)^{-1}\mu)$.
The problem (6) does not admit any dynamic programming principal due the average term in the variance definition so that no PDE or regression method can be used directly to solve it. The derivation of the analytic solution is based on a LQ auxiliary equivalent formulation as shown in Zhou and Li 2000. Indeed the solution of the problem (6) is solution of

$$\xi^* = \underset{\xi \in \mathcal{L}^2_{\mathcal{F}_t}(0,T,\mathbb{R}^d)}{\text{argmin}} \ \mathbb{E}[(X^\xi_T - \gamma)^2] \tag{8}$$

where

$$\gamma = \frac{1}{2\beta} + \mathbb{E}[X^{\xi^*}_T]. \tag{9}$$

It is then possible to estimate the efficient frontier by the resolution of the problem (8) by letting γ vary. This kind of formulation is generally used by conventional methods such as regressions Cong and Oosterlee 2016 and PDEs Wang and Forsyth 2010, Dang and Forsyth 2014 as the dynamic programming principle can be used.

3.1 Neural network approximations

Equation (4) is first discretized on grid of dates $t_i, 0 \leq i < N$ such that $t_0 = 0$, $0 < t_i < T$ for $0 < i < N$ and we note $t_N = T$. We note ϕ^i a position (as a fraction of the wealth invested in each asset) at date t_i, and $\phi = (\phi^i)_{i=0,\dots,N-1}$. The portfolio value is then given by

$$X_T^\phi = X_0 + \sum_{i=0}^{N-1} \phi^i X_{t_i}^\phi . \frac{S_{t_{i+1}} - S_{t_i}}{S_{t_i}} \tag{10}$$

We first present the methodology used to solve (6) for a given β (or (8) for a given γ). The state of the system only depends on t and the wealth x as shown by equation (4), then we classically introduce a single network with parameters $\theta \in \mathbb{R}^{\hat{\kappa}}$ taking t and the wealth x as input (so in dimension 2) and with output $\hat{\phi}^\theta(t,x)$ in dimension d where $\hat{\phi}_j^\theta(t_i,.)$ is an approximation of ϕ_j^i.

Remark 3.1 *In the general case, where the asset is not modeled by a geometric brownian motion, the value of the asset has to be included in the state.*

No activation function is used on the final output such that the network gives an output potentially covering $\mathbb{R}^d$. Then we solve

$$\theta^* = \underset{\theta \in \mathbb{R}^{\hat{\kappa}}}{\operatorname{argmin}} \; -\mathbb{E}[X_T^{\hat{\phi}^\theta}] + \beta \mathbb{E}[(X_T^{\hat{\phi}^\theta} - \mathbb{E}[X_T^{\hat{\phi}^\theta}])^2] \tag{11}$$

for problem (6) or

$$\theta^* = \underset{\theta \in \mathbb{R}^{\hat{\kappa}}}{\operatorname{argmin}} \, \mathbb{E}[(X_T^{\hat{\phi}^\theta} - \gamma)^2] \tag{12}$$

for problem (8).
Instead of evaluating the frontier by solving (11) for each β chosen (or (12) for each γ), a second resolution method aims at evaluating the global frontier in one calculation. In this case, taking for example problem (6), we introduce a network with input (t,x,β) in dimension 3 and still with an output in dimension d. The associated strategy is noted $\hat{\phi}^{\theta;\beta}$ to emphasize the dependence on β and we solve :

$$\theta^* = \underset{\theta \in \hat{\kappa}}{\operatorname{argmin}} \, \mathbb{E}[-\mathbb{E}[X_T^{\hat{\phi}^{\theta;\hat{\beta}}} | \hat{\beta}] + \hat{\beta} \mathbb{E}[(X_T^{\hat{\phi}^{\theta;\hat{\beta}}} - \mathbb{E}[X_T^{\hat{\phi}^{\theta;\hat{\beta}}} | \hat{\beta}])^2 | \hat{\beta}]] \tag{13}$$

where $E[|\hat{\beta}]$ stands for the conditional expectation with respect to $\hat{\beta}$ which is a random variable with density that can be taken

- either with discrete values so $p(x) = \frac{1}{K} \sum_j^K \delta_{\beta_j}(x)$ where $(\beta_i)_{i=1,K}$ is a set of values where we want to approximate the frontier (generally $\beta_1 = 0$) and in this case it is equivalent to minimize

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \sum_{k=1}^{K} -\mathbb{E}[X_T^{\hat{\phi}^{\theta;\beta_k}}] + \beta_k \mathbb{E}[(X_T^{\hat{\phi}^{\theta;\beta_k}} - \mathbb{E}[X_T^{\hat{\phi}^{\theta;\beta_k}}])^2] \tag{14}$$

- or $p(x)$ is for example an uniform law on $[\underline{\beta}, \bar{\beta}]$ representing where we want to approximate the frontier.

All results in the following section are obtained using some batch of size 300, the linear rate is taken linearly decreasing from 25×10^{-4} to 25×10^{-5} with gradient iterations. The number of iterations is set to 15000. After training using 40 points ($K = 40$ in the global estimation in equation (14), or 40 points to approximate the frontier point by point), each point of the frontier is plotted calculating mean and variance using $1e5$ simulations.

Remark 3.2 *We use a single network for the resolution. It is also possible to use a neural network at each date as used in Han, Jentzen, and Weinan 2018 for the resolution of non linear PDEs. It turns out that the use of a single network as proposed in Chan-Wai-Nam, Mikael, and Warin 2019 for the same method highly improves the results in all the tested configurations in this article. Note that there are real life problems in finance such as gas storage valuation where the use of a single network is not sufficient Warin 2021.*

3.2 Results in dimension 4

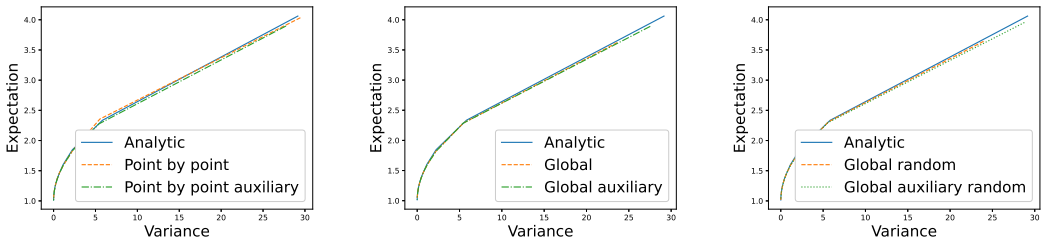
We take $\mu = (0.01, 0.0225, 0.035, 0.0475)^T$, the diagonal of σ is given by

$$(0.05, 0.1, 0.15, 0.2)$$

and we take

$$\rho = \begin{pmatrix} 1. & 0.26 & -0.43 & 0.233 \\ 0.26 & 1. & 0.003 & 0.06 \\ -0.43 & 0.003 & 1. & -0.33 \\ 0.233 & 0.06 & -0.33 & 1. \end{pmatrix}.$$

We suppose that $T = 1$ year and that the rebalancing is achieved twice a week ($N = 104$) to approach a continuous rebalancing. We plot the frontier obtained with β values between 0.05 to 2.7. The analytic solution is obtained by applying the continuous optimal control (7) at each rebalancing date. In the sequel, in the figures, "Point by point" stands for an approximation using (11), "point by point auxiliary" for an approximation using (12), "global" stands for an approximation using (14), "global random" stands for an approximation using (13) and an uniform law for $\hat{\beta}$. At last "global auxiliary" and "global auxiliary random" stand respectively for a global resolution with deterministic and random γ values for the auxiliary problem (8).



Point by point Global Global random

Figure 1: Efficient frontier in dimension 4.

Results on figure 1 show that the whole frontiers obtained are on the analytic frontier but global random estimation on the direct problem (6) tend to fail to reproduce the whole curve. Portfolio with very high returns are not found (the part of the curve with very high returns is missing). Nevertheless results are very good.

Remark 3.3 *For the smallest $\beta = 0.05$ of the curve, the global random result obtained with the direct formulation is rather unstable and obtained doubling the learning rates. An increase of the number of gradient iterations does not improve the results.*

In table 1 and 2, we give the results obtained for 3 points of the curve corresponding to a low, a medium and a high variance result. Results are all very good except with the global random

method associated with the direct approach (11) for the high variance case, or associated to the auxiliary approach (12) for the low variance case.

β	Analytical		Point by point		Global		Global random	
	Mean	Variance	Mean	Variance	Mean	Variance	Mean	Variance
0.05	4.056	31.445	3.997	29.868	4.021	30.281	3.83	27.4
0.2	1.779	1.949	1.746	1.845	1.76	1.914	1.765	1.915
2.0	1.077	0.019	1.08	0.021	1.078	0.02	1.075	0.018

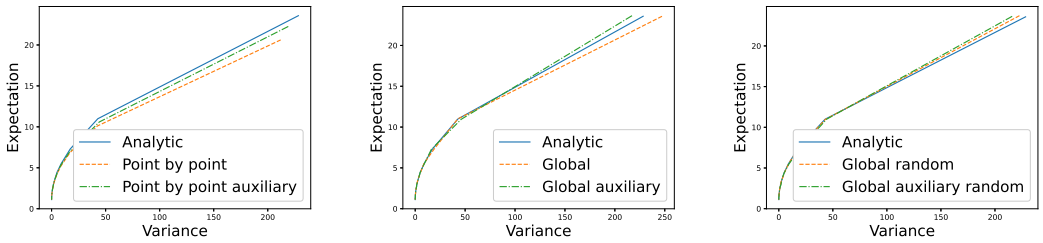
Table 1: Resolution of (11) for some β values in a continuous setting in dimension 4.

γ	Analytical		Point by point		Global		Global random	
	Mean	Variance	Mean	Variance	Mean	Variance	Mean	Variance
14.097	4.107	30.324	3.992	29.941	4.033	30.748	4.108	32.068
4.274	1.778	1.951	1.758	1.889	1.749	1.863	1.822	2.882
1.327	1.077	0.019	1.075	0.019	1.077	0.021	1.099	0.906

Table 2: Resolution of (12) for some γ values in a continuous setting in dimension 4.

3.3 Results in dimension 20

We take $\mu_i = 0.01 + \frac{i-1}{400}$ and the diagonal matrix σ such as $\sigma_{i,i} = 0.05 + \frac{i-1}{100}$ for $i = 1, \ldots, 20$. The correlation is picked up randomly by a random generator but avoiding too high correlation leading to huge positions in the portfolio. Results on figure 2 show that the point by point approximation especially for the direct approach is less effective for very small β. The global approach can slightly outperform the analytic solution as the continuous formula is applied on a discrete-time problem.



Point by point Global Global random

Figure 2: Efficient frontier in dimension 20.

Remark 3.4 *For the global random direct approach, the point (with high variance) on the right picture in figure 2 not aligned with the points of the other curves is not improved while changing the hyper parameters.*

As in dimension 4, in table 3 and 4, we give the results obtained for 3 points of the curve corresponding to a low, a medium and a high variance result.

β	Analytical		Point by point		Global		Global random	
	Mean	Variance	Mean	Variance	Mean	Variance	Mean	Variance
0.05	24.204	238.117	21.094	193.111	20.278	179.205	14.301	107.822
0.2	6.774	14.265	6.349	13.287	6.426	13.063	6.23	12.925
2.0	1.583	0.145	1.541	0.13	1.562	0.142	1.564	0.142

Table 3: Resolution of (11) for some β values in a continuous setting in dimension 20.

γ	Analytical		Point by point		Global		Global random	
	Mean	Variance	Mean	Variance	Mean	Variance	Mean	Variance
34.146	24.096	241.119	22.472	228.249	23.669	236.393	23.898	236.101
9.286	6.802	13.695	6.556	14.524	6.731	14.641	6.971	18.877
1.829	1.579	0.151	1.569	0.146	1.587	0.158	1.83	3.935

Table 4: Resolution of (12) for some γ values in a continuous setting in dimension 20.

Using results both in dimension 4 and 20, it seems that:

- for high variance cases, formulation (12) should be preferred,
- the global random method appears to have difficulties to catch different extreme points of the curve depending on the formulation selected.

4 Mean-Variance on the discrete case adding constraints

In the section, we focus on the discrete case still trying to solve (6) or (8).

Remark 4.1 *We always suppose that the allocation weights belong to a convex set and then* (8) *and* (6) *remain equivalent.*

In the whole section, we impose that there is no short selling, no borrowing and that all the whole wealth is invested. Then in the two sections below, the ϕ^i in equation (10) satisfy:

$$\begin{aligned} &0 \leq \phi_j^i \leq 1, \quad \forall i = 0, \dots, N-1, \quad j = 1, \dots, d \\ &\sum_{j=1}^{d} \phi_j^i = 1, \quad \forall i = 0, \dots, N-1 \end{aligned} \tag{15}$$

In this case, we can rewrite equation (10) by introducing the yield vector $\mathbb{Y}_i = \frac{S_{t_{i+1}} - S_{t_i}}{S_{t_i}}$ so that:

$$X_T^{\phi} = X_0 \prod_{i=0}^{N-1} (1 + \phi_i . \mathbb{Y}_i) \tag{16}$$

In all the cases in this section we take $T = 10$ years and we suppose that rebalancing is achieved once a month. In dimension 4 and 20, trends and volatilities are the same as in the continuous case but correlations are again picked up randomly and for example in dimension 4:

$$\rho = \begin{pmatrix} 1. & 0.805 & -0.894 & 0.59 \\ 0.805 & 1. & -0.571 & 0.473 \\ -0.894 & -0.571 & 1. & -0.772 \\ 0.59 & 0.473 & -0.772 & 1. \end{pmatrix}.$$

Remark 4.2 *No limits are imposed anymore on the correlation used as the weights are naturally bounded.*

4.1 Dynamic versus static optimization.

The most interesting problem consists in finding the weights satisfying the equation (15) and resulting from strategies letting the weights vary with time and adapt to the state of the world. Another strategy widely used by the practitioners consists in taking the weights constant leading to the "constant mix" strategy. The value of the weights can be chosen by expert or optimized to minimize a given criterion. Then an efficient frontier in the class of strategies with constant weights can be calculated too.
In practice, engineers, trying to optimize these constant weights, draw randomly the weights and, for a set a weights drawn, test the given strategy leading to a single point (Variance, Expectation) using a very high number of asset trajectories. When the number of assets is not too important, this brute force calculation permits to identify the sets of weights near the efficient frontier.
As the methodology is easy to implement, this method is widely used. As the weights are constant, the rebalancing of the portfolio is generally limited and practitioners do not have to justify the high rebalancing of the portfolio that a dynamic strategy may propose.
In the sequel of the article, we will note by static optimization ("Static" on figures) the optimal constant mix strategy such that the weights are kept constant during the whole period. When "Static" is not specified, a plot is carried out with a dynamic optimization.

Remark 4.3 *For a static optimization no neural network is used and the problem is reduced to an optimization in dimension d.*

4.2 No other constraints

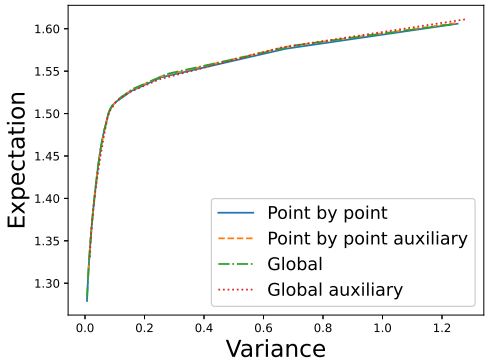
In this section, we suppose that only constraints (15) are imposed. We use a similar network as in the previous section with parameter θ except that we use a sigmoid activation function at the output giving a network $\zeta^\theta(t, X)$ for the point by point evaluation with values in $[0, 1]^d$. Investment weights are then given by

$$\hat{\phi}^\theta(t, X) = \frac{\zeta^\theta(t, X)}{\sum_{i=1}^{d} \zeta_i^\theta(t, X)} \tag{17}$$

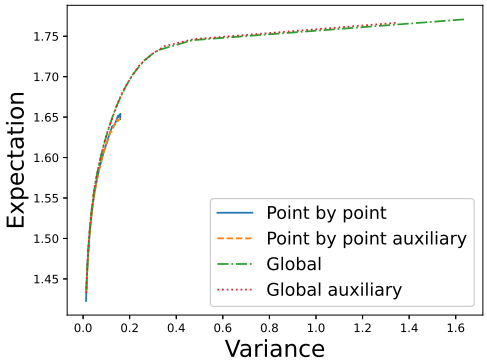
Then equation (11) or (12) can be solved.
Similarly for a global formulation the network for the direct problem (6) take (t, x, β) as input and the weights are defined as

$$\hat{\phi}^{\theta;\beta}(t, X, \beta) = \frac{\zeta^\theta(t, X, \beta)}{\sum_{i=1}^{d} \zeta_i^\theta(t, X, \beta)}. \tag{18}$$

Then it is possible to solve (13) or to minimize the objective function corresponding to the auxiliary problem.
In this section we take an initial learning rate equal to $\frac{1}{2}\frac{10^{-2}}{d}$ and linearly decreasing with iterations to $\frac{1}{2}\frac{10^{-3}}{d}$. The number of gradient iterations is set to 15000 in dimension 4 and 30000 in dimension 20, the batch size equal to 300.



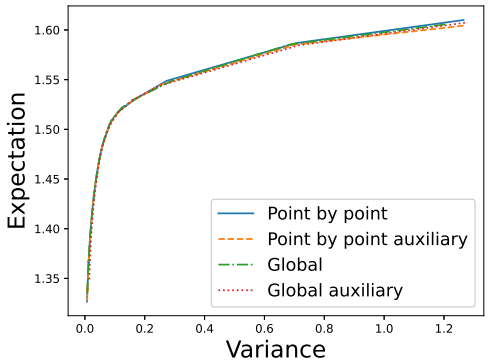
Dimension 4



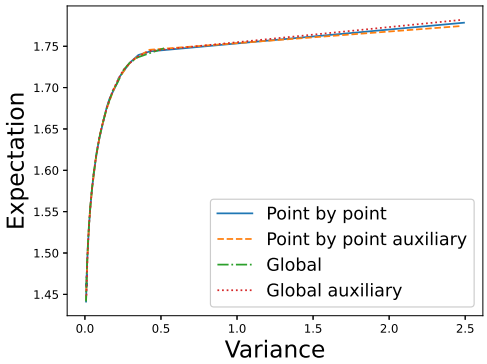
Dimension 20

Figure 3: Efficient frontier with static optimization.

On figure 3, we plot the efficient frontier calculated by static optimization for different methods. In dimension 4, all methods seem to get back the whole frontier while in dimension 20 very high returns are hard to catch : the global methods can get the part of the curve with very high returns quite easily while point by point methods fail even when increasing the number of gradient iterations.



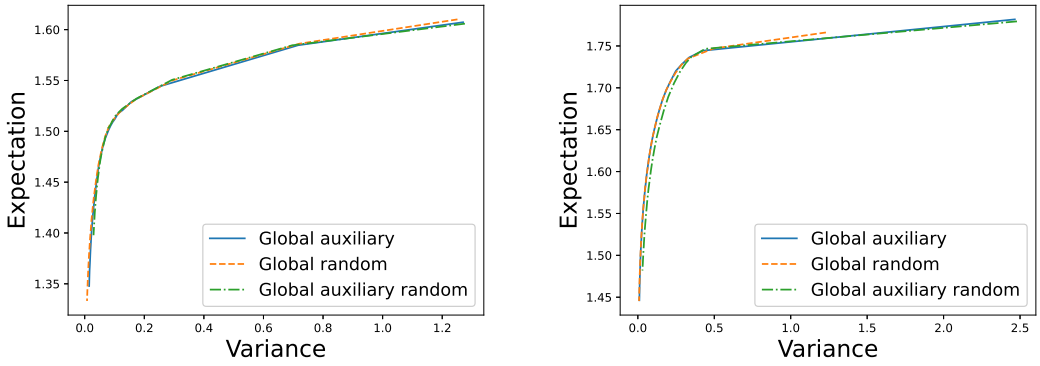
Dimension 4



Dimension 20

Figure 4: Efficient frontier with dynamic optimization (point by point and global methods).

As seen on figure 4 dealing with the dynamic case, global with deterministic β (respectively γ) coefficients and point by point methods give very similar results.



Dimension 4 Dimension 20

Figure 5: Efficient frontier with dynamic optimization for global random approach : methods with stochastic risk coefficients compared to reference calculated with global auxiliary method.

As in the analytical case, the randomized version of the global approach has difficulties to represent the whole curve especially in high dimension as shown on figure 5: in dimension 20, the direct global approach with randomization only gives a part of the curve (permitting to recover the curve only for β leading a low variance portfolio) while the auxiliary global random version gives another part of the optimal curve (permitting to recover the curve only for β leading a high variance portfolio).

Remark 4.4 *As we sample the parameters to represent the same part of the curve, the use of a uniform law in the random method to explore the β or γ values may give too much importance on a given part of the curve (it depends if a direct or auxiliary approach is used). However results are not improved trying to change the law used to sample the β or γ.*

Nevertheless, the different methods give very similar results for different values of β/γ not corresponding to extreme points of the curve as shown on tables 5, 6, 7, 8.

β	Point by point		Global		Global random	
	Mean	Variance	Mean	Variance	Mean	Variance
0.062	1.585	0.703	1.587	0.747	1.59	0.766
0.821	1.503	0.08	1.497	0.073	1.497	0.072
5.04	1.379	0.015	1.379	0.015	1.381	0.015

Table 5: Resolution of (11) for some β values with constraint on sum of weights in dimension 4.

γ	Point by point		Global		Global random	
	Mean	Variance	Mean	Variance	Mean	Variance
9.677	1.583	0.685	1.585	0.694	1.584	0.675
2.111	1.503	0.08	1.499	0.073	1.498	0.073
1.478	1.38	0.015	1.386	0.019	1.378	0.015

Table 6: Resolution of (12) for some γ values with constraint on sum of weights in dimension 4.

β	Point by point		Global		Global random	
	Mean	Variance	Mean	Variance	Mean	Variance
0.062	1.745	0.431	1.739	0.371	1.744	0.442
0.821	1.65	0.126	1.651	0.11	1.651	0.11
5.04	1.488	0.016	1.489	0.015	1.49	0.015

Table 7: Resolution of (11) for some β values with constraint on sum of weights in dimension 20.

γ	Point by point		Global		Global random	
	Mean	Variance	Mean	Variance	Mean	Variance
9.837	1.745	0.435	1.745	0.431	1.745	0.431
2.259	1.65	0.108	1.651	0.11	1.653	0.11
1.588	1.488	0.015	1.503	0.022	1.496	0.018

Table 8: Resolution of (12) for some γ values with constraint on sum of weights in dimension 20.

4.3 Adding constraints on the weights

Investors often impose other constraints on portfolio:

- Some are local constraints: the variations of the weights are limited from one step to another in order to face liquidity constraints and to reduce transaction cost:

$$|\phi_j(t_{i+1}, X_{t_{i+1}}) - \phi_j(t_i, X_{t_i})| \leq \eta_j, \text{ for } j = 1, \ldots, d, \text{ and } i = 0, \ldots, N-1 \tag{19}$$

- Some are global constraints : weights are only allowed to stay in a convex compact:

$$\underline{\phi}_j \leq \phi_j(t_i, X_{t_i}) \leq \bar{\phi}_j, \quad \text{for } j = 1, \ldots, d. \tag{20}$$

We first compare different formulations to solve the problem with the previous constraints. We test them on the point by point estimation of the frontier in dimension 4 in the next subsection. Then we use the best model to achieve an exhaustive comparison of the point by point and global approaches.

4.3.1 Presentation of the different models on the point by point approach in dimension 4

- The first model consists in taking the same representation for the weights as in equation (17). Then constraints (19), (20) are imposed by penalization of the objective function and the parameters θ minimize:

$$J_1(\hat{\phi}^\theta) + \beta J_2(\hat{\phi}^\theta) + \frac{1}{\epsilon} \sum_{j=1}^{d} \sum_{i=0}^{N-2} \mathbb{E}\left[(|\hat{\phi}_j^\theta(t_{i+1}, X_{t_{i+1}}^{\hat{\phi}^\theta}) - \hat{\phi}_j^\theta(t_i, X_{t_i}^{\hat{\phi}^\theta})| - \eta_j)^+ \right] +$$
$$\frac{1}{\epsilon} \sum_{j=1}^{d} \sum_{i=0}^{N-1} \mathbb{E}\left((\hat{\phi}_j^\theta(t_i, X_{t_i}^{\hat{\phi}^\theta}) - \bar{\phi}_j)^+ + (\underline{\phi}_j - \hat{\phi}_j^\theta(t_i, X_{t_i}^{\hat{\phi}^\theta}))^+ \right) \tag{21}$$

  where ϵ is a small penalization parameter.

- The second model consists in introducing the variation of weights between two dates similarly as in Fécamp, Mikael, and Warin 2019. We introduce $\xi^\theta(t,X)$ a neural network taking time and the portfolio value as input with a tanh activation function as output so with values in $[-1,1]^d$. The initial portfolio weight is represented by a vector $\tilde{\theta}$ in $\mathbb{R}^d$ and the weight in the portfolio at a given date is given by:

$$\hat{\phi}^{\theta,\tilde{\theta}}(t_i, X_{t_i}^{\hat{\phi}^{\theta,\tilde{\theta}}}) = \tilde{\theta} + \eta \sum_{l=1}^{i} \xi^\theta(t_l, X_{t_l}^{\hat{\phi}^{\theta,\tilde{\theta}}}) \tag{22}$$

We note $\bar{\theta} = (\theta, \tilde{\theta})$. Local constraints are taken into account in this formulation. It remains to impose global bounds on the weights and the constraint on the summation of the weights. It leads to the minimization in $\bar{\theta}$ of

$$J_1(\hat{\phi}^{\bar{\theta}}) + \beta J_2(\hat{\phi}^{\bar{\theta}}) + \frac{1}{\epsilon} \sum_{i=0}^{N-1} \mathbb{E}\left(|\sum_{j=1}^{d} \hat{\phi}_j^{\bar{\theta}}(t_i, X_{t_i}^{\hat{\phi}^{\bar{\theta}}}) - 1| \right)$$
$$\frac{1}{\epsilon} \sum_{j=1}^{d} \sum_{i=0}^{N-1} \mathbb{E}\left((\hat{\phi}_j^{\bar{\theta}}(t_i, X_{t_i}^{\hat{\phi}^{\bar{\theta}}}) - \bar{\phi}_j)^+ + (\underline{\phi}_j - \hat{\phi}_j^{\bar{\theta}}(t_i, X_{t_i}^{\hat{\phi}^{\bar{\theta}}}))^+ \right) \tag{23}$$

- The third model consist in imposing directly the global constraints at the output of the network (22) by introducing:

$$\hat{\phi}^{\theta,\tilde{\theta}}(t_i, X_{t_i}^{\hat{\phi}^{\theta,\tilde{\theta}}}) = ((\tilde{\theta} + \eta \sum_{l=1}^{i} \xi^\theta(t_l, X_{t_l}^{\hat{\phi}^{\theta,\tilde{\theta}}})) \wedge \bar{\phi}) \vee \underline{\phi} \tag{24}$$

Local and global constraint on the weights are taken into account and it remains to impose that the sum of the weights is equal to one giving the following expression to minimize in θ:

$$J_1(\hat{\phi}^{\bar{\theta}}) + \beta J_2(\hat{\phi}^{\bar{\theta}}) + \frac{1}{\epsilon} \sum_{i=0}^{N-1} \mathbb{E}\left(|\sum_{j=1}^{d} \hat{\phi}_j^{\bar{\theta}}(t_i, X_{t_i}^{\hat{\phi}^{\bar{\theta}}}) - 1| \right) \tag{25}$$

- The fourth and last model consists in using a feedforward network giving as output $\zeta^\theta(t,X)$ in $[0,1]^d$ by using a sigmoid activation function at the output of the network. A rescaling is achieved to get the weights in $[\underline{\phi}, \bar{\phi}]$ by

$$\hat{\phi}^\theta(t,X) = \underline{\phi} + \zeta^\theta(t,X)(\bar{\phi} - \underline{\phi}). \tag{26}$$

It remains to get an output in the following hyperplane

$$1 = \sum_{i=1}^{d} \hat{\phi}^\theta(t,X), \tag{27}$$

which can be achieved with the following projection algorithm applied on the network:

Algorithm 1 Projection algorithm applied on the output of the network

1: Input : $\hat{\phi}^\theta(t,X)$ with values in $[\underline{\phi}, \bar{\phi}]$
2: **for** $i = 1, d$ **do**
3: $\quad \hat{\phi}_i^\theta(t,X) = [(\hat{\phi}_i^\theta(t,X) + (1 - \sum_{j=1}^{d} \hat{\phi}_j^\theta(t,X))) \wedge \bar{\phi}_i] \vee \underline{\phi}_i$
4: **end for**
5: Return : $\hat{\phi}^\theta(t,X)$ satisfying (27)

The projections are carried out successively in the different directions. In order to avoid having a preferential direction, it is also possible to randomize the loop of the previous algorithm by performing a random permutation of the different visited dimensions.
The objective function to minimize in θ is

$$J_1(\hat{\phi}^\theta) + \beta J_2(\hat{\phi}^\theta) + \frac{1}{\epsilon}\sum_{j=1}^{d}\sum_{i=0}^{N-1}\mathbb{E}\left[(|\hat{\phi}_j^\theta(t_{i+1}, X_{t_{i+1}}^{\hat{\phi}^\theta}) - \hat{\phi}_j^\theta(t_i, X_{t_i}^{\hat{\phi}^\theta})| - \eta_j)^+\right] \tag{28}$$

Remark 4.5 *In the previous formulations, we have supposed that the initial weights in the portfolio had to be optimized. It is often a data in the problem and then only weights after the initial date have to be optimized.*

We test the four previous model on the four dimensional test case described at the beginning of the section. We impose the additional constraints : the weights cannot vary more than 0.05 in absolute value between two rebalancing date and we impose that weights are between 0.1 and 0.6. Besides in this case, we impose that initial weights are the same for each asset.
We take the following resolution parameters: we take $\epsilon = 10^{-4}$, the initial learning rate is set to 10^{-3} and the learning rate decreases to 10^{-5} with the gradient iterations. The batch size is equal to 300 and the number of gradient iterations equal to 10000. For each case, we carry out 4 optimizations and keep the best result (the one with the minimal objective function). In the table 9, we give $\mathbb{E}(X_T) - \beta\mathbb{E}((X_T - \mathbb{E}(X_T))^2)$ for the four models for different values of β.

β	0	0.23	0.479	0.719	0.959	1.198	1.427	1.678	2.158
Opt 1	1.468	1.419	1.402	1.392	1.382	1.370	1.365	1.354	1.337
Opt 2	1.336	1.309	1.322	1.334	1.322	1.313	1.298	1.303	1.292
Opt 3	1.397	1.290	1.224	1.206	1.213	1.306	1.167	1.245	0.791
Opt 4	1.474	1.426	1.409	1.398	1.387	1.378	1.368	1.360	1.344

Table 9: Comparison of the 4 models taking into account the constraints giving $E(X_T) - \beta E((X_T - E(X_T))^2)$ obtained.

Models 1 and 4 give the best results. Besides, models 2 and 3 have difficulties to satisfy the constraints: in the following table 9 we give the opposite of the objective function calculated. The difference between results in table below with results in table 9 indicate a violation of the constraints.

β	0	0.23	0.479	0.719	0.959	1.198	1.427	1.678	2.158
Opt 1	1.427	1.371	1.396	1.380	1.373	1.370	1.362	1.354	1.337
Opt 2	1.335	1.307	1.320	1.331	1.318	1.313	1.280	1.281	1.246
Opt 3	-620	-660	-942	-568	-479	-1336	-670	-611	-750
Opt 4	1.474	1.425	1.409	1.397	1.386	1.378	1.368	1.360	1.341

Table 10: Opposite of the objective function.

Model 3 seems be to totally unable to take into account the constraints, while model 2 satisfies globally the constraints. We may think that the results depends on some hyper parameters, and we test the different algorithms with a smaller $\epsilon = 10^{-5}$. Results are given in table 11 and 12: they confirm our first results.

β	0	0.23	0.479	0.719	0.959	1.198	1.427	1.678	2.158
Opt 1	1.465	1.415	1.395	1.389	1.380	1.372	1.364	1.349	1.331
Opt 2	1.362	1.331	1.337	1.342	1.322	1.329	1.290	1.282	1.289
Opt 3	1.431	1.370	1.308	1.259	1.192	1.189	1.163	1.245	1.198
Opt 4	1.474	1.428	1.409	1.398	1.388	1.378	1.368	1.360	1.345

Table 11: Comparison of the 4 models taking into account the constraints giving $E(X_T) - \beta E((X_T - E(X_T))^2)$ obtained using $\epsilon = 10^{-5}$.

β	0	0.23	0.479	0.719	0.959	1.198	1.427	1.678	2.158
Opt 1	1.448	1.414	1.394	1.377	1.380	1.363	1.353	1.342	1.331
Opt 2	1.310	1.328	1.306	1.295	1.316	1.300	1.287	1.236	1.268
Opt 3	-15502	-12847	-7690	-6891	-14510	-6284	-9096	-8022	-7221
Opt 4	1.472	1.426	1.408	1.397	1.387	1.377	1.368	1.360	1.344

Table 12: Opposite of the objective function using $\epsilon = 10^{-5}$.

At last we test the influence of some other parameters: we multiply the learning rate by two, take a batch size of 500, and a use 20000 gradient iterations. Results are presented in tables 13 and 13. Other tests are not presented here but, using some smaller learning rates, they give the same results.

β	0	0.23	0.479	0.719	0.959	1.198	1.427	1.678	2.158
Opt 1	1.457	1.417	1.402	1.395	1.385	1.375	1.366	1.354	1.337
Opt 2	1.348	1.374	1.348	1.328	1.324	1.297	1.303	1.310	1.302
Opt 3	1.2283	1.369	1.346	1.224	1.198	1.195	0.989	1.204	1.083
Opt 4	1.475	1.427	1.410	1.398	1.388	1.378	1.368	1.360	1.345

Table 13: Comparison of the 4 models taking into account the constraints giving $E(X_T) - \beta E((X_T - E(X_T))^2)$ obtained multiplying learning rates by 2, using a batch size of 500 and 20000 gradient iterations.

β	0	0.23	0.479	0.719	0.959	1.198	1.427	1.678	2.158
Opt 1	1.457	1.369	1.400	1.395	1.385	1.344	1.323	1.344	1.335
Opt 2	1.297	1.280	1.346	1.304	1.265	1.296	1.270	1.237	1.230
Opt 3	-284	-208	-335	-309	-454	-336	-185	-239	-148
Opt 4	1.474	1.425	1.409	1.397	1.386	1.377	1.368	1.357	1.345

Table 14: Opposite of the objective function obtained multiplying learning rates by 2, using a batch size of 500 and 20000 gradient iterations.

Once again models 1 and 4 are the best to respect the constraints and results for model 3 (always very bad) vary a lot with the hyper parameters taken.

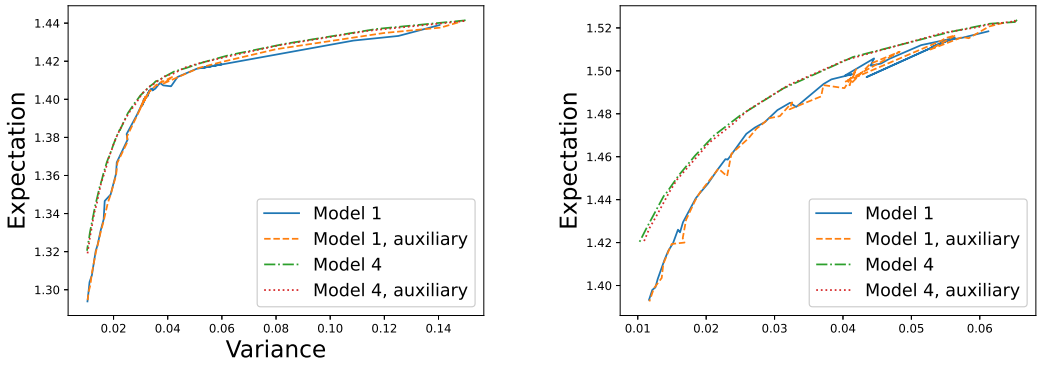
4.3.2 Point by point approximation for efficient frontier

We keep model 1 and model 4 to test them further in different dimensions. We suppose that the weights are between $0.5p_{Init}$ and $2p_{Init}$ where $p_{Init} = \frac{1}{d}$ is the initial equal weight used in the portfolio. Local constraints are kept as in the previous subsection with variations below 0.05 in absolute value between two rebalancing date. Optimization parameters are:

- Model 1: initial learning rate equal to $\frac{0.810^{-3}}{d}$ and linearly decreasing to $\frac{0.810^{-4}}{d}$ with gradient iterations,
- Model 4: initial learning rate equal to 10^{-3} and linearly decreasing to 10^{-5}.

Remark 4.6 *Model 1 has a convergence more sensitive than Model 4 to the learning rate depending on the dimension. We found that such a scaling gives good results.*

25000 iterations of gradient are used. The batch size is equal to 100 to reduce the computational time as we found that this is sufficient. In dimension 4, the efficient frontier is estimated using 40 discretization points. In dimension 20, 40 discretization points are used with model 1 while 16 points are used only with model 4: it is important to notice that the projection algorithm leads to a computational time far more important as the dimension of the problem increases.



Dimension 4 Dimension 20

Figure 6: Comparing model 1 and 4 taking into account constraints in a point by point approximation of the efficient frontier.

In figure 6, we compare model 1 and 4 to calculate the efficient frontier using the direct or auxiliary problem. Direct and auxiliary problems give the same frontier for both methods and model 4 is clearly superior to model 1. In the following model 4 will be used to deal with the global/local constraints.

4.3.3 Global optimization and comparison with point by point optimization of the efficient frontier.

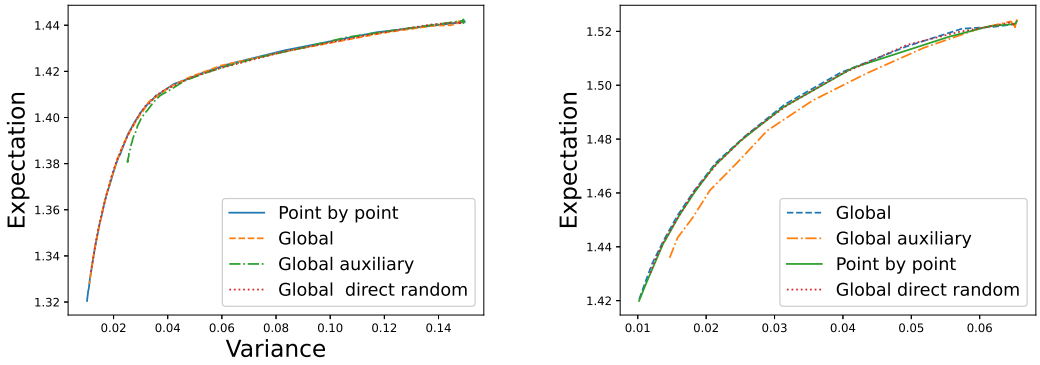
We keep the constraints defined in section 4.3.2. The model 4 can be extended easily to estimate the frontier globally: the network ζ^θ with parameters θ is now a function of t, X, β still with values in $[0,1]^d$. The weight approximation is carried out by:

$$\hat{\phi}^{\theta;\beta}(t,X,\beta) = \underline{\phi} + \zeta^\theta(t,X,\beta)(\bar{\phi} - \underline{\phi}). \tag{29}$$

The projection algorithm (27) is used and the objective function (13) is replaced by:

$$\begin{aligned}\theta^* =& \underset{\theta}{\operatorname{argmin}}\, \mathbb{E}[-\mathbb{E}[X_T^{\hat{\phi}^{\theta;\hat{\beta}}}|\hat{\beta}] + \hat{\beta}\mathbb{E}[(X_T^{\hat{\phi}^{\theta;\hat{\beta}}} + \mathbb{E}[X_T^{\hat{\phi}^{\theta;\hat{\beta}}}|\hat{\beta}])^2|\hat{\beta}]] + \\ & \frac{1}{\epsilon}\sum_{j=1}^{d}\sum_{i=0}^{N-2}\mathbb{E}[|\hat{\phi}_j^\theta(t_{i+1}, X_{t_{i+1}}^{\hat{\phi}^{\theta};\hat{\beta}}, \hat{\beta}) - \hat{\phi}_j^\theta(t_i, X_{t_i}^{\hat{\phi}^{\theta};\hat{\beta}}, \hat{\beta})| - \eta_j)^+]\end{aligned} \tag{30}$$

Of course a similar objective function can be written for the auxiliary problem.
We only present the results obtained using the deterministic (for β and γ) global method as the randomized approach tends to represent only a part of the curve or is suboptimal as seen on the case without constraints. As before, in dimension 4, 40 points are used to estimate the efficient frontier while only 16 are used in dimension 20.



Dimension 4 Dimension 20

Figure 7: Point by point versus global efficient frontier estimation.

On figure 7, we plot the efficient frontier obtained by point by point and global estimation. The point by point calculation and the direct global calculations (deterministic and randomized version) give the same curves in dimension 4 and 20. The global auxiliary approach gives a suboptimal curve in dimension 4 and 20 for its deterministic version while the randomize version gives solution with a flat variance in both cases so is not reported.

Remark 4.7 *In the whole section, we have added local constraints in the strategies. It is also possible to remove the local constraints and take into account the transaction costs directly in the dynamic of the asset. Supposing that half of spread bid ask for asset i is given by p_i, and still supposing that we deal with the discrete-time optimization* (10)*, then the final value of the assets is given with the convention $\phi_{-1} = 0$ by*

$$X_T^\phi = X_0 + \sum_{i=0}^{N-1} \phi_i X_{t_i}^\phi . \frac{S_{t_{i+1}} - S_{t_i}}{S_{t_i}} + \sum_{i=0}^{N-1} \sum_{j=1}^{d} |\frac{\phi_{i,j} X_{t_i}^\phi}{S_{t_i,j}} - \frac{\phi_{i-1,j} X_{t_{i-1}}^\phi}{S_{t_{i-1},j}}| p_j \tag{31}$$

Now the control depends not only on the wealth but also on S_t and the weight at the previous date and they have to be included in the state at the input of the network.

5 Time discrete optimization under an Heston model

We now suppose that the assets follow the Heston model Heston 1993:

$$\begin{aligned} dS_t &= \mu S_t dt + \sqrt{V_t} S_t dW_t^1, \\ dV_t &= \kappa(\bar{V} - V_t)dt + \bar{\sigma}\sqrt{V_t} dW_t^2 \end{aligned} \tag{32}$$

where S_t, V_t with values in $\mathbb{R}^d$, (W_t^1, W_t^2) is a vector of $2d$ Brownian correlated with a correlation matrix ρ. As there is no analytic solution to equation (32), we rely on a Milstein scheme on the volatility Kahl and Jäckel 2006 to assure positivity of the volatility under the Feller condition

$2\kappa\bar{V} \leq \bar{\sigma}^2$.
We are only interested in the discrete problem with no short selling and no borrowing : weights are all in $[0,1]$ and the summation of the weights is equal to one. This problem is interesting as the state of the problem now involve not only the wealth X_t but also the variance V_t, such that not only the control but also the global state has a dimension increasing with the number of assets in the portfolio.
We first deal with the case without additional constraints and then the case with local and global constraints. For all optimizations, we take a learning rate initially equal to 10^{-3} and linearly decreasing to 10^{-4} with stochastic gradient iterations. We take 25000 gradient iterations except when specified. The batch size is equal to 100.

5.1 No additional constraints

Using the previous algorithm equation (17) and (18) are now replaced by

$$\hat{\phi}^{\theta}(t, X, V_1, \ldots, V_d) = \frac{\zeta^{\theta}(t, X, V_1, \ldots, V_d)}{\sum_{i=1}^{d} \zeta_i^{\theta}(t, X, V_1, \ldots, V_d)} \tag{33}$$

for the point by point approximation and by

$$\hat{\phi}^{\theta;\beta}(t, X, V_1, \ldots, V_d, \beta) = \frac{\zeta^{\theta}(t, X, V_1, \ldots, V_d, \beta)}{\sum_{i=1}^{d} \zeta_i^{\theta}(t, X, V_1, \ldots, V_d, \beta)} \tag{34}$$

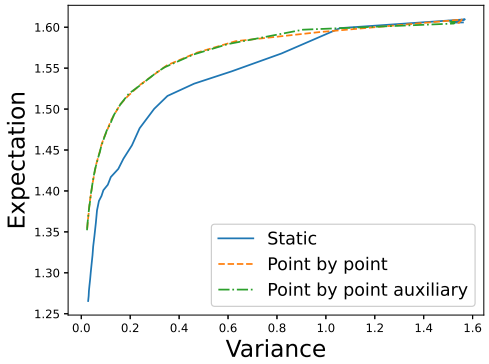
in the global method where V_j is the variance of the asset j.
In the whole section we still take $T = 10$ years and a rebalancing every month. In dimension 4 we take the following parameters:

$$\mu = (0.01, 0.0225, 0.035, 0.0475)^T,$$

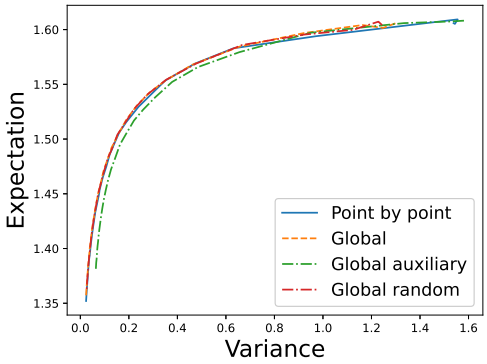
$\bar{V}$ is taken equal to V_0 and given by $(0.0025, 0.01, 0.0225, 0.04)^T$. Volatility of the variance $\bar{\sigma}$ is given by the diagonal matrix with coefficients $(0.05, 0.1, 0.15, 0.2)$. All the κ values are equal to 0.5. The initial asset values are all equal to one. The correlation matrix associated with $(W^1_{t,1}, \ldots, W^1_{t,d}, W^2_{t,1}, \ldots, W^2_{t,d})$ is given by

$$\rho = \begin{pmatrix} 1. & -0.383 & 0.378 & -0.324 & -0.751 & -0.110 & 0.272 & 0.465 \\ -0.383 & 1. & -0.938 & -0.411 & -0.053 & 0.276 & -0.226 & -0.349 \\ 0.378 & -0.938 & 1. & 0.145 & -0.051 & -0.398 & 0.249 & 0.401 \\ -0.324 & -0.411 & 0.145 & 1. & 0.655 & 0.329 & -0.264 & -0.048 \\ -0.751 & -0.053 & -0.051 & 0.655 & 1. & -0.172 & -0.464 & -0.105 \\ -0.110 & 0.276 & -0.398 & 0.329 & -0.172 & 1. & 0.044 & -0.348 \\ 0.272 & -0.226 & 0.249 & -0.264 & -0.464 & 0.044 & 1. & -0.580 \\ 0.465 & -0.349 & 0.401 & -0.048 & -0.105 & -0.348 & -0.580 & 1. \end{pmatrix}.$$

In dimension 10 a similar test case is created with quite high correlations. All curves are plotted using 30 points. The convergence of the "global auxiliary" being very slow, 50000 gradient iterations have been used specially for the curves "global auxiliary".



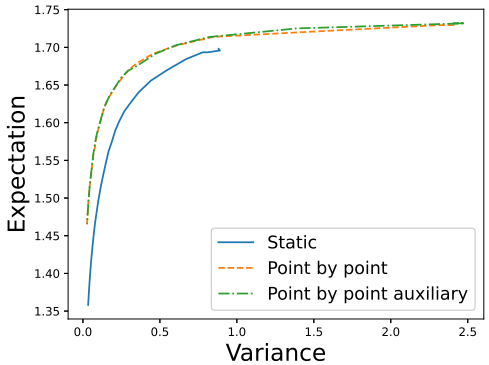
Point by point



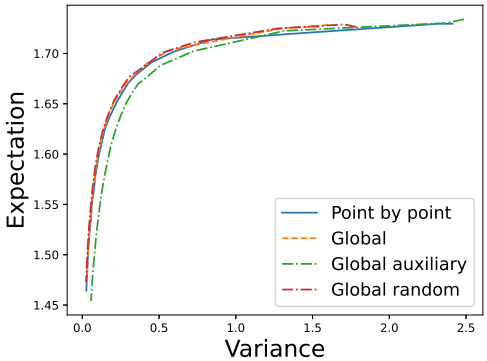
Point by point versus global

Figure 8: Results in dimension 4 for the Heston model without additional constraints.

In figure 8, we first plot in dimension 4 the efficient frontier obtained by direct optimization and using auxiliary equations. Both approaches give the same curve, well above the static curve which surprisingly doesn't seem to be very well estimated (by a point by point approach) as convexity of the curve is not totally respected. In figure 8, we also compare global estimations to point by point estimations: using the direct approach (deterministic and randomized version) we get the same curve as the one given by the point by point estimation. Using the auxiliary version with the global approach, we get similar results as in the Black-Scholes case: the deterministic version gives a suboptimal curve while the randomized version is not reported as it gives bad results.
In figure 9, we give the same results in dimension 10. We get very similar results, except that the static point by point optimization curve is more realistic.



Point by point



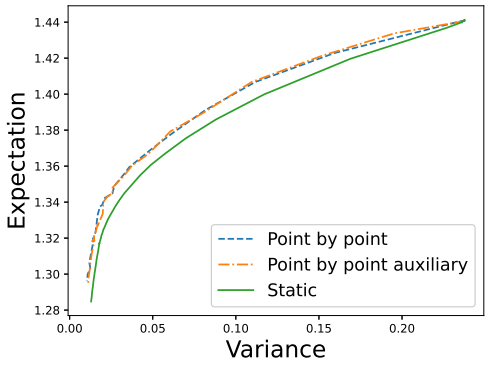
Point by point versus global

Figure 9: Results in dimension 10 for the Heston model without additional constraints.

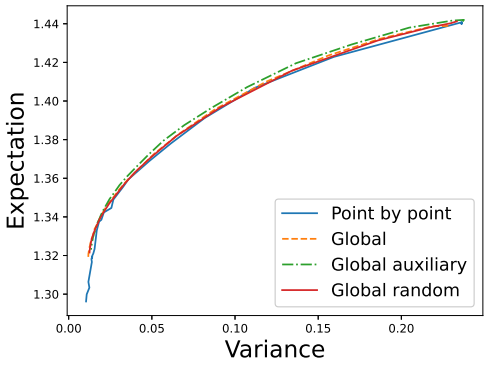
5.2 Imposing additional local and global constraints

We keep the constraints used in the Black-Scholes model in section 4.3.2. The equation (26) is modified taking into account the fact that the state includes the variance of the different assets

for the point by point optimization and the equation (29) is modified in the same way for global optimization.



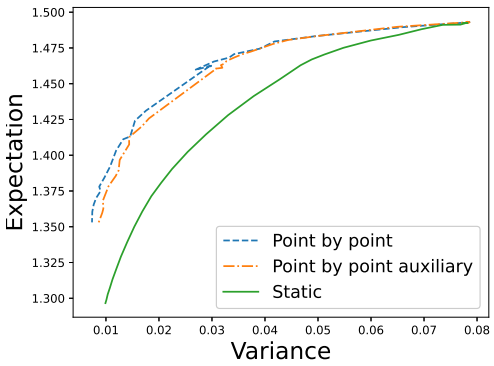
Point by point



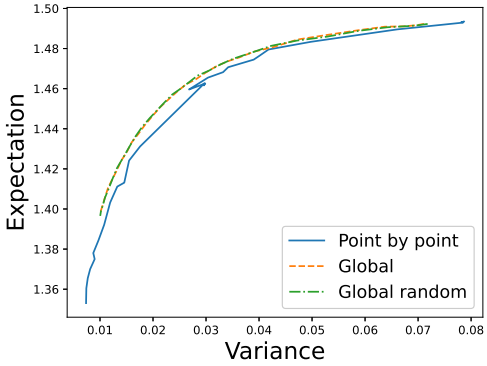
Point by point versus global

Figure 10: Results in dimension 4 for the Heston model with additional constraints.

Figure 10 seems to indicate that we are able to recover the frontier with a good accuracy both by the point by point and the global method.



Point by point



Point by point versus global

Figure 11: Results in dimension 8 for the Heston model with additional constraints.

In higher dimension (for example in figure 11 in dimension 8), the convergence is harder to achieve and small oscillations appear for the point by point approach. In this case, the global auxiliary results are not reported as the curve obtained is not satisfactory. Globally, the global version on the direct approach appears to be the most effective.

6 Mean-CVaR risk optimization

In the whole section we suppose that the assets follow a Black-Scholes model given by equation (3). As the Mean-Variance case penalizes both gains and losses, practitioners prefer to use some downside risk only penalizing the losses (or small gains). In this part we focus on the Mean-CVaR criterion.

We first recall the VaR definition: if X is a distribution of gain with a continuous cumulative distribution $F_X(x)$ and $\alpha \in]0,1[$, we note

$$VaR(X,\alpha) = \min(y \in \mathbb{R} / P(-X \leq y) \geq \alpha) \tag{35}$$

where $P(-X \leq x)$ is the probability that the random variable $-X$ is below x, so that $P(-X \leq x) = F_{-X}(x)$.
CVaR is then the average loss conditionally to the fact that the losses are above the VaR:

$$CVaR(X,\alpha) = \mathbb{E}[-X/-X \geq VaR(X,\alpha)] \tag{36}$$

The VaR criterion is not convex but the CVaR is convex Rockafellar, Uryasev, et al. 2000. The Mean-CVaR problem consists in finding strategies adapted to the available information and minimizing

$$(J_1(\phi), J_2(\phi)) = (-\mathbb{E}[X_T^\phi], CVaR(X_T^\phi - X_0, \alpha)). \tag{37}$$

where X_T^ϕ is the value of portfolio managed with strategy ϕ and given by (10).
As for the Mean-Variance case, an admissible strategy ϕ^* is efficient if there is no other admissible strategy ψ such that

$$J_1(\psi) \leq J_1(\phi^*), \quad J_2(\psi) \leq J_2(\phi^*)$$

with at least one of the last inequality being strict. The set of efficient points $(J_1(\phi^*), J_2(\phi^*))$ defines the efficient frontier.
As recalled in the introduction, the continuous optimization problem is not well posed and we will only focus on the discrete-time optimization problem.
It is also possible to recast the problem as minimizing the CVaR under a constraint that the expected value of the gains is above a threshold M:

$$\min_{\phi \in \mathcal{L}^2_{\mathcal{F}_t}(0,T,\mathbb{R}^d)} CVar(X_T^\phi - X_0) \tag{38}$$

$$\text{with } E[X_T^\phi] \geq M \tag{39}$$

so that using Rockafellar, Uryasev, et al. 2000 representation,

$$\min_{y,\phi \in \mathcal{L}^2_{\mathcal{F}_t}(0,T,\mathbb{R}^d)} \mathbb{E}[y + \frac{1}{1-\alpha}(-X_T^\phi + X_0 - y)^+] \tag{40}$$

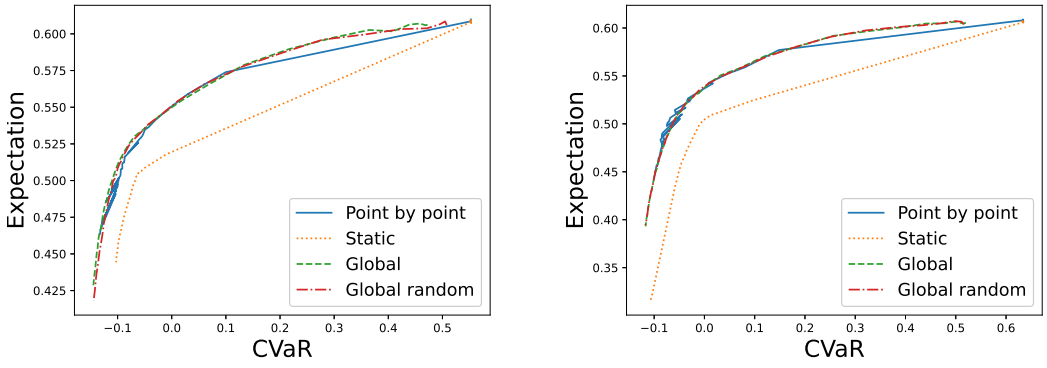
$$\text{with } E[X_T^\phi] \geq M \tag{41}$$

and when optimality is reached, y corresponds to the VaR of the portfolio.
Conventional numerical methods generally prefer this formulation as if y is fixed, the problem can be solved by dynamic programming. Then combining a gradient descent method with this optimization where y is fixed, it possible to optimize the portfolio.
The interest of this formulation is not obvious with neural network problem as it adds a fictitious dimension on the problem and it turns out that using this formulation goes not give good results such that we don't report them in the article.
In the sequel use the same formulation as in the Mean-Variance case: the efficient frontier can be calculated by minimizing (6) using the definition (37) for a fixed β and let β vary to describe the frontier.
As in the Mean-Variance case, it is possible to use a neural network to optimize the strategies by minimizing equation (11) for a point by point approximation of the frontier or minimizing the single problem (13) to learn the global curve.
In all the tests, the parameters of the model are the same as in section 4. As before, $T = 10$

years and rebalancing is achieved every month.
As for the convergence of the stochastic gradient, the learning rate starts at 10^{-4} and decreases linearly with gradient iterations to 10^{-5}. The number of stochastic gradient iterations is fixed at 15000. The batch size is chosen equal to 2000 to have a correct assessment of the CVaR. While calculating the curve point by point or globally with deterministic β, the following values $\beta_i = (i2/K)^2$, $i = 0, \dots, K-1$ are used. The value for K depends on the case. When a randomization of the parameter β is used, $\hat{\beta} \sim \beta_{K-1} U^2$ where U is a uniform random variable on $[0, 1]$. Other distributions have been tested trying to increase the density of points where the portfolio is more risky, but the global curve was not as good as with the uniform distribution.

6.1 Not short selling, no borrowing

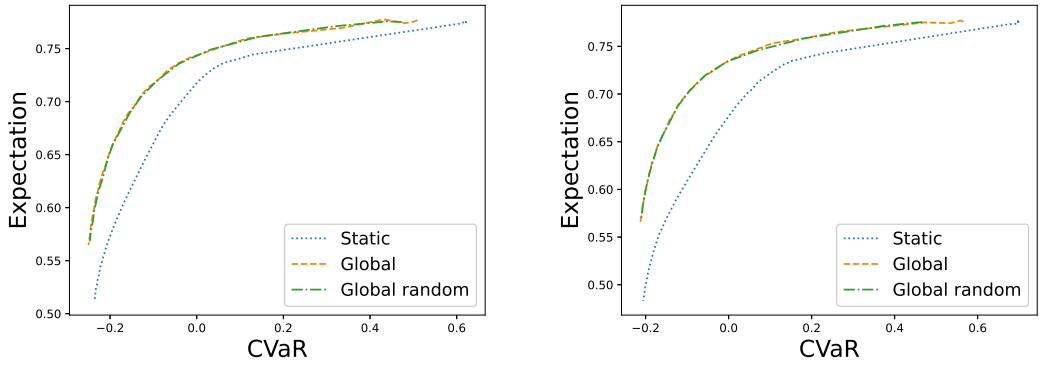
Similarly to the Mean-Variance we can impose constraints on the portfolio: using the weights formulation (17) while optimizing (6), the weights formulation (18) optimizing (13), it is possible to impose that all weights are positive, between 0 and 1 and with sum equal to one.
We train the network and plot the resulting efficient frontier using 40 points first in dimension 4 on figure 12. The global approaches with deterministic and stochastic β give the same curve. The point by point curve appears to be oscillating and may not be very accurate.



$\alpha = 0.9$ $\alpha = 0.95$

Figure 12: Mean-CVaR optimization comparison between point by point optimization with global optimization with deterministic and stochastic β in dimension 4.

In dimension 20, results with point by point approximation are not reported as they are not usable. Only global solution with deterministic and stochastic β are given on figure 13.



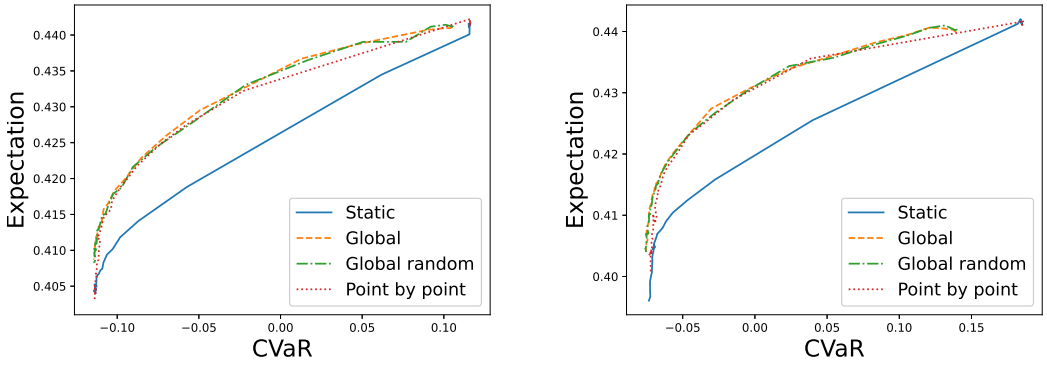
$\alpha = 0.9$ $\alpha = 0.95$

Figure 13: Mean-CVaR optimization comparison between static optimization and global optimization with deterministic and stochastic β in dimension 20.

For the Mean-CVaR criterion, it seems that the global approximation gives the best results.

6.2 Adding other constraints

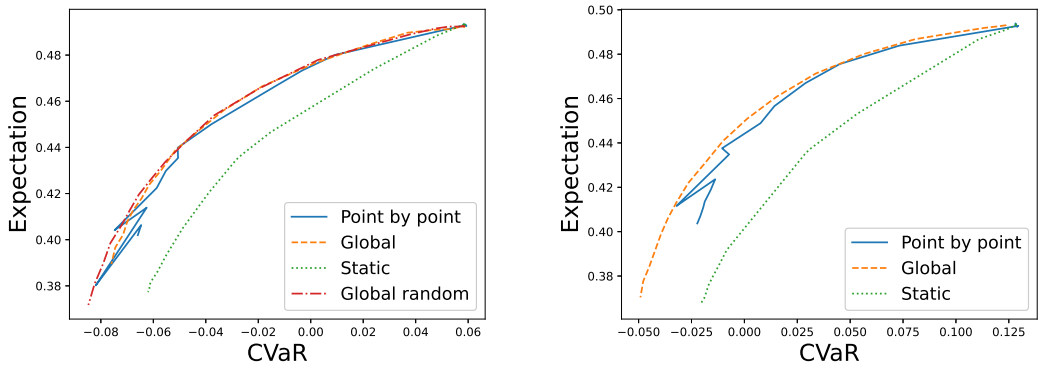
Using equation (26) with algorithm 1 in equation (28), or equation (29) for the weights while optimizing (30), it is possible to add the constraints given by equations (19) and (20). The parameters for the additional constraints are the same as in section 4.3.2. Results are reported in dimension 4 and only in dimension 8 as computing cost grows significantly with the dimension of the problem. Curves are reconstructed with 20 points. On figure 14, we plot the curves obtained with a static optimization, and the dynamic optimization with the global approaches and the point by point approach. The global approaches seem to give slightly better results.



$\alpha = 0.9$ $\alpha = 0.95$

Figure 14: Mean-CVaR in dimension 4 with additional constraints on weights : global, point by point, and static estimations.

In higher dimension, the point by point approach appears to give oscillations. It turns out that the best solution is given by the global approach with random β coefficients as shown on figure 15.



$\alpha = 0.9$ $\alpha = 0.95$

Figure 15: Mean CVaR in dimension 8 with additional constraints on weights : global, point by point, and static estimations.

For example, with $\alpha = 0.95$, the global resolution can lead to a suboptimal curve as shown on figure 16. The optimizer is trapped in a local minimum away from the solution and point by point estimations oscillates between the two curves.

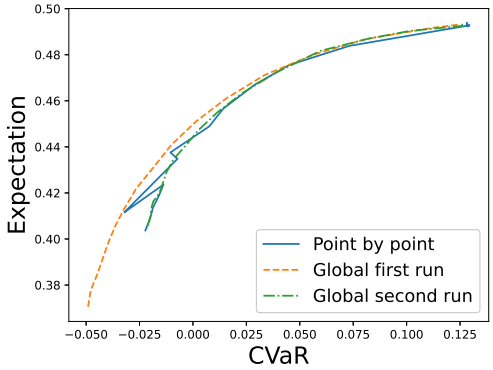


Figure 16: Different Mean CVaR runs with deterministic global method compared to point by point estimation with $\alpha = 0.95$.

So it seems that with or without additional constraints, the global approach generally gives more satisfactory curves than the point by point approach and it is smoother by construction.

7 Conclusion

We have shown that neural network can be used to efficiently compute the Markovitz frontier with or without weight constraints.
In the Mean-Variance case, all methods seem to approximate the Markovitz frontier very well for portfolios with not a too high or too low variance. Global random methods may have difficulties to either catch very low variance portfolios or very high variance portfolios depending on the formulation taken. When there are local and global constraints on the weights, the new projection algorithm presented in the article has to be used to get accurate results. Nevertheless,

the accuracy of the curve obtained depends on the asset model and the dimension of the model. Therefore we cannot say which formulation (direct (11) or auxiliary (12)) is the best and which resolution method "global" or "point by point" has to be used to get the best results.
In the Mean-CVaR case, the batch size to estimate the CVaR has to be very high, and the computational time is currently a limiting factor. The global methods appears to be more effective than point by point methods: imposing no borrowing, and no short selling, the curves obtained seem to be realistic. Adding additional constraints increasing a lot the computational time, the number of assets has to be limited and the solver can be trapped by local minima rather different depending on the run: it gives high oscillations with point by point methods and sometimes converges to a sub-optimal curve with global methods.
All the results in this article have been obtained using classical feedforward networks: some work on the structure of the networks used could perhaps improve the results in the difficult Mean-CVaR case.
Another interesting approach would be to directly optimize the expected gain with given bounds on the variance of the portfolio during the whole period or at some given dates:

$$\begin{aligned} \max_{\xi} \quad & \mathbb{E}[X_T^{\xi}] \\ & \mathbb{E}[(X_{T_i}^{\xi} - \mathbb{E}[X_{T_i}^{\xi}])^2] \leq M, \text{ for } i = 1, \dots, N, \end{aligned}$$

where $T_1 < \cdots < T_N = T$, leading to a better control of the risk of the portfolio during the whole period. The resolution of this problem using Neural Networks is a current subject of research.

References

Buehler, Hans, Lukas Gonon, Josef Teichmann, and Ben Wood. 2019. "Deep hedging." *Quantitative Finance* 19 (8): 1271–1291.

Chan-Wai-Nam, Quentin, Joseph Mikael, and Xavier Warin. 2019. "Machine learning for semi linear PDEs." *Journal of Scientific Computing* 79 (3): 1667–1712.

Chiu, Mei Choi, and Hoi Ying Wong. 2014. "Mean–variance portfolio selection with correlation risk." *Journal of Computational and Applied Mathematics* 263:432–444.

Cong, Fei, and Cornelis W Oosterlee. 2016. "Multi-period mean–variance portfolio optimization based on Monte-Carlo simulation." *Journal of Economic Dynamics and Control* 64:23–38.

Dang, Duy-Minh, and Peter A Forsyth. 2014. "Continuous time mean-variance optimal portfolio allocation under jump diffusion: An numerical impulse control approach." *Numerical Methods for Partial Differential Equations* 30 (2): 664–698.

Dembo, Ron, and Dan Rosen. 1999. "The practice of portfolio replication. A practical overview of forward and inverse problems." *Annals of Operations Research* 85:267–284.

Fécamp, Simon, Joseph Mikael, and Xavier Warin. 2019. "Risk management with machine-learning-based algorithms." *arXiv preprint arXiv:1902.05287.*

Fishburn, Peter C. 1977. "Mean-risk analysis with risk associated with below-target returns." *The American Economic Review* 67 (2): 116–126.

Gao, Jianjun, Ke Zhou, Duan Li, and Xiren Cao. 2017. "Dynamic mean-LPM and mean-CVaR portfolio optimization in continuous-time." *SIAM Journal on Control and Optimization* 55 (3): 1377–1397.

Gatheral, Jim, Thibault Jaisson, and Mathieu Rosenbaum. 2018. "Volatility is rough." *Quantitative Finance* 18 (6): 933–949.

Germain, Maximilien, Huyen Pham, and Xavier Warin. 2020. "Deep backward multistep schemes for nonlinear PDEs and approximation error analysis." *arXiv preprint arXiv:2006.01496.*

Han, Jiequn, Arnulf Jentzen, and E Weinan. 2018. "Solving high-dimensional partial differential equations using deep learning." *Proceedings of the National Academy of Sciences* 115 (34): 8505–8510.

Heston, Steven L. 1993. "A closed-form solution for options with stochastic volatility with applications to bond and currency options." *The review of financial studies* 6 (2): 327–343.

Hornik, K., M. Stinchcombe, and H. White. 1989. "Multilayer feedforward networks are universal approximators." *Neural Networks* 2 (5): 359–366.

Huré, Côme, Huyên Pham, and Xavier Warin. 2020. "Deep backward schemes for high-dimensional nonlinear PDEs." *Mathematics of Computation* 89 (324): 1547–1579.

Ismail, A., and H. Pham. 2019. "Robust Markowitz mean-variance portfolio selection under ambiguous covariance matrix." *Mathematical Finance* 29 (174-207).

Jaber, Eduardo Abi, Enzo Miller, and Huyên Pham. 2020. "Markowitz portfolio selection for multivariate affine and quadratic Volterra models." *Available at SSRN 3636794.*

Jin, Hanqing, Jia-An Yan, and Xun Yu Zhou. 2005. "Continuous-time mean–risk portfolio selection." In *Annales de l'Institut Henri Poincare (B) Probability and Statistics,* 41:559–580. 3. Elsevier.

Kahl, Christian, and Peter Jäckel. 2006. "Fast strong approximation Monte Carlo schemes for stochastic volatility models." *Quantitative Finance* 6 (6): 513–536.

Kingma, Diederik P, and Jimmy Ba. 2014. "Adam: A method for stochastic optimization." *arXiv preprint arXiv:1412.6980.*

Krabichler, Thomas, and Josef Teichmann. 2020. "Deep Replication of a Runoff Portfolio." *arXiv preprint arXiv:2009.05034.*

Li, Duan, and Wan-Lung Ng. 2000. "Optimal dynamic portfolio selection: Multiperiod mean-variance formulation." *Mathematical finance* 10 (3): 387–406.

Li, Xun, Xun Yu Zhou, and Andrew EB Lim. 2002. "Dynamic mean-variance portfolio selection with no-shorting constraints." *SIAM Journal on Control and Optimization* 40 (5): 1540–1555.

Lim, Andrew EB. 2004. "Quadratic hedging and mean-variance portfolio selection with random parameters in an incomplete market." *Mathematics of Operations Research* 29 (1): 132–161.

Markowitz, H. 1952. "Portfolio selection." *J Finance* 7 (77–91).

———. 1959. *Portfolio Selection: Efficient Diversification of Investment.* Wiley, New York.

Martin Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, et al. 2015. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems.* Software available from tensorflow.org. https://www.tensorflow.org/.

Merton, Robert C. 1972. "An analytic derivation of the efficient portfolio frontier." *Journal of financial and quantitative analysis,* 1851–1872.

Miller, Christopher W, and Insoon Yang. 2017. "Optimal control of conditional value-at-risk in continuous time." *SIAM Journal on Control and Optimization* 55 (2): 856–884.

Pham, Huyen, Xavier Warin, and Maximilien Germain. 2021. "Neural networks-based backward scheme for fully nonlinear PDEs." *SN Partial Differential Equations and Applications* 2 (1): 1–24.

Rockafellar, R Tyrrell, Stanislav Uryasev, et al. 2000. "Optimization of conditional value-at-risk." *Journal of risk* 2:21–42.

Roy, Andrew Donald. 1952. "Safety first and the holding of assets." *Econometrica: Journal of the econometric society,* 431–449.

Wang, Jian, and Peter A Forsyth. 2010. "Numerical solution of the Hamilton–Jacobi–Bellman formulation for continuous time mean variance asset allocation." *Journal of Economic Dynamics and control* 34 (2): 207–230.

Warin, Xavier. 2021. "Reservoir optimization and Machine Learning methods." *arXiv preprint arXiv:2106.08097.*

Zhou, Xun Yu, and Duan Li. 2000. "Continuous-time mean-variance portfolio selection: A stochastic LQ framework." *Applied Mathematics and Optimization* 42 (1): 19–33.