Quantile and moment neural networks for learning functionals of distributions. [*]

Xavier WARIN [†]

March 17, 2023

Abstract

We study news neural networks to approximate function of distributions in a probability space. Two classes of neural networks based on quantile and moment approximation are proposed to learn these functions and are theoretically supported by universal approximation theorems. By mixing the quantile and moment features in other new networks, we develop schemes that outperform existing networks on numerical test cases involving univariate distributions. For bivariate distributions, the moment neural network outperforms all other networks.

1 Introduction

The deep neural networks have been successfully used to solve high dimensional PDEs either by solving the PDE using physics informed methods, or by using backward stochastic differential equations (see [2], [6] for an overview). Recently the mean field game and control theory has allowed the formalization of problems involving large populations of interacting agents. The solution of such problems is a function depending on the probability distribution of the population and can be obtained by solving a PDE in the Wasserstein space of probability measures (called the Master equation) or by solving BSDEs of McKean-Vlasov (MKV) (see [3, 4]). In this case, the resulting PDE is infinite dimensional and must be reduced to a (high) finite dimensional problem to be tractable.
To solve such problems, [11] has developed two schemes approximating functions depending on both X a random variable and μ a probability distribution, where $X \sim \mu$. The first scheme is based on a bin representation of the density and the second uses a neural network to automatically extract the key features of the distribution. In both cases, these key features and the X values are used as input to a neural network permitting to approximate the value function. The first approach is the bin network and the second one is the cylinder network. Both schemes have been successfully applied to various toy cases in the case of one-dimensional distributions and used to solve the master equation using its semi-lagrangian representation in [10]. As we explain in the next section, the X dependence on the functional is not relevant for testing the various networks developed, and we focus in this article only on the dependence on the distribution.
In this article we propose new different networks to approximate functions depending on distributions:

- The first one, limited to one-dimensional distributions, uses the quantile of the distribution as key features : the resulting scheme gives the quantile network scheme.
- The second ones uses the moments of the distribution as key features and leads to the moment network scheme.
- Finally, the two previous features can be mixed to take advantage of the first two networks.

We give some universal approximation theorems for the first two networks. We test the developed networks on functions of univariate and bivariate distributions, where possible. We compare the proposed networks with the bin network and the cylinder network and show that:

[*]This work is supported by FiME, Laboratoire de Finance des Marchés de l'Energie, and the "Finance and Sustainable Development" EDF - CACIB Chair.

[†]EDF R&D & FiME xavier.warin at edf.fr

- The moment network **or** the quantile network outperform the cylinder network and the bin network in the case of univariate distributions: the best solution obtained by the two networks always gives better results (on our tests) than the cylinder network and the bin network.
- By combining quantile and moment, we obtain a neural network that always outperforms the cylinder and the bin networks.
- In the case of bivariate distributions, the bin network fails and the moment network outperforms all other networks.

The structure of the article is as follows. In a first section, we formalize our problem as a minimization problem on the distribution space using a formal neural network. We give the general methodology for sampling distributions in the general multivariate case, and show how to solve the previous minimization problem using a stochastic gradient method. The second section is dedicated to the different proposed neural networks. The last one is dedicated to numerical results for univariate and bivariate distributions. A final conclusion is given.

Notations. Denote by $\mathcal{P}_2(\mathbb{R}^d)$ the Wasserstein space of square integrable probability measures equipped with the 2-Wasserstein distance $\mathcal{W}_2$. Given some $\mu \in \mathcal{P}_2(\mathbb{R}^d)$, and ϕ a measurable function on $\mathbb{R}^d$ with quadratic growth condition, hence in $L^2(\mu)$, we set: $\mathbb{E}_{X\sim\mu}[\phi(X)] := \int \phi(x)\mu(\mathrm{d}x)$.

2 Learning distribution functions

Given a function V on $\mathcal{P}_2(\mathbb{R}^d)$, valued on $\mathbb{R}^p$, we want to approximate the infinite-dimensional mapping

$$\mathcal{V} : \mu \in \mathcal{P}_2(\mathbb{R}^d) \longmapsto V(\mu) \in \mathbb{R}^p,$$

called the distribution function, by a map $\mathcal{N}$ constructed from suitable classes of neural networks. The distribution network $\mathcal{N}$ takes input μ a probability measure and outputs $\mathcal{N}(\mu)$. The quality of this approximation is measured by the error:

$$L(\mathcal{N}) := \int_{\mathcal{P}_2(\mathbb{R}^d)} \big|\mathcal{V}(\mu) - \mathcal{N}(\mu)\big|^2 \nu(\mathrm{d}\mu) \tag{2.1}$$

where ν is a probability measure on $\mathcal{P}_2(\mathbb{R}^d)$, called the training measure. The distribution function $\mathcal{V}$ is learned by minimizing the loss function over the parameters of the neural network operator $\mathcal{N}$.
In the article [11], the authors learn what they call a mean-field function, a function $\hat{V}$ depending on both μ and x. The network is a function $\hat{\mathcal{N}}$ that takes as input μ a probability measure and x in the support of μ, and outputs $\hat{\mathcal{N}}(\mu)(x)$. The solution is found by minimizing:

$$\hat{L}(\hat{\mathcal{N}}) := \int_{\mathcal{P}_2(\mathbb{R}^d)} \mathbb{E}_{X\sim\mu}\big|\hat{V}(X,\mu) - \hat{\mathcal{N}}(\mu)(X)\big|^2 \nu(\mathrm{d}\mu). \tag{2.2}$$

The resolution of the equation (2.2) is more general than the resolution of the equation (2.1), but in fact the result is simply obtained by concatenating x and the representation of the distribution μ as input to the neural network, similarly as suggested in [5]. Therefore we focus on the resolution of the problem (2.1).
In the following, we explain how to sample distributions and how to generate samples for a given distribution in the multivariate case. Then we explain the global methodology used to train the neural networks. This methodology is used for all the networks developed in the next sections.

2.1 Sampling distribution on a compact set

To learn a function of a distribution μ with support in $\mathcal{K} = [\underline{\mathcal{K}}_1, \bar{\mathcal{K}}_1] \times \ldots \times [\underline{\mathcal{K}}_d, \bar{\mathcal{K}}_d] \subset \mathbb{R}^d$, we must be able to "generate" distributions and, having chosen the distribution μ , to efficiently sample $X \sim \mu$ in $\mathbb{R}^d$.
As done in [11], we use a bin representation but propose a different algorithm to tackle the multivariate

case. By tensorization, a multivariate bin representation for a lattice $(J_1, \dots, J_d)$ is given, for $(j_1, \dots, j_d) \in [1, J_1] \times \dots \times [1, J_d]$, by

$$\mathrm{Bin}(j_1, \dots, j_d) = \prod_{i=1}^{d} [\underline{\mathcal{K}}_i + (j_i - 1)\frac{\bar{\mathcal{K}}_i - \underline{\mathcal{K}}_i}{J_i}, \underline{\mathcal{K}}_i + j_i \frac{\bar{\mathcal{K}}_i - \underline{\mathcal{K}}_i}{J_i}].$$

- First, we generate a distribution μ by sampling $e_1, \dots, e_{\prod_{i=1}^{d} J_i}$, positive random variables according to an exponential law, and set for $(j_1, \dots, j_d) \in [1, J_1] \times \dots \times [1, J_d]$

$$p(j_1, \dots, j_d) = \frac{e_{j_1 + j_2 J_1 + \dots j_d \prod_{i=1}^{d-1} J_i}}{\sum_{i=1}^{\prod_{i=1}^{d} J_i} e_i}$$

  which gives a constant per bin probability measure where the probability of sampling in $\mathrm{Bin}(j_1, \dots, j_d)$ is given by $p(j_1, \dots, j_d)$.

- Now that we have chosen μ, we can generate N samples of d dimensional coordinates $(j_1^n, \dots, j_d^n) \in [1, J_1] \times \dots \times [1, J_d]$ for $n \in [1, N]$ such that

$$\mathrm{proba}[(j_1^n, \dots, j_d^n) = (j_1, \dots, j_d)] = p(j_1, \dots, j_d).$$

  Finally, we sample $Y^n \sim \mathbf{U}([0,1]^d)$ for $n = 1, N$, and set

$$\begin{aligned} X^n =& (X_1^n, \dots, X_d^n), \\ \text{where} \quad X_i^n =& \underline{\mathcal{K}}_i + (j_i^n - 1 + Y_i^n)\frac{\bar{\mathcal{K}}_i - \underline{\mathcal{K}}_i}{J_i}, \text{for} \quad i = 1, \dots, d. \end{aligned}$$

Remark 2.1. *This procedure allows to generate points according to a constant density function per bin. In dimension one, it is equivalent to the algorithm proposed in [11] which generates points with a linear representation of the cumulative distribution function.*

2.2 The training methodolody

Since the equation (2.1) is infinite dimensional, we need to introduce a discretization of the measure. We note $R^K(\mu) := (R_k^K(\mu))_{k=1,\dots,K}$ the K features estimated from the law μ. The features selected depend on the method developed and will be detailed in the following sections.
The neural network $\mathcal{N}(\mu) := \Phi_\theta(R^K(\mu))$ is such that Φ_θ is an operator from $\mathbb{R}^K$ to $\mathbb{R}^p$ depending on some parameters θ and we use a gradient descent algorithm (ADAM [8]) with Tensorflow software [1] to minimize the loss

$$\bar{L}(\theta) := \int_{\mathcal{P}_2(\mathbb{R}^d)} \left| \mathcal{V}(\mu) - \Phi_\theta(R^K(\mu)) \right|^2 \nu(\mathrm{d}\mu) \tag{2.3}$$

with respect to the parameters θ.
At each iteration of the stochastic gradient,

- M distributions $(\mu^m)_{m=1,M}$ are generated and for each μ^m, $X^{m,n} \sim \mu^m$ are generated for $n = 1, \dots, N$, following the methodology given in section 2.1.

- The K features representing the law are estimated from the N samples for a given estimator $R^{K,N}(\mu^m) := (R_k^{K,N}((X^{m,n})_{n=1,N}))_{k=1,\dots,K}$.

The discretized version of the loss function (2.3) is then

$$\tilde{L}(\theta) := \frac{1}{M} \sum_{m=1}^{M} \left| V(\mu^m) - \Phi_\theta(R^{K,N}(\mu^m)) \right|^2.$$

The learning rate associated with the gradient method is equal to 5×10^{-3} in all the experiments.

3 The networks

3.1 The quantile network for one dimensional distribution

Let $\mathcal{D}_2(\mathbb{R})$ be the subset of probability measures μ in $\mathcal{P}_2(\mathbb{R})$ admitting a density function p^μ with respect to the Lebesgue measure λ on $\mathbb{R}$. Fixing $\mathcal{K}$ as a bounded segment in $\mathbb{R}$, we want to approximate the functions of the distributions with support in $\mathcal{K}$.
For a distribution μ, we note F_μ its cumulative distribution function and we note Q_μ the quantile function defined as $Q_\mu(p) = \inf\{x \in \mathcal{K} : p \leq F_\mu(x)\}$. In the sequel, we also use the notation Q_X for Q_μ if $X \sim \mu$.
Choosing $K > 0$, the main characteristics of the distribution μ are given by

$$\mathrm{Q}_\mu^K = (Q_\mu(\frac{k}{K+1}))_{k=1,K}$$

which lies on $\mathcal{D}_K := \{Q := (q_k)_{k=1,K} : q_1 < \cdots < q_K\}$.
A quantile network is thus an operator on $\mathcal{D}_2(\mathbb{R})$ in the form

$$\mathcal{N}_Q(\mu) = \Phi_\theta(\mathrm{Q}_\mu^K),$$

so setting $R^K(\mu) = \mathrm{Q}_\mu^K$ in equation (2.3).
Let us denote by $\mathcal{D}_{C^1}(\mathcal{K})$ the subset of elements μ in $\mathcal{D}_2(\mathbb{R})$ with support in $\mathcal{K}$, with continuously derivable density functions p^μ. We get the following universal approximation theorem:

Theorem 3.1. *Let $\mathcal{K} = [\underline{\mathcal{K}}, \bar{\mathcal{K}}]$ be a bounded segment in $\mathbb{R}$, V a continuous function from $\mathcal{P}_2(\mathbb{R})$ to $\mathbb{R}$. Then, for all $\varepsilon > 0$, there exists $K \in \mathbb{N}^*$, and Φ a neural network on $\mathbb{R}^K$ with values in $\mathbb{R}$ such that*

$$\left|V(\mu) - \Phi(\mathrm{Q}_\mu^K)\right| \leq \varepsilon, \quad \forall \mu \in \mathcal{D}_{C^1}(\mathcal{K}).$$

Proof. The proof is very similar to the proof of theorem 2.1 in [11]. The only difference is the modification of step one in the proof. From the quantile representation of the density function, we get the following density step approximation:

$$p_{\mu,K}^{\mathrm{Q}}(x) = \frac{1}{K(Q_{\mu,k+1}^K - Q_{\mu,k}^K)}, \text{ for } x \in]Q_{\mu,k}^K, Q_{\mu,k+1}^K], \quad 0 \leq k \leq K$$

where $Q_{\mu,k}^K = Q_\mu^K(\frac{k}{K+1})$, for $k = 1, \ldots, K$, $Q_{\mu,0}^K = \underline{\mathcal{K}}$, $Q_{\mu,K+1}^K = \bar{\mathcal{K}}$.
For $\mu \in \mathcal{D}_{C^1}(\mathcal{K})$ with density p^μ, denote by $\hat{\mu}^K = \mathcal{L}_D(p_{\mu,K}^{\mathrm{Q}})$ the probability measure with density representation $p_{\mu,K}^{\mathrm{Q}}$.

Since μ, $\hat{\mu}^K$ are supported on the compact set $\mathcal{K}$, they lie in $\mathcal{P}_1(\mathbb{R}^d)$ the set of probability measures with finite first moment. From the Kantorovich-Rubinstein dual representation of the 1-Wasserstein distance, we have

$$\mathcal{W}_1(\mu, \hat{\mu}^K) \quad = \quad \sup_\phi \int_{\mathcal{K}} \phi(x)(\mathrm{p}^\mu(x) - p_{\mu,K}^{\mathrm{Q}}(x))\mathrm{d}x,$$

where the supremum is taken over all Lipschitz continuous functions ϕ on $\mathcal{K}$ with Lipschitz constant bounded by 1, and where we can assume w.l.o.g. that $\phi(x_0) = 0$ for some fixed point x_0 in $\mathcal{K}$.

Noting $\bar{\mathrm{p}}^{\mu}_k := \frac{1}{(Q^K_{\mu,k+1}-Q^K_{\mu,k})}\int_{Q^K_{\mu,k}}^{Q^K_{\mu,k+1}} \mathrm{p}^{\mu}(s)ds = \mathrm{p}^{\mu}(\tilde{x}_k)$ with $\tilde{x}_k \in [Q^K_{\mu,k}, Q^K_{\mu,k+1}]$ due to mean value theorem:

$$\begin{aligned}
\mathcal{W}_1(\mu,\hat{\mu}^K) \leq & \sup_{\phi} \sum_{k=1}^{K} \int_{Q^K_{\mu,k}}^{Q^K_{\mu,k+1}} |\phi(x)|\big|\big(\mathrm{p}^{\mu}(x) - \frac{1}{K(Q^K_{\mu,k+1}-Q^K_{\mu,k})}\big)\big|\mathrm{d}x \\
\leq & \operatorname{diam}(\mathcal{K}) \sum_{k=1}^{K} \int_{Q^K_{\mu,k}}^{Q^K_{\mu,k+1}} \big|\big(\mathrm{p}^{\mu}(x) - \frac{1}{K(Q^K_{\mu,k+1}-Q^K_{\mu,k})}\big)\big|\mathrm{d}x \\
= & \operatorname{diam}(\mathcal{K}) \sum_{k=1}^{K} \int_{Q^K_{\mu,k}}^{Q^K_{\mu,k+1}} |\mathrm{p}^{\mu}(x) - \bar{\mathrm{p}}^{\mu}_k| dx \\
= & \operatorname{diam}(\mathcal{K}) \sum_{k=1}^{K} (1_{\bar{\mathrm{p}}^{\mu}_k<\epsilon} + 1_{\bar{\mathrm{p}}^{\mu}_k\geq\epsilon}) \int_{Q^K_{\mu,k}}^{Q^K_{\mu,k+1}} |\mathrm{p}^{\mu}(x) - \bar{\mathrm{p}}^{\mu}_k| dx \qquad (3.1)
\end{aligned}$$

where we used that $|\phi(x)| \leq |x - x_0| \leq \operatorname{diam}(\mathcal{K})$.
Then :

$$\begin{aligned}
\sum_{k=1}^{K} 1_{\bar{\mathrm{p}}^{\mu}_k<\epsilon} \int_{Q^K_{\mu,k}}^{Q^K_{\mu,k+1}} |\mathrm{p}^{\mu}(x) - \bar{\mathrm{p}}^{\mu}_k| dx \leq & \sum_{k=1}^{K} 1_{\bar{\mathrm{p}}^{\mu}_k<\epsilon} \int_{Q^K_{\mu,k}}^{Q^K_{\mu,k+1}} (\mathrm{p}^{\mu}(x) + \bar{\mathrm{p}}^{\mu}_k) dx \\
= & \sum_{k=1}^{K} 1_{\bar{\mathrm{p}}^{\mu}_k<\epsilon} (Q^K_{\mu,k+1} - Q^K_{\mu,k}) 2\bar{\mathrm{p}}^{\mu}_k \leq 2\epsilon \qquad (3.2)
\end{aligned}$$

Notice that if $\mathrm{p}^{\mu}_k \geq \epsilon$, $Q^K_{\mu,k+1} - Q^K_{\mu,k} \leq \frac{1}{\epsilon K}$ and again due to mean value theorem and noting $C = \sup_y \mathrm{p}^{\mu'}(y)$:

$$\begin{aligned}
\sum_{k=1}^{K} 1_{\bar{\mathrm{p}}^{\mu}_k\geq\epsilon} \int_{Q^K_{\mu,k}}^{Q^K_{\mu,k+1}} |\mathrm{p}^{\mu}(x) - \bar{\mathrm{p}}^{\mu}_k| dx = & \sum_{k=1}^{K} 1_{\bar{\mathrm{p}}^{\mu}_k\geq\epsilon} \int_{Q^K_{\mu,k}}^{Q^K_{\mu,k+1}} |\mathrm{p}^{\mu}(x) - \mathrm{p}^{\mu}_k(\tilde{x}_k)| dx \\
\leq & C \sum_{k=1}^{K} 1_{\bar{\mathrm{p}}^{\mu}_k\geq\epsilon} \int_{Q^K_{\mu,k}}^{Q^K_{\mu,k+1}} |x - \tilde{x}_k| dx \\
\leq & C \sum_{k=1}^{K} 1_{\bar{\mathrm{p}}^{\mu}_k\geq\epsilon} (Q^K_{\mu,k+1} - Q^K_{\mu,k})^2 \\
\leq & C \sum_{k=1}^{K} (Q^K_{\mu,k+1} - Q^K_{\mu,k}) \frac{1}{\epsilon K} = C\frac{1}{\epsilon K} \qquad (3.3)
\end{aligned}$$

Pluging equations (3.2), (3.3) in (3.1) gives:

$$\mathcal{W}_1(\mu,\hat{\mu}^K) \leq \operatorname{diam}(\mathcal{K})(2\epsilon + C\frac{1}{\epsilon K})$$

For ϵ given, it is possible to have K high enough to get $\mathcal{W}_1(\mu,\hat{\mu}^K) \leq \operatorname{diam}(\mathcal{K})3\epsilon$ and noting that $\mathcal{W}_2(\mu,\hat{\mu}^K) \leq \sqrt{\operatorname{diam}(\mathcal{K})\mathcal{W}_1(\mu,\hat{\mu}^K)}$ by Hölder inequality, we get that $\mathcal{W}_2(\mu,\hat{\mu}^K) \leq \operatorname{diam}(\mathcal{K})\sqrt{3\epsilon}$.
Therefore we have shown that

$$\sup_{\mu\in\mathcal{D}_{C^1}(\mathcal{K})} \mathcal{W}_2(\mu,\hat{\mu}^K) \to 0, \quad \text{as } K \to \infty.$$

Then we use the same argument as in [11] to get that, for a given ϵ, it is possible to set K such that

$$|V(\mu) - V(\hat{\mu}^K)| \leq \frac{\varepsilon}{2}, \quad \forall\, \mu \in \mathcal{D}_{C^1}(\mathcal{K}).$$

At last using a classical universal theorem, we can conclude as in Step 2 of [11].

□

3.2 The moment network

Let $\mathcal{D}_2(\mathbb{R}^d)$ be the subset of probability measures μ in $\mathcal{P}_2(\mathbb{R}^d)$ that admit a density function p^μ with respect to the Lebesgue measure λ_d on $\mathbb{R}^d$. Fixing $\mathcal{K}$ as a bounded rectangle in $\mathbb{R}^d$, we want to approximate the function of the distribution with support in $\mathcal{K}$. By choosing $K > 0$, the main features of the distribution μ are approximated by choosing the lowest moments of the distribution so:

$$\boldsymbol{M}_\mu^K = (\mathbb{E}_{X\sim\mu}[\prod_{i=1,\dots,d} X_i^{k_i}])_{\sum_{i=1}^d k_i \le K}$$

with values in $\mathbb{R}^{\hat{K}}$, with $\hat{K} = \#\{p \in \mathbb{N}^d / \sum_{i=1}^d p_i \le K\}$.

Remark 3.2. *Since the support of the distribution is bounded all moments are well defined.*

A moment network is an operator on $\mathcal{D}_2(\mathbb{R}^d)$ in the form

$$\mathcal{N}_Q(\mu) = \Phi_\theta(\boldsymbol{M}_\mu^K),$$

so setting $R^{\hat{K}}(\mu) = \boldsymbol{M}_\mu^K$ in the equation (2.3).

Remark 3.3. *This approach is closely related to the moment problem which consists in determining a distribution from its moments if they exist. If the support of the distribution is $[0, \infty[$, this is the Stieltjes moment problem, and if the support is $\mathbb{R}$, this is the Hamburger moment problem. If μ is a positive measure with all moments defined, we say that μ is a solution to the moment problem. If the solution to the moment problem is unique, the moment problem is called determinate. Otherwise the moment problem is said to be indeterminate. In our case, where the support is compact, this problem is known as the Haussdorf moment problem and it is determinate. The connection with the moment problem and the reconstruction of an approximation of the quantile has been studied for example in [9].*

We now give a universal approximation theorem for this neural network:

Theorem 3.4. *Let $\mathcal{K}$ be a bounded rectangle in $\mathbb{R}^d$, and V be a continuous function from $\mathcal{P}_2(\mathcal{K})$ into $\mathbb{R}^p$, then, for all $\varepsilon > 0$, there exists K and Ψ a neural network from $\mathbb{R}^{\hat{K}}$ to $\mathbb{R}^p$ such that*

$$\left|V(\mu) - \Psi(\boldsymbol{M}_\mu^K)\right| \le \varepsilon \quad \forall \mu \in \mathcal{P}(\mathcal{K})$$

Proof. By the density of the cylindrical polynomial function with respect to distribution functions, see Lemma 3.12 in [7], for all $\varepsilon > 0$, there exists $K \in \mathbb{N}^*$, P a linear function from $\mathbb{R}^{\hat{K}}$ into $\mathbb{R}^p$, s.t.

$$\left|V(\mu) - P(\boldsymbol{M}_\mu^K)\right| \le \frac{\varepsilon}{2}, \quad \forall \mu \in \mathcal{P}(\mathcal{K}).$$

Note that since $\mathcal{K}$ is bounded, $\boldsymbol{M}_\mu^K$ is in a compact $\mathcal{Y}$ and we use the classical universal approximation theorem for finite-dimensional functions to obtain the existence of a feedforward neural network Ψ on $R^{\hat{K}}$ such that

$$\left|P(x) - \Psi(x)\right| \le \frac{\varepsilon}{2}, \quad \forall x \in \mathcal{Y}.$$

We conclude that for all $\mu \in \mathcal{P}(\mathcal{K})$,

$$\begin{aligned}&\left|V(\mu) - \Psi(\boldsymbol{M}_\mu^K)\right| \\ \le\ &\left|V(\mu) - P(\boldsymbol{M}_\mu^K)\right| + \left|P(\boldsymbol{M}_\mu^K) - \Psi(\boldsymbol{M}_\mu^K)\right| \ \le\ \varepsilon.\end{aligned}$$

□

4 Numerical tests

4.1 Univariate case

All functions are tested with the developed networks, and the results are compared with those obtained using the bin and cylinder methods in [11]. In dimension one, we propose to learn the following functions V with support in $[-2, 2]$.

A. The moment case

$$V(\mu) = \mathbb{E}_{X\sim\mu}[X]\mathbb{E}_{X\sim\mu}[X^4] - \mathbb{E}_{X\sim\mu}[X^2]$$

B. The pure quantile case

$$V(\mu) = Q_\mu(q)$$

and we take $q = 0.7$.

C. The quantile-moment case

$$V(\mu) = \mathbb{E}_{X\sim\mu}[X^3](1 + Q_\mu(q))$$

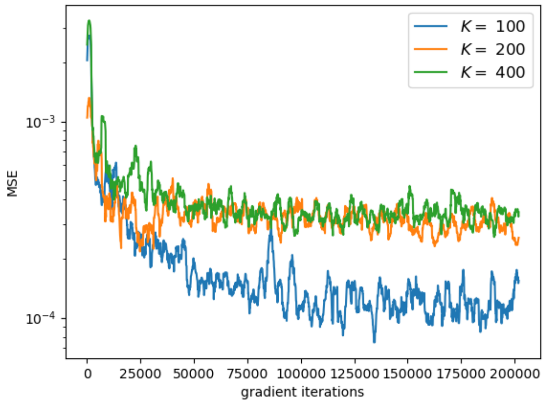
taking $q = 0.9$.

D. The quantile-superquantile case

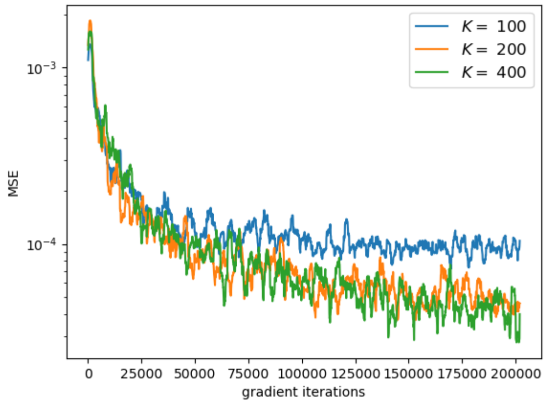
$$V(\mu) = \mathbb{E}_{X\sim\mu}[X/X > Q_\mu(q)] + Q_\mu(q)$$

and we take $q = 0.3$.

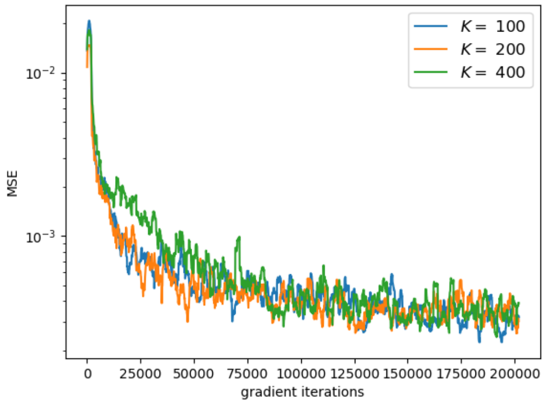
All distribution features are estimated with $N = 200000$ samples, and the distribution is sampled using the method in the section 2.1 using $J_1 = 400$ bins. During the training, $M = 20$ distributions (the batch size) are used. All curves plot the MSE obtained during gradient iterations as follows : every 100 iterations, the MSE is estimated using 1000 distributions and the results are plotted using a window averaging the estimates over 20 consecutive iterations. The ReLU activation function is used for all networks. Similar results are obtained using the tanh activation function. The quantile, and moment networks (and the networks derived from these features) use 2 hidden layers with 20 neurons. The cylinder network, which uses 2 networks, has 3 layers and 20 neurons for the "inner" network and 2 layers with 20 neurons for the "outer" network (see [11]).



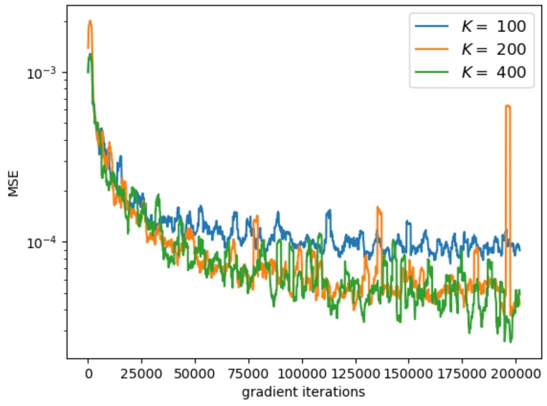
Case A



Case B



Case C



Case D

Figure 1: Quantile network convergence depending on the number of quantiles.

The neural network seems to be less accurate for functions involving moments (cases A and C) (see figure 1). A number of quantiles equal to 200 seems to be sufficient to obtain a good accuracy.

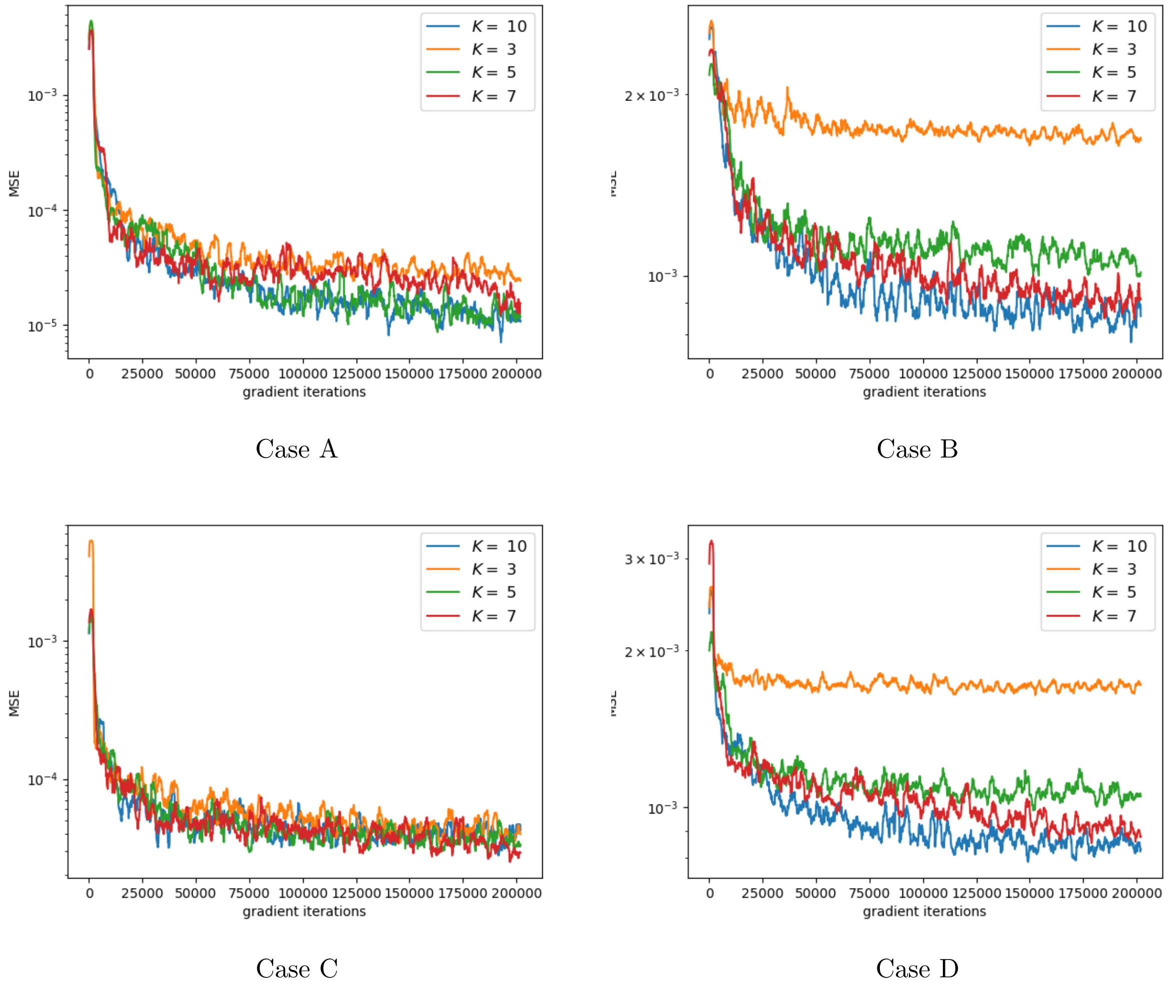


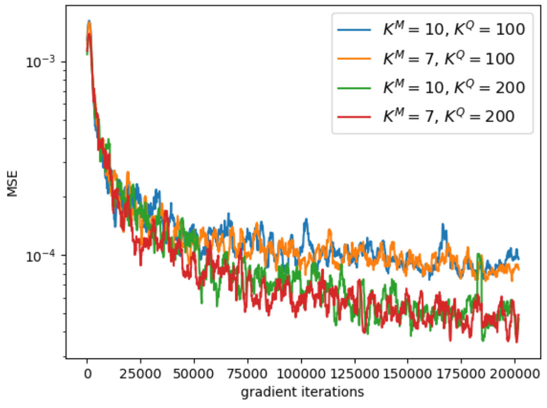
Figure 2: Moment network convergence depending on the number of moments.

In contrast to the quantile networks, the results are not surprisingly better when the functional to be approximated is mainly a function of the moments (see Figure 2). In case A, it is optimal to use a small number of moments, since the functional is only a function of moments with degrees less than 5.
Since the best network depends on the case, we can develop new networks based on moments and quantiles:

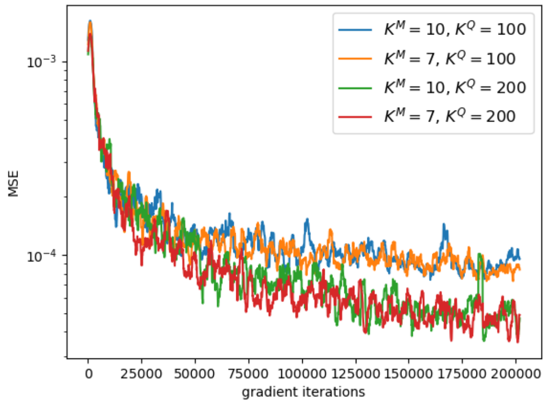
- A first one uses a concatenation of the features of the two proposed networks. Using the same notation as in the section 3,

$$\mathcal{N}_{QM}(\mu) = \Phi_\theta(\boldsymbol{M}_\mu^{K^M}, \mathrm{Q}_\mu^{K^Q}),$$

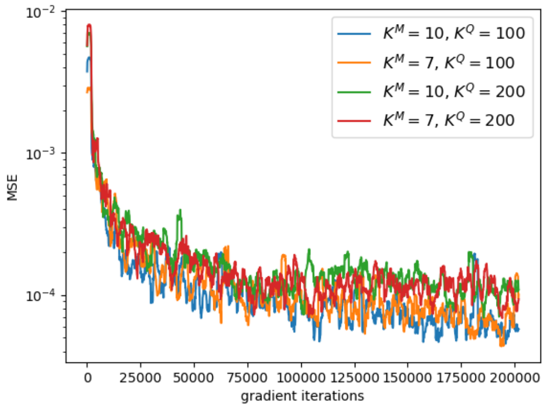
  where now K^M is the number of moments retained in the approximation and K^Q is the number of quantiles. This neural network is the **moment and quantile network**. The results obtained for this network are shown in Figure 3. Overall, it seems that a moment number of $K^M = 7$ and a quantile number of $K^Q = 200$ is a good choice.



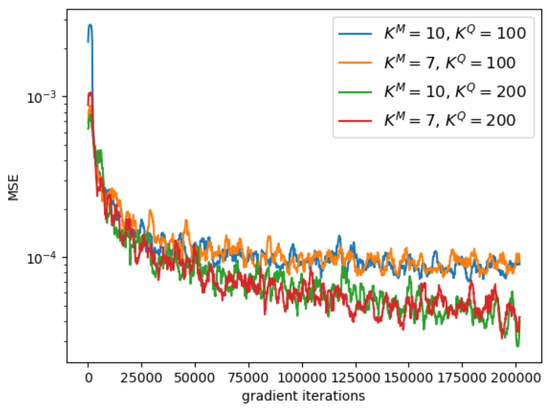
Case A



Case B



Case C



Case D

Figure 3: Moment and quantile network convergence depending on K^M and K^Q.

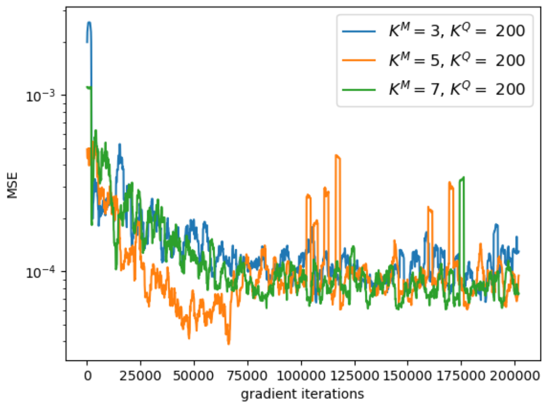
- In a second one, instead of concatenating some moments expectations and quantiles, we can take some quantiles of the moments by defining:

$$\boldsymbol{L}_\mu^{K_M,K_Q} = [Q_{\prod_{i=1,\dots,d} X_i^{\tilde{k}_i}/X\sim\mu}(\frac{\hat{k}}{K_Q+1})]_{\sum_{i=1}^d \tilde{k}_i \le K_M, 1\le\hat{k}\le K_Q}$$

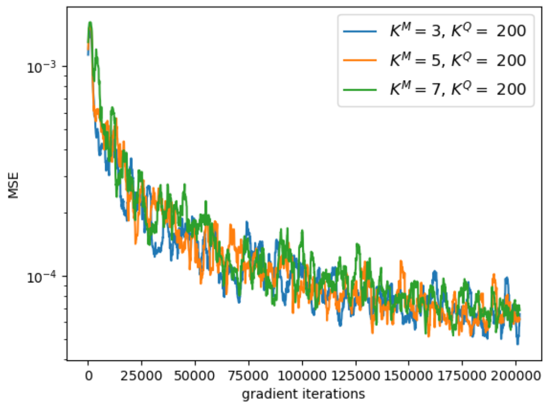
A **quantile of moments network** is an operator on $\mathcal{D}_2(\mathbb{R}^d)$ in the form

$$\mathcal{N}_Q(\mu) = \Phi_\theta(\boldsymbol{L}_\mu^{K_M,K_Q}),$$

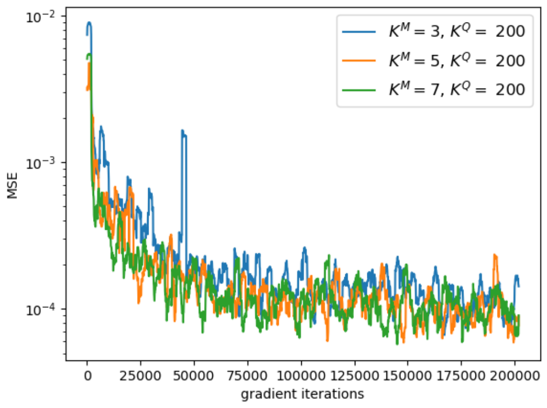
thus setting $R^K(\mu) = \boldsymbol{L}_\mu^{K_M,K_Q}$ in the equation (2.3). The results for this network are shown in the figure 4. The convergence seems to be good in all cases, but we observe that this convergence is less regular than with the previous neural network.



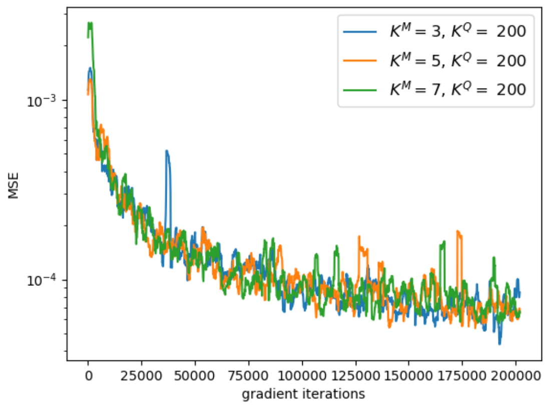
Case A



Case B



Case C



Case D

Figure 4: Quantile of moments network convergence.

Remark 4.1. *We have also tested networks based on superquantiles or superquantiles of moments.*

- *Defining for one-dimensional distributions*

$$\boldsymbol{V}_\mu^K = \left[E_{X\sim\mu}[X \geq Q_\mu(\frac{\hat{k}}{K+1})]\right]_{0\leq\hat{k}\leq K},$$

a superquantile network is an operator on $\mathcal{D}_2(\mathbb{R}^d)$ in the form

$$\mathcal{N}_Q(\mu) = \Phi_\theta(\boldsymbol{V}_\mu^K),$$

- *and defining for potentially multivariate distributions:*

$$\boldsymbol{S}_\mu^{K_M,K_Q} = \left[E_{X\sim\mu}[\prod_{i=1,\dots,d} X_i^{\tilde{k}_i} / \prod_{i=1,\dots,d} X_i^{\tilde{k}_i} \geq Q_{\prod_{i=1,\dots,d} X_i^{\tilde{k}_i}/X\sim\mu}(\frac{\hat{k}}{K_Q+1})]\right]_{\sum_{i=1}^d \tilde{k}_i \leq K_M, 0\leq\hat{k}\leq K_Q},$$

a superquantile of moment network is an operator on $\mathcal{D}_2(\mathbb{R}^d)$ in the form

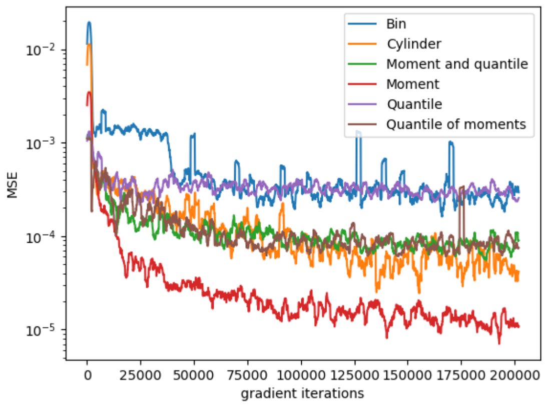
$$\mathcal{N}_Q(\mu) = \Phi_\theta(\boldsymbol{S}_\mu^{K_M,K_Q}).$$

Both neural networks give good results but never better than the cylinder network. We don't report them.

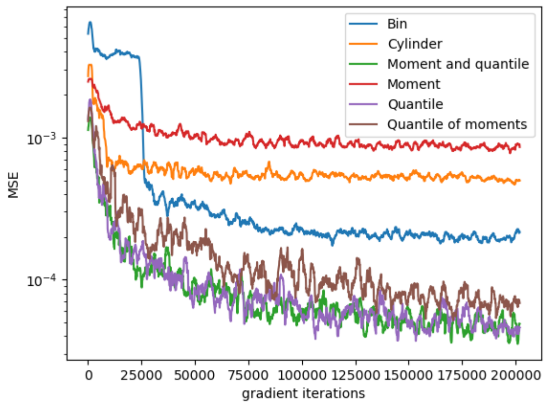
We now compare all the networks together on the figure 5 using

- $K = 200$ quantiles for the quantile network,
- $K = 10$ moments for the moment network,
- $K^M = 7$ moments and $K^Q = 200$ quantiles for the "quantile of moments" and "moment and quantile" networks,
- 200 bins for the bin network.

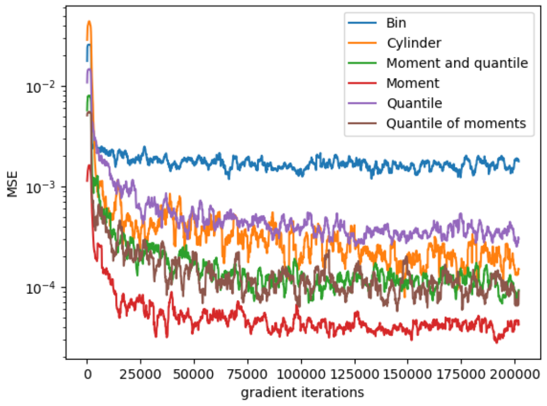
Overall, the "moment and quantile" network and the "quantile of moments" network seem to be the best choices.



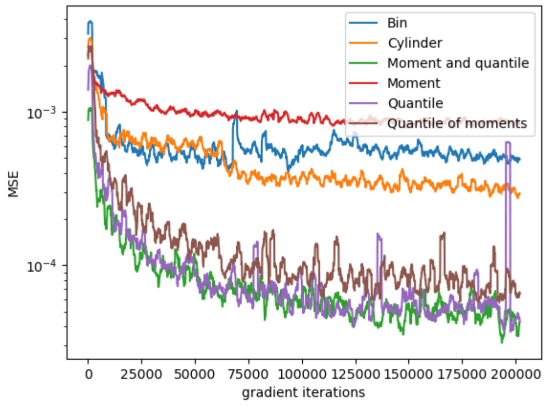
Case A



Case B



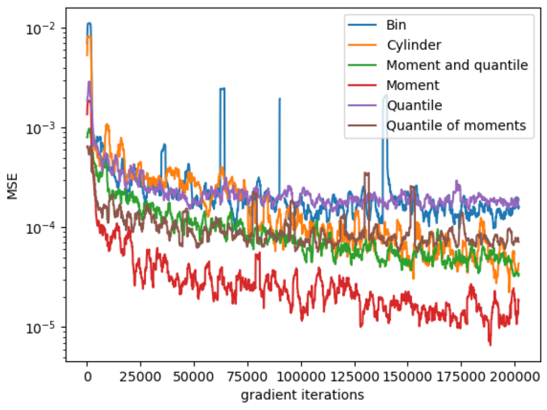
Case C



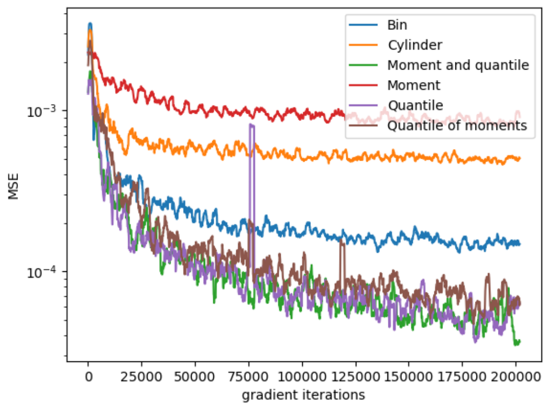
Case D

Figure 5: Convergence of the different networks in 1D.

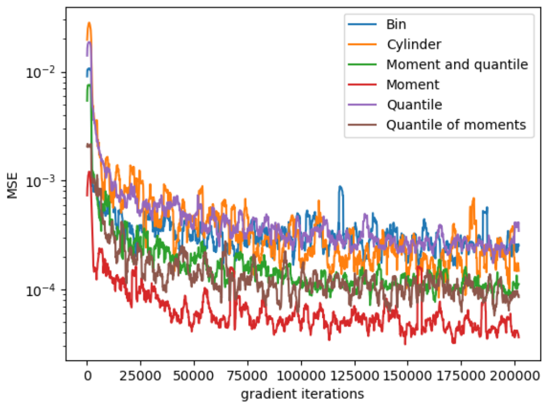
Finally, we compare the different networks, taking for all networks 3 layers and 40 neurons.



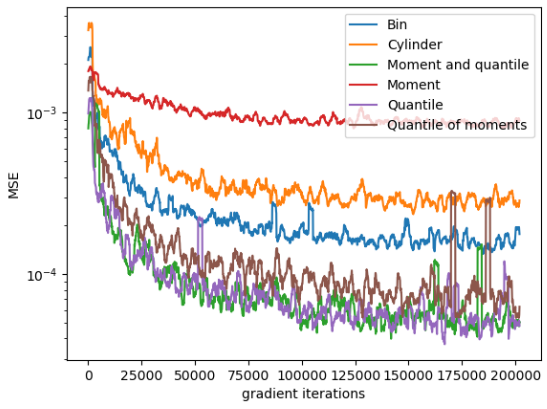
Case A



Case B



Case C



Case D

Figure 6: Convergence of the different networks in 1D taking 3 layers and 40 neurons.

The results for the bin network are improved, but the conclusions remain the same.

4.2 Bivariate case

We assume that the support is in $[-2,2]^2$. For a distribution $\mu \in \mathcal{P}_2(\mathbb{R}^d)$, $(j,m) \in \mathbb{N}^* \times \mathbb{N}^*$, we note $\hat{F}_{\mu,j,m}$ the cumulative distribution function of $X_1^j X_2^m$ where $X \sim \mu$ and $\hat{Q}_{\mu,j,m}(p) = \inf\{x \in \mathbb{R} : p \leq \hat{F}_{\mu,j,m}(x)\}$. We define the following test cases:

A. The moment case

$$V(\mu) = \sum_{i=1}^{2} \left[\mathbb{E}_{X\sim\mu_i}[X]\mathbb{E}_{X\sim\mu_i}[X^4] - \mathbb{E}_{X\sim\mu_i}[X^2]\right].$$

B. The quantile-superquantile case

$$V(\mu) = \sum_{i=1}^{2} [\mathbb{E}_{X\sim\mu_i}[X/X > Q_{\mu_i}(q)] + Q_{\mu_i}(q)] + \\ \mathbb{E}_{X\sim\mu}[X_1X_2/X_1X_2 > \hat{Q}_{\mu,1,1}(q)]$$

with $q = 0.7$.

C. The quantile moment case

$$V(\mu) = (1 + Q_{\mu_1}(q))\mathbb{E}_{X\sim\mu_1}[X^3] + \mathbb{E}_{X\sim\mu_2}[X^3] + \hat{Q}_{\mu,2,1}(q) + \hat{Q}_{\mu,1,2}(q)$$

with $q = 0.9$.

D. The quantile-superquantile marginal case

$$V(\mu) = \sum_{i=1}^{2} [\mathbb{E}_{X\sim\mu_i}[X/X > Q_{\mu_i}(q_i)] + Q_{\mu_i}(q_i)]$$

with $q = (0.6, 0.3)$.

E. The quantile-cross-superquantile case

$$V(\mu) = \mathbb{E}_{X\sim\mu}[X_2/X_2 > Q_{\mu_1}(q)] + Q_{\mu_1}(q)$$

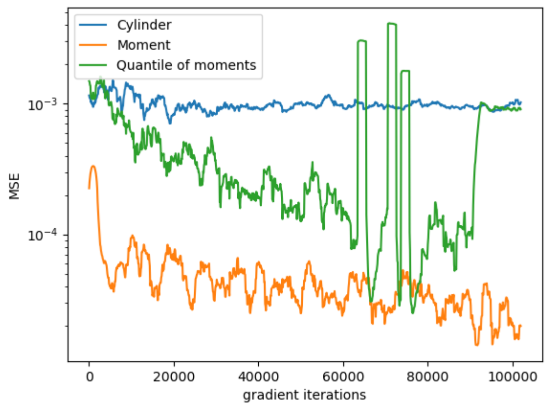
with $q = 0.2$.

F. The quantile marginal case

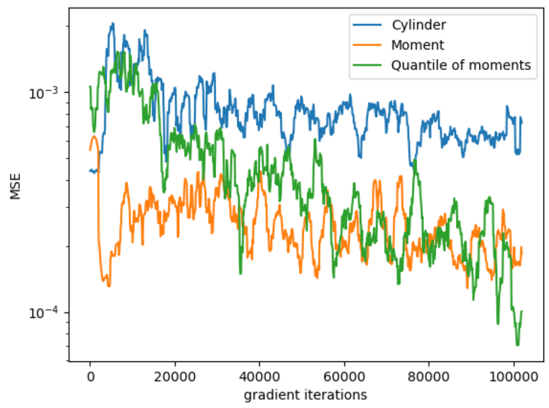
$$V(\mu) = Q_{\mu_1}(q) + Q_{\mu_2}(q)$$

with $q = 0.8$.

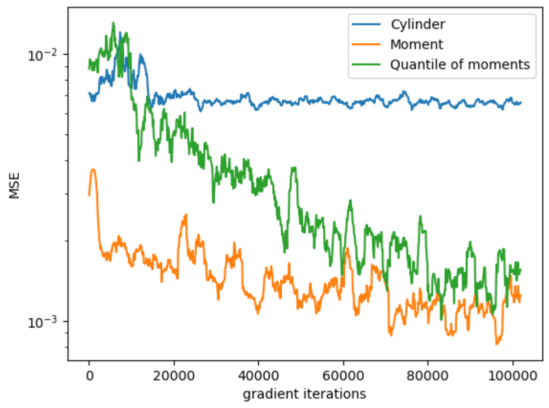
We test the bin network, the cylinder network, the moment network, and the quantile of moments network on the different cases. The bin network fails in all the test cases with a number of layers equal to 2 or 3 and a number of neurons taken equal to 20, 40 and 80. As for the other networks, we keep the same number of layers and neurons as in the previous section. For the moment network we use $K = 7$, while for the quantile of moment network we use $K^M = 5$ and $K^Q = 200$. The distribution features are estimated using $N = 400000$ samples, and we take $(J_1, J_2) = (200, 200)$ to sample a given distribution.



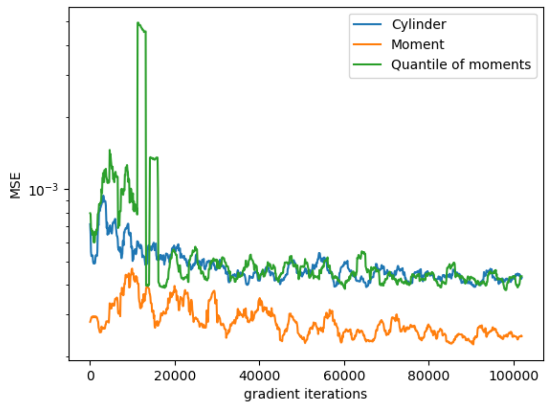
Case A



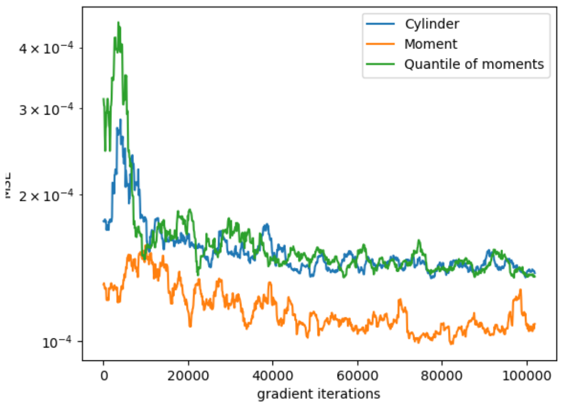
Case B



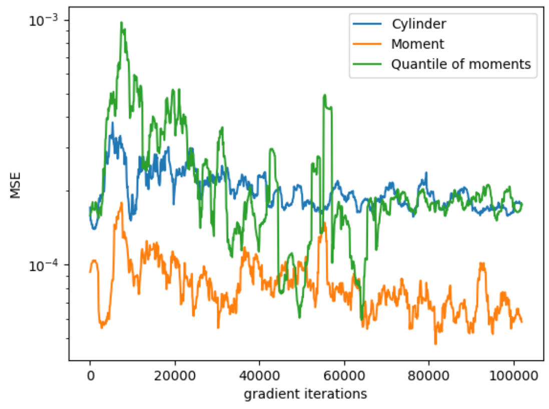
Case C



Case D



Case E



Case F

Figure 7: Convergence of the different networks in 2D.

The tests are shown in figure 7. In all cases, the moment network gives the best results. We observe a loss not as good as in dimension one for the case C.

5 Conclusion

New networks have been developed to learn functions of distributions: some of them outperform the existing ones. In all cases, the best networks are based on some moments of the distribution. For univariate distributions, it is optimal to add some information, for example based on quantiles, to get an effective scheme on all the test cases. For bivariate distributions, it is sufficient to take the expectation of the moments to get the best scheme. Using this moment scheme, the resolution of the PDE in Wasserstein space becomes possible in the multivariate case.

References

[1] Martín Abadi et al. "Tensorflow: a system for large-scale machine learning." In: *Osdi.* Vol. 16. 2016. Savannah, GA, USA. 2016, pp. 265–283.

[2] Christian Beck, Martin Hutzenthaler, Arnulf Jentzen, and Benno Kuckuck. "An overview on deep learning-based approximation methods for partial differential equations". In: *arXiv preprint arXiv:2012.12348* (2020).

[3] R. Carmona and F. Delarue. *Probabilistic Theory of Mean Field Games: vol. I, Mean Field FBSDEs, Control, and Games,* Springer, 2018.

[4] R. Carmona and F. Delarue. *Probabilistic Theory of Mean Field Games: vol. II, Mean Field FBSDEs, Control, and Games,* Springer, 2018.

[5] M. Germain, M. Laurière, H. Pham, and X. Warin. "DeepSets and derivative networks for solving symmetric PDEs". In: *Journal of Scientific Computing* 91 (2022).

[6] M. Germain, H. Pham, and X. Warin. "Neural networks based algorithms for stochastic control and PDEs in finance". In: *arXiv:2101.08068 to appear in Machine Learning And Data Sciences For Financial Markets: A Guide To Contemporary Practices.* Ed. by A. Capponi and C.A. Lehalle. Cambridge University Press, 2022.

[7] X. Guo, H. Pham, and X. Wei. "Itô's formula for flows of semimartingales". In: *arXiv: 2010.05288* (2022).

[8] Diederik P Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[9] Robert M Mnatsakanov and Artak S Hakobyan. "Recovery of distributions via moments". In: *Lecture Notes-Monograph Series* (2009), pp. 252–265.

[10] Huyên Pham and Xavier Warin. "Mean-field neural networks-based algorithms for McKean-Vlasov control problems". In: *arXiv preprint arXiv:2212.11518* (2022).

[11] Huyên Pham and Xavier Warin. "Mean-field neural networks: learning mappings on Wasserstein space". In: *arXiv preprint arXiv:2210.15179* (2022).