Numerical resolution of McKean-Vlasov Forward Backward Stochastic Differential Equations using neural networks

Maximilien GERMAIN
Joint work with Joseph MIKAEL, Xavier WARIN

EDF R&D, LPSM, Université Paris Diderot

8th January 2020
McKean-Vlasov FBSDEs

We solve general McKean-Vlasov FBSDEs of the form

\[
\begin{align*}
X_t &= \xi + \int_0^t b(s, X_s, Y_s, Z_s, \mathcal{L}(X_s), \mathcal{L}(Y_s), \mathcal{L}(Z_s)) \, ds + \int_0^t \sigma(s, X_s) \, dW_s \\
Y_t &= g(X_T, \mathcal{L}(X_T)) + \int_t^T f(s, X_s, Y_s, Z_s, \mathcal{L}(X_s), \mathcal{L}(Y_s), \mathcal{L}(Z_s)) \, ds - \int_t^T Z_s \, dW_s
\end{align*}
\]

with machine learning techniques. $W_s$ is a standard Brownian motion.

The processes dynamics depend on their laws $\mathcal{L}(\cdot)$. In practise we restrict to cases when the law dependence only concerns probability moments.
Main motivation: Mean Field Games (MFG)

Stochastic games introduced by (Lasry and Lions 2006), (Huang, Malhame, and Caines 2006) followed by (Carmona and Delarue 2012) dealing with a large number of interacting players. The empirical law of players states influences both the dynamics and cost.

Applications

- Population dynamics (crowd, traffic jam, bird flocking...).
- Market interactions (Cardaliaguet and Lehalle 2019).
- Energy storage (Matoussi, Alasseur, and Ben Taher 2018), electric cars management.
- Social networks.
$N$ players stochastic game

Player $i$ minimises a cost $J^i(\alpha^1, \cdots, \alpha^i, \cdots, \alpha^N)$ depending on the empirical distribution $\mu_t = \frac{1}{N} \sum_{k=1}^{N} \delta_{X^k_t} \rightarrow$ mean field interaction.

$$\min_{\alpha^i \in \mathbb{A}} \mathbb{E} \left[ \int_0^T f(t, X^i_t, \mu_t, \alpha^i_t) \, dt + g(X^i_T, \mu_T) \right]$$

subject to

$$dX^i_t = b(t, X^i_t, \mu_t, \alpha^i_t) \, dt + \sigma(t, X^i_t, \mu_t) \, dW^i_t.$$ 

Difficult problem in general.
Asymptotic control problem when $N \to +\infty$

Given a family $(\mu_t)_{t \in [0,T]}$ of probability measures we solve for a representative player

$$
\min_{\alpha} \mathbb{E} \left[ \int_0^T f(t, X_t^\mu, \mu_t, \alpha_t) \, dt + g(X_T^\mu, \mu) \right]
$$

subject to

$$
dX_t^\mu = b(t, X_t^\mu, \mu_t, \alpha_t) \, dt + \sigma(t, X_t^\mu, \mu_t) \, dW_t.
$$

Fixed point on probability measures

$$
\mu_t = \mathcal{L}(X_t^\mu).
$$

**Simpler** than the $N$ players game $\to$ unique player in the limit!

Value function given by

$$
v^\mu(t, x) = \inf_{\alpha \in \mathcal{A}_t} \mathbb{E} \left[ \int_t^T f(s, X_s, \mu_s, \alpha_s) \, ds + g(X_T, \mu) | X_t = x \right].
$$
Optimality conditions

HJB equation

\[
\begin{align*}
\partial_t v + \frac{1}{2} \text{Tr}(\sigma \sigma^\top \partial_{xx} v) + \min_{\alpha \in \mathcal{A}} H^{\nu_t}(t, x, \partial_x v, \alpha) &= 0 \\
v(T, x) &= g(x, \nu_T)
\end{align*}
\]

coupled with Fokker-Planck equation:

\[
\begin{align*}
\partial_t \nu - \frac{1}{2} \text{Tr}(\sigma \sigma^\top \partial_{xx} \nu) - \text{div}(b(t, x, \nu_t, \hat{\alpha}^{\nu_t}(t, x, \partial_x v))\nu) &= 0 \\
\nu(0, \cdot) &= \nu_0.
\end{align*}
\]

Solved in (Achdou and Capuzzo-Dolcetta 2010) with finite differences methods.
\[ Y_t = \partial_x v(t, X_t) \rightarrow \text{stochastic Pontryagin principle} \]

\[
\begin{align*}
\text{d}X_t &= b(t, X_t, \mathcal{L}(X_t), \hat{\alpha}^{\mathcal{L}(X_t)}(t, X_t, Y_t)) \text{ d}t + \sigma \text{ d}W_t \\
X_0 &= \xi \\
\text{d}Y_t &= -\partial_x H^{\mathcal{L}(X_t)}(t, X_t, Y_t, \hat{\alpha}^{\mathcal{L}(X_t)}(t, X_t, Y_t)) \text{ d}t + Z_t \text{ d}W_t \\
Y_T &= \partial_x g(X_T, \mu_T).
\end{align*}
\]

In the form

\[
\begin{align*}
X_t &= \xi + \int_0^t b(s, X_s, Y_s, Z_s, \mathcal{L}(X_s), \mathcal{L}(Y_s), \mathcal{L}(Z_s)) \text{ d}s + \int_0^t \sigma(s, X_s) \text{ d}W_s \\
Y_t &= g(X_T, \mathcal{L}(X_T)) + \int_t^T f(s, X_s, Y_s, Z_s, \mathcal{L}(X_s), \mathcal{L}(Y_s), \mathcal{L}(Z_s)) \text{ d}s - \int_t^T Z_s \text{ d}W_s
\end{align*}
\]
Advantages of machine learning techniques

Curse of dimensionnality with finite differences methods if the state dimension is high ($\geq 3$ or $4$).

→ Some machine learning schemes solve nonlinear PDE in high dimension (10, 50 or even 100): - Deep BSDE method of (Han, Jentzen, and E 2017)  
- Deep Galerkin method of (Sirignano and Spiliopoulos 2017)  
- Deep Backward Dynamic Programming of (Huré, Pham, and Warin 2020).

→ Open source libraries such as Tensorflow or Pytorch.

→ Efficient computation on GPU nodes.
A global method

Based on the Deep BSDE method of (Han, Jentzen, and E 2017)

\[
\begin{align*}
X_t &= \xi + \int_0^t b(s, X_s, Y_s, Z_s, \mathcal{L}(X_s), \mathcal{L}(Y_s), \mathcal{L}(Z_s)) \, ds + \int_0^t \sigma(s, X_s) \, dW_s \\
Y_t &= g(X_T, \mathcal{L}(X_T)) + \int_t^T f(s, X_s, Y_s, Z_s, \mathcal{L}(X_s), \mathcal{L}(Y_s), \mathcal{L}(Z_s)) \, ds - \int_t^T Z_s \, dW_s
\end{align*}
\]

Discretization:

\[
\begin{align*}
X_{t_i+1} &= X_{t_i} + b \left( t_i, X_{t_i}, Y_{t_i}, Z_{t_i}, \mu_{t_i} \right) \, \Delta t_i + \sigma \left( t_i, X_{t_i} \right) \, \Delta W_{t_i} \\
Y_{t_i+1} &= Y_{t_i} - f \left( t_i, X_{t_i}, Y_{t_i}, Z_{t_i}, \mu_{t_i} \right) \, \Delta t_i + Z^\theta \left( t_i, X_{t_i} \right) \, \Delta W_{t_i}.
\end{align*}
\]

\(Y_0\) is a variable and \(Z_{t_i}\) is approached by a neural network \(Z^\theta(t_i, \cdot)\) which minimizes the loss function \(\mathbb{E} \left[ (Y_T - g(X_T, \mu_T))^2 \right]\). The forward backward system is transformed into a \textbf{forward form} and an \textbf{optimization} problem. Also studied in (Fouque and Zhang 2019), (Carmona and Laurière 2019) in dimension 1 and with less generality.
Fixed point on probability measures

Estimation of state and value function moments:

- **Direct**: use of current particles empirical law

\[
\mathbb{E}[X_{t,i}]^{(k+1)} \simeq \mu_{t_i}^{(k+1)} = \frac{1}{N_b} \sum_{j=1}^{N_b} X_{t_i}^j. \quad (1)
\]

- **Dynamic**: keep in memory previously computed moments and average them with current particle moments

\[
\mathbb{E}[X_{t,i}]^{(k+1)} \simeq \frac{N_m N_b \mu_{t_i}^{(k)} + \sum_{j=1}^{N_b} X_{t_i}^j}{N_b + N_m N_b}. \quad (2)
\]

- **Expectation**: estimate \(\mathbb{E}[X_t]\) by an additional neural network.

\[
\mathbb{E}[X_t] \simeq \Psi_\kappa(t). \quad (3)
\]

We add to the loss a term \(\mathbb{E} \left[ \lambda \sum_{i=0}^{N_t} \left( \Psi_\kappa(t_i) - X_{t_i} \right)^2 \right].\)
A local approach

Inspired by (Huré, Pham, and Warin 2020).

$Z$ and $Y$ are approximated by neural networks $(Z^i_{\theta_m^i}(\cdot), Y^i_{\theta_m^i}(\cdot)) \{i \in [0,N-1]\}$. At iteration $m$ we simulate $X_i$ with the previously computed parameters $\theta_m^i$. The $X_i$’s dynamics being frozen with parameters $\theta_m^i$:

- First $Y^N$ is set to the terminal condition $g(X_N, \mu_N)$.
- Then, we solve successively the local backward problems for $i$ from $N - 1$ to 0

$$\min_{\theta_m^i} \mathbb{E} \left[ \left( Y^{i+1}_{\theta_m^i+1} (X_{i+1}) - Y^i_{\theta_m^i} (X_i) + f(t_i, X_i, Y^i_{\theta_m^i} (X_i), Z^i_{\theta_m^i} (X_i), \mu_{t_i}) \Delta t - Z^i_{\theta_m^i} (X_i) \Delta W_{t_i} \right)^2 \right].$$
A linear quadratic test case: price impact

Large number of traders who want to sell a portfolio at the same time. Example from (Carmona and Delarue 2018).

\[
\min_{\alpha \in \mathcal{A}} \mathbb{E} \left[ \int_0^T \left( \frac{c_\alpha}{2} \|\alpha_t\|^2 + \frac{c_X}{2} \|X_t\|^2 - \gamma X_t \cdot \mu_t \right) \, dt + \frac{c_g}{2} \|X_T\|^2 \right].
\]

subject to \( X_t = x_0 + \int_0^t \alpha_s \, ds + \sigma \, W_t \)

and the fixed point \( \mathbb{E}[\alpha_t] = \mu_t \). Optimality system:

\[
\begin{cases}
    dX_t &= -\frac{1}{c_\alpha} Y_t \, dt + \sigma \, dW_t \\
    X_0 &= x_0 \\
    dY_t &= -(c_X X_t + \frac{\gamma}{c_\alpha} \mathbb{E}[Y_t]) \, dt + Z_t \, dW_t \\
    Y_T &= c_g X_T.
\end{cases}
\]

We take \( c_X = 2, x_0 = 1, \sigma = 0.7, \gamma = 2, c_\alpha = 2/3, c_g = 0.3 \). The simulations are conducted with \( d = 10, \Delta t = 0.01 \).
Results 1/4: Local method/Price Impact model

**Figure:** Computed optimal control after 6000 iterations and expectation of the state at terminal state for the local method.
Figure: Computed optimal control after 2000 iterations and expectation of the state at terminal state for the global method.
A non linear quadratic test case

Take $X_t$ log normal, $Y_t = \exp(\alpha t) \log(\prod_i X_t^i)$ and quadratic dynamics.

\[
\begin{aligned}
\frac{dX_t^i}{dt} &= (a^i X_t^i + b(Y_t + Z_t^i + \mathbb{E}[X_t^i] + \mathbb{E}[Y_t] + \mathbb{E}[Z_t^i])) \\
&\quad - b(e^{\alpha t} \log(\prod_{i=1}^d X_t^i) + \sigma_t^i e^{\alpha t} + g_t^i + c_t + e_t^i) \\
&\quad + c\left(Y_t^2 + (Z_t^i)^2 + \mathbb{E}[(X_t^i)^2] + \mathbb{E}[Y_t^2] + \mathbb{E}[(Z_t^i)^2]\right) \\
X_0^i &= \xi^i \\
\frac{dY_t}{dt} &= \left(\phi(t, X_t) + b(Y_t + \frac{1}{d} \sum_{i=1}^d Z_t^i + \frac{1}{d} \sum_{i=1}^d \mathbb{E}[X_t^i] + \mathbb{E}[Y_t] + \frac{1}{d} \sum_{i=1}^d \mathbb{E}[Z_t^i])ight) \\
&\quad - b\left(e^{\alpha t} \log(\prod_{i=1}^d X_t^i) + \frac{1}{d} \sum_{i=1}^d \sigma_t^i e^{\alpha t} + \frac{1}{d} \sum_{i=1}^d g_t^i + c_t + \frac{1}{d} \sum_{i=1}^d e_t^i\right) \\
&\quad + c\left(Y_t^2 + \frac{1}{d} \sum_{i=1}^d (Z_t^i)^2 + \frac{1}{d} \sum_{i=1}^d \mathbb{E}[(X_t^i)^2] + \mathbb{E}[Y_t^2] + \frac{1}{d} \sum_{i=1}^d \mathbb{E}[(Z_t^i)^2]\right) \\
&\quad - c\left(e^{2\alpha t} \log(\prod_{i=1}^d X_t^i)^2 + c_t^2 + \frac{1}{d} \sum_{i=1}^d (\sigma_t^i)^2 e^{2\alpha t} + (g_t^i)^2 + (e_t^i)^2\right)\right) dt + Z_t\, dW_t \\
Y_T &= e^{\alpha T} \log(\prod_{i=1}^d X_T^i).
\end{aligned}
\]

We take $a = b = c = 0.1$, $\alpha = 0.5$, $\sigma = 0.4$, $\xi = 1$. The simulations are conducted with $d = 10$, $\Delta t = 0.01$. 
Figure: $Y$ and $Z$ for the local method after 6000 iterations
Results 4/4: Global method/Non linear quadratic equation

Figure: $Y$ and $Z$ for the global method after 2000 iterations


