Mean Field Games: Numerical Methods and Applications in Machine Learning

Part 4: Methods Based on the Probabilistic Approach

Mathieu Laurière

https://mlauriere.github.io/teaching/MFG-PKU-4.pdf

Peking University
Summer School on Applied Mathematics
July 26 – August 6, 2021

RECAP

Outline

1. A Picard Scheme for MKV FBSDE
   - Picard Scheme & Continuation Method
   - Tree-Based Algorithm
   - Grid-Based Algorithm

2. Stochastic Methods for some Finite-Dimensional MFC Problems

Outline

MKV FBSDE System

- Recall: generic form:

$$\begin{cases} dX_t = B(X_t, \mathcal{L}(X_t), Y_t, Z_t)dt + \sigma dW_t, & 0 \le t \le T \\ dY_t = -F(X_t, \mathcal{L}(X_t), Y_t, Z_t)dt + Z_t dW_t, & 0 \le t \le T \\ X_0 \sim m_0, \qquad Y_T = G(X_T, \mathcal{L}(X_T)) \end{cases}$$

- Decouple:
  - Given $(\mathcal{L}(X), Y, Z)$, solve for X
  - Given $(X, \mathcal{L}(X))$ solve for (Y, Z)
- Iterate
- Algorithm proposed by [Chassagneux et al.'19][1], [Angiuli et al.'19][2]

[1] Chassagneux, J.-F., Crisan, D., & Delarue, F.. Numerical method for FBSDEs of McKean–Vlasov type. *The Annals of Applied Probability* 29.3 (2019): 1640-1684.

[2] Angiuli, A., et al. Cemracs 2017: numerical probabilistic approach to MFG. *ESAIM: Proceedings and Surveys* 65 (2019): 84-113.

Picard Scheme for MKV FBSDE System

Input: Initial guess (ξ, ζ); initial condition ξ; terminal condition ζ; time horizon T; number of iterations $\mathtt{K}$

Output: Approximation of (X, Y, Z) solving the MKV FBSDE system

1 Initialize $X_t^{(0)} = \xi, Y_t^{(0)} = 0, Z_t^{(0)} = 0, 0 \leq t \leq T$

2 **for** $\mathtt{k} = 0, 1, 2, \dots, \mathtt{K} - 1$ **do**

3 Let $X^{(\mathtt{k}+1)}$ be the solution to:

$$\begin{cases} dX_t = B(X_t^{(\mathtt{k})}, \mathcal{L}(X_t^{(\mathtt{k})}), Y_t^{(\mathtt{k})}, Z_t^{(\mathtt{k})})dt + \sigma dW_t, & 0 \leq t \leq T \\ X_0 = \xi \end{cases}$$

Picard Scheme for MKV FBSDE System

Input: Initial guess (ξ, ζ); initial condition ξ; terminal condition ζ; time horizon T; number of iterations $\mathtt{K}$

Output: Approximation of (X, Y, Z) solving the MKV FBSDE system

Initialize $X_t^{(0)} = \xi, Y_t^{(0)} = 0, Z_t^{(0)} = 0, 0 \le t \le T$

for $\mathtt{k} = 0, 1, 2, \ldots, \mathtt{K} - 1$ **do**

Let $X^{(\mathtt{k}+1)}$ be the solution to:

$$\begin{cases} dX_t = B(X_t^{(\mathtt{k})}, \mathcal{L}(X_t^{(\mathtt{k})}), Y_t^{(\mathtt{k})}, Z_t^{(\mathtt{k})})dt + \sigma dW_t, & 0 \le t \le T \\ X_0 = \xi \end{cases}$$

Let $(Y^{(\mathtt{k}+1)}, Z^{(\mathtt{k}+1)})$ be the solution to:

$$\begin{cases} dY_t = -F(X_t^{(\mathtt{k}+1)}, \mathcal{L}(X_t^{(\mathtt{k}+1)}), Y_t^{(\mathtt{k})}, Z_t^{(\mathtt{k})})dt + Z_t^{(\mathtt{k})} dW_t, & 0 \le t \le T \\ Y_T = \zeta \end{cases}$$

return $\mathtt{Picard}[T](\xi, \zeta) = (X^{(\mathtt{K})}, Y^{(\mathtt{K})}, Z^{(\mathtt{K})})$

Picard Scheme for MKV FBSDE System

Input: Initial guess (ξ, ζ); initial condition ξ; terminal condition ζ; time horizon T; number of iterations $\mathtt{K}$

Output: Approximation of (X, Y, Z) solving the MKV FBSDE system

Initialize $X_t^{(0)} = \xi, Y_t^{(0)} = 0, Z_t^{(0)} = 0, 0 \leq t \leq T$

for $\mathtt{k} = 0, 1, 2, \ldots, \mathtt{K} - 1$ **do**

Let $X^{(\mathtt{k}+1)}$ be the solution to:

$$\begin{cases} dX_t = B(X_t^{(\mathtt{k})}, \mathcal{L}(X_t^{(\mathtt{k})}), Y_t^{(\mathtt{k})}, Z_t^{(\mathtt{k})})dt + \sigma dW_t, & 0 \leq t \leq T \\ X_0 = \xi \end{cases}$$

Let $(Y^{(\mathtt{k}+1)}, Z^{(\mathtt{k}+1)})$ be the solution to:

$$\begin{cases} dY_t = -F(X_t^{(\mathtt{k}+1)}, \mathcal{L}(X_t^{(\mathtt{k}+1)}), Y_t^{(\mathtt{k})}, Z_t^{(\mathtt{k})})dt + Z_t^{(\mathtt{k})} dW_t, & 0 \leq t \leq T \\ Y_T = \zeta \end{cases}$$

return $\mathtt{Picard}[T](\xi, \zeta) = (X^{(\mathtt{K})}, Y^{(\mathtt{K})}, Z^{(\mathtt{K})})$

Notation: $\Phi_{\xi,\zeta} : (X^{(\mathtt{k})}, \mathcal{L}(X^{(\mathtt{k})}), Y^{(\mathtt{k})}, Z^{(\mathtt{k})}) \mapsto (X^{(\mathtt{k}+1)}, \mathcal{L}(X^{(\mathtt{k}+1)}), Y^{(\mathtt{k}+1)}, Z^{(\mathtt{k}+1)})$

Picard Scheme for MKV FBSDE System

Input: Initial guess (ξ, ζ); initial condition ξ; terminal condition ζ; time horizon T; number of iterations $\mathtt{K}$

Output: Approximation of (X, Y, Z) solving the MKV FBSDE system

Initialize $X_t^{(0)} = \xi, Y_t^{(0)} = 0, Z_t^{(0)} = 0, 0 \leq t \leq T$

for $\mathtt{k} = 0, 1, 2, \ldots, \mathtt{K} - 1$ **do**

Let $X^{(\mathtt{k}+1)}$ be the solution to:

$$\begin{cases} dX_t = B(X_t^{(\mathtt{k})}, \mathcal{L}(X_t^{(\mathtt{k})}), Y_t^{(\mathtt{k})}, Z_t^{(\mathtt{k})})dt + \sigma dW_t, & 0 \leq t \leq T \\ X_0 = \xi \end{cases}$$

Let $(Y^{(\mathtt{k}+1)}, Z^{(\mathtt{k}+1)})$ be the solution to:

$$\begin{cases} dY_t = -F(X_t^{(\mathtt{k}+1)}, \mathcal{L}(X_t^{(\mathtt{k}+1)}), Y_t^{(\mathtt{k})}, Z_t^{(\mathtt{k})})dt + Z_t^{(\mathtt{k})} dW_t, & 0 \leq t \leq T \\ Y_T = \zeta \end{cases}$$

return $\mathtt{Picard}[T](\xi, \zeta) = (X^{(\mathtt{K})}, Y^{(\mathtt{K})}, Z^{(\mathtt{K})})$

Notation: $\Phi_{\xi,\zeta} : (X^{(\mathtt{k})}, \mathcal{L}(X^{(\mathtt{k})}), Y^{(\mathtt{k})}, Z^{(\mathtt{k})}) \mapsto (X^{(\mathtt{k}+1)}, \mathcal{L}(X^{(\mathtt{k}+1)}), Y^{(\mathtt{k}+1)}, Z^{(\mathtt{k}+1)})$

Contraction? Small T or small Lipschitz constants for B, F, G

Continuation Method

- If T is big: Solve FBSDE on small intervals & “patch” the solutions together

Continuation Method

- If T is big: Solve FBSDE on small intervals & "patch" the solutions together
- Grid: $0 = T_0 < T_1 < \cdots < T_{M-1} < T_M = T$
- Subproblem: Given $(\xi_{T_m}, \mathcal{L}(\xi_{T_m}))$ and $\zeta_{T_{m+1}}$, solve:

$$\begin{cases} dX_t = B(X_t, \mathcal{L}(X_t), Y_t, Z_t)dt + \sigma dW_t, & T_m \leq t \leq T_{m+1} \\ dY_t = -F(X_t, \mathcal{L}(X_t), Y_t, Z_t)dt + Z_t dW_t, & T_m \leq t \leq T_{m+1} \\ X_{T_m} = \xi_{T_m}, \qquad Y_{T_{m+1}} = \zeta_{T_{m+1}} \end{cases}$$

Continuation Method

- If T is big: Solve FBSDE on small intervals & "patch" the solutions together
- Grid: $0 = T_0 < T_1 < \cdots < T_{M-1} < T_M = T$
- Subproblem: Given $(\xi_{T_m}, \mathcal{L}(\xi_{T_m}))$ and $\zeta_{T_{m+1}}$, solve:

$$\begin{cases} dX_t = B(X_t, \mathcal{L}(X_t), Y_t, Z_t)dt + \sigma dW_t, & T_m \le t \le T_{m+1} \\ dY_t = -F(X_t, \mathcal{L}(X_t), Y_t, Z_t)dt + Z_t dW_t, & T_m \le t \le T_{m+1} \\ X_{T_m} = \xi_{T_m}, \qquad Y_{T_{m+1}} = \zeta_{T_{m+1}} \end{cases}$$

- How to find ξ_{T_m} and $\zeta_{T_{m+1}}$?
  - $\rightarrow$ ξ_{T_m} from previous problem's solution (or initial condition)
  - $\rightarrow$ $\zeta_{T_{m+1}}$ from next problem's solution (or terminal condition)

Global Solver for MKV FBSDE System

Following [Chassagneux et al.'19], define a global solver recursively, and then call:

$$\mathtt{Solver}[m](\xi_0, \mu_0)$$

with ξ_0 a random variable with distribution μ_0

Input: Initial guess $(\xi, \mathcal{L}(\xi))$; time step index m; number of iterations $\mathtt{K}$
Output: Approximation of Y_{T_m} where (X, Y, Z) solves the MKV FBSDE system on $[T_m, T]$ starting with $(\xi, \mathcal{L}(\xi))$ at time T_m

Initialize $X_t^{(0)} = \xi, \mathcal{L}(X_t^{(0)}) = \mathcal{L}(\xi)$ for all $T_m \le t \le T_{m+1}$
for $\mathtt{k} = 0, 1, 2, \ldots, \mathtt{K} - 1$ **do**
    If $T_{m+1} = T$, $Y_{T_{m+1}}^{(\mathtt{k}+1)} = G(X_{T_{m+1}}^{(\mathtt{k})}, \mathcal{L}(X_{T_{m+1}}^{(\mathtt{k})}))$

Global Solver for MKV FBSDE System

Following [Chassagneux et al.'19], define a global solver recursively, and then call:

$$\texttt{Solver}[m](\xi_0, \mu_0)$$

with ξ_0 a random variable with distribution μ_0

Input: Initial guess $(\xi, \mathcal{L}(\xi))$; time step index m; number of iterations $\texttt{K}$

Output: Approximation of Y_{T_m} where (X, Y, Z) solves the MKV FBSDE system on $[T_m, T]$ starting with $(\xi, \mathcal{L}(\xi))$ at time T_m

Initialize $X_t^{(0)} = \xi, \mathcal{L}(X_t^{(0)}) = \mathcal{L}(\xi)$ for all $T_m \leq t \leq T_{m+1}$

for $\texttt{k} = 0, 1, 2, \ldots, \texttt{K} - 1$ **do**

If $T_{m+1} = T$, $Y_{T_{m+1}}^{(\texttt{k}+1)} = G(X_{T_{m+1}}^{(\texttt{k})}, \mathcal{L}(X_{T_{m+1}}^{(\texttt{k})}))$

Else: compute recursively:

$$Y_{T_{m+1}}^{(\texttt{k}+1)} = \texttt{Solver}[m+1](X_{T_{m+1}}^{(\texttt{k})}, \mathcal{L}(X_{T_{m+1}}^{(\texttt{k})}))$$

Global Solver for MKV FBSDE System

Following [Chassagneux et al.'19], define a global solver recursively, and then call:

$$\texttt{Solver}[m](\xi_0, \mu_0)$$

with ξ_0 a random variable with distribution μ_0

Input: Initial guess $(\xi, \mathcal{L}(\xi))$; time step index m; number of iterations $\mathtt{K}$

Output: Approximation of Y_{T_m} where (X, Y, Z) solves the MKV FBSDE system on $[T_m, T]$ starting with $(\xi, \mathcal{L}(\xi))$ at time T_m

Initialize $X_t^{(0)} = \xi, \mathcal{L}(X_t^{(0)}) = \mathcal{L}(\xi)$ for all $T_m \leq t \leq T_{m+1}$

for $\mathtt{k} = 0, 1, 2, \dots, \mathtt{K} - 1$ **do**

  If $T_{m+1} = T$, $Y_{T_{m+1}}^{(\mathtt{k}+1)} = G(X_{T_{m+1}}^{(\mathtt{k})}, \mathcal{L}(X_{T_{m+1}}^{(\mathtt{k})}))$

  Else: compute recursively:

$$Y_{T_{m+1}}^{(\mathtt{k}+1)} = \texttt{Solver}[m+1](X_{T_{m+1}}^{(\mathtt{k})}, \mathcal{L}(X_{T_{m+1}}^{(\mathtt{k})}))$$

  Compute:

$$(X_t^{(\mathtt{k}+1)}, \mathcal{L}(X_t^{(\mathtt{k}+1)}), Y_t^{(\mathtt{k}+1)}, Z_t^{(\mathtt{k}+1)})_{T_m \leq t \leq T_{m+1}} = \texttt{Picard}[T_{m+1} - T_m](X_{T_m}^{(\mathtt{k})}, Y_{T_{m+1}}^{(\mathtt{k}+1)})$$

return $\texttt{Solver}[m](\xi, \mathcal{L}(\xi)) := Y_{T_m}^{(\mathtt{K})}$

Implementation: Discretizations

Following [Angiuli et al.'19][3]

- Tree algorithm:
  - Time discretization
  - Space discretization: binomial tree structure
  - Look at trajectories
- Grid algorithm:
  - Time and space discretization on a grid
  - Look at time marginals

[3] Angiuli, A., et al. Cemracs 2017: numerical probabilistic approach to MFG. *ESAIM: Proceedings and Surveys* 65 (2019): 84-113.

Outline

1. A Picard Scheme for MKV FBSDE
   - Picard Scheme & Continuation Method
   - Tree-Based Algorithm
   - Grid-Based Algorithm

2. Stochastic Methods for some Finite-Dimensional MFC Problems

Time Discretization

- Focus on an interval $[0, T]$ with small enough T (otherwise: call recursive solver)

- Focus on an interval $[0, T]$ with small enough T (otherwise: call recursive solver)
- Time discretization: $0 = t_0 < t_1 < \cdots < t_{N_t} = T$, $t_{i+1} - t_i = \Delta t$
- Euler Scheme: $0 \le i \le N_t - 1$

$$
\begin{cases}
X_{t_{i+1}}^{(\mathtt{k+1})} = X_{t_i}^{(\mathtt{k+1})} + B(X_{t_i}^{(\mathtt{k+1})}, \mathcal{L}(X_{t_i}^{(\mathtt{k+1})}), Y_{t_i}^{(\mathtt{k})}, Z_{t_i}^{(\mathtt{k})})\Delta t + \sigma \Delta W_{t_{i+1}} \\
X_0^{(\mathtt{k+1})} = \xi \\
Y_{t_i}^{(\mathtt{k+1})} = \mathbb{E}_{t_i}[Y_{t_{i+1}}^{(\mathtt{k+1})}] + F(X_{t_i}^{(\mathtt{k+1})}, \mathcal{L}(X_{t_i}^{(\mathtt{k+1})}), Y_{t_i}^{(\mathtt{k})}, Z_{t_i}^{(\mathtt{k})})\Delta t \\
\qquad \approx Y_{t_{i+1}}^{(\mathtt{k+1})} + F(X_{t_i}^{(\mathtt{k+1})}, \mathcal{L}(X_{t_i}^{(\mathtt{k+1})}), Y_{t_i}^{(\mathtt{k})}, Z_{t_i}^{(\mathtt{k})})\Delta t - Z_{t_i}^{(\mathtt{k+1})} \Delta W_{t_{i+1}} \\
Y_T^{(\mathtt{k+1})} = G(X_T^{(\mathtt{k+1})}, \mathcal{L}(X_T^{(\mathtt{k+1})})) \\
Z_{t_i}^{(\mathtt{k+1})} = \frac{1}{\Delta t} \mathbb{E}_{t_i}[Y_{t_{i+1}}^{(\mathtt{k+1})} \Delta W_{t_{i+1}}] \\
Z_T^{(\mathtt{k+1})} = 0
\end{cases}
$$

Time Discretization

- Focus on an interval $[0, T]$ with small enough T (otherwise: call recursive solver)
- Time discretization: $0 = t_0 < t_1 < \cdots < t_{N_t} = T$, $t_{i+1} - t_i = \Delta t$
- Euler Scheme: $0 \le i \le N_t - 1$

$$
\begin{cases}
X_{t_{i+1}}^{(\mathtt{k+1})} = X_{t_i}^{(\mathtt{k+1})} + B(X_{t_i}^{(\mathtt{k+1})}, \mathcal{L}(X_{t_i}^{(\mathtt{k+1})}), Y_{t_i}^{(\mathtt{k})}, Z_{t_i}^{(\mathtt{k})})\Delta t + \sigma \Delta W_{t_{i+1}} \\
X_0^{(\mathtt{k+1})} = \xi \\
Y_{t_i}^{(\mathtt{k+1})} = \mathbb{E}_{t_i}[Y_{t_{i+1}}^{(\mathtt{k+1})}] + F(X_{t_i}^{(\mathtt{k+1})}, \mathcal{L}(X_{t_i}^{(\mathtt{k+1})}), Y_{t_i}^{(\mathtt{k})}, Z_{t_i}^{(\mathtt{k})})\Delta t \\
\qquad\quad \approx Y_{t_{i+1}}^{(\mathtt{k+1})} + F(X_{t_i}^{(\mathtt{k+1})}, \mathcal{L}(X_{t_i}^{(\mathtt{k+1})}), Y_{t_i}^{(\mathtt{k})}, Z_{t_i}^{(\mathtt{k})})\Delta t - Z_{t_i}^{(\mathtt{k+1})} \Delta W_{t_{i+1}} \\
Y_T^{(\mathtt{k+1})} = G(X_T^{(\mathtt{k+1})}, \mathcal{L}(X_T^{(\mathtt{k+1})})) \\
Z_{t_i}^{(\mathtt{k+1})} = \frac{1}{\Delta t}\mathbb{E}_{t_i}[Y_{t_{i+1}}^{(\mathtt{k+1})} \Delta W_{t_{i+1}}] \\
Z_T^{(\mathtt{k+1})} = 0
\end{cases}
$$

- Questions:
  - How to represent $\mathcal{L}(X_{t_i}^{(\mathtt{k+1})})$?
  - How to compute the conditional expectation $\mathbb{E}_{t_i}[Y_{t_{i+1}}^{(\mathtt{k+1})}]$?

Tree-Based Approach

- At each t_i, replace $\Delta W_{t_{i+1}}$ by a branch with 2 values: $\pm\sqrt{\Delta t}$ w.p. $1/2$
- Answers:
  - $\mathcal{L}(X_{t_i}^{(\mathtt{k}+1)}) \approx$ weighted empirical distribution:

$$\mathcal{L}(X_{t_0}^{(\mathtt{k}+1)}) \approx \sum_{n=1}^{N_{x_0}} p_0^k \delta_{x_0^k},$$

    and at time $t_i, i \geq 1$: look at values on the nodes at depth i
  - $\mathbb{E}_{t_i}[Y_{t_{i+1}}^{(\mathtt{k}+1)}] \approx$ weighted average of values on the two next branches

Tree-Based Approach

- At each t_i, replace $\Delta W_{t_{i+1}}$ by a branch with 2 values: $\pm\sqrt{\Delta t}$ w.p. $1/2$
- Answers:
  - $\mathcal{L}(X_{t_i}^{(\mathtt{k}+1)}) \approx$ weighted empirical distribution:

$$\mathcal{L}(X_{t_0}^{(\mathtt{k}+1)}) \approx \sum_{n=1}^{N_{x_0}} p_0^k \delta_{x_0^k},$$

    and at time $t_i, i \geq 1$: look at values on the nodes at depth i
  - $\mathbb{E}_{t_i}[Y_{t_{i+1}}^{(\mathtt{k}+1)}] \approx$ weighted average of values on the two next branches
- Starting from some x_0, doing N_t steps: 2^{N_t} paths
- N_{x_0} starting points i.i.d. $\sim \mu_0$: $N_{x_0} \times 2^{N_t}$ paths !

Tree-Based Approach

- At each t_i, replace $\Delta W_{t_{i+1}}$ by a branch with 2 values: $\pm\sqrt{\Delta t}$ w.p. $1/2$
- Answers:
  - $\mathcal{L}(X_{t_i}^{(\mathtt{k}+1)}) \approx$ weighted empirical distribution:

$$\mathcal{L}(X_{t_0}^{(\mathtt{k}+1)}) \approx \sum_{n=1}^{N_{x_0}} p_0^k \delta_{x_0^k},$$

    and at time $t_i, i \geq 1$: look at values on the nodes at depth i
  - $\mathbb{E}_{t_i}[Y_{t_{i+1}}^{(\mathtt{k}+1)}] \approx$ weighted average of values on the two next branches
- Starting from some x_0, doing N_t steps: 2^{N_t} paths
- N_{x_0} starting points i.i.d. $\sim \mu_0$: $N_{x_0} \times 2^{N_t}$ paths !
- Save space thanks to recombinations?

Tree-Based Approach

- At each t_i, replace $\Delta W_{t_{i+1}}$ by a branch with 2 values: $\pm\sqrt{\Delta t}$ w.p. $1/2$
- Answers:
  - $\mathcal{L}(X_{t_i}^{(\mathtt{k}+1)}) \approx$ weighted empirical distribution:

$$\mathcal{L}(X_{t_0}^{(\mathtt{k}+1)}) \approx \sum_{n=1}^{N_{x_0}} p_0^k \delta_{x_0^k},$$

    and at time $t_i, i \geq 1$: look at values on the nodes at depth i
  - $\mathbb{E}_{t_i}[Y_{t_{i+1}}^{(\mathtt{k}+1)}] \approx$ weighted average of values on the two next branches
- Starting from some x_0, doing N_t steps: 2^{N_t} paths
- N_{x_0} starting points i.i.d. $\sim \mu_0$: $N_{x_0} \times 2^{N_t}$ paths !
- Save space thanks to recombinations? *Not really but ...*

Outline

1. A Picard Scheme for MKV FBSDE
   - Picard Scheme & Continuation Method
   - Tree-Based Algorithm
   - Grid-Based Algorithm

2. Stochastic Methods for some Finite-Dimensional MFC Problems

Time & Space Discretization

- Decoupling functions (see e.g., [CD'18, Vol. I, Section 6.4]):

$$Y_t = u(t, X_t, \mathcal{L}(X_t)), \qquad Z_t = v(t, X_t, \mathcal{L}(X_t))$$

$\rightarrow$ Approximate $u(\cdot,\cdot,\cdot), v(\cdot,\cdot,\cdot)$ instead of $(Y_t, Z_t)_{t\in[0,T]}$

Time & Space Discretization

- Decoupling functions (see e.g., [CD'18, Vol. I, Section 6.4]):

$$Y_t = u(t, X_t, \mathcal{L}(X_t)), \qquad Z_t = v(t, X_t, \mathcal{L}(X_t))$$

  $\rightarrow$ Approximate $u(\cdot,\cdot,\cdot), v(\cdot,\cdot,\cdot)$ instead of $(Y_t, Z_t)_{t\in[0,T]}$

- Difficulty: space of $\mathcal{L}(X_t)$ is infinite dimensional
  $\rightarrow$ Freeze it during each Picard iteration:

$$Y_t^{(\mathtt{k}+1)} = u^{(\mathtt{k}+1)}(t, X_t^{(\mathtt{k}+1)}), \qquad Z_t^{(\mathtt{k}+1)} = v^{(\mathtt{k}+1)}(t, X_t^{(\mathtt{k}+1)}) \qquad (\star)$$

Time & Space Discretization

- Decoupling functions (see e.g., [CD'18, Vol. I, Section 6.4]):

$$Y_t = u(t, X_t, \mathcal{L}(X_t)), \qquad Z_t = v(t, X_t, \mathcal{L}(X_t))$$

  $\rightarrow$ Approximate $u(\cdot,\cdot,\cdot), v(\cdot,\cdot,\cdot)$ instead of $(Y_t, Z_t)_{t\in[0,T]}$

- Difficulty: space of $\mathcal{L}(X_t)$ is infinite dimensional
  $\rightarrow$ Freeze it during each Picard iteration:

$$Y_t^{(\mathtt{k}+1)} = u^{(\mathtt{k}+1)}(t, X_t^{(\mathtt{k}+1)}), \qquad Z_t^{(\mathtt{k}+1)} = v^{(\mathtt{k}+1)}(t, X_t^{(\mathtt{k}+1)}) \tag{$\star$}$$

- Picard iterations for distribution & decoupling functions:
  - **Step 1:** Given $(\mu^{(\mathtt{k})}, u^{(\mathtt{k})}, v^{(\mathtt{k})})$, compute $\mu_t^{(\mathtt{k}+1)} = \mathcal{L}(X_t^{(\mathtt{k}+1)}), 0 \le t \le T$, where

$$dX_t^{(\mathtt{k}+1)} = B\Big(X_t^{(\mathtt{k}+1)}, \mu_t^{(\mathtt{k})}, u^{(\mathtt{k})}(t, X_t^{(\mathtt{k}+1)}), v^{(\mathtt{k})}(t, X_t^{(\mathtt{k}+1)})\Big)dt + \sigma dW_t$$

  - **Step 2:** Given $(X^{(\mathtt{k})}, \mu^{(\mathtt{k}+1)})$, compute $(u^{(\mathtt{k}+1)}, v^{(\mathtt{k}+1)})$ such that ($\star$) holds, where

$$dY_t^{(\mathtt{k}+1)} = -F\Big(X_t^{(\mathtt{k}+1)}, \mu_t^{(\mathtt{k}+1)}, Y_t^{(\mathtt{k}+1)}, Z_t^{(\mathtt{k}+1)}\Big)dt + Z_t^{(\mathtt{k}+1)}dW_t$$

  - Return $(\mu^{(\mathtt{k}+1)}, u^{(\mathtt{k}+1)}, v^{(\mathtt{k}+1)})$

Time & Space Discretization: Forward Equation

- Focus on an interval $[0, T]$ with small enough T (otherwise: call recursive solver)
- Time discretization: $0 = t_0 < t_1 < \cdots < t_{N_t} = T$, $t_{i+1} - t_i = \Delta t$
- Space discretization ($d = 1$): Grid Γ: $x_0 < x_1 < \cdots < x_{N_x}$, $x_{j+1} - x_j = \Delta x$

Time & Space Discretization: Forward Equation

- Focus on an interval $[0, T]$ with small enough T (otherwise: call recursive solver)
- Time discretization: $0 = t_0 < t_1 < \cdots < t_{N_t} = T$, $t_{i+1} - t_i = \Delta t$
- Space discretization ($d = 1$): Grid Γ: $x_0 < x_1 < \cdots < x_{N_x}$, $x_{j+1} - x_j = \Delta x$
- Use projection Π to stay on Γ at every t_i: $\mathcal{L}(X_{t_i}^{(\mathtt{k}+1)}) \approx$ vector of weights

Time & Space Discretization: Forward Equation

- Focus on an interval $[0, T]$ with small enough T (otherwise: call recursive solver)
- Time discretization: $0 = t_0 < t_1 < \cdots < t_{N_t} = T$, $t_{i+1} - t_i = \Delta t$
- Space discretization ($d = 1$): Grid Γ: $x_0 < x_1 < \cdots < x_{N_x}$, $x_{j+1} - x_j = \Delta x$
- Use projection Π to stay on Γ at every t_i: $\mathcal{L}(X_{t_i}^{(\mathtt{k}+1)}) \approx$ vector of weights
- Picard iterations for distribution & decoupling functions:
  - **Step 1:** Given $(\mu^{(\mathtt{k})}, u^{(\mathtt{k})}, v^{(\mathtt{k})})$, compute $\mu_{t_i}^{(\mathtt{k}+1)} = \mathcal{L}(X_{t_i}^{(\mathtt{k}+1)}), i = 0, \ldots, N_t$, where

$$X_{t_{i+1}}^{(\mathtt{k}+1)} = \Pi\Big[X_{t_i}^{(\mathtt{k}+1)} + B\Big(X_{t_i}^{(\mathtt{k}+1)}, \mu_{t_i}^{(\mathtt{k})}, u_{t_i}^{(\mathtt{k})}(X_{t_i}^{(\mathtt{k}+1)}), v_{t_i}^{(\mathtt{k})}(X_{t_i}^{(\mathtt{k}+1)})\Big)dt + \sigma \Delta W_{t_{i+1}}\Big]$$

Time & Space Discretization: Forward Equation

- Focus on an interval $[0, T]$ with small enough T (otherwise: call recursive solver)
- Time discretization: $0 = t_0 < t_1 < \cdots < t_{N_t} = T$, $t_{i+1} - t_i = \Delta t$
- Space discretization ($d = 1$): Grid Γ: $x_0 < x_1 < \cdots < x_{N_x}$, $x_{j+1} - x_j = \Delta x$
- Use projection Π to stay on Γ at every t_i: $\mathcal{L}(X_{t_i}^{(\mathtt{k}+1)}) \approx$ vector of weights
- Picard iterations for distribution & decoupling functions:
  - **Step 1:** Given $(\mu^{(\mathtt{k})}, u^{(\mathtt{k})}, v^{(\mathtt{k})})$, compute $\mu_{t_i}^{(\mathtt{k}+1)} = \mathcal{L}(X_{t_i}^{(\mathtt{k}+1)}), i = 0, \ldots, N_t$, where

$$X_{t_{i+1}}^{(\mathtt{k}+1)} = \Pi\Big[X_{t_i}^{(\mathtt{k}+1)} + B\Big(X_{t_i}^{(\mathtt{k}+1)}, \mu_{t_i}^{(\mathtt{k})}, u_{t_i}^{(\mathtt{k})}(X_{t_i}^{(\mathtt{k}+1)}), v_{t_i}^{(\mathtt{k})}(X_{t_i}^{(\mathtt{k}+1)})\Big)dt$$
$$+ \sigma \Delta W_{t_{i+1}}\Big]$$

  - In fact $\mu_{t_{i+1}}^{(\mathtt{k}+1)}$ can be expressed in terms of $\mu_{t_i}^{(\mathtt{k}+1)}$ and a transition kernel
  - Ex: binomial approx. of $W \to$ efficient computation using quantization

Time & Space Discretization: Backward Equation

- Picard iterations for distribution & decoupling functions (continued):
  - **Step 2:** Update u, v: for all $0 \leq i \leq N_t$, $x \in \Gamma$,

$$\begin{cases} u_{t_i}^{(\mathtt{k}+1)}(x) = \mathbb{E}\Big[u_{t_{i+1}}^{(\mathtt{k}+1)}(X_{t_i}^{(\mathtt{k}+1)}) \\ \qquad\qquad +F\Big(X_{t_i}^{(\mathtt{k}+1)}, \mu_{t_i}^{(\mathtt{k}+1)}, u_{t_i}^{(\mathtt{k})}(X_{t_i}^{(\mathtt{k}+1)}), v_{t_i}^{(\mathtt{k})}(X_{t_i}^{(\mathtt{k}+1)})\Big)\Delta t \,\Big|\, X_{t_i}^{(\mathtt{k}+1)} = x\Big] \\ u_T^{(\mathtt{k}+1)}(x) = G(x, \mu_{t_i}^{(\mathtt{k}+1)}) \\ v_{t_i}^{(\mathtt{k}+1)}(x) = \mathbb{E}\Big[\frac{1}{\Delta t} u_{t_{i+1}}^{(\mathtt{k}+1)}(X_{t_i}^{(\mathtt{k}+1)}) \,\Big|\, X_{t_i}^{(\mathtt{k}+1)} = x\Big] \\ v_T^{(\mathtt{k}+1)}(x) = 0 \end{cases}$$

  - Ex.: binomial approximation of W $\rightarrow$ more explicit formulas

Time & Space Discretization: Backward Equation

- Picard iterations for distribution & decoupling functions (continued):
  - **Step 2:** Update u, v: for all $0 \leq i \leq N_t$, $x \in \Gamma$,

$$
\begin{cases}
u_{t_i}^{(\mathtt{k}+1)}(x) = \mathbb{E}\Big[u_{t_{i+1}}^{(\mathtt{k}+1)}(X_{t_i}^{(\mathtt{k}+1)}) \\
\qquad\qquad + F\Big(X_{t_i}^{(\mathtt{k}+1)}, \mu_{t_i}^{(\mathtt{k}+1)}, u_{t_i}^{(\mathtt{k})}(X_{t_i}^{(\mathtt{k}+1)}), v_{t_i}^{(\mathtt{k})}(X_{t_i}^{(\mathtt{k}+1)})\Big)\Delta t \,\Big|\, X_{t_i}^{(\mathtt{k}+1)} = x\Big] \\
u_T^{(\mathtt{k}+1)}(x) = G(x, \mu_{t_i}^{(\mathtt{k}+1)}) \\
v_{t_i}^{(\mathtt{k}+1)}(x) = \mathbb{E}\Big[\frac{1}{\Delta t} u_{t_{i+1}}^{(\mathtt{k}+1)}(X_{t_i}^{(\mathtt{k}+1)}) \,\Big|\, X_{t_i}^{(\mathtt{k}+1)} = x\Big] \\
v_T^{(\mathtt{k}+1)}(x) = 0
\end{cases}
$$

  - Ex.: binomial approximation of W $\to$ more explicit formulas
- Summary:
  - Forward: $(\mu^{(\mathtt{k})}, u^{(\mathtt{k})}, v^{(\mathtt{k})}) \mapsto \mu^{(\mathtt{k}+1)} = \mathcal{L}(X^{(\mathtt{k}+1)})$
  - Backward: $(\mu^{(\mathtt{k}+1)}, u^{(\mathtt{k})}, v^{(\mathtt{k})}) \mapsto (u^{(\mathtt{k}+1)}, v^{(\mathtt{k}+1)})$

Time & Space Discretization: Backward Equation

- Picard iterations for distribution & decoupling functions (continued):
  - **Step 2:** Update u, v: for all $0 \leq i \leq N_t$, $x \in \Gamma$,

$$\begin{cases} u_{t_i}^{(\mathtt{k}+1)}(x) = \mathbb{E}\Big[u_{t_{i+1}}^{(\mathtt{k}+1)}(X_{t_i}^{(\mathtt{k}+1)}) \\ \qquad\qquad + F\Big(X_{t_i}^{(\mathtt{k}+1)}, \mu_{t_i}^{(\mathtt{k}+1)}, u_{t_i}^{(\mathtt{k})}(X_{t_i}^{(\mathtt{k}+1)}), v_{t_i}^{(\mathtt{k})}(X_{t_i}^{(\mathtt{k}+1)})\Big)\Delta t \,\Big|\, X_{t_i}^{(\mathtt{k}+1)} = x \Big] \\ u_T^{(\mathtt{k}+1)}(x) = G(x, \mu_{t_i}^{(\mathtt{k}+1)}) \\ v_{t_i}^{(\mathtt{k}+1)}(x) = \mathbb{E}\Big[\frac{1}{\Delta t} u_{t_{i+1}}^{(\mathtt{k}+1)}(X_{t_i}^{(\mathtt{k}+1)}) \,\Big|\, X_{t_i}^{(\mathtt{k}+1)} = x \Big] \\ v_T^{(\mathtt{k}+1)}(x) = 0 \end{cases}$$

  - Ex.: binomial approximation of W $\rightarrow$ more explicit formulas
- Summary:
  - Forward: $(\mu^{(\mathtt{k})}, u^{(\mathtt{k})}, v^{(\mathtt{k})}) \mapsto \mu^{(\mathtt{k}+1)} = \mathcal{L}(X^{(\mathtt{k}+1)})$
  - Backward: $(\mu^{(\mathtt{k}+1)}, u^{(\mathtt{k})}, v^{(\mathtt{k})}) \mapsto (u^{(\mathtt{k}+1)}, v^{(\mathtt{k}+1)})$

For more details and numerical examples, see [Chassagneux et al.'19; Angiuli et al.'19]

Outline

1. A Picard Scheme for MKV FBSDE

2. Stochastic Methods for some Finite-Dimensional MFC Problems
   - Finite-Dimensional Structure
   - Conditional Expectation Estimation

Outline

1. A Picard Scheme for MKV FBSDE

2. Stochastic Methods for some Finite-Dimensional MFC Problems
   - Finite-Dimensional Structure
   - Conditional Expectation Estimation

Dependence on the Moments

- In general: b, f, g involve the whole distribution $\mu_t = \mathcal{L}(X_t)$ (infinite dim.)
- What if they involve only the first moment $\overline{\mu}_t = \mathbb{E}[X_t]$?

Dependence on the Moments

- In general: b, f, g involve the whole distribution $\mu_t = \mathcal{L}(X_t)$ (infinite dim.)
- What if they involve only the first moment $\overline{\mu}_t = \mathbb{E}[X_t]$?
- Ex. 1: LQ (see Part 1)
  - optimal control is a function of X_t and $\overline{\mu}_t = \mathbb{E}[X_t]$
  - ODE for $\overline{\mu}_t$ of the form $\frac{d}{dt}\overline{\mu}_t = \varphi(t, \overline{\mu}_t)$

Dependence on the Moments

- In general: b, f, g involve the whole distribution $\mu_t = \mathcal{L}(X_t)$ (infinite dim.)
- What if they involve only the first moment $\overline{\mu}_t = \mathbb{E}[X_t]$?
- Ex. 1: LQ (see Part 1)
  - optimal control is a function of X_t and $\overline{\mu}_t = \mathbb{E}[X_t]$
  - ODE for $\overline{\mu}_t$ of the form $\frac{d}{dt}\overline{\mu}_t = \varphi(t, \overline{\mu}_t)$
- Ex. 2:

$$\begin{cases} b(x, \mu, v) = b(x, \overline{\mu}, v) = (\cos(x) + \cos(\overline{\mu}))v \\ f(x, \mu, v) = |v|^2, \qquad g(x, \mu) = 0 \end{cases}$$

  - Can the optimal control be expressed as a function of $X_t, \mathbb{E}[X_t]$ only?
  - ODE for $\overline{\mu}_t$?

Dependence on the Moments

- In general: b, f, g involve the whole distribution $\mu_t = \mathcal{L}(X_t)$ (infinite dim.)

- What if they involve only the first moment $\overline{\mu}_t = \mathbb{E}[X_t]$?

- Ex. 1: LQ (see Part 1)
  - optimal control is a function of X_t and $\overline{\mu}_t = \mathbb{E}[X_t]$
  - ODE for $\overline{\mu}_t$ of the form $\frac{d}{dt}\overline{\mu}_t = \varphi(t, \overline{\mu}_t)$

- Ex. 2:

$$\begin{cases} b(x, \mu, v) = b(x, \overline{\mu}, v) = (\cos(x) + \cos(\overline{\mu}))v \\ f(x, \mu, v) = |v|^2, \qquad g(x, \mu) = 0 \end{cases}$$

  - Can the optimal control be expressed as a function of $X_t, \mathbb{E}[X_t]$ only?
  - ODE for $\overline{\mu}_t$?

$$\frac{d}{dt}\overline{\mu}_t = \mathbb{E}\left[(\cos(X_t) + \cos(\overline{\mu}_t))v(t, X_t)\right]$$

⚠ It involves not only $\mathbb{E}[X_t] = \overline{\mu}_t$ but also $\mathbb{E}[\cos(X_t)]$

Dependence on the Moments

- In general: b, f, g involve the whole distribution $\mu_t = \mathcal{L}(X_t)$ (infinite dim.)

- What if they involve only the first moment $\overline{\mu}_t = \mathbb{E}[X_t]$?

- Ex. 1: LQ (see Part 1)
  - optimal control is a function of X_t and $\overline{\mu}_t = \mathbb{E}[X_t]$
  - ODE for $\overline{\mu}_t$ of the form $\frac{d}{dt}\overline{\mu}_t = \varphi(t, \overline{\mu}_t)$

- Ex. 2:

$$\begin{cases} b(x, \mu, v) = b(x, \overline{\mu}, v) = (\cos(x) + \cos(\overline{\mu}))v \\ f(x, \mu, v) = |v|^2, \qquad g(x, \mu) = 0 \end{cases}$$

  - Can the optimal control be expressed as a function of $X_t, \mathbb{E}[X_t]$ only?
  - ODE for $\overline{\mu}_t$?

$$\frac{d}{dt}\overline{\mu}_t = \mathbb{E}\left[(\cos(X_t) + \cos(\overline{\mu}_t))v(t, X_t)\right]$$

  ⚠ It involves not only $\mathbb{E}[X_t] = \overline{\mu}_t$ but also $\mathbb{E}[\cos(X_t)]$

- Class of MFC s.t. the problem can be solved with a finite number of moments?

Finite-Dimensional Reformulation

Following [Balata et al.'19][4]:

- In some cases, MFC problems can be written as:

$$J(v) = \mathbb{E}\left[\int_0^T \mathcal{F}(\underline{X}_t, v_t)dt + \mathcal{G}(\underline{X}_T)\right]$$

subject to:

$$d\underline{X}_t = \mathcal{B}(\underline{X}_t, v_t)dt + \Sigma d\mathbb{W}_t$$

where the state is: $\underline{X}_t = (\mathbb{E}[X_t], \mathbb{E}[|X_t|^2], \dots, \mathbb{E}[|X_t|^p]) \in (\mathbb{R}^d)^p$

[4] Balata, A., Huré, C., Laurière, M., Pham, H., & Pimentel, I. (2019). A class of finite-dimensional numerically solvable McKean-Vlasov control problems. *ESAIM: Proceedings and Surveys*, 65, 114-144.

Finite-Dimensional Reformulation

Following [Balata et al.'19][4]:

- In some cases, MFC problems can be written as:

$$J(v) = \mathbb{E}\left[\int_0^T \mathcal{F}(\underline{X}_t, v_t)dt + \mathcal{G}(\underline{X}_T)\right]$$

  subject to:

$$d\underline{X}_t = \mathcal{B}(\underline{X}_t, v_t)dt + \Sigma d\mathbb{W}_t$$

  where the state is: $\underline{X}_t = (\mathbb{E}[X_t], \mathbb{E}[|X_t|^2], \ldots, \mathbb{E}[|X_t|^p]) \in (\mathbb{R}^d)^p$

- Time discretization: $0 = t_0 < t_1 < \cdots < t_{N_t} = T$, $t_{i+1} - t_i = \Delta t$

[4] Balata, A., Huré, C., Laurière, M., Pham, H., & Pimentel, I. (2019). A class of finite-dimensional numerically solvable McKean-Vlasov control problems. *ESAIM: Proceedings and Surveys*, 65, 114-144.

Finite-Dimensional Reformulation

Following [Balata et al.'19][4]:

- In some cases, MFC problems can be written as:

$$J(v) = \mathbb{E}\left[\int_0^T \mathcal{F}(\underline{X}_t, v_t)dt + \mathcal{G}(\underline{X}_T)\right]$$

  subject to:

$$d\underline{X}_t = \mathcal{B}(\underline{X}_t, v_t)dt + \Sigma d\mathbb{W}_t$$

  where the state is: $\underline{X}_t = (\mathbb{E}[X_t], \mathbb{E}[|X_t|^2], \dots, \mathbb{E}[|X_t|^p]) \in (\mathbb{R}^d)^p$

- Time discretization: $0 = t_0 < t_1 < \cdots < t_{N_t} = T$, $t_{i+1} - t_i = \Delta t$
- DPP for $V : [0,T] \times (\mathbb{R}^d)^p \to \mathbb{R}$ or rather $V_{\Delta t} : \{t_0, \dots, t_{N_t}\} \times (\mathbb{R}^d)^p \to \mathbb{R}$:

$$\begin{cases} V_{\Delta t}(T, \underline{x}) = \mathcal{G}(\underline{x}) \\ V_{\Delta t}(t_n, \underline{x}) = \sup_v \left\{ \mathcal{F}(\underline{x}, v)\Delta t + \mathbb{E}^{t_n, \underline{x}, v}\left[V_{\Delta t}(t_{n+1}, \underline{X}_{t_{n+1}})\right]\right\}, n = N_t - 1, \dots, 1, 0 \end{cases}$$

  where $\mathbb{E}^{t_n, \underline{x}, v}\left[V_{\Delta t}(t_{n+1}, \underline{X}_{t_{n+1}})\right] = \mathbb{E}\left[V_{\Delta t}(t_{n+1}, \underline{X}^v_{t_{n+1}}) \,|\, \underline{X}^v_{t_n} = \underline{x}\right]$

[4] Balata, A., Huré, C., Laurière, M., Pham, H., & Pimentel, I. (2019). A class of finite-dimensional numerically solvable McKean-Vlasov control problems. *ESAIM: Proceedings and Surveys*, 65, 114-144.

Finite-Dimensional Reformulation

Following [Balata et al.'19][4]:

- In some cases, MFC problems can be written as:

$$J(v) = \mathbb{E}\left[\int_0^T \mathcal{F}(\underline{X}_t, v_t)dt + \mathcal{G}(\underline{X}_T)\right]$$

  subject to:

$$d\underline{X}_t = \mathcal{B}(\underline{X}_t, v_t)dt + \Sigma d\mathbb{W}_t$$

  where the state is: $\underline{X}_t = (\mathbb{E}[X_t], \mathbb{E}[|X_t|^2], \dots, \mathbb{E}[|X_t|^p]) \in (\mathbb{R}^d)^p$

- Time discretization: $0 = t_0 < t_1 < \cdots < t_{N_t} = T$, $t_{i+1} - t_i = \Delta t$
- DPP for $V : [0,T] \times (\mathbb{R}^d)^p \to \mathbb{R}$ or rather $V_{\Delta t} : \{t_0, \dots, t_{N_t}\} \times (\mathbb{R}^d)^p \to \mathbb{R}$:

$$\begin{cases} V_{\Delta t}(T, \underline{x}) = \mathcal{G}(\underline{x}) \\ V_{\Delta t}(t_n, \underline{x}) = \sup_v \left\{\mathcal{F}(\underline{x}, v)\Delta t + \mathbb{E}^{t_n, \underline{x}, v}\left[V_{\Delta t}(t_{n+1}, \underline{X}_{t_{n+1}})\right]\right\}, n = N_t - 1, \dots, 1, 0 \end{cases}$$

  where $\mathbb{E}^{t_n, \underline{x}, v}\left[V_{\Delta t}(t_{n+1}, \underline{X}_{t_{n+1}})\right] = \mathbb{E}\left[V_{\Delta t}(t_{n+1}, \underline{X}^v_{t_{n+1}}) \,|\, \underline{X}^v_{t_n} = \underline{x}\right]$

  $\to$ **Key difficulty:** estimation of the conditional expectation

[4] Balata, A., Huré, C., Laurière, M., Pham, H., & Pimentel, I. (2019). A class of finite-dimensional numerically solvable McKean-Vlasov control problems. *ESAIM: Proceedings and Surveys*, 65, 114-144.

Outline

Estimation Method 1: Regression Monte Carlo

- Family of basis functions $\phi = (\phi^m)_{m=1,\ldots,M}$
- Projection:

$$\mathbb{E}\left[V_{\Delta t}(t_{n+1}, \underline{X}^{v}_{t_{n+1}}) \,|\, \underline{X}^{v}_{t_n}\right] \approx \sum_{m=1}^{M} \beta^m_{t_n} \phi^m(\underline{X}^{v}_{t_n})$$

where

$$\beta^m_{t_n} = \operatorname*{argmin}_{\beta \in \mathbb{R}^M} \mathbb{E}\left[\left|V_{\Delta t}(t_{n+1}, \underline{X}^{v}_{t_{n+1}}) - \sum_{m=1}^{M} \beta^m \phi^m(\underline{X}^{v}_{t_n})\right|^2\right]$$

Estimation Method 1: Regression Monte Carlo

- Family of basis functions $\phi = (\phi^m)_{m=1,\dots,M}$
- Projection:

$$\mathbb{E}\left[V_{\Delta t}(t_{n+1}, \underline{X}^v_{t_{n+1}}) \mid \underline{X}^v_{t_n}\right] \approx \sum_{m=1}^{M} \beta^m_{t_n} \phi^m(\underline{X}^v_{t_n})$$

where

$$\beta^m_{t_n} = \underset{\beta \in \mathbb{R}^M}{\operatorname{argmin}} \, \mathbb{E}\left[\left| V_{\Delta t}(t_{n+1}, \underline{X}^v_{t_{n+1}}) - \sum_{m=1}^{M} \beta^m \phi^m(\underline{X}^v_{t_n}) \right|^2\right]$$

- Explicit expression:

$$\beta^m_{t_n} = \mathbb{E}[\phi(\underline{X}^v_{t_n})\phi(\underline{X}^v_{t_n})^\top]^{-1} \, \mathbb{E}[V_{\Delta t}(t_{n+1}, \underline{X}^v_{t_{n+1}})\phi(\underline{X}^v_{t_n})]$$

Estimation Method 1: Regression Monte Carlo

- Family of basis functions $\phi = (\phi^m)_{m=1,\dots,M}$
- Projection:

$$\mathbb{E}\left[V_{\Delta t}(t_{n+1}, \underline{X}^v_{t_{n+1}}) \,|\, \underline{X}^v_{t_n}\right] \approx \sum_{m=1}^{M} \beta^m_{t_n} \phi^m(\underline{X}^v_{t_n})$$

where

$$\beta^m_{t_n} = \underset{\beta \in \mathbb{R}^M}{\operatorname{argmin}} \, \mathbb{E}\left[\left| V_{\Delta t}(t_{n+1}, \underline{X}^v_{t_{n+1}}) - \sum_{m=1}^{M} \beta^m \phi^m(\underline{X}^v_{t_n}) \right|^2\right]$$

- Explicit expression:

$$\beta^m_{t_n} = \mathbb{E}[\phi(\underline{X}^v_{t_n})\phi(\underline{X}^v_{t_n})^\top]^{-1}\,\mathbb{E}[V_{\Delta t}(t_{n+1}, \underline{X}^v_{t_{n+1}})\phi(\underline{X}^v_{t_n})]$$

- Estimation with N_{MC} Monte Carlo samples:

$$\mathbb{E}[\phi(\underline{X}^{\ell,v}_{t_n})\phi(\underline{X}^{\ell,v}_{t_n})^\top] \approx \frac{1}{N_{MC}} \sum_{\ell=1}^{N_{MC}} \phi(\underline{X}^{\ell,v}_{t_n})\phi(\underline{X}^{\ell,v}_{t_n})^\top$$

and

$$\mathbb{E}[V_{\Delta t}(t_{n+1}, \underline{X}^{\ell,v}_{t_{n+1}})\phi(\underline{X}^{\ell,v}_{t_n})] \approx \frac{1}{N_{MC}} \sum_{\ell=1}^{N_{MC}} V_{\Delta t}(t_{n+1}, \underline{X}^{\ell,v}_{t_{n+1}})\phi(\underline{X}^{\ell,v}_{t_n})$$

with training set $\{(\underline{X}^{\ell,v}_{t_n}, \underline{X}^{\ell,v}_{t_{n+1}}); \ell = 1, \dots, N_{MC}\}$

Estimation Method 1: Regression Monte Carlo

- Family of basis functions $\phi = (\phi^m)_{m=1,\ldots,M}$ ⚠ *Not always easy to choose !*
- Projection:

$$\mathbb{E}\left[V_{\Delta t}(t_{n+1}, \underline{X}^{v}_{t_{n+1}}) \,|\, \underline{X}^{v}_{t_n}\right] \approx \sum_{m=1}^{M} \beta^m_{t_n} \phi^m(\underline{X}^{v}_{t_n})$$

where

$$\beta^m_{t_n} = \underset{\beta \in \mathbb{R}^M}{\operatorname{argmin}} \, \mathbb{E}\left[\left| V_{\Delta t}(t_{n+1}, \underline{X}^{v}_{t_{n+1}}) - \sum_{m=1}^{M} \beta^m \phi^m(\underline{X}^{v}_{t_n}) \right|^2\right]$$

- Explicit expression:

$$\beta^m_{t_n} = \mathbb{E}[\phi(\underline{X}^{v}_{t_n})\phi(\underline{X}^{v}_{t_n})^\top]^{-1}\, \mathbb{E}[V_{\Delta t}(t_{n+1}, \underline{X}^{v}_{t_{n+1}})\phi(\underline{X}^{v}_{t_n})]$$

- Estimation with N_{MC} Monte Carlo samples:

$$\mathbb{E}[\phi(\underline{X}^{\ell,v}_{t_n})\phi(\underline{X}^{\ell,v}_{t_n})^\top] \approx \frac{1}{N_{MC}} \sum_{\ell=1}^{N_{MC}} \phi(\underline{X}^{\ell,v}_{t_n})\phi(\underline{X}^{\ell,v}_{t_n})^\top$$

and

$$\mathbb{E}[V_{\Delta t}(t_{n+1}, \underline{X}^{\ell,v}_{t_{n+1}})\phi(\underline{X}^{\ell,v}_{t_n})] \approx \frac{1}{N_{MC}} \sum_{\ell=1}^{N_{MC}} V_{\Delta t}(t_{n+1}, \underline{X}^{\ell,v}_{t_{n+1}})\phi(\underline{X}^{\ell,v}_{t_n})$$

with training set $\left\{\left(\underline{X}^{\ell,v}_{t_n}, \underline{X}^{\ell,v}_{t_{n+1}}\right); \ell = 1, \ldots, N_{MC}\right\}$

Estimation Method 2: Quantization

- Two space discretizations:
  - Set of points Γ on which we want to approximate $V_{\Delta t}$; projection Π_Γ
  - Quantization of noise (see e.g. [Pagès'18][5]):
    - ★ Set of cells $\mathcal{C}_Q = \{C_j; j = 1, \ldots, J_Q\}$
    - ★ Associated grid points $\mathcal{G}_Q = \{\zeta_j; j = 1, \ldots, J_Q\}$
    - ★ Weights for Gaussian r.v. $\Delta\mathbb{W} \sim \mathcal{N}(0, \Delta t)$: $p_j = \mathbb{P}(\Delta\mathbb{W} \in C_j)$
    - ★ Discrete version: $\Delta\hat{\mathbb{W}} \in \mathcal{G}_Q$: $\mathbb{P}(\Delta\hat{\mathbb{W}} = \zeta_j) = p_j$
    - ★ Can be optimized[6]; particularly helpful when $d > 1$

[5] Pagès, G. (2018). Numerical probability. In Universitext. Springer Cham.

[6] Optimal grids/weights available here: `http://www.quantize.maths-fi.com`

Estimation Method 2: Quantization

- Two space discretizations:
  - Set of points Γ on which we want to approximate $V_{\Delta t}$; projection Π_Γ
  - Quantization of noise (see e.g. [Pagès'18][5]):
    - ★ Set of cells $\mathcal{C}_Q = \{C_j; j = 1, \ldots, J_Q\}$
    - ★ Associated grid points $\mathcal{G}_Q = \{\zeta_j; j = 1, \ldots, J_Q\}$
    - ★ Weights for Gaussian r.v. $\Delta\mathbb{W} \sim \mathcal{N}(0, \Delta t)$: $p_j = \mathbb{P}(\Delta\mathbb{W} \in C_j)$
    - ★ Discrete version: $\Delta\hat{\mathbb{W}} \in \mathcal{G}_Q$: $\mathbb{P}(\Delta\hat{\mathbb{W}} = \zeta_j) = p_j$
    - ★ Can be optimized[6]; particularly helpful when $d > 1$
- Estimation with piecewise constant interpolation: $\bar{V}_{\Delta t} : \{t_0, \ldots, t_{N_t}\} \times \Gamma \to \mathbb{R}$

$$\mathbb{E}\left[V_{\Delta t}(t_{n+1}, \underline{X}^v_{t_{n+1}}) \,|\, \underline{X}^v_{t_n} = \underline{x}\right] \approx \sum_{j=1}^{J_Q} p_j \bar{V}_{\Delta t}\Big(t_{n+1}, \Pi_\Gamma\Big(\mathcal{B}(\underline{x}, v_{t_n})\Delta t + \Sigma\zeta_j\Big)\Big)$$

for all $\underline{x} \in \Gamma$

[5] Pagès, G. (2018). Numerical probability. In Universitext. Springer Cham.

[6] Optimal grids/weights available here: `http://www.quantize.maths-fi.com`

Estimation Method 2: Quantization

- Two space discretizations:
  - Set of points Γ on which we want to approximate $V_{\Delta t}$; projection Π_Γ
  - Quantization of noise (see e.g. [Pagès'18][5]):
    - ★ Set of cells $\mathcal{C}_Q = \{C_j; j = 1, \ldots, J_Q\}$
    - ★ Associated grid points $\mathcal{G}_Q = \{\zeta_j; j = 1, \ldots, J_Q\}$
    - ★ Weights for Gaussian r.v. $\Delta\mathbb{W} \sim \mathcal{N}(0, \Delta t)$: $p_j = \mathbb{P}(\Delta\mathbb{W} \in C_j)$
    - ★ Discrete version: $\Delta\hat{\mathbb{W}} \in \mathcal{G}_Q$: $\mathbb{P}(\Delta\hat{\mathbb{W}} = \zeta_j) = p_j$
    - ★ Can be optimized[6]; particularly helpful when $d > 1$
- Estimation with piecewise constant interpolation: $\bar{V}_{\Delta t} : \{t_0, \ldots, t_{N_t}\} \times \Gamma \to \mathbb{R}$

$$\mathbb{E}\left[V_{\Delta t}(t_{n+1}, \underline{X}^v_{t_{n+1}}) \,|\, \underline{X}^v_{t_n} = \underline{x}\right] \approx \sum_{j=1}^{J_Q} p_j \bar{V}_{\Delta t}\Big(t_{n+1}, \Pi_\Gamma\Big(\mathcal{B}(\underline{x}, v_{t_n})\Delta t + \Sigma\zeta_j\Big)\Big)$$

  for all $\underline{x} \in \Gamma$
- Other interpolations are possible

[5] Pagès, G. (2018). Numerical probability. In Universitext. Springer Cham.

[6] Optimal grids/weights available here: `http://www.quantize.maths-fi.com`

Estimation Method 2: Quantization

- Two space discretizations:
  - Set of points Γ on which we want to approximate $V_{\Delta t}$; projection Π_Γ
  - Quantization of noise (see e.g. [Pagès'18][5]):
    - ★ Set of cells $\mathcal{C}_Q = \{C_j; j = 1, \dots, J_Q\}$
    - ★ Associated grid points $\mathcal{G}_Q = \{\zeta_j; j = 1, \dots, J_Q\}$
    - ★ Weights for Gaussian r.v. $\Delta\mathbb{W} \sim \mathcal{N}(0, \Delta t)$: $p_j = \mathbb{P}(\Delta\mathbb{W} \in C_j)$
    - ★ Discrete version: $\Delta\hat{\mathbb{W}} \in \mathcal{G}_Q$: $\mathbb{P}(\Delta\hat{\mathbb{W}} = \zeta_j) = p_j$
    - ★ Can be optimized[6]; particularly helpful when $d > 1$
- Estimation with piecewise constant interpolation: $\bar{V}_{\Delta t} : \{t_0, \dots, t_{N_t}\} \times \Gamma \to \mathbb{R}$

$$\mathbb{E}\left[V_{\Delta t}(t_{n+1}, \underline{X}^v_{t_{n+1}}) \,|\, \underline{X}^v_{t_n} = \underline{x}\right] \approx \sum_{j=1}^{J_Q} p_j \bar{V}_{\Delta t}\Big(t_{n+1}, \Pi_\Gamma\Big(\mathcal{B}(\underline{x}, v_{t_n})\Delta t + \Sigma\zeta_j\Big)\Big)$$

  for all $\underline{x} \in \Gamma$
- Other interpolations are possible

For more details and numerical examples, see [Balata et al.'19]

[5] Pagès, G. (2018). Numerical probability. In Universitext. Springer Cham.

[6] Optimal grids/weights available here: `http://www.quantize.maths-fi.com`

Summary

Numerical Methods for MFG: Some references

Methods based on a deterministic approach:

- Finite diff. & Newton meth.: [Achdou, Capuzzo-Dolcetta'10; Achdou, Capuzzo-Dolcetta, Camilli'13; . . .]
- Gradient descent: [L., Pironneau'14; Pfeiffer'16]
- Semi-Lagrangian scheme: [Carlini, Silva'14; Carlini, Silva'15]
- Augmented Lagrangian & ADMM: [Benamou, Carlier'14; Achdou, L.'16; Andreev'17]
- Primal-dual algo.: [Briceño-Arias, Kalise, Silva'18; BAKS + Kobeissi, L., Mateos González'18]
- Monotone operators: [Almulla *et al.*'17; Gomes, Saúde'18; Gomes, Yang'18]

Methods based on a probabilistic approach:

- Cubature: [Chaudru de Raynal, Garcia Trillos'15]
- Recursion: [Chassagneux *et al.*'17; Angiuli *et al.*'18]
- MC & Regression: [Balata, Huré, L., Pham, Pimentel'18]

Surveys and lecture notes: [Achdou'13 (LNM); Achdou, L.'20 (Cetraro); L.'21 (AMS)]

Numerical Methods for MFG: Some references

Methods based on a deterministic approach:

- Finite diff. & Newton meth.: [Achdou, Capuzzo-Dolcetta'10; Achdou, Capuzzo-Dolcetta, Camilli'13; . . .]
- Gradient descent: [L., Pironneau'14; Pfeiffer'16]
- Semi-Lagrangian scheme: [Carlini, Silva'14; Carlini, Silva'15]
- Augmented Lagrangian & ADMM: [Benamou, Carlier'14; Achdou, L.'16; Andreev'17]
- Primal-dual algo.: [Briceño-Arias, Kalise, Silva'18; BAKS + Kobeissi, L., Mateos González'18]
- Monotone operators: [Almulla *et al.*'17; Gomes, Saúde'18; Gomes, Yang'18]

Methods based on a probabilistic approach:

- Cubature: [Chaudru de Raynal, Garcia Trillos'15]
- Recursion: [Chassagneux *et al.*'17; Angiuli *et al.*'18]
- MC & Regression: [Balata, Huré, L., Pham, Pimentel'18]

Surveys and lecture notes: [Achdou'13 (LNM); Achdou, L.'20 (Cetraro); L.'21 (AMS)]

Limitations:

- dimensionality (typically: state in dimension ≤ 3)
- structure of the problem (typically: simple costs, dynamics and noises)

Numerical Methods for MFG: Some references

Methods based on a deterministic approach:

- Finite diff. & Newton meth.: [Achdou, Capuzzo-Dolcetta'10; Achdou, Capuzzo-Dolcetta, Camilli'13; ...]
- Gradient descent: [L., Pironneau'14; Pfeiffer'16]
- Semi-Lagrangian scheme: [Carlini, Silva'14; Carlini, Silva'15]
- Augmented Lagrangian & ADMM: [Benamou, Carlier'14; Achdou, L.'16; Andreev'17]
- Primal-dual algo.: [Briceño-Arias, Kalise, Silva'18; BAKS + Kobeissi, L., Mateos González'18]
- Monotone operators: [Almulla *et al.*'17; Gomes, Saúde'18; Gomes, Yang'18]

Methods based on a probabilistic approach:

- Cubature: [Chaudru de Raynal, Garcia Trillos'15]
- Recursion: [Chassagneux *et al.*'17; Angiuli *et al.*'18]
- MC & Regression: [Balata, Huré, L., Pham, Pimentel'18]

Surveys and lecture notes: [Achdou'13 (LNM); Achdou, L.'20 (Cetraro); L.'21 (AMS)]

Limitations:

- dimensionality (typically: state in dimension ≤ 3)
- structure of the problem (typically: simple costs, dynamics and noises)

Recent progress: extending the toolbox with tools from **machine learning**:

- approximation without a grid (mesh-free methods): **opt. control** & **distribution**
  $\rightarrow$ [Carmona, L.; Al-Aradi *et al.*; Fouque *et al.*; Germain *et al.*; Ruthotto *et al.*; Agram *et al.*; ...]
- even when the **dynamics / cost are not known** (model-free methods)
  $\rightarrow$ [Guo *et al.*; Subramanian *et al.*; Elie *et al.*; Carmona *et al.*; Pham *et al.*; Yang *et al.*; ...]

Some Code Samples

- ODE solvers for LQ MFC:

  `https://colab.research.google.com/drive/1jac1M1zFBlY6j6BY1ocwgmkNTflpRQYY?usp=sharing`

- PDE solver with Semi-Lagrangian approach

  `https://colab.research.google.com/drive/18Oj6cKlvfe5UlMnm_Lm0klyuYrJKc0k4?usp=sharing`

- PDE solver with Finite Difference scheme & Picard iterations + Newton

  `(coming soon)`

- ...