Deep Learning for Stackelberg Mean Field Games via Single-Level Reformulation

Mathieu Laurière
NYU Shanghai, Mathematics & Data Science

joint work with Gökçe Dayanıklı

Karlstad Applied Analysis Seminar
November 19, 2024

Motivation

Goal: A principal wants to design optimal policies to get the best outcomes from a large population of agents who prioritize their own objectives

Some examples:

→ **Systemic risk:** A **regulator** incentivizes **large number of banks** borrowing and lending from each other to minimize the expected number of defaults.

→ **Contract theory:** An **employer** (principal) writes a payment contract for a **large number of employees** to maximize their expected return.

→ **Carbon emissions:** A **regulator** wants to find optimal carbon tax levels for **electricity producers** to attain the targeted reduction in the carbon emission levels.

→ **Advertisement:** A **company** wants to optimize its advertisement strategies while interacting with **consumers** to maximize their profits.

→ **Management of epidemics:** A **government** chooses nonpharmaceutical policies to mitigate an epidemic in a **country**.

Outline of this Talk

→ Brief Review of **Stochastic Optimal Control** & Solving it with Deep Learning

Outline of this Talk

→ Brief Review of **Stochastic Optimal Control** & Solving it with Deep Learning

→ Nash Equilibrium in Large Populations
  → Approximating Nash Equilibrium for Large Populations: **Mean Field Games**
  → Deep Learning for Solving Mean Field Games

Outline of this Talk

- → Brief Review of **Stochastic Optimal Control** & Solving it with Deep Learning
- → Nash Equilibrium in Large Populations
  - → Approximating Nash Equilibrium for Large Populations: **Mean Field Games**
  - → Deep Learning for Solving Mean Field Games
- → Stackelberg Equilibrium
  - → Introduction to Stackelberg Equilibrium
  - → Optimal Policies for Large Populations: **Stackelberg Mean Field Games**
  - → Rewriting Bi-level Stackelberg Mean Field Game Problem as a Single-level Problem
  - → Single-level Deep Learning for Solving Stackelberg Mean Field Games
  - → Numerical Examples

Brief Review of
Stochastic Optimal Control Problems

Stochastic Optimal Control Problems

We have 1 agent.

She chooses her control to minimize her expected costs (or maximize her rewards) between time $t = 0$ and $t = T$.

She has:

→ State: $(X_t)_{t \in [0,T]}$

→ Control: $(\alpha_t)_{t \in [0,T]}$

→ Objectives: running cost & terminal cost

Stochastic Optimal Control Problems

We have 1 agent.

She chooses her control to minimize her expected costs (or maximize her rewards) between time $t = 0$ and $t = T$.

She has:

- $\rightarrow$ State: $(X_t)_{t \in [0,T]}$
- $\rightarrow$ Control: $(\alpha_t)_{t \in [0,T]}$
- $\rightarrow$ Objectives: running cost & terminal cost

Example: The agent works in a company and she chooses her effort level that affects the value of the project she is working on:

- $\rightarrow$ X_t: Value of the project at time t
- $\rightarrow$ α_t: Effort level at time t
- $\rightarrow$ Objectives: effort's cost & utility from the value of the project

Stochastic Optimal Control Problems: Mathematical Formulation (I)

Example: The agent works in a company and she chooses her effort level that affects the value of the project she is working on.

Mathematical Formulation:

$$\min_{(\alpha_t)_t} \mathbb{E}\Big[\int_0^T \underbrace{\Big(c_1\alpha_t^2 - c_2 U(X_t)\Big)}_{\text{Running Cost}} dt \underbrace{-c_3 U(X_T)}_{\text{Terminal Cost}}\Big]$$

$$dX_t = \underbrace{\alpha_t}_{\text{Drift}} dt + \sigma dW_t, \quad X_0 = \zeta$$

- $\rightarrow$ $U(\cdot)$ is a utility function
- $\rightarrow$ c_1, c_2, c_3, σ are positive constants (weights)
- $\rightarrow$ W_t is the Brownian motion
- $\rightarrow$ $\zeta \sim \mu_0$ is the initial condition

Stochastic Optimal Control Problems: Mathematical Formulation (II)

Agent's problem:

$$\min_{(\alpha_t)_t} \mathbb{E}\Big[\int_0^T \underbrace{\Big(c_1 \alpha_t^2 - c_2 U(X_t)\Big)}_{\text{Running Cost}} dt \underbrace{- c_3 U(X_T)}_{\text{Terminal Cost}} \Big]$$

$$dX_t = \underbrace{\alpha_t}_{\text{Drift}} dt + \sigma dW_t, \quad X_0 = \zeta$$

More generally: stochastic optimal control (SOC) problem:

$$\min_{(\alpha_t)_t} \mathbb{E}\Big[\int_0^T \underbrace{f(t, X_t, \alpha_t)}_{\text{Running Cost}} dt + \underbrace{g(X_T)}_{\text{Terminal Cost}} \Big]$$

$$dX_t = \underbrace{b(t, X_t, \alpha_t)}_{\text{Drift}} dt + \sigma dW_t, \quad X_0 = \zeta$$

Stochastic Optimal Control Problems: Mathematical Formulation (II)

Agent's problem:

$$\min_{(\alpha_t)_t} \mathbb{E}\Big[\int_0^T \underbrace{\Big(c_1 \alpha_t^2 - c_2 U(X_t)\Big)}_{\text{Running Cost}} dt \underbrace{-c_3 U(X_T)}_{\text{Terminal Cost}} \Big]$$

$$dX_t = \underbrace{\alpha_t}_{\text{Drift}} dt + \sigma dW_t, \quad X_0 = \zeta$$

More generally: stochastic optimal control (SOC) problem:

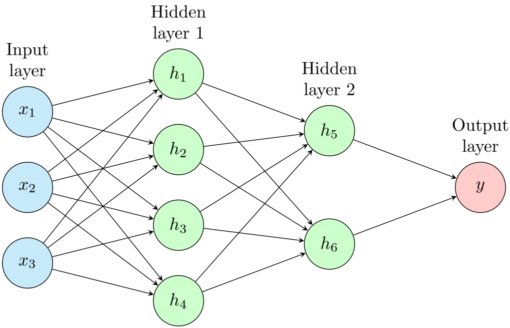
$$\min_{(\alpha_t)_t} \mathbb{E}\Big[\int_0^T \underbrace{f(t, X_t, \alpha_t)}_{\text{Running Cost}} dt + \underbrace{g(X_T)}_{\text{Terminal Cost}} \Big]$$

$$dX_t = \underbrace{b(t, X_t, \alpha_t)}_{\text{Drift}} dt + \sigma dW_t, \quad X_0 = \zeta$$

Later: several interacting agents; not just SOC but game theory.

Using Deep Learning to Solve Stochastic Optimal Control Problems

Neural Networks as Function Approximators



→ Neural networks (NNs) can be used to approximate functions

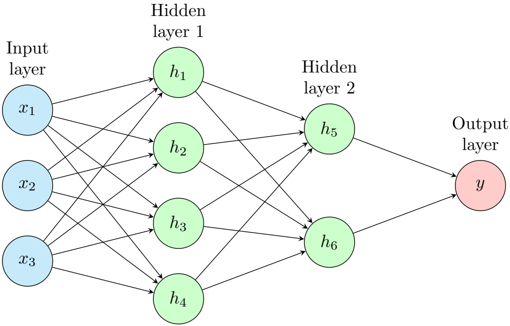
→ Empirically efficient in **high dimension**

→ Provably breaks the curse of dimensionality in some cases

→ Ex.: **Regression**: To approximate a function $f(x)$, we can use a NN that outputs $f_\theta(x)$ and train it (i.e., adjust θ) to minimize the loss given by the MSE:

$$L(\theta) = \mathbb{E}|f(x) - f_\theta(x)|^2$$

Neural Networks as Function Approximators



→ Neural networks (NNs) can be used to approximate functions

→ Empirically efficient in **high dimension**

→ Provably breaks the curse of dimensionality in some cases

→ Ex.: **Regression**: To approximate a function $f(x)$, we can use a NN that outputs $f_\theta(x)$ and train it (i.e., adjust θ) to minimize the loss given by the MSE:

$$L(\theta) = \mathbb{E}|f(x) - f_\theta(x)|^2$$

→ In the sequel, we will use NN to minimize other **loss functions** $L(\theta)$

Deep Learning for Stochastic Optimal Control Problem

SOC problem:

$$\min_{(\alpha_t)_t} \mathbb{E}\Big[\int_0^T f(t, X_t, \alpha_t)dt + g(X_T)\Big]$$

$$dX_t = b(t, X_t, \alpha_t)dt + \sigma dW_t, \quad X_0 \sim \mu_0$$

Numerical approach with deep learning:

→ Consider the control as a function of time and the current state: $\alpha_t = \varphi(t, X_t)$

→ Use NN approximation $\varphi_\theta(t, X_t)$ for the control function

[1] Similar to Han & E (2016), extended to MFC problems in Carmona, Laurière (2022) and Dayanıklı, Laurière, Zhang (2023). See Hu, R., & Laurière, M. (2022) for a survey.

Deep Learning for Stochastic Optimal Control Problem

SOC problem:

$$\min_{(\alpha_t)_t} \mathbb{E}\Big[\int_0^T f(t, X_t, \alpha_t)dt + g(X_T)\Big]$$

$$dX_t = b(t, X_t, \alpha_t)dt + \sigma dW_t, \quad X_0 \sim \mu_0$$

Numerical approach with deep learning:

→ Consider the control as a function of time and the current state: $\alpha_t = \varphi(t, X_t)$

→ Use NN approximation $\varphi_\theta(t, X_t)$ for the control function

→ Discretize the time: $t = \{0, \Delta t, 2\Delta t, \ldots, n\Delta t\}$, where $T = n\Delta t$:

$$L(\theta) = \mathbb{E}\Big[\sum_t f(t, X_t, \varphi_\theta(t, X_t)) \times \Delta t + g(X_T)\Big]$$

$$X_{t+\Delta t} = b(t, X_t, \varphi_\theta(t, X_t)) \times \Delta t + \sigma W_{\Delta t}, \quad X_0 \sim \mu_0$$

[1] Similar to Han & E (2016), extended to MFC problems in Carmona, Laurière (2022) and Dayanıklı, Laurière, Zhang (2023). See Hu, R., & Laurière, M. (2022) for a survey.

Deep Learning for Stochastic Optimal Control Problem

SOC problem:

$$\min_{(\alpha_t)_t} \mathbb{E}\Big[\int_0^T f(t, X_t, \alpha_t)dt + g(X_T)\Big]$$

$$dX_t = b(t, X_t, \alpha_t)dt + \sigma dW_t, \quad X_0 \sim \mu_0$$

Numerical approach with deep learning:

→ Consider the control as a function of time and the current state: $\alpha_t = \varphi(t, X_t)$

→ Use NN approximation $\varphi_\theta(t, X_t)$ for the control function

→ Discretize the time: $t = \{0, \Delta t, 2\Delta t, \ldots, n\Delta t\}$, where $T = n\Delta t$:

$$L(\theta) = \mathbb{E}\Big[\sum_t f(t, X_t, \varphi_\theta(t, X_t)) \times \Delta t + g(X_T)\Big]$$

$$X_{t+\Delta t} = b(t, X_t, \varphi_\theta(t, X_t)) \times \Delta t + \sigma W_{\Delta t}, \quad X_0 \sim \mu_0$$

→ Sample X_0 and Brownian motion increments; simulate a trajectory

→ Train to minimize the loss (cost) $L(\theta)$ over the parameters θ. [1]

We want to use deep learning to solve more complex problems.

[1] Similar to Han & E (2016), extended to MFC problems in Carmona, Laurière (2022) and Dayanıklı, Laurière, Zhang (2023). See Hu, R., & Laurière, M. (2022) for a survey.

Nash Equilibrium in Large Populations

Overview of the Approach: Mean Field Game

One of the most studied solution concept in game theory: **Nash equilibrium**.

In this talk: Dynamic, stochastic, continuous time, (possibly) continuous space.

→ **Challenge:** Large number N of agents.

→ **Approach:** Approximate the game with a Mean Field Game.

[2] Huang-Malhamé-Caines (2006), Lasry-Lions (2006).
Image credit: https://gbxglobal.org/the-importance-of-the-network/

Overview of the Approach: Mean Field Game

One of the most studied solution concept in game theory: **Nash equilibrium**.

In this talk: Dynamic, stochastic, continuous time, (possibly) continuous space.

→ **Challenge:** Large number N of agents.

→ **Approach:** Approximate the game with a Mean Field Game.

In Mean Field Games (MFGs):[2]

→ Assume $N \to \infty$.

→ Agents are identical and infinitesimal.

→ Agents interact through the distribution.

→ **Idea:** Focus on

- a representative agent
- and her interactions with the distribution

[2] Huang-Malhamé-Caines (2006), Lasry-Lions (2006).
Image credit: https://gbxglobal.org/the-importance-of-the-network/

Mathematical Formulation of Mean Field Game

The **cost** for the **representative agent** using control $\boldsymbol{\alpha} \in \mathbb{A}$ when facing a population with state distribution $\boldsymbol{\mu}$ is

$$J(\boldsymbol{\alpha}; \boldsymbol{\mu}) := \mathbb{E}\left[\int_0^T \underbrace{f(t, X_t, \alpha_t, \mu_t)}_{\text{Running Cost}} dt + \underbrace{g(X_T, \mu_T)}_{\text{Terminal Cost}}\right].$$

The agent's state X_t has the following dynamics:

$$dX_t = \underbrace{b(t, X_t, \alpha_t, \mu_t)}_{\text{Drift}} dt + \sigma dW_t, \qquad X_0 = \zeta \sim \mu_0.$$

Mathematical Formulation of Mean Field Game

The **cost** for the **representative agent** using control $\boldsymbol{\alpha} \in \mathbb{A}$ when facing a population with state distribution $\boldsymbol{\mu}$ is

$$J(\boldsymbol{\alpha}; \boldsymbol{\mu}) := \mathbb{E}\left[\int_0^T \underbrace{f(t, X_t, \alpha_t, \mu_t)}_{\text{Running Cost}} dt + \underbrace{g(X_T, \mu_T)}_{\text{Terminal Cost}}\right].$$

The agent's state X_t has the following dynamics:

$$dX_t = \underbrace{b(t, X_t, \alpha_t, \mu_t)}_{\text{Drift}} dt + \sigma dW_t, \qquad X_0 = \zeta \sim \mu_0.$$

Definition: The pair $(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\mu}})$ is a **Mean Field Game Nash equilibrium** if it satisfies:

(i) $\hat{\boldsymbol{\alpha}}$ minimizes the cost of representative agent given population distribution $\hat{\boldsymbol{\mu}}$;

(ii) $\forall t \in [0, T]$, $\hat{\mu}_t$ is the distribution of the representative agent's state X_t.

Mathematical Formulation of Mean Field Game

The **cost** for the **representative agent** using control $\boldsymbol{\alpha} \in \mathbb{A}$ when facing a population with state distribution $\boldsymbol{\mu}$ is

$$J(\boldsymbol{\alpha};\boldsymbol{\mu}) := \mathbb{E}\left[\int_0^T \underbrace{f(t, X_t, \alpha_t, \mu_t)}_{\text{Running Cost}}\, dt + \underbrace{g(X_T, \mu_T)}_{\text{Terminal Cost}}\right].$$

The agent's state X_t has the following dynamics:

$$dX_t = \underbrace{b(t, X_t, \alpha_t, \mu_t)}_{\text{Drift}}\, dt + \sigma dW_t, \qquad X_0 = \zeta \sim \mu_0.$$

Definition: The pair $(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\mu}})$ is a **Mean Field Game Nash equilibrium** if it satisfies:

(i) $\hat{\boldsymbol{\alpha}}$ minimizes the cost of representative agent given population distribution $\hat{\boldsymbol{\mu}}$;

(ii) $\forall t \in [0, T]$, $\hat{\mu}_t$ is the distribution of the representative agent's state X_t.

It can be characterized by a **forward-backward stochastic differential equation** (FBSDE) system of McKean-Vlasov (MKV) type.

Mean Field Game Example

→ Instead of 1 agent: there is a large population of agents.

→ Each agent

- chooses her effort level
- aims at minimizing their total cost
- interacts with other agents through the average project value

$$\min_{(\alpha_t)_t} \mathbb{E}\Big[\int_0^T \underbrace{\left(\frac{1}{2}\alpha_t^2 - U(X_t) \right)}_{\text{Running Cost}} dt + G(X_T) \Big]$$

$$dX_t = \underbrace{(\alpha_t + \bar{X}_t)}_{\text{Drift}} dt + \sigma dW_t, \quad X_0 = \zeta$$

Mean Field Game Example

→ Instead of 1 agent: there is a large population of agents.

→ Each agent

- chooses her effort level
- aims at minimizing their total cost
- interacts with other agents through the average project value

$$\min_{(\alpha_t)_t} \mathbb{E}\Big[\int_0^T \underbrace{\left(\frac{1}{2}\alpha_t^2 - U(X_t) \right)}_{\text{Running Cost}} dt + G(X_T) \Big]$$

$$dX_t = \underbrace{(\alpha_t + \bar{X}_t)}_{\text{Drift}} dt + \sigma dW_t, \quad X_0 = \zeta$$

The **Nash equilibrium** control is

$$\hat{\alpha}_t = -\tfrac{1}{\sigma} Z_t$$

where $(X_t, Y_t, Z_t)_t$ solves the FBSDE:

$$\begin{aligned} dX_t &= (-Z_t/\sigma + \bar{X}_t)dt + \sigma dW_t, & X_0 &= \zeta \\ dY_t &= \left(\frac{1}{2\sigma^2} Z_t^2 - U(X_t) \right) dt + Z_t dW_t, & Y_T &= G(X_T). \end{aligned}$$

Using Deep Learning to Solve Mean Field Games

Using Deep Learning to Find Mean Field Nash Equilibrium (1/3)

There are various MFG numerical methods (finite diff. schemes, ML methods, . . .). [3]

[3] See e.g. Achdou & Laurière (2020) and Laurière (2021) for surveys.

Using Deep Learning to Find Mean Field Nash Equilibrium (1/3)

There are various MFG numerical methods (finite diff. schemes, ML methods, . . .). [3]

Here, we want to solve the **FBSDE** that characterizes the mean field Nash equilibrium:

$\rightarrow$ **Challenges:** Coupled, McKean-Vlasov (interactions through the law)

[3] See e.g. Achdou & Laurière (2020) and Laurière (2021) for surveys.

Using Deep Learning to Find Mean Field Nash Equilibrium (1/3)

There are various MFG numerical methods (finite diff. schemes, ML methods, ...). [3]

Here, we want to solve the **FBSDE** that characterizes the mean field Nash equilibrium:

→ **Challenges:** Coupled, McKean-Vlasov (interactions through the law)

→ Y_t represents the value function of a representative player (i.e., the minimized expected cost between time t and T when the player starts from $x = X_t$ and the population follows the equilibrium).

$$\text{State dynamics} \leftarrow X_t = \zeta + \int_0^t b(s, X_s, \hat{\alpha}_s, \mu_s) ds + \int_0^t \sigma dW_s$$

$$\text{Value function} \leftarrow Y_t = g(X_T, \mu_T) + \int_t^T f(s, X_s, \hat{\alpha}_s, \mu_s) ds - \int_t^T Z_s dW_s$$

where $\mu_t = \mathcal{L}(X_t)$ and $\hat{\alpha}_s = \hat{\alpha}_s(X_s, \mu_s, Z_s)$.

[3] See e.g. Achdou & Laurière (2020) and Laurière (2021) for surveys.

Using Deep Learning to Find Mean Field Nash Equilibrium (2/3)

In order to solve the coupled FBSDE, we are going to use a shooting method:[4]

→ Instead of:

$$\text{Value function} \leftarrow Y_t = g(X_T, \mu_T) + \int_t^T f(s, X_s, \hat{\alpha}_s, \mu_s) ds - \int_t^T Z_s dW_s$$

→ We write:

$$Y_t = Y_0 - \int_0^t f(s, X_s, \hat{\alpha}_s, \mu_s) ds + \int_0^t Z_s dW_s$$

→ **Goal:** Find Y_0 and $(Z_t)_t$ s.t. the terminal condition $Y_T = g(X_T, \mu_T)$ is satisfied

[4] Han, Jentzen, E (2019); extended to McKean-Vlasov FBSDEs in Carmona, Laurière (2022).

Using Deep Learning to Find Mean Field Nash Equilibrium (2/3)

In order to solve the coupled FBSDE, we are going to use a shooting method:[4]

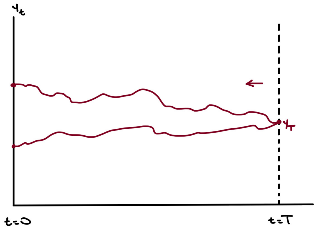
→ Instead of:

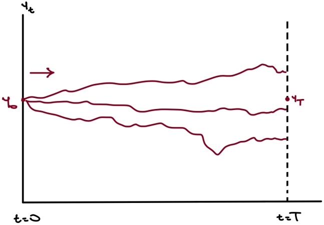
$$\text{Value function} \leftarrow Y_t = g(X_T, \mu_T) + \int_t^T f(s, X_s, \hat{\alpha}_s, \mu_s)ds - \int_t^T Z_s dW_s$$

→ We write:

$$Y_t = Y_0 - \int_0^t f(s, X_s, \hat{\alpha}_s, \mu_s)ds + \int_0^t Z_s dW_s$$

→ **Goal:** Find Y_0 and $(Z_t)_t$ s.t. the terminal condition $Y_T = g(X_T, \mu_T)$ is satisfied





[4]Han, Jentzen, E (2019); extended to McKean-Vlasov FBSDEs in Carmona, Laurière (2022).

Using Deep Learning to Find Mean Field Nash Equilibrium (3/3)

→ Now we have forward-forward SDEs:

$$X_t = \zeta + \int_0^t b(s, X_s, \hat{\alpha}_s, \mu_s) ds + \int_0^t \sigma dW_s$$
$$Y_t = Y_0 - \int_0^t f(s, X_s, \hat{\alpha}_s, \mu_s) ds + \int_0^t Z_s dW_s$$

where $\mu_t = \mathcal{L}(X_t)$ and $\hat{\alpha}_s = \hat{\alpha}_s(X_s, \mu_s, Z_s)$ and we need to shoot $Y_T = g(X_T, \mu_T)$.

Application in finite state MFG: Aurell-Carmona-Dayanıklı-Laurière (2022a)

Graphon game application: Aurell-Carmona-Dayanıklı-Laurière (2022b)

Using Deep Learning to Find Mean Field Nash Equilibrium (3/3)

→ Now we have forward-forward SDEs:

$$X_t = \zeta + \int_0^t b(s, X_s, \hat{\alpha}_s, \mu_s) ds + \int_0^t \sigma dW_s$$
$$Y_t = Y_0 - \int_0^t f(s, X_s, \hat{\alpha}_s, \mu_s) ds + \int_0^t Z_s dW_s$$

where $\mu_t = \mathcal{L}(X_t)$ and $\hat{\alpha}_s = \hat{\alpha}_s(X_s, \mu_s, Z_s)$ and we need to shoot $Y_T = g(X_T, \mu_T)$.

→ **As for SOC:** Discretize the time.

→ **But here:**

- There is a **distribution**: we approximate it by an empirical distribution μ^N, obtained by simulating a system of N particles: $(X_t^i, Y_t^i)_{t\in[0,T], i\in[N]}$
- The **controls** are: $Y_0 = y_{0,\theta_1}(X_0)$ and $Z_t = z_{\theta_2}(t, X_t)$
- The goal is to **shoot** the **terminal condition**: $Y_T = g(X_T, \mu_T)$.

Application in finite state MFG: Aurell-Carmona-Dayanıklı-Laurière (2022a)

Graphon game application: Aurell-Carmona-Dayanıklı-Laurière (2022b)

Using Deep Learning to Find Mean Field Nash Equilibrium (3/3)

→ Now we have forward-forward SDEs:

$$X_t = \zeta + \int_0^t b(s, X_s, \hat{\alpha}_s, \mu_s) ds + \int_0^t \sigma dW_s$$

$$Y_t = Y_0 - \int_0^t f(s, X_s, \hat{\alpha}_s, \mu_s) ds + \int_0^t Z_s dW_s$$

where $\mu_t = \mathcal{L}(X_t)$ and $\hat{\alpha}_s = \hat{\alpha}_s(X_s, \mu_s, Z_s)$ and we need to shoot $Y_T = g(X_T, \mu_T)$.

→ **As for SOC:** Discretize the time.

→ **But here:**

- There is a **distribution**: we approximate it by an empirical distribution μ^N, obtained by simulating a system of N particles: $(X_t^i, Y_t^i)_{t\in[0,T], i\in[N]}$
- The **controls** are: $Y_0 = y_{0,\theta_1}(X_0)$ and $Z_t = z_{\theta_2}(t, X_t)$
- The goal is to **shoot** the **terminal condition**: $Y_T = g(X_T, \mu_T)$.

→ The problem is to minimize over $\boldsymbol{\theta} = (\theta_1, \theta_2)$ the loss:

$$\boxed{L(\boldsymbol{\theta}) = \frac{1}{N}\mathbb{E}\sum_{i=1}^{N}\left(Y_T^{i,\boldsymbol{\theta}} - g(X_T^{i,\boldsymbol{\theta}}, \mu_T^{N,\boldsymbol{\theta}})\right)^2}$$

Application in finite state MFG: Aurell-Carmona-Dayanıklı-Laurière (2022a)

Graphon game application: Aurell-Carmona-Dayanıklı-Laurière (2022b)

Stackelberg Equilibrium &
Stackelberg Mean Field Games

What is a Stackelberg Equilibrium?

Our aim is to design optimal policies/incentives in order to get the best outcomes when we interact with many rational agents who prioritize their own.

- $\rightarrow$ There is a leader (principal) and a follower (agent).
- $\rightarrow$ The leader chooses incentives.
- $\rightarrow$ The follower gives their best response to these incentives.
- $\rightarrow$ The leader optimizes incentives by anticipating the follower's reaction.
- $\rightarrow$ **Bi-level** optimization problem.

[5] Başar (1984, 1989), Holmström-Milgrom (1987), Sannikov (2008, 2013), Cvitanić-Possamaï-Touzi (2018) Ljungqvist, Sargent (Chapter 19: Dynamic Stackelberg Problems)

What is a Stackelberg Equilibrium?

Our aim is to design optimal policies/incentives in order to get the best outcomes when we interact with many rational agents who prioritize their own.

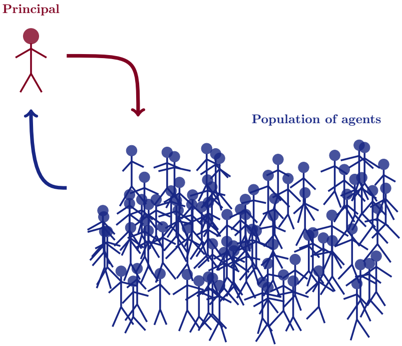
- → There is a leader (principal) and a follower (agent).
- → The leader chooses incentives.
- → The follower gives their best response to these incentives.
- → The leader optimizes incentives by anticipating the follower's reaction.
- → **Bi-level** optimization problem.
- → **Stackelberg equilibrium[5] is different from Nash Equilibrium**

[5] Başar (1984, 1989), Holmström-Milgrom (1987), Sannikov (2008, 2013), Cvitanić-Possamaï-Touzi (2018) Ljungqvist, Sargent (Chapter 19: Dynamic Stackelberg Problems)

From Stackelberg Equilibrium to Stackelberg MFG

In our setup:

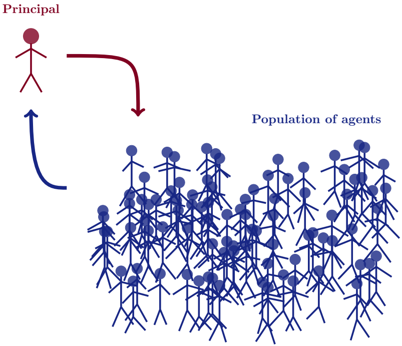
→ Not just one follower, but a **large population** of followers (agents).
→ They are **noncooperative** agents.
→ So the population of agents will be in a **Nash equilibrium**.
→ The Nash equilibrium depends on the **incentives given by the principal**.



From Stackelberg Equilibrium to Stackelberg MFG

In our setup:

- → Not just one follower, but a **large population** of followers (agents).
- → They are **noncooperative** agents.
- → So the population of agents will be in a **Nash equilibrium**.
- → The Nash equilibrium depends on the **incentives given by the principal**.



Nash eq. in the population will be approximated with a **Mean Field Game**.

Stackelberg Mean Field Games

Brief Literature Review

Some related references:

→ Contract theory models with large number of agents:

- Elie, Mastrolia, and Possamaï (2019): Continuous state space
- Carmona and Wang (2018): Finite state space
- Incentives through a terminal payment only

→ Numerical approaches:

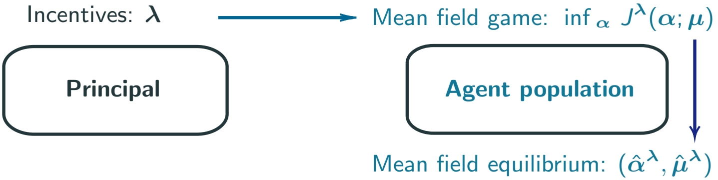
- Aurell, Carmona, Dayanıklı, and Laurière (SICON, 2022)
- Campbell, Chen, Shrivats and Jaimungal (2021)

→ **In the rest of the talk:**

- *A Machine Learning Method for Stackelberg Mean Field Games.* Dayanıklı, Laurière (2023, to appear in MOR).

See **Gökçe Dayanıklı**'s papers for more examples!

Agent Population: Mean Field Game Given Principal's Incentives



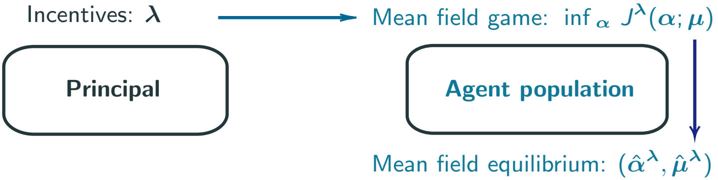
The **cost** for the **representative agent** using control $\boldsymbol{\alpha} \in \mathbb{A}$ when facing a population with state distribution $\boldsymbol{\mu}$ is

$$J^{\boldsymbol{\lambda}}(\boldsymbol{\alpha}; \boldsymbol{\mu}) := \mathbb{E}\left[\int_0^T f(t, X_t, \alpha_t, \mu_t; \lambda_t)dt + g(X_T, \mu_T; \lambda_T)\right],$$

where $\boldsymbol{\lambda}$ is incentive chosen by the principal, and the representative agent's state X_t has the following dynamics:

$$dX_t = b(t, X_t, \alpha_t, \mu_t; \lambda_t)dt + \sigma dW_t, \qquad X_0 = \zeta \sim \mu_0.$$

Agent Population: Mean Field Game Given Principal's Incentives



The **cost** for the **representative agent** using control $\boldsymbol{\alpha} \in \mathbb{A}$ when facing a population with state distribution $\boldsymbol{\mu}$ is

$$J^{\boldsymbol{\lambda}}(\boldsymbol{\alpha}; \boldsymbol{\mu}) := \mathbb{E}\left[\int_0^T f(t, X_t, \alpha_t, \mu_t; \lambda_t) dt + g(X_T, \mu_T; \lambda_T)\right],$$

where $\boldsymbol{\lambda}$ is incentive chosen by the principal, and the representative agent's state X_t has the following dynamics:

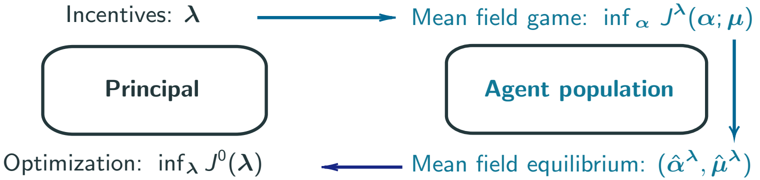
$$dX_t = b(t, X_t, \alpha_t, \mu_t; \lambda_t) dt + \sigma dW_t, \qquad X_0 = \zeta \sim \mu_0.$$

Different than before: Impact of the principal's incentive.

Given $\boldsymbol{\lambda}$, the MFG solution can still be characterized with an **FBSDE**.

Remark: Principal's incentive, λ_t, can be in the form of $\lambda(t, X_t, \mu_t)$.

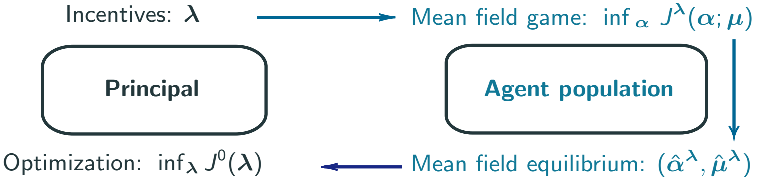
Principal: Defining Stackelberg Equilibrium



The principal's **cost** for using incentive $\boldsymbol{\lambda}$ is

$$J^0(\boldsymbol{\lambda}) := \int_0^T f_0(t, \hat{\mu}_t^{\boldsymbol{\lambda}}, \lambda_t)dt + g_0(\hat{\mu}_T^{\boldsymbol{\lambda}}, \lambda_T)$$

Principal: Defining Stackelberg Equilibrium



The principal's **cost** for using incentive $\boldsymbol{\lambda}$ is

$$J^0(\boldsymbol{\lambda}) := \int_0^T f_0(t, \hat{\mu}_t^{\boldsymbol{\lambda}}, \lambda_t)dt + g_0(\hat{\mu}_T^{\boldsymbol{\lambda}}, \lambda_T)$$

The principal's **optimization problem** is

$$\inf_{\boldsymbol{\lambda}} J^0(\boldsymbol{\lambda}).$$

subject to the constraint: the population is in MFG Nash equilibrium: $(\hat{\boldsymbol{\alpha}}^{\boldsymbol{\lambda}}, \hat{\boldsymbol{\mu}}^{\boldsymbol{\lambda}})$

Stackelberg Mean Field Game Problem

The full problem becomes:

$$\left.\inf_{\boldsymbol{\lambda}\in\boldsymbol{\Lambda}} \int_0^T \underbrace{f_0(t,\mu_t^{\boldsymbol{\lambda}},\lambda_t)}_{\text{Running cost of principal}} dt + \underbrace{g_0(\mu_T^{\boldsymbol{\lambda}},\lambda_T)}_{\text{Terminal cost of principal}}\right\} \text{Optimization of Principal}$$

$$\left.\begin{aligned}
\underbrace{X_t^{\boldsymbol{\lambda}}}_{\text{State of agent}} &= \zeta + \int_0^t \underbrace{b(s,X_s^{\boldsymbol{\lambda}},\hat{\alpha}_s^{\boldsymbol{\lambda}},\mu_s^{\boldsymbol{\lambda}};\lambda_s)}_{\text{Drift of agent}} ds + \int_0^t \sigma dW_s \\
\underbrace{Y_t^{\boldsymbol{\lambda}}}_{\text{Value function}} &= \underbrace{g(X_T^{\boldsymbol{\lambda}},\mu_T^{\boldsymbol{\lambda}};\lambda_T)}_{\text{Terminal cost of agent}} + \int_t^T \underbrace{f(s,X_s^{\boldsymbol{\lambda}},\hat{\alpha}_s^{\boldsymbol{\lambda}},\mu_s^{\boldsymbol{\lambda}};\lambda_s)}_{\text{Running cost of agent}} ds - \int_t^T Z_s dW_s
\end{aligned}\right\} \text{Equilibrium in the Population}$$

where $\mu_t^{\boldsymbol{\lambda}} = \mathcal{L}(X_t^{\boldsymbol{\lambda}})$ and $\zeta \sim \mu_0$.

Stackelberg Mean Field Game Problem

The full problem becomes:

$$\left.\inf_{\boldsymbol{\lambda}\in\boldsymbol{\Lambda}} \int_0^T \underbrace{f_0(t,\mu_t^{\boldsymbol{\lambda}},\lambda_t)}_{\text{Running cost of principal}} dt + \underbrace{g_0(\mu_T^{\boldsymbol{\lambda}},\lambda_T)}_{\text{Terminal cost of principal}} \right\} \text{Optimization of Principal}$$

$$\left.\begin{aligned} \underbrace{X_t^{\boldsymbol{\lambda}}}_{\text{State of agent}} &= \zeta + \int_0^t \underbrace{b(s,X_s^{\boldsymbol{\lambda}},\hat{\alpha}_s^{\boldsymbol{\lambda}},\mu_s^{\boldsymbol{\lambda}};\lambda_s)}_{\text{Drift of agent}} ds + \int_0^t \sigma dW_s \\ \underbrace{Y_t^{\boldsymbol{\lambda}}}_{\text{Value function}} &= \underbrace{g(X_T^{\boldsymbol{\lambda}},\mu_T^{\boldsymbol{\lambda}};\lambda_T)}_{\text{Terminal cost of agent}} + \int_t^T \underbrace{f(s,X_s^{\boldsymbol{\lambda}},\hat{\alpha}_s^{\boldsymbol{\lambda}},\mu_s^{\boldsymbol{\lambda}};\lambda_s)}_{\text{Running cost of agent}} ds - \int_t^T Z_s dW_s \end{aligned}\right\} \text{Equilibrium in the Population}$$

where $\mu_t^{\boldsymbol{\lambda}} = \mathcal{L}(X_t^{\boldsymbol{\lambda}})$ and $\zeta \sim \mu_0$.

This is a bi-level problem!

$\rightarrow$ We will rewrite the problem as a single level problem, to solve it more efficiently.

How to rewrite this problem

as a single-level optimization problem?

Rewriting the Problem I: Rewriting Backward Equation

We have the following objective

$$\inf_{\boldsymbol{\lambda}} \int_0^T f_0(t, \mu_t^{\boldsymbol{\lambda}}, \lambda_t)dt + g_0(\mu_T^{\boldsymbol{\lambda}}, \lambda_T)$$

where the trajectories of $X_t^{\boldsymbol{\lambda}}$ and $Y_t^{\boldsymbol{\lambda}}$ are determined by the forward **backward** SDEs:

$$X_t^{\boldsymbol{\lambda}} = \zeta + \int_0^t b(s, X_s^{\boldsymbol{\lambda}}, \hat{\alpha}_s^{\boldsymbol{\lambda}}, \mu_s^{\boldsymbol{\lambda}}; \lambda_s)ds + \int_0^t \sigma dW_t$$

$$\boxed{Y_t^{\boldsymbol{\lambda}} = g(X_T^{\boldsymbol{\lambda}}, \mu_T^{\boldsymbol{\lambda}}; \lambda_T) + \int_t^T f(s, X_s^{\boldsymbol{\lambda}}, \hat{\alpha}_s^{\boldsymbol{\lambda}}, \mu_s^{\boldsymbol{\lambda}}; \lambda_s)ds - \int_t^T Z_s^{\boldsymbol{\lambda}} dW_s}$$

Rewriting the Problem I: Rewriting Backward Equation

We have the following objective

$$\inf_{\substack{\boldsymbol{\lambda} \\ \boldsymbol{Z}, Y_0}} \int_0^T f_0(t, \mu_t^{\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0}, \lambda_t) dt + g_0(\mu_T^{\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0}, \lambda_T)$$

where the trajectories of $X_t^{\boldsymbol{\lambda}}$ and $Y_t^{\boldsymbol{\lambda}}$ are determined by the forward **forward** SDEs:

$$X_t^{\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0} = \zeta + \int_0^t b(s, X_s^{\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0}, \hat{\alpha}_s^{\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0}, \mu_s^{\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0}; \lambda_s) ds + \int_0^t \sigma dW_s$$

$$\boxed{Y_t^{\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0} = Y_0 - \int_0^t f(s, X_s^{\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0}, \hat{\alpha}_s^{\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0}, \mu_s^{\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0}; \lambda_s) ds + \int_0^t Z_s dW_s}$$

with the constraint

$$Y_T^{\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0} = g(X_T^{\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0}, \mu_T^{\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0}; \lambda_T).$$

Controls of the problem: $\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0$

Rewriting the Problem II: Introducing the Penalty

Idea: Instead of solving a constrained optimization problem, introduce the penalized objective function and directly minimize it

$\rightarrow$ Our **constrained** problem is:

$$\inf_{\boldsymbol{\lambda},\boldsymbol{Z},Y_0} \int_0^T f_0(t, \mu_t^{\boldsymbol{\lambda},\boldsymbol{Z},Y_0}, \lambda_t)dt + g_0(\mu_T^{\boldsymbol{\lambda},\boldsymbol{Z},Y_0}, \lambda_T)$$

with the constraint

$$\boxed{Y_T^{\boldsymbol{\lambda},\boldsymbol{Z},Y_0} = g(X_T^{\boldsymbol{\lambda},\boldsymbol{Z},Y_0}, \mu_T^{\boldsymbol{\lambda},\boldsymbol{Z},Y_0}; \lambda_T).}$$

and where the trajectories of X_t and Y_t are determined by the previously introduced forward **forward** SDEs.

Rewriting the Problem II: Introducing the Penalty

Idea: Instead of solving a constrained optimization problem, introduce the penalized objective function and directly minimize it

→ Our **constrained** problem is:

$$\inf_{\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0} \int_0^T f_0(t, \mu_t^{\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0}, \lambda_t) dt + g_0(\mu_T^{\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0}, \lambda_T)$$

with the constraint

$$\boxed{Y_T^{\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0} = g(X_T^{\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0}, \mu_T^{\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0}; \lambda_T).}$$

and where the trajectories of X_t and Y_t are determined by the previously introduced forward **forward** SDEs.

→ Introduce the **penalized problem**:

$$\inf_{\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0} \int_0^T f_0(t, \mu_t^{\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0}, \lambda_t) dt + g_0(\mu_T^{\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0}, \lambda_T) + \nu \boxed{\mathbb{E}\left[\mathbf{P}\Big(Y_T^{\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0} - g\big(X_T^{\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0}, \mu_T^{\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0}; \lambda_T\big)\Big)\right]},$$

where $\mathbf{P}$ is a penalty function such that $\mathbf{P}(0) = 0$ and $\mathbf{P}(x) > 0$ for all $x \neq 0$.

Rewritten Optimization Problem

The rewritten penalized problem becomes:

$$\inf_{\boldsymbol{\lambda}, Z, Y_0} \underbrace{\int_0^T f_0(t, \mu_t^{\boldsymbol{\lambda}, Z, Y_0}, \lambda_t) dt + g_0(\mu_T^{\boldsymbol{\lambda}, Z, Y_0}, \lambda_T)}_{\text{Cost of the principal: } J^0}$$

$$+ \nu \underbrace{\mathbb{E}\left[\mathbf{P}\big(Y_T^{\boldsymbol{\lambda}, Z, Y_0} - g\big(X_T^{\boldsymbol{\lambda}, Z, Y_0}, \mu_T^{\boldsymbol{\lambda}, Z, Y_0}; \lambda_T\big)\big)\right]}_{\text{Penalty: } \bar{\mathbf{P}}},$$

where

$$\left.\begin{aligned} X_t^{\boldsymbol{\lambda}, Z, Y_0} &= \zeta + \int_0^t b(s, X_s^{\boldsymbol{\lambda}, Z, Y_0}, \hat{\alpha}_s^{\boldsymbol{\lambda}, Z, Y_0}, \mu_s^{\boldsymbol{\lambda}, Z, Y_0}; \lambda_s) ds + \int_0^t \sigma dW_s, \\ Y_t^{\boldsymbol{\lambda}, Z, Y_0} &= Y_0 - \int_0^t f(s, X_s^{\boldsymbol{\lambda}, Z, Y_0}, \hat{\alpha}_s^{\boldsymbol{\lambda}, Z, Y_0}, \mu_s^{\boldsymbol{\lambda}, Z, Y_0}; \lambda_s) ds + \int_0^t Z_s dW_s, \end{aligned}\right\} \text{FFSDE}$$

and $\mu_t^{\boldsymbol{\lambda}, Z, Y_0} = \mathcal{L}(X_t^{\boldsymbol{\lambda}, Z, Y_0})$.

⇒ This is a **single-level** problem.

Using Deep Learning to Solve Stackelberg Mean Field Games

`DeepStackelbergMFG` **Idea:** Similar to the ideas introduced earlier, utilize neural networks (NN) to approximate functions for the controls of the problem.

Steps:

→ Approximate the new controls $(\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0)$ by NNs.

→ Approximate the MF distribution by an **empirical distribution**.

→ Discretize time.

→ Simulate trajectories of (X_t, Y_t) by Monte Carlo using the forward **forward** SDEs.

→ Loss function = **penalized** cost.

Using Deep Learning to Solve Stackelberg Mean Field Games

`DeepStackelbergMFG` **Idea:** Similar to the ideas introduced earlier, utilize neural networks (NN) to approximate functions for the controls of the problem.

Steps:

- → Approximate the new controls $(\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0)$ by NNs.
- → Approximate the MF distribution by an **empirical distribution**.
- → Discretize time.
- → Simulate trajectories of (X_t, Y_t) by Monte Carlo using the forward **forward** SDEs.
- → Loss function = **penalized** cost.

Theorem (Dayanıklı, Laurière, 2023): Under suitable assumptions, the solution of the parameterized, time discretized, empirically approximated, and penalized problem converges to the solution of the original problem.

Remark: Still holds if policies are in the form $\lambda(t, X_t)$ or $\lambda(t, X_t, \mu_t)$.

Numerical Approach: Extensions

Numerically, we can implement this approach for models with more complexity:

→ For example, we can have a path dependent terminal payment as a control for the principal as in *contract theory*.

→ We can have interactions through the distribution of control and state instead of just the distribution of state in the spirit of *extended mean field games*.

Mathematical Formulation of the Contract Theory Type Model

The representative agent's model:

$$\inf_{\boldsymbol{\alpha}} J^{\boldsymbol{\lambda},\xi}(\boldsymbol{\alpha},\boldsymbol{\mu}) := \mathbb{E}\left[\int_0^T f(t, X_t, \alpha_t, \mu_t; \lambda_t)dt + g(X_T, \mu_T; \lambda_T) - U(\xi)\right]$$

$$dX_t = b(t, X_t, \alpha_t, \mu_t; \lambda_t)dt + \sigma dW_t, \qquad X_0 = \zeta,$$

→ Mean field Nash equilibrium can be characterized with an FBSDE.

Mathematical Formulation of the Contract Theory Type Model

The representative agent's model:

$$\inf_{\boldsymbol{\alpha}} J^{\boldsymbol{\lambda},\xi}(\boldsymbol{\alpha},\boldsymbol{\mu}) := \mathbb{E}\left[\int_0^T f(t,X_t,\alpha_t,\mu_t;\lambda_t)dt + g(X_T,\mu_T;\lambda_T) - U(\xi)\right]$$

$$dX_t = b(t,X_t,\alpha_t,\mu_t;\lambda_t)dt + \sigma dW_t, \qquad X_0 = \zeta,$$

→ Mean field Nash equilibrium can be characterized with an FBSDE.

The principal's problem:

$$\inf_{\boldsymbol{\lambda},\xi} \mathbb{E}\left[\int_0^T f_0(t,\hat{\mu}_t,\lambda_t)dt + g_0(\hat{\mu}_T,\lambda_T) + \xi\right],$$

s.t:

→ $(\hat{\boldsymbol{\alpha}},\hat{\boldsymbol{\mu}})$ is a mean field Nash equilibrium given $(\boldsymbol{\lambda},\xi)$

→ Introduce the *walkaway* option for the agents: $J^{\boldsymbol{\lambda},\xi}(\hat{\boldsymbol{\alpha}},\hat{\boldsymbol{\mu}}) \leq \kappa$

Mathematical Formulation of the Contract Theory Type Model

The representative agent's model:

$$\inf_{\boldsymbol{\alpha}} J^{\boldsymbol{\lambda},\xi}(\boldsymbol{\alpha},\boldsymbol{\mu}) := \mathbb{E}\left[\int_0^T f(t, X_t, \alpha_t, \mu_t; \lambda_t)dt + g(X_T, \mu_T; \lambda_T) - U(\xi)\right]$$

$$dX_t = b(t, X_t, \alpha_t, \mu_t; \lambda_t)dt + \sigma dW_t, \qquad X_0 = \zeta,$$

→ Mean field Nash equilibrium can be characterized with an FBSDE.

The principal's problem:

$$\inf_{\boldsymbol{\lambda},\xi} \mathbb{E}\left[\int_0^T f_0(t, \hat{\mu}_t, \lambda_t)dt + g_0(\hat{\mu}_T, \lambda_T) + \xi\right],$$

s.t:

→ $(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\mu}})$ is a mean field Nash equilibrium given $(\boldsymbol{\lambda}, \xi)$

→ Introduce the *walkaway* option for the agents: $J^{\boldsymbol{\lambda},\xi}(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\mu}}) \le \kappa$

The constraint becomes:

$$\boxed{Y_T = g(X_T, \mu_T; \lambda_T) - U(\xi)}$$

Rewritten Extended Model

With the same idea, the model can be written as:

$$\inf_{Y_0:\mathbb{E}[Y_0]\leq\kappa}\inf_{\boldsymbol{Z},\boldsymbol{\lambda},\xi}\mathbb{E}\left[\int_0^T f_0(t,\mu_t^{\boldsymbol{\lambda},\boldsymbol{Z},Y_0,\xi},\lambda_t)dt+g_0(\mu_T^{\boldsymbol{\lambda},\boldsymbol{Z},Y_0,\xi},\lambda_T)+\xi\right] \\ +\nu\mathbb{E}\left[\mathbf{P}\left(Y_T^{\boldsymbol{\lambda},\boldsymbol{Z},Y_0,\xi}-g\left(X_T^{\boldsymbol{\lambda},\boldsymbol{Z},Y_0,\xi},\mu_T^{\boldsymbol{\lambda},\boldsymbol{Z},Y_0,\xi};\lambda_T\right)+U(\xi)\right)\right]$$

where the trajectories of X_t and Y_t are determined by the forward forward SDE.

Rewritten Extended Model

With the same idea, the model can be written as:

$$\inf_{Y_0:\mathbb{E}[Y_0]\le\kappa}\inf_{\boldsymbol{Z},\boldsymbol{\lambda},\xi}\mathbb{E}\left[\int_0^T f_0(t,\mu_t^{\boldsymbol{\lambda},\boldsymbol{Z},Y_0,\xi},\lambda_t)dt + g_0(\mu_T^{\boldsymbol{\lambda},\boldsymbol{Z},Y_0,\xi},\lambda_T)+\xi\right]$$
$$+\nu\mathbb{E}\left[\mathbf{P}\left(Y_T^{\boldsymbol{\lambda},\boldsymbol{Z},Y_0,\xi}-g(X_T^{\boldsymbol{\lambda},\boldsymbol{Z},Y_0,\xi},\mu_T^{\boldsymbol{\lambda},\boldsymbol{Z},Y_0,\xi};\lambda_T)+U(\xi)\right)\right]$$

where the trajectories of X_t and Y_t are determined by the forward forward SDE.

Special Case: Assume $g(X_T,\rho_T;\lambda_T)=0$ and $U(\cdot)$ is invertible:

→ Terminal condition of (previously) backward SDE gives

$$Y_T=-U(\xi)\quad\Rightarrow\quad \xi=U^{-1}(-Y_T)$$

→ Then focus on minimizing:

$$\inf_{Y_0:\mathbb{E}[Y_0]\le\kappa}\inf_{\boldsymbol{Z},\boldsymbol{\lambda}}\mathbb{E}\left[\int_0^T f_0(t,\mu_t^{\boldsymbol{\lambda},\boldsymbol{Z},Y_0},\lambda_t)dt + g_0(\mu_T^{\boldsymbol{\lambda},\boldsymbol{Z},Y_0},\lambda_T)+U^{-1}(-Y_T^{\boldsymbol{\lambda},\boldsymbol{Z},Y_0})\right]$$

No penalty function is needed!

Numerical Results

Example 1: Systemic Risk Model with a Regulator[6]

Principal (Regulator): Proposes incentive $\boldsymbol{\lambda}$ and has the objective:

$$\inf_{\lambda} \int_0^T (\lambda_t - \lambda_t^{\text{aim}})^2 dt + \gamma \mathbb{P}\Big[X_T < D\Big]$$

for exogenous λ_t^{aim} = aimed level and D = Default threshold < 0.

Agent Population (Banks): Control = lending/borrowing rate α_t.

The objective of the representative bank is given as

$$\inf_{\alpha} \mathbb{E}\left[\int_0^T \left(\frac{\alpha_t^2}{2} - \lambda_t \alpha_t (\bar{X}_t - X_t) + \frac{\epsilon}{2}(\bar{X}_t - X_t)^2\right) dt + \frac{c}{2}(\bar{X}_T - X_T)^2\right]$$

where $\epsilon, c, \lambda > 0$ are exogenous constants and

$$dX_t = \big[a(\bar{X}_t - X_t) + \alpha_t\big] dt + dW_t$$

where W_t is the idiosyncratic noise and $a > 0$ is an exogenous constant.

[6] Carmona, Fouque, and Sun (2013)

Solutions: Systemic Risk with a Regulator

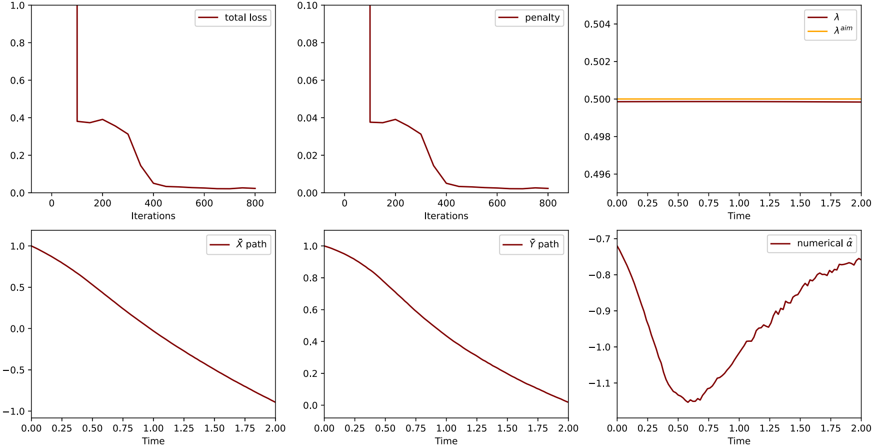


Figure 2: Systemic Risk Model with $\gamma = 0$.

T	Δt	μ_0	a	c	ϵ	λ^{aim}	γ	D
2.0	0.02	δ_1	1.0	1.0	1.0	0.5	0.0	-0.001

Solutions: Systemic Risk with a Regulator

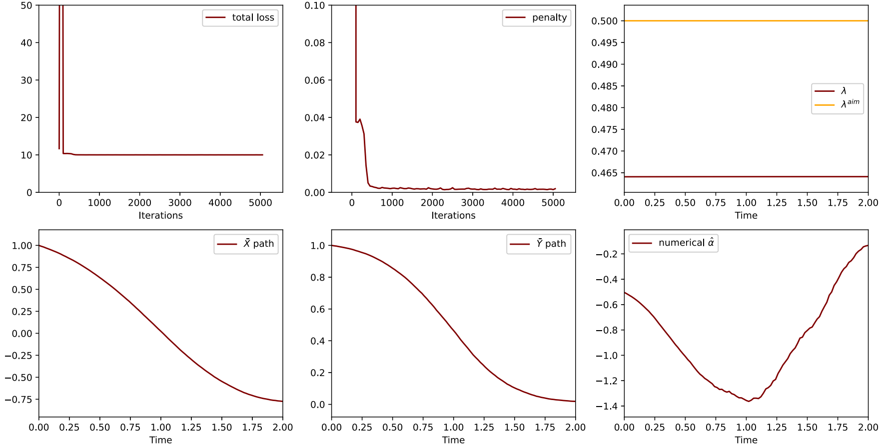


Figure 3: Systemic Risk Model with $\gamma = 10$.

T	Δt	μ_0	a	c	ϵ	λ^{aim}	γ	D
2.0	0.02	δ_1	1.0	1.0	1.0	0.5	10.0	-0.001

Solutions: Systemic Risk with a Regulator

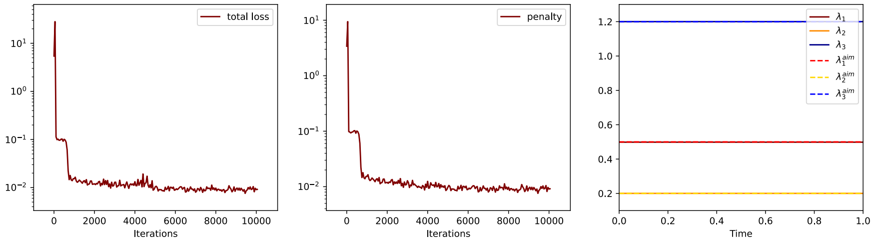


Figure 4: Systemic Risk Model with $\gamma = 0$ and multiple assets.

Example 2: Contract Theory Model with a Principal and Many Agents[7]

Principal: Proposes terminal payment ξ and has the objective

$$\inf_{\xi} \mathbb{E}\big[\xi - X_T\big]$$

Agent Population: Controls the effort level α_t.

The objective of the representative agent is given as

$$\inf_{\alpha} \ \mathbb{E}\Big[\int_0^T k\frac{\alpha_t^2}{2}dt - \xi\Big]$$

where $k > 0$ is an exogeneous constant and

$$dX_t = \Big(\alpha_t + aX_t + \beta_1 \bar{X}_t + \beta_2 \bar{\alpha}_t\Big)dt + dW_t$$

where $\beta_1, \beta_2 \geq 0$ are constants, and W_t is the idiosyncratic noise.

Remark: Optimal Effort of the agent is given by

$$\boxed{\alpha_t^* = (1+\beta_2)\frac{e^{(a+\beta_1)(T-t)}}{k}}$$

[7] Elie, Mastrolia, and Possamaï (2019)

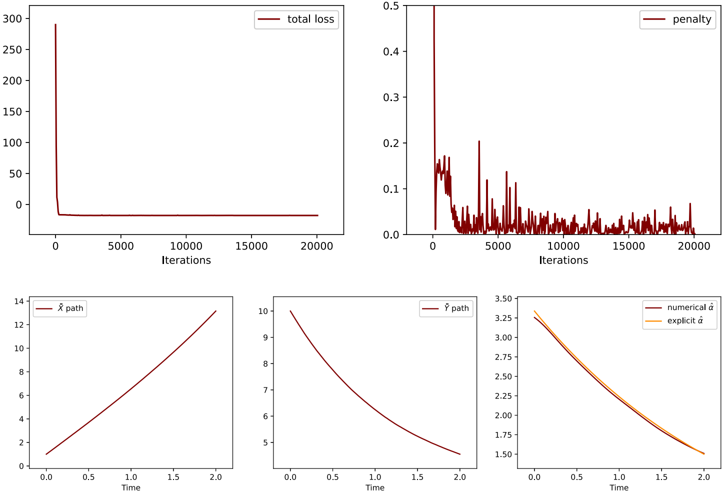


Figure 5: $dX_t = \left(\alpha_t + aX_t + \beta_2 \bar{\alpha} dt\right) dt + dW_t$

T	Δt	μ_0	σ	k	a	β_1	β_2	γ
2.0	0.02	δ_1	1.0	1.0	0.4	0.0	0.5	0

Solutions: Interactions through the mean of the controls (Special case)

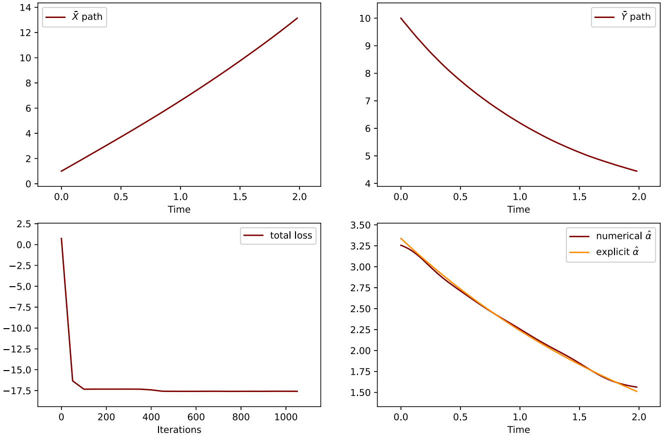


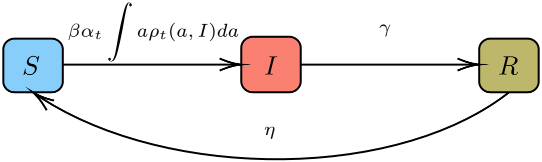
Figure 6: $dX_t = \Big(\alpha_t + aX_t + \beta_2 \bar{\alpha} dt\Big) dt + dW_t$

T	Δt	μ_0	σ	k	a	β_1	β_2	γ
2.0	0.02	δ_1	1.0	1.0	0.4	0.0	0.5	0

Example 3: Mitigating Epidemics (I): Intuitions[8]

→ **Agent Population:**

- → **Control:** Socialization levels
- → **Objectives:** Follow the policies & minimize the cost (infection/treatment)



→ **Principal:**

- → **Control:** Social distancing measures, stimulus payment
- → **Objectives:** Follow the recommendations from healthcare professionals & flatten the *curve*

[8] Aurell, Carmona, Dayanıklı, Laurière (2022)

Example 3: Mitigating Epidemics (II): Agent's Model

Control: Socialization Level: α_t

States: Health Conditions: Susceptible (S), Infected (I), Recovered (R)

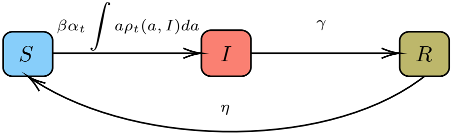
Objective:

$$\inf_{(\alpha_t)_t} \mathbb{E}\Big[\int_0^T \frac{c_\lambda}{2} \left(\lambda_t^{(\mathsf{S})} - \alpha_t\right)^2 \mathbb{1}_{\mathsf{S}}(x) + \left(\frac{1}{2} \left(\lambda_t^{(\mathsf{I})} - \alpha_t\right)^2 + \underbrace{c_{\mathsf{I}}}_{\text{treatment cost}} \right) \mathbb{1}_{\mathsf{I}}(x)$$

$$+ \frac{1}{2} \underbrace{\left(\lambda_t^{(\mathsf{R})} - \alpha_t\right)^2 \mathbb{1}_{\mathsf{R}}(x)}_{\text{cost of not following the policy}} dt - \xi \Big]$$

where $c_\lambda, c_{\mathsf{I}} \in \mathbb{R}_+$ are constants.

State Dynamics:



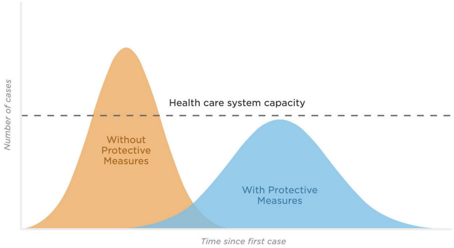
Example 3: Mitigating Epidemics (III): Principal's Model

Controls: Social Distancing Policy λ, stimulus check ξ

Objective:

$$\inf_{(\lambda_t)_t,\xi} \mathbb{E}\Big[\int_0^T \underbrace{c_{\text{Inf}}\, p_t(\mathsf{I})^2}_{\text{flattening the curve}} + \sum_{i\in\{\mathsf{S},\mathsf{I},\mathsf{R}\}} \frac{\bar{\beta}^{(i)}}{2}\Big(\lambda_t^{(i)} - \underbrace{\bar{\lambda}_t^{(i)}}_{\substack{\text{recommended}\\ \text{policy}}}\Big)^2 dt + \xi\Big]$$

for constant $\bar{\lambda}, \bar{\beta} \in \mathbb{R}_+^m$ and $c_{\text{Inf}} > 0$.



https://www.npr.org/sections/health-shots/2020/03/13/815502262/flattening-a-pandemics-curve-why-staying-home-now-can-save-lives

Solution: SIR Mean Field Game

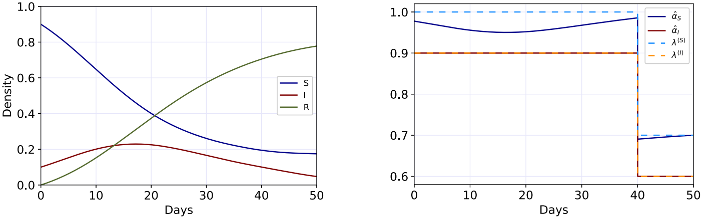


Figure 9: Late lockdown, **explicit** solution. Evolution of the population state distribution (left), evolution of the controls (right).

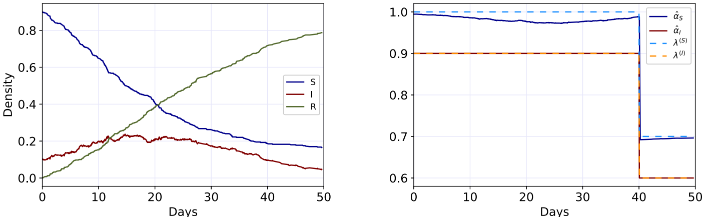


Figure 10: Late lockdown, **numerical** solution. Evolution of the population state distribution (left), evolution of the controls (right).

Solutions: SEIRD Stackelberg Mean Field Game

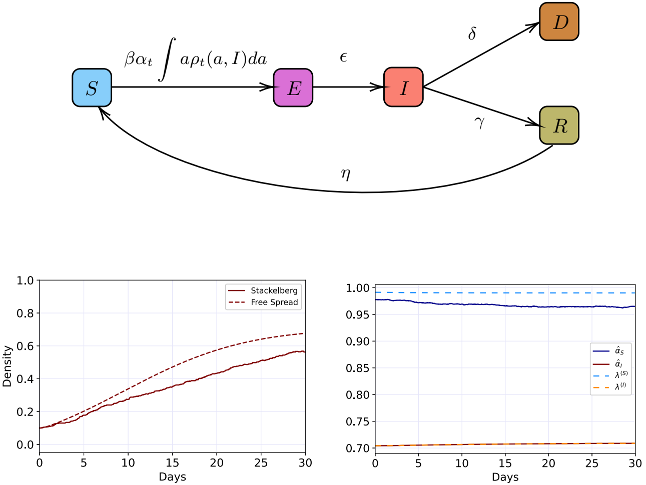


Figure 11: SEIRD Dynamics (top). SEIRD Stackelberg MFG vs free spread SEIRD dynamics (bottom): Comparison of the Cumulative Density of Infected agents (left); Evolution of the controls (right).

T	ρ^0	c_λ	c_I	c_{Inf}	$\bar{\beta}$	$\bar{\lambda}$	β	γ	η	ϵ	δ	c_d	c_D
30	[0.9, 0, 0.1, 0, 0]	10	1	1	[0.2, 0.2, 1, 0]	[1, 1, 0.7, 1]	0.25	0.1	0.01	2	0.01	20	20

Conclusions

This talk: Optimal policies for a large population of noncooperative agents

- Introduction to SOC and deep learning for such problems
- Equilibrium notions
- MFGs & FBSDEs
- Stackelberg MFGs
- Bi-level optim. $\rightarrow$ constrained optim. $\rightarrow$ single-level optim
- Deep learning algorithm & numerical examples

Future directions:

- Existence & uniqueness of solutions to general Stackelberg MFG
- Convergence rate to Nash equilibrium for the shooting method
- Real-world applications (e.g., in economics)

Thank you!

ArXiv: 2302.10440, 2011.03105, 2106.07859, 2306.04788

mathieu.lauriere@nyu.edu, https://mlauriere.github.io