Machine Learning for Mean Field Games and its Extensions

Gökçe Dayanıklı
Department of Statistics, University of Illinois Urbana-Champaign

21st International Symposium on Dynamic Games and Applications
Boğaziçi University, Istanbul, Turkey
July 20, 2026

Motivation for this Tutorial

→ We want to understand the emergent behavior in large populations of interacting agents (individuals, banks, traders, employees, electricity producers, etc.).

→ Some examples:

- Modeling and understanding the borrowing and lending behavior of interacting banks.
- Modeling and understanding the trading behavior of traders who want to maximize their expected return.
- Modeling and understanding of renewable energy investment decisions of the large number of electricity producers that are interacting through the price of the electricity.
- Modeling the response of individuals to social-distancing or vaccination policies in the epidemics.

Motivation for this Tutorial

→ We want to understand the emergent behavior in large populations of interacting agents (individuals, banks, traders, employees, electricity producers, etc.).

→ Some examples:

- Modeling and understanding the borrowing and lending behavior of interacting banks.
- Modeling and understanding the trading behavior of traders who want to maximize their expected return.
- Modeling and understanding of renewable energy investment decisions of the large number of electricity producers that are interacting through the price of the electricity.
- Modeling the response of individuals to social-distancing or vaccination policies in the epidemics.

→ **Idea:** These problems are highly complex (dynamic, stochastic, and with many decision makers).
For models with realistic features, **machine learning** can greatly help.

→ We will focus on continuous time setting for this tutorial.

In This Tutorial

How to use deep learning for mean field games and related problems?

- → Problems with one agent: Stochastic Optimal Control
- → Problems with large number of agents
  - → Mean Field Approximations
  - → Cooperative agents: Deep learning for Mean Field Control
  - → Crash course on Nash equilibrium
  - → Non-cooperative agents: Deep learning for Mean Field Games
- → Extensions
  - → Multi-populations
  - → Complex network interactions: Graphon games
  - → Addition of significant players: Stackelberg Mean Field Games

Starting with one agent:

Stochastic Optimal Control Problems

Stochastic Optimal Control Problems

We have 1 agent. She tries to minimize her costs/ maximize her rewards while choosing her controls between time $t = 0$ to $t = T$. She has

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

Stochastic Optimal Control Problems

We have 1 agent. She tries to minimize her costs/ maximize her rewards while choosing her controls between time $t = 0$ to $t = T$. She has

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

Let's assume that Sibel works in a company and she chooses her effort level that affects the value of the project she is working on.

Then Sibel's:

- $\rightarrow$ X_t: Value of the project at time t
- $\rightarrow$ α_t: Effort level at time t
- $\rightarrow$ Objectives: utility from the value of the project, cost of putting too much effort.

Stochastic Optimal Control Problems: Mathematical Formulation (I)

Let's assume that Sibel works in a company and she chooses her effort level that affects the value of the project she is working on.

Then Sibel's:

- $\rightarrow$ X_t: Value of the project at time t
- $\rightarrow$ α_t: Effort level at time t
- $\rightarrow$ Objectives: utility from the value of the project, cost of putting too much effort.

How can we write this model mathematically?

Stochastic Optimal Control Problems: Mathematical Formulation (I)

Let's assume that Sibel works in a company and she chooses her effort level that affects the value of the project she is working on.

Then Sibel's:

- $\rightarrow$ X_t: Value of the project at time t
- $\rightarrow$ α_t: Effort level at time t
- $\rightarrow$ Objectives: utility from the value of the project, cost of putting too much effort.

Mathematical Formulation:

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

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

where $U(\cdot)$ is a utility function, c_1, c_2, c_3, σ are positive constants and W_t is the Brownian Motion.

Stochastic Optimal Control Problems: Mathematical Formulation (II)

Sibel's problem:

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

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

Writing it as a general stochastic optimal control problem:

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

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

Another example: Merton's problem

An individual wants to maximize

→ their utility from consumption over the time interval $t \in [0, T]$ and

→ their utility from the terminal wealth.

She controls at time t

→ fraction of wealth to be invested in stocks: π_t

→ consumption level: c_t

Her state is **?**

Another example: Merton's problem

An individual wants to maximize

→ their utility from consumption over the time interval $t \in [0, T]$ and

→ their utility from the terminal wealth.

She controls at time t

→ fraction of wealth to be invested in stocks: π_t

→ consumption level: c_t

Her state is **wealth**: X_t

How can we write this problem mathematically?

Another example: Merton's problem

An individual wants to maximize

→ their utility from consumption over the time interval $t \in [0, T]$ and

→ their utility from the terminal wealth.

She controls at time t

→ fraction of wealth to be invested in stocks: π_t

→ consumption level: c_t

Her state is **wealth**: X_t

How can we write this problem mathematically?

$$\max_{(\pi_t, c_t)_t} E\left[\int_0^T e^{-\rho s} U(c_s)\, ds + \epsilon^\gamma e^{-\rho T} U(X_T)\right]$$

$$dX_t = \big[(r + \pi_t(\mu - r))X_t - c_t\big]dt + X_t \pi_t \sigma dW_t$$

Another example: Merton's problem

An individual wants to maximize

→ their utility from consumption over the time interval $t \in [0, T]$ and

→ their utility from the terminal wealth.

She controls at time t

→ fraction of wealth to be invested in stocks: π_t

→ consumption level: c_t

Her state is **wealth**: X_t

How can we write this problem mathematically?

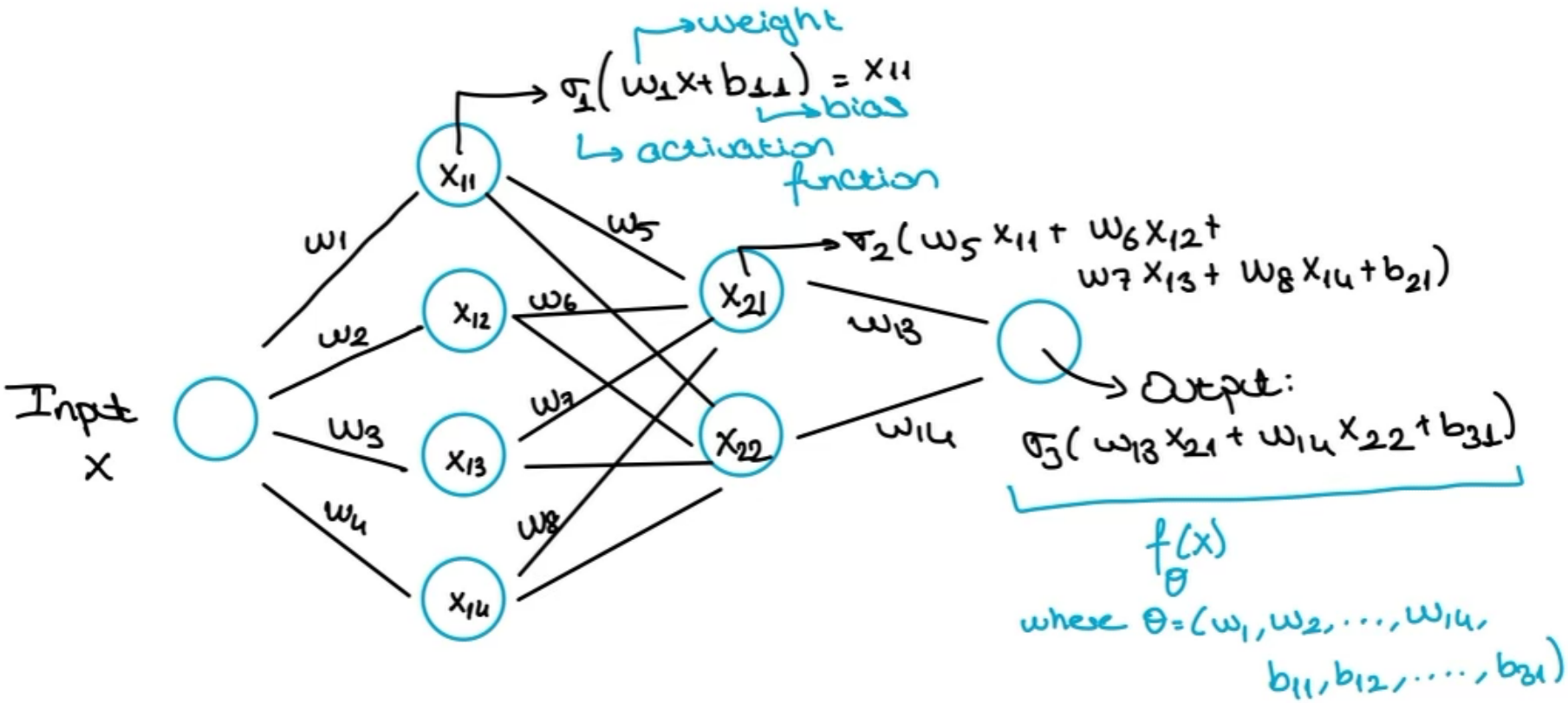
$$\max_{(\pi_t, c_t)_t} E\left[\int_0^T e^{-\rho s} U(c_s)\, ds + \epsilon^\gamma e^{-\rho T} U(X_T)\right]$$

$$dX_t = \big[(r + \pi_t(\mu - r))X_t - c_t\big]dt + X_t \pi_t \sigma dW_t$$

Before going to large populations: let's discuss how we can utilize deep learning to solve the stochastic optimal control problem.

Using Deep Learning to Solve Stochastic Optimal Control Problems

Side Note: Neural Networks as Function Approximators



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

→ Assume you want to approximate a function $f(x)$. You can build a neural network that outputs $f_\theta(x)$ and train it over the parameters θ by using objective:

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

Deep Learning for the Stochastic Optimal Control

Remember stochastic optimal control problems:

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

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

Numerical solution approach with deep learning:

→ **Time discretization:** Assume the control is a function of time and the current state: $\alpha_t = \varphi(t, X_t)$ and discretize the time: $t = \{0, \Delta t, 2\Delta t, \ldots, n\Delta t\}$, where $T = n\Delta t$:

$$\min_{\varphi} \mathbb{E}\Big[\sum_t f(t, X_t, \varphi(t, X_t)) \times \Delta t + g(X_T)\Big]$$

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

Deep Learning for the Stochastic Optimal Control

Remember stochastic optimal control problems:

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

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

Numerical solution approach with deep learning:

→ **Time discretization:** Assume the control is a function of time and the current state: $\alpha_t = \varphi(t, X_t)$ and discretize the time: $t = \{0, \Delta t, 2\Delta t, \ldots, n\Delta t\}$, where $T = n\Delta t$:

$$\min_{\varphi} \mathbb{E}\Big[\sum_t f(t, X_t, \varphi(t, X_t)) \times \Delta t + g(X_T)\Big]$$

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

→ **Control parameterization & Simulation:** Initialize the state X_0 and simulate the trajectory of X_t. Use NN approximation $\varphi_\theta(t, X_t)$ for the control function and aim to minimize the cost over the parameters θ in the training.

Deep Learning for the Stochastic Optimal Control

Remember stochastic optimal control problems:

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

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

Numerical solution approach with deep learning:

→ **Time discretization:** Assume the control is a function of time and the current state: $\alpha_t = \varphi(t, X_t)$ and discretize the time: $t = \{0, \Delta t, 2\Delta t, \ldots, n\Delta t\}$, where $T = n\Delta t$:

$$\min_{\varphi} \mathbb{E}\Big[\sum_t f(t, X_t, \varphi(t, X_t)) \times \Delta t + g(X_T)\Big]$$

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

→ **Control parameterization & Simulation:** Initialize the state X_0 and simulate the trajectory of X_t. Use NN approximation $\varphi_\theta(t, X_t)$ for the control function and aim to minimize the cost over the parameters θ in the training.

→ Similar to Han & E (2016).

We want to use the deep learning to solve more complex problems. But first let's check the pseudo-code.

Pseudo-Code for Stochastic Optimal Control

Algorithm 1 Simulation of state trajectories

Input: Time horizon T; time increment Δt; initial distribution μ_0
Output: Trajectories of state and control.

Let $n = 0, t_0 = 0$, sample $X_0 \sim \mu_0$
while $n \times \Delta t = t_n < T$ **do**
    Set $\alpha_{t_n} = \varphi_\theta(t_n, X_{t_n})$
    Sample $\Delta W_{t_n} \sim \mathcal{N}(0, \Delta t)$
    Let $X_{t_{n+1}} = X_{t_n} + b(t_n, X_{t_n}, \alpha_{t_n})\Delta t + \sigma \Delta W_{t_n}$
    Set $t_n = t_n + \Delta t$ and $n = n + 1$
end while
Set $N_T = n$, $t_{N_T} = T$
return $(X_{t_n}, \alpha_{t_n})_{n=0,\dots,N_T}$

Algorithm 2 Stochastic Gradient Descent for solving Stochastic Optimal Control

Input: Initial parameter θ_0; number of iterations K; sequence $(\beta_k)_{k=0,\dots,K-1}$ of learning rates; time horizon T; time increment Δt; initial distribution μ_0
Output: Approximation of θ^* minimizing $\mathbb{J}(\theta) = \mathbb{E}\Big[\sum_t f(t, X_t, \varphi_\theta(t, X_t)) \times \Delta t + g(X_T)\Big]$

for $k = 0, 1, 2, \dots, K-1$ **do**
    Simulate state trajectory with control functions $\boldsymbol{\alpha} = \varphi_{\theta_k}$ and parameters: T, Δt, μ_0
    Compute the gradient of the expected cost: $\nabla \mathbb{J}(\theta_k)$ of $\mathbb{J}(\theta_k)$
    Set $\theta_{k+1} = \theta_k - \beta_k \nabla \mathbb{J}(\theta_k)$
end for
return θ_K and corresponding trajectories $(X_{t_n}, \alpha_{t_n})_{n=0,\dots,N_T}$

Large Populations:

Mean Field Approximations

Overview of the Approach

→ Solution notions:
- Global optimization for finding the social optimum (Cooperative agents) or
- Game theory tools for finding the Nash equilibrium (Non-cooperative agents).

→ **Difficulty:** Curse of dimensionality when there is a large number of agents.

→ **Approach:** Mean field approximations:
- Cooperative agents: Mean field control
- Non-cooperative agents: Mean field game

[1] Huang-Malhamé-Caines (2006), Lasry-Lions (2006).

[2] Picture Credit: https://gbxglobal.org/the-importance-of-the-network/

Overview of the Approach

→ Solution notions:
- Global optimization for finding the social optimum (Cooperative agents) or
- Game theory tools for finding the Nash equilibrium (Non-cooperative agents).

→ **Difficulty:** Curse of dimensionality when there is a large number of agents.

→ **Approach:** Mean field approximations:
- Cooperative agents: Mean field control
- Non-cooperative agents: Mean field game

In Mean Field Control & Games[1]:

→ Assume $N \to \infty$.

→ Agents are identical and infinitesimal.

→ Agents interact symmetrically.

→ Agents interact through mean field.

→ **Idea:** Focus on representative agent and her interactions with the distribution.

[1] Huang-Malhamé-Caines (2006), Lasry-Lions (2006).

[2] Picture Credit: https://gbxglobal.org/the-importance-of-the-network/

Cooperative Agents:

Deep learning for Mean Field Control

Mathematical Formulation of Mean Field Control

For very large N, we approximate the finite player stochastic optimal control with **Mean Field Control**.

The average **cost** of the agents using control $\boldsymbol{\alpha} \in \mathbb{A}$ is

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

The agent's state X_t has the following dynamics:

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

Mathematical Formulation of Mean Field Control

For very large N, we approximate the finite player stochastic optimal control with **Mean Field Control**.

The average **cost** of the agents using control $\boldsymbol{\alpha} \in \mathbb{A}$ is

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

The agent's state X_t has the following dynamics:

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

Definition: Control $\hat{\boldsymbol{\alpha}}$ is called a social optimum if it minimizes the cost $J(\boldsymbol{\alpha})$.

Using Deep Learning to Solve Mean Field Control

Some remarks:

- $\rightarrow$ The ideas from solving the stochastic optimal control with deep learning can be extended.
- $\rightarrow$ However, now we have large number of interacting agents. $\Rightarrow$ we will implement **Monte-Carlo Simulation**.
- $\rightarrow$ **Q:** How about the population distribution?

Using Deep Learning to Solve Mean Field Control

Some remarks:

- → The ideas from solving the stochastic optimal control with deep learning can be extended.
- → However, now we have large number of interacting agents. ⇒ we will implement **Monte-Carlo Simulation**.
- → **Q:** How about the population distribution?
  - **A:** A population of size N will be simulated and the empirical distribution will be used.

Pseudo-Code for Mean Field Control

Algorithm 3 Simulation of state trajectories of interacting agents

Input: Time horizon T; time increment Δt; initial distribution μ_0; population size N

Output: Trajectories of state and control.

Let $n = 0, t_0 = 0$, sample $X_0^i \sim \mu_0$ for all $i \in \{1, 2, \ldots, N\}$

while $n \times \Delta t = t_n < T$ **do**

  Set $\alpha_{t_n}^i = \varphi_\theta(t_n, X_{t_n}^i)$ for all $i \in \{1, 2, \ldots, N\}$

  Sample $\Delta W_{t_n}^i \sim \mathcal{N}(0, \Delta t)$ for all $i \in \{1, 2, \ldots, N\}$

  Approximate $\mathcal{L}(X_{t_n})$ using the empirical distribution of $X_{t_n}^i$ and call it μ_{t_n}

  Let $X_{t_{n+1}}^i = X_{t_n}^i + b(t_n, X_{t_n}^i, \alpha_{t_n}^i, \mu_{t_n})\Delta t + \sigma \Delta W_{t_n}^i$ for all $i \in \{1, 2, \ldots, N\}$

  Set $t_n = t_n + \Delta t$ and $n = n + 1$

end while

Set $N_T = n$, $t_{N_T} = T$

return $(X_{t_n}^i, \alpha_{t_n}^i)_{n=0,\ldots,N_T; i=1,\ldots,N}$

Remark: Parameter updates are done similar to stochastic optimal control.

An example: Systemic Risk[3]

The agents are the banks. They decide how much to borrow or lend from the central bank.

The objective of the representative bank is given as

$$\inf_{\alpha} \int_0^T \Big[\frac{\alpha_t^2}{2} - q\alpha_t(\bar{X}_t - X_t) + \frac{\epsilon}{2}(\bar{X}_t - X_t)^2 dt + \frac{c}{2}(\bar{X}_T - X_T)^2\Big]$$

where $\epsilon, c, \lambda > 0$ are exogenous constants, $\bar{X}_t = \mathbb{E}[X_t]$ and

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

where W_t is the idiosyncratic noise and $a, \sigma > 0$ are exogenous constants.

Remark: This problem has an explicit solution where:

$$\alpha_t^i = (q + \eta_t)(\bar{X}_t - X_t^i).$$

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

Solutions with Deep Learning

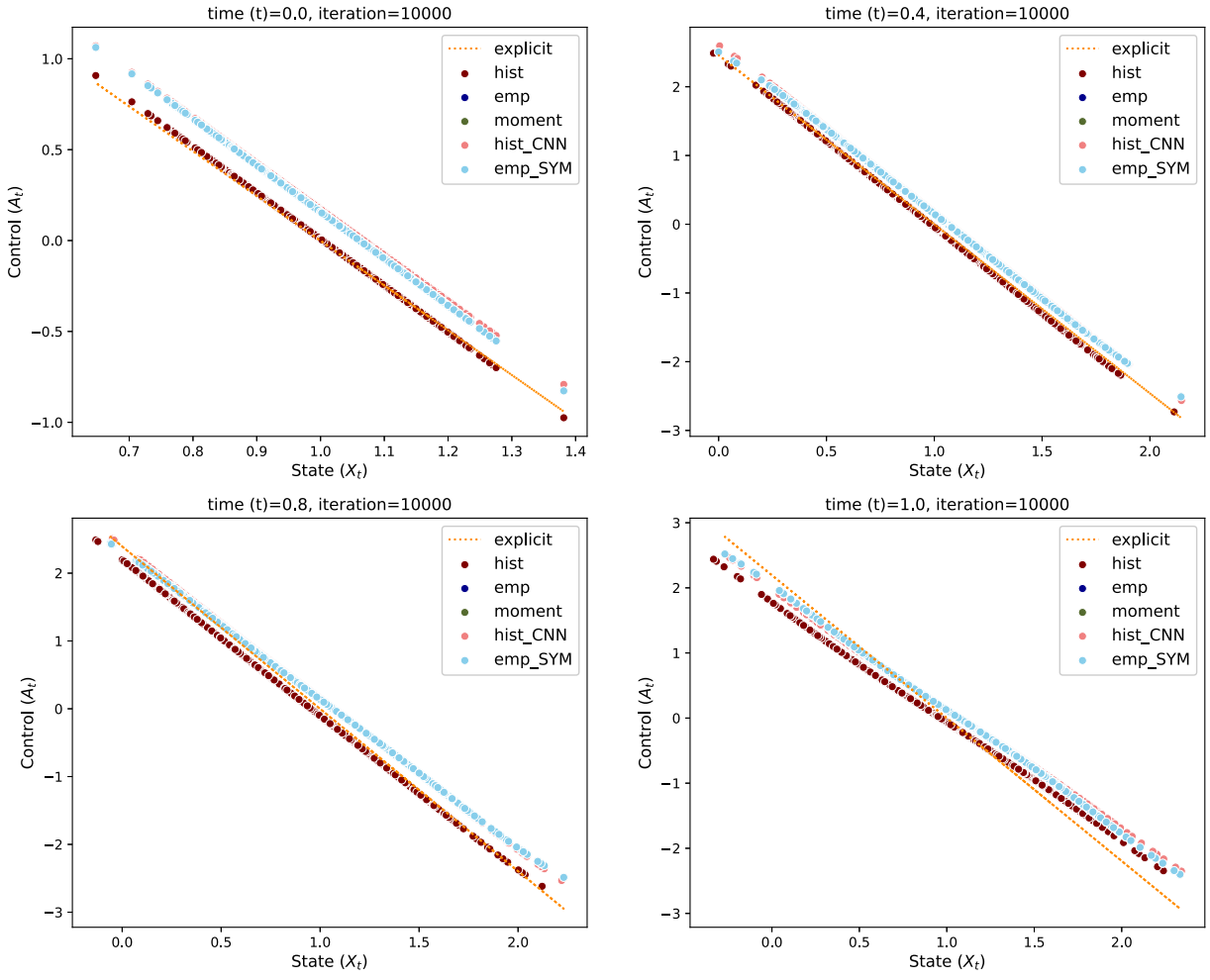


Figure 1: Systemic Risk Results with the Deep Learning Approach from **GD.**-Lauriere-Zhang (2023)

Non-cooperative Agents:

Deep Learning for Mean Field Game

Non-cooperative Agents:

Deep Learning for Mean Field Game

But first: A Crash Course on Nash Equilibrium

Crash Course on Game Theory & Nash Equilibrium

→ We aim to find an equilibrium such as **Nash equilibrium**.

- Agents can't benefit from deviating from their equilibrium action.
- At equilibrium each agent's action is the best response to others.
- **Mathematically:** Assume there are N agents, then an action profile $\alpha^* = (\alpha_1^*, \ldots, \alpha_N^*)$ is a Nash equilibrium if for any other action of agent i, α_i:

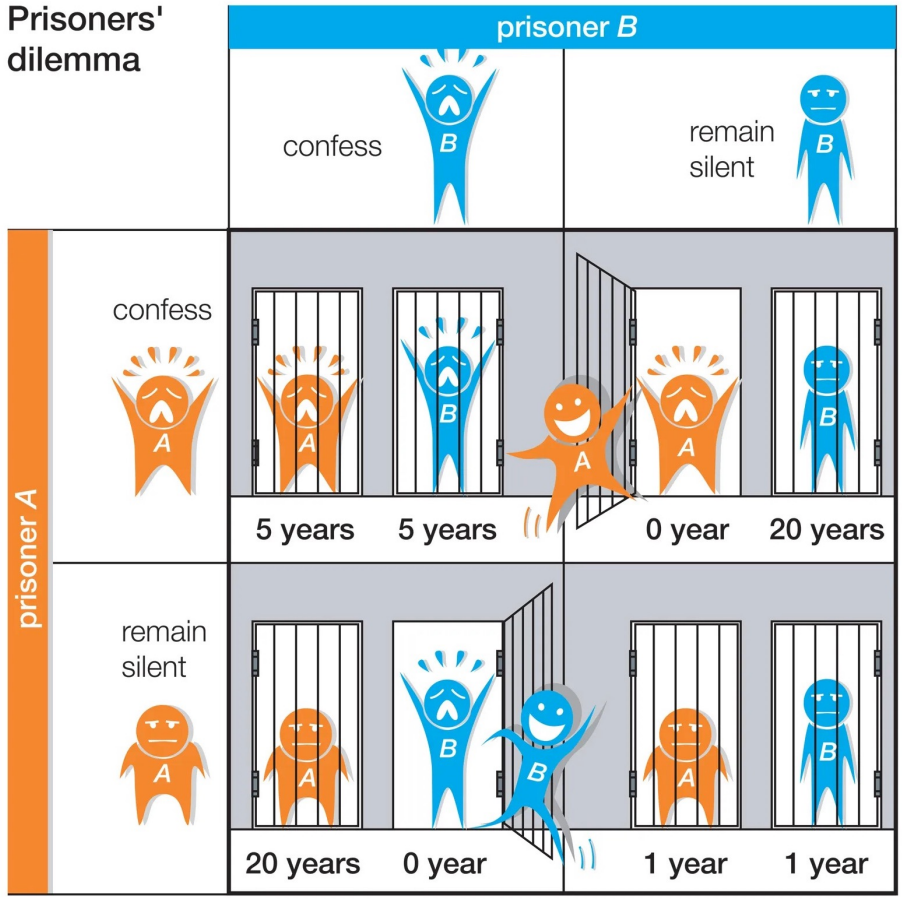
$$J_i(\alpha^*) \leq J_i((\alpha_i, \alpha_{-i}^*)) \qquad \forall i = 1, \ldots, N$$

where $(\alpha_i, \alpha_{-i}^*) = (\alpha_1^*, \ldots, \alpha_i, \ldots, \alpha_N^*)$ and J_i is the cost function of agent i.

Crash Course on Non-cooperative Game Theory & Nash Equilibrium

→ We aim to find the **Nash equilibrium**.

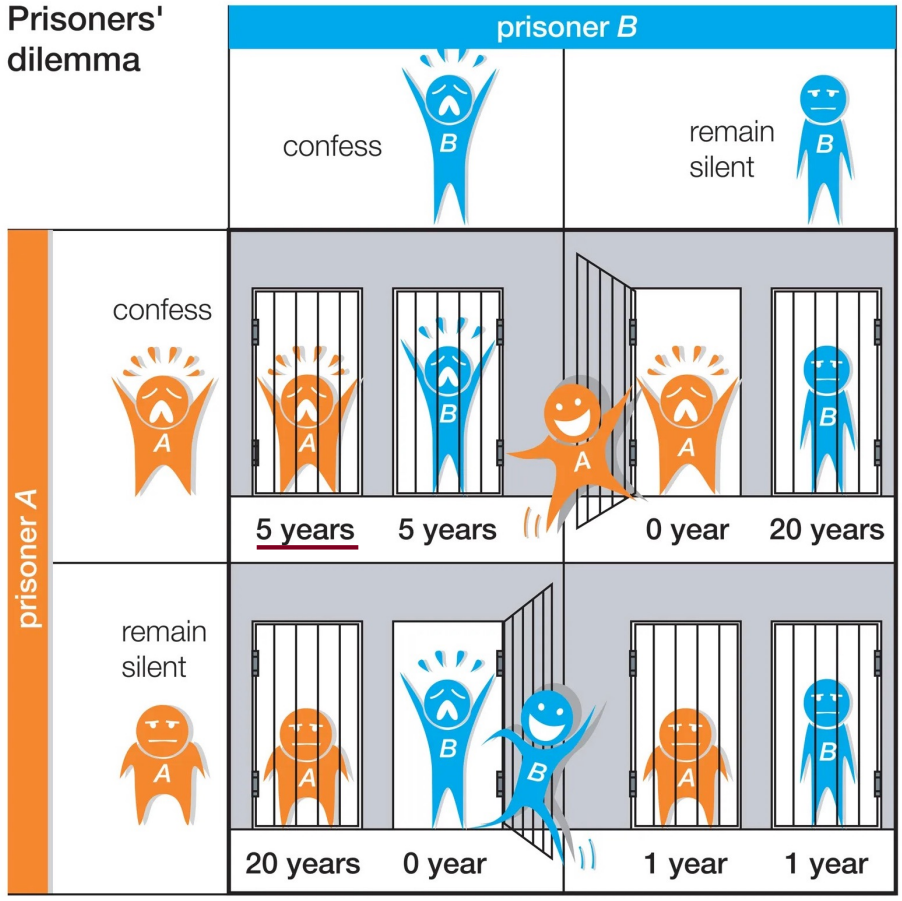
- Agents can't benefit from deviating from their equilibrium action
- At equilibrium each agent's action is the best response to others.



Crash Course on Non-cooperative Game Theory & Nash Equilibrium

→ We aim to find the **Nash equilibrium**.

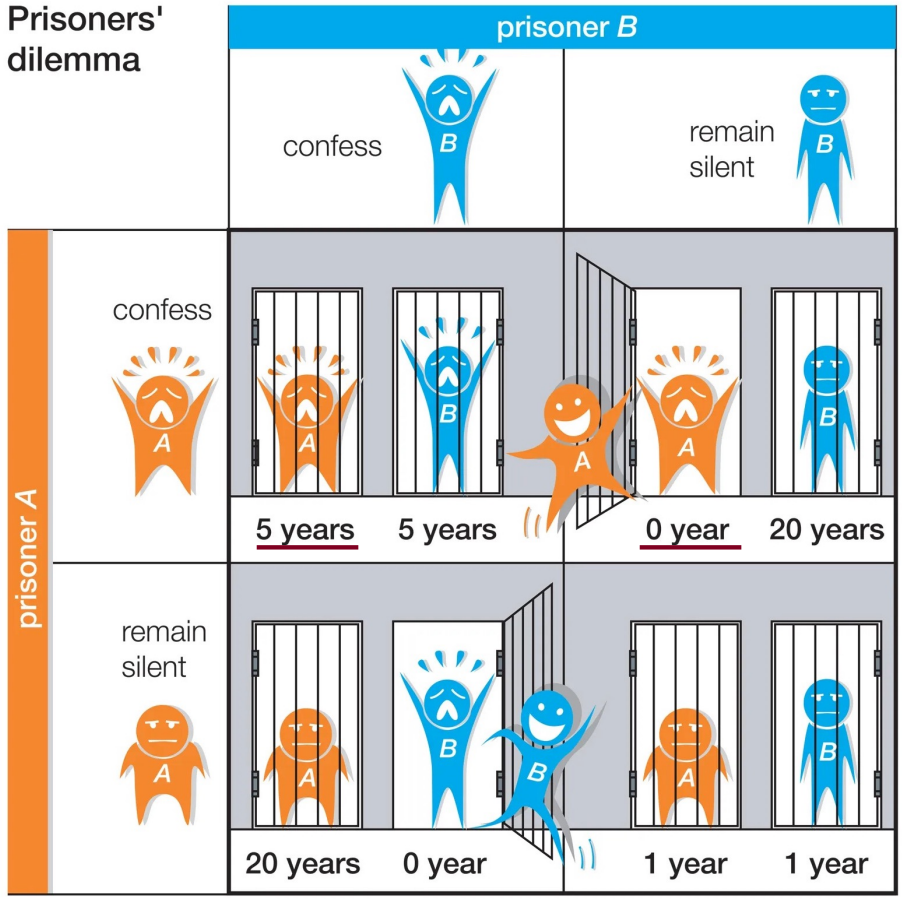
- Agents can't benefit from deviating from their equilibrium action
- At equilibrium each agent's action is the best response to others.



Crash Course on Non-cooperative Game Theory & Nash Equilibrium

→ We aim to find the **Nash equilibrium**.

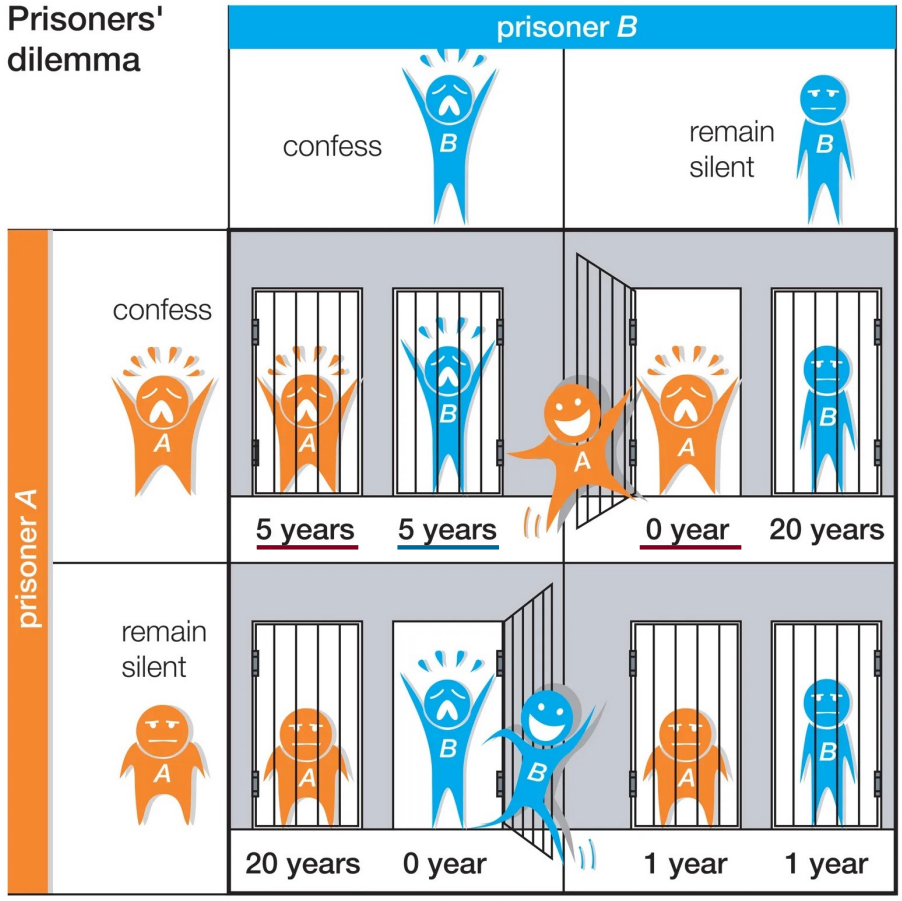
- Agents can't benefit from deviating from their equilibrium action
- At equilibrium each agent's action is the best response to others.



Crash Course on Non-cooperative Game Theory & Nash Equilibrium

→ We aim to find the **Nash equilibrium**.

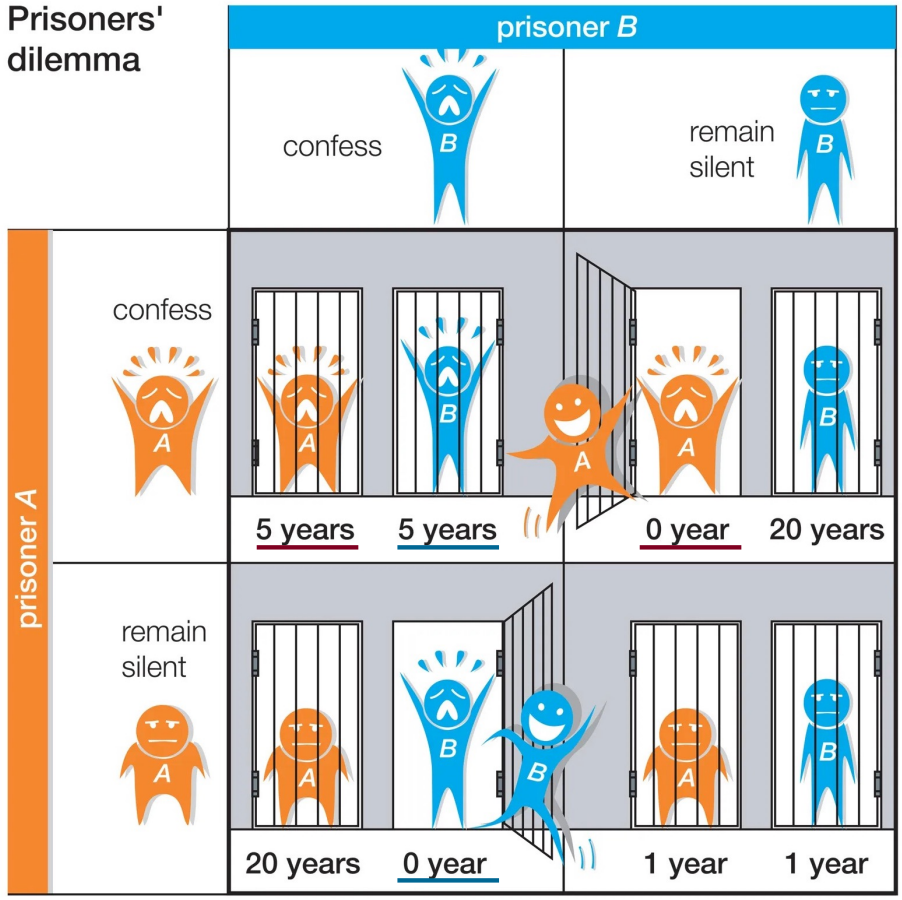
- Agents can't benefit from deviating from their equilibrium action
- At equilibrium each agent's action is the best response to others.



Crash Course on Non-cooperative Game Theory & Nash Equilibrium

→ We aim to find the **Nash equilibrium**.

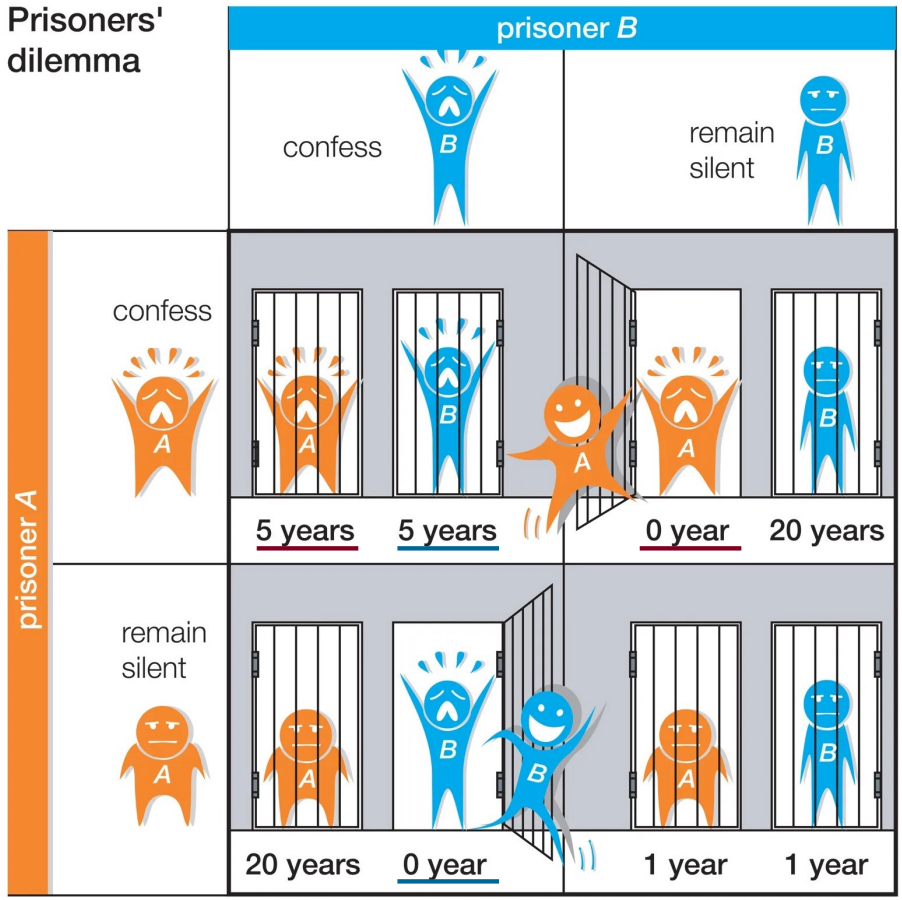
- Agents can't benefit from deviating from their equilibrium action
- At equilibrium each agent's action is the best response to others.



Crash Course on Non-cooperative Game Theory & Nash Equilibrium

→ We aim to find the **Nash equilibrium**.

- Agents can't benefit from deviating from their equilibrium action
- At equilibrium each agent's action is the best response to others.



→ This is a static, 2-player, finite action game: Life gets harder in higher dimensions!

Picture Credit: https://cdn.britannica.com/55/91955-050-F444D10D/dilemma-prisoners-participants-game-theory-communication-strategy.jpg

Going Back to the Mean Field Games

Our problems of interest: Dynamic, Stochastic, Large number of agents, Continuous action space.

Remember in Mean Field Games:

- → Assume $N \to \infty$.
- → Agents are identical and infinitesimal.
- → Agents interact symmetrically.
- → There are mean field interactions.
- → **Idea:** Focus on representative agent and her interactions with the distribution.

Going Back to the Mean Field Games

Our problems of interest: Dynamic, Stochastic, Large number of agents, Continuous action space.

Remember in Mean Field Games:

- $\rightarrow$ Assume $N \rightarrow \infty$.
- $\rightarrow$ Agents are identical and infinitesimal.
- $\rightarrow$ Agents interact symmetrically.
- $\rightarrow$ There are mean field interactions.
- $\rightarrow$ **Idea:** Focus on representative agent and her interactions with the distribution.
  - Since the representative agent is **infinitesimal**, the mean field can be taken as fixed when finding the best response for the representative agent. **(Optimality)**
  - Since all the agents are **identical**, we can assume everyone will behave similarly at the equilibrium. **(Fixed Point)**

Mathematical Formulation of Mean Field Game

For very large N, approximate the game between agents with **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

For very large N, approximate the game between agents with **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: If the pair $(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\mu}})$ 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,

then $(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\mu}})$ is a **Mean Field Nash equilibrium**.

Mathematical Formulation of Mean Field Game

For very large N, approximate the game between agents with **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: If the pair $(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\mu}})$ 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,

then $(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\mu}})$ is a **Mean Field Nash equilibrium**.

A **forward-backward stochastic differential equation system** (FBSDE) characterizes the Mean Field Nash equilibrium.

Mean Field Game Example

Let's extend Sibel's problem to the large population setup.

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

→ Everyone minimizes their costs and chooses their own effort levels.

→ Their project value dynamics depend on 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 Example: Nash Equilibrium Characterization

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

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

The Nash equilibrium control can be written as the minimizer of the Hamiltonian:

$$\hat{\alpha}_t = \arg\min_{\alpha_t} (\alpha_t + \bar{X}_t)\frac{Z_t}{\sigma} + \frac{1}{2}\alpha_t^2 - U(X_t)$$

We can see $\hat{\alpha}_t = -Z_t/\sigma$, where (X_t, Y_t, Z_t) solves forward backward stochastic differential equation (FBSDE) system:

$$dX_t = (-Z_t/\sigma + \bar{X}_t)dt + \sigma dW_t, \qquad X_0 = \zeta$$

$$dY_t = \left(\frac{1}{2\sigma^2}Z_t^2 - U(X_t)\right) dt + Z_t dW_t, \qquad Y_T = G(X_T).$$

Using Deep Learning to Solve Mean Field Games

Using Deep Learning to Find Mean Field Nash Equilibrium

We cannot use the same ideas from mean field control!

Instead, we need to solve the FBSDE that characterizes the Mean field Nash Equilibrium:

→ **Challenges:** Coupled, McKean-Vlasov type equations

→ Backward dynamics represents the dynamics of the value function (i.e., the expected minimized cost between time t and T when player starts from $X_t = x$).

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

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

where $\mu_t = \mathcal{L}(X_t)$ and $\hat{\alpha}_s = \hat{\alpha}_s(X_s, \mu_s, Z_s)$ is the minimizer of the Hamiltonian:

$$b(s, X_s, \alpha_s, \mu_s)\frac{Z_s}{\sigma} + f(s, X_s, \alpha_s, \mu_s)$$

Using Deep Learning to Find Mean Field Nash Equilibrium

We need to solve the FBSDE that characterizes the Mean field Nash Equilibrium:

→ Backward dynamics represents the dynamics of the value function (i.e., the expected minimized cost between time t and T when player starts from $X_t = x$).

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

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

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

Using Deep Learning to Find Mean Field Nash Equilibrium

We need to solve the FBSDE that characterizes the Mean field Nash Equilibrium:

→ Backward dynamics represents the dynamics of the value function (i.e., the expected minimized cost between time t and T when player starts from $X_t = x$).

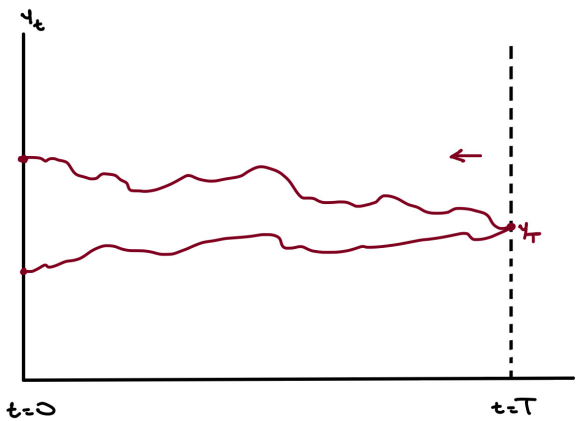
$$\text{State dynamics} \leftarrow X_t = \zeta + \int_0^t b(s, X_s, \hat{\alpha}_s, \mu_s)ds + \int_0^t \sigma dW_s$$

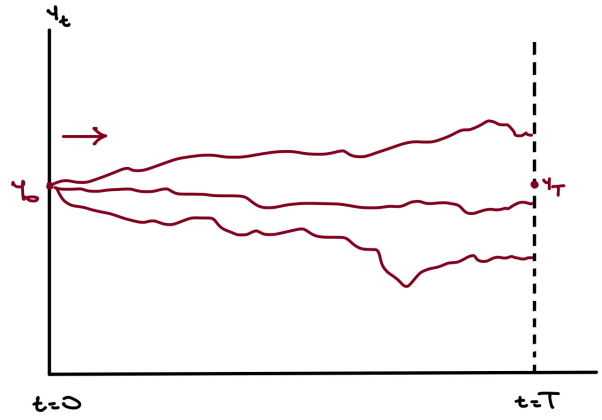
$$\text{Value function} \leftarrow Y_t = g(X_T, \mu_T) + \int_t^T f(s, X_s, \hat{\alpha}_s, \mu_s)ds - \int_t^T Z_s dW_s$$

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

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

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





Using Deep Learning to Find Mean Field Nash Equilibrium

→ Now we have forward-forward stochastic differential equations:

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

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

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 before:** Discretize the time.

Using Deep Learning to Find Mean Field Nash Equilibrium

→ Now we have forward-forward stochastic differential equations:

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

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

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 before:** Discretize the time.

→ **Similar to MFC:** There is a distribution: Monte Carlo simulate a population of size N and use the empirical distribution: μ^N.

Using Deep Learning to Find Mean Field Nash Equilibrium

→ Now we have forward-forward stochastic differential equations:

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

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

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 before:** Discretize the time.

→ **Similar to MFC:** There is a distribution: Monte Carlo simulate a population of size N and use the empirical distribution: μ^N.

→ Approximate **controls** $Y_0 = y_{0,\theta_1}(X_0)$ and $Z_t = z_{\theta_2}(t, X_t)$ while trying to **shoot** the **terminal condition**: $Y_T = g(X_T, \mu_T)$, i.e.:

$$\min_{\boldsymbol{\theta}=(\theta_1,\theta_2)} \sum_{i=1}^{N} \left(Y_T^{i,\boldsymbol{\theta}} - g(X_T^{i,\boldsymbol{\theta}}, \mu_T^{N,\boldsymbol{\theta}})\right)^2$$

→ Continuous state application: Carmona-Laurière (2021);
Finite state application: Aurell-Carmona-GD.-Laurière (2022a).

Pseudo-Code for Mean Field Game

Algorithm 4 Simulation of FFSDE trajectories

Input: Time horizon T; time increment Δt; initial distribution μ_0; population size N

Output: Trajectories of X, Y, and Z processes.

Let $n = 0$, $t_0 = 0$, sample $X_0^i \sim \mu_0$, and set $Y_0^i = y_{0,\theta_1}(X_0^i)$ for all $i \in \{1, 2, \ldots, N\}$

while $n \times \Delta t = t_n < T$ **do**

  Set $Z_{t_n}^i = z_{\theta_2}(t_n, X_{t_n}^i)$ for all $i \in \{1, 2, \ldots, N\}$

  Approximate the mean field using the empirical distribution of $X_{t_n}^i$ and call it $\mu_{t_n}^N$

  Set $\hat{\alpha}_{t_n}^i = \hat{\alpha}(t_n, X_{t_n}^i, \mu_{t_n}^N, Z_{t_n}^i)$, for all $i \in \{1, 2, \ldots, N\}$

  Sample $\Delta W_{t_n}^i \sim \mathcal{N}(0, \Delta t)$, for all $i \in \{1, 2, \ldots, N\}$

  Let $X_{t_{n+1}}^i = X_{t_n}^i + b(t_n, X_{t_n}^i, \hat{\alpha}_{t_n}^i, \mu_{t_n}^N)\Delta t + \sigma \Delta W_{t_n}^i$ for all $i \in \{1, 2, \ldots, N\}$

  Let $Y_{t_{n+1}}^i = Y_{t_n}^i - f(t_n, X_{t_n}^i, \hat{\alpha}_{t_n}^i, \mu_{t_n}^N)\Delta t + Z_{t_n}^i \Delta W_{t_n}^i$, for all $i \in \{1, 2, \ldots, N\}$

  Set $t_n = t_n + \Delta t$ and $n = n + 1$

end while

Set $N_T = n$, $t_{N_T} = T$

return $(X_{t_n}^i, Y_{t_n}^i, Z_{t_n}^i)_{n=0,\ldots,N_T; i=1,\ldots,N}$

Remark: We update the parameters of neural networks by using stochastic gradient descent to minimize the shooting cost:

$$\min_{\boldsymbol{\theta}=(\theta_1,\theta_2)} \sum_{i=1}^{N} \left(Y_T^{i,\boldsymbol{\theta}} - g(X_T^{i,\boldsymbol{\theta}}, \mu_T^{N,\boldsymbol{\theta}}) \right)^2$$

Extensions

Many Extensions are Possible

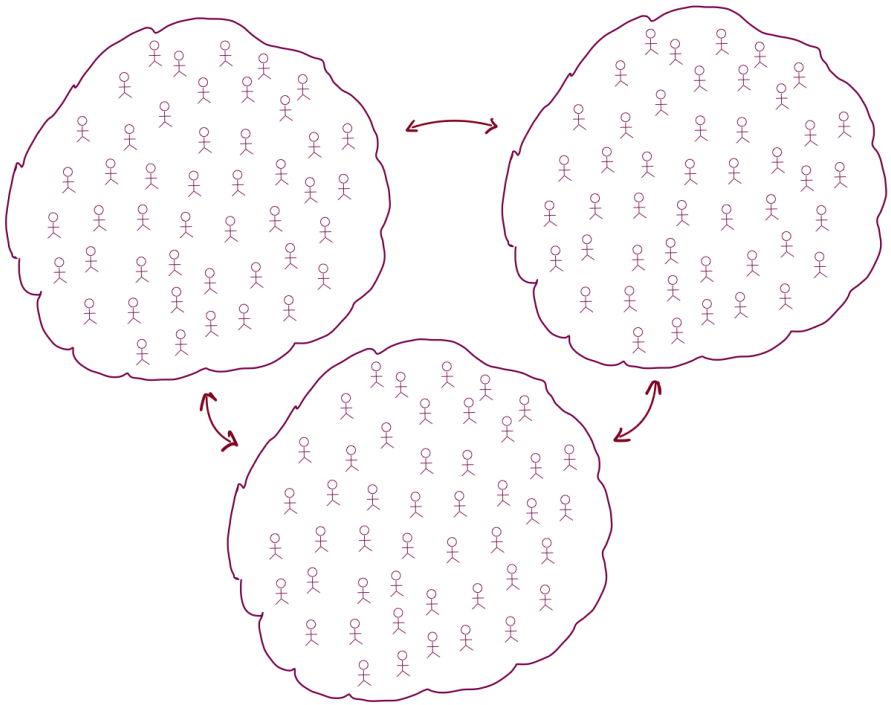
For example:

- → Multiple large populations of agents
- → More complex interactions among the agents: Graphon Games
- → Addition of significant agents
  - Major-minor mean field games
  - Stackelberg mean field games

Multiple large populations & Complex Interactions

→ Multiple large populations: Previous ideas can be extended.

→ More complex interactions: **Graphon games** can be used for modeling.



Graphon Games[4]

→ It is used to model:

- Heterogeneous agents
- Non-symmetric interactions

[4] Delarue (2017), Parise-Özdağlar (2019), Carmona-Cooney-Graves-Laurière (2019), Caines-Huang (2018, 2019), Bayraktar-Chakraborty-Wu (2020), Aurell-Carmona-Laurière (2021), Aurell-Carmona-GD.-Laurière (2021)

Graphon Games[4]

→ It is used to model:

- Heterogeneous agents
- Non-symmetric interactions

→ We can't focus on a representative player anymore. The agents are indexed by $x \in I := [0, 1]$.

[4] Delarue (2017), Parise-Özdağlar (2019), Carmona-Cooney-Graves-Laurière (2019), Caines-Huang (2018, 2019), Bayraktar-Chakraborty-Wu (2020), Aurell-Carmona-Laurière (2021), Aurell-Carmona-GD.-Laurière (2021)

Graphon Games[4]

→ It is used to model:
- Heterogeneous agents
- Non-symmetric interactions

→ We can't focus on a representative player anymore. The agents are indexed by $x \in I := [0,1]$.

→ The **graphon function** $w : I \times I \to [0,1]$ gives the connection strength between two agents.

[4] Delarue (2017), Parise-Özdağlar (2019), Carmona-Cooney-Graves-Laurière (2019), Caines-Huang (2018, 2019), Bayraktar-Chakraborty-Wu (2020), Aurell-Carmona-Laurière (2021), Aurell-Carmona-GD.-Laurière (2021)

Graphon Games[4]

→ It is used to model:
- Heterogeneous agents
- Non-symmetric interactions

→ We can't focus on a representative player anymore. The agents are indexed by $x \in I := [0,1]$.

→ The **graphon function** $w : I \times I \to [0,1]$ gives the connection strength between two agents.

→ **Example:**
- In a mean field game, interaction of the representative agent with the population can be through $\bar{X}$.
- In graphon games agent x interacts with the *weighted* population: $\int_I w(x,y)\bar{X}^y dy$.

→ The solution is characterized by a **continuum** of FBSDE. We can still apply the **shooting method**.

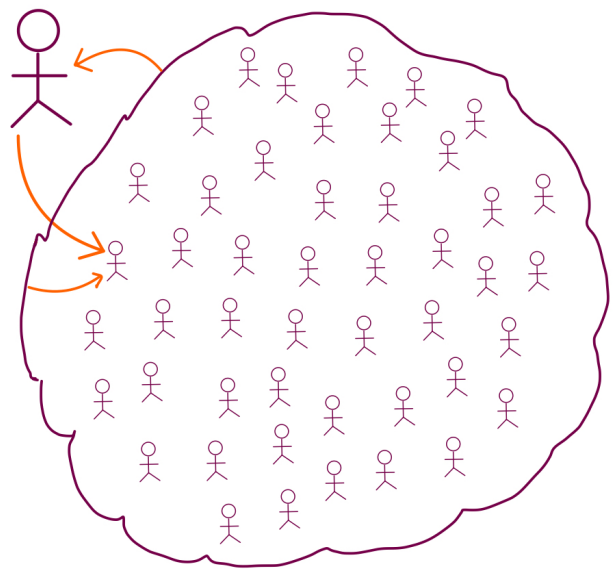
[4] Delarue (2017), Parise-Özdağlar (2019), Carmona-Cooney-Graves-Laurière (2019), Caines-Huang (2018, 2019), Bayraktar-Chakraborty-Wu (2020), Aurell-Carmona-Laurière (2021), Aurell-Carmona-GD.-Laurière (2021)

Addition of a Significant Player

→ **Situation:** A player can have a **significant effect** on the whole system or they can be in the role of a **regulator**.

→ Examples of a player with a significant effect:

- Modeling hedge funds and individual (small) traders.
- Modeling large banks and regional banks together.
- A public-health authority deciding on epidemic mitigation policies for a large society.



Addition of a Significant Player

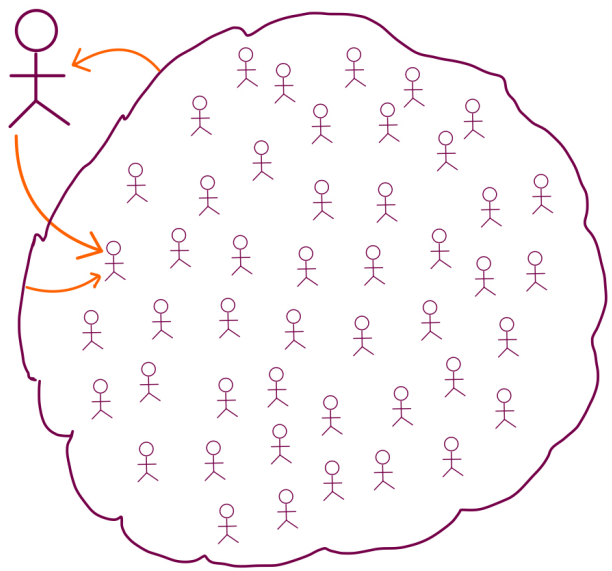
→ **Situation:** A player can have a **significant effect** on the whole system or they can be in the role of a **regulator**.

→ Examples of a player with a significant effect:

- Modeling hedge funds and individual (small) traders.
- Modeling large banks and regional banks together.
- A public-health authority deciding on epidemic mitigation policies for a large society.

→ Two ways to model:

- Major-minor mean field games
- Stackelberg mean field games



Addition of a Significant Player

→ **Situation:** A player can have a **significant effect** on the whole system or they can be in the role of a **regulator**.

→ Examples of a player with a significant effect:

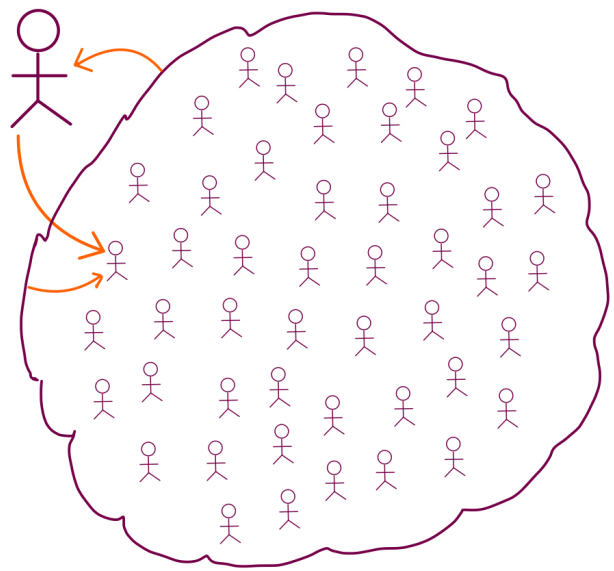
- Modeling hedge funds and individual (small) traders.
- Modeling large banks and regional banks together.
- A public-health authority deciding on epidemic mitigation policies for a large society.

→ Two ways to model:

- Major-minor mean field games
- Stackelberg mean field games

→ Major-minor mean field games:

- All agents are in Nash equilibrium.
- Technical challenges if the major agent's state has noise.
- Ideas still can be extended.



Addition of a Significant Player

→ **Situation:** A player can have a **significant effect** on the whole system or they can be in the role of a **regulator**.

→ Examples of a player with a significant effect:

- Modeling hedge funds and individual (small) traders.
- Modeling large banks and regional banks together.
- A public-health authority deciding on epidemic mitigation policies for a large society.

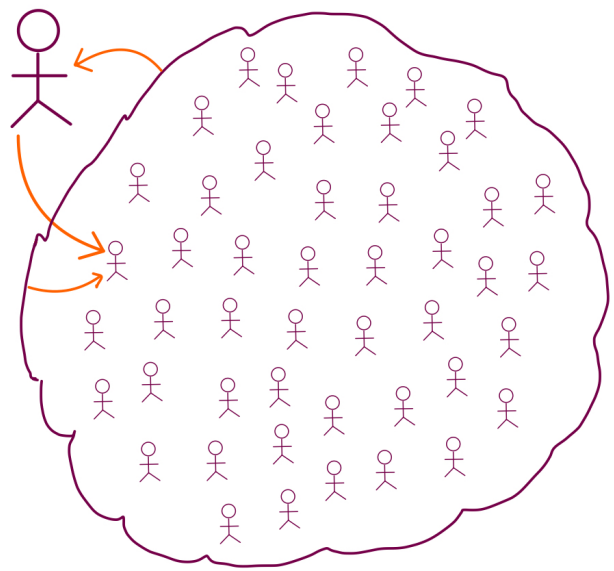
→ Two ways to model:

- Major-minor mean field games
- Stackelberg mean field games

→ Major-minor mean field games:

- All agents are in Nash equilibrium.
- Technical challenges if the major agent's state has noise.
- Ideas still can be extended.

→ **We will discuss Stackelberg mean field games.**



Stackelberg Mean Field Games:

Definition & Deep Learning Approach

Stackelberg Mean Field Games Motivation

How can a regulator design the optimal policies/incentives in order to get the best outcomes when she interacts with many agents who prioritize their own objectives?

→ Some examples:

- A **regulator** incentivizes **large number of banks** borrowing and lending from a central bank to minimize the expected number of defaults.
- A **principal** wants to write an optimal payment contract for a **large number of employees** to maximize their expected return.
- A **government** wants to find optimal carbon tax levels for **electricity producers** to attain the targeted reduction in the carbon emission levels.
- **Companies** want to optimize their advertisement decisions while interacting with **consumers** to maximize their profits.
- **Public-health authorities** want to choose policies to mitigate an epidemic in a **society**.

→ This type of problems require introduction of another equilibrium notion between the regulators and the large populations: **Stackelberg equilibrium**

What is Stackelberg Equilibrium?

In Stackelberg equilibrium:

→ There is a leader (principal) and a follower (agent).

→ The leader chooses incentives and follower gives the best response to these incentives.

→ The leader optimizes incentives by taking into account the best response reaction of the follower.

[5] Başar (1984, 1989), Holmström-Milgrom (1987), Sannikov (2008, 2013), Cvitanić-Possamaï-Touzi (2018)

What is Stackelberg Equilibrium?

In Stackelberg equilibrium:

→ There is a leader (principal) and a follower (agent).

→ The leader chooses incentives and follower gives the best response to these incentives.

→ The leader optimizes incentives by taking into account the best response reaction of the follower.

→ **Stackelberg Equilibrium[5] is different than Nash Equilibrium!**

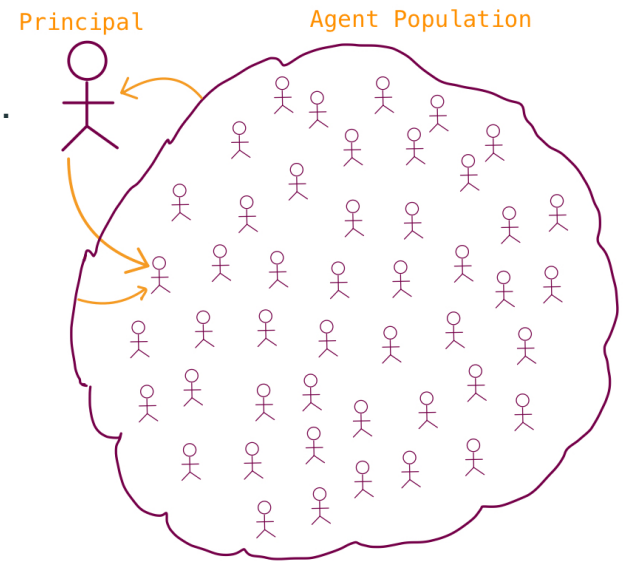
[5] Başar (1984, 1989), Holmström-Milgrom (1987), Sannikov (2008, 2013), Cvitanić-Possamaï-Touzi (2018)

What is Stackelberg Equilibrium?

In Stackelberg equilibrium:

- → There is a leader (principal) and a follower (agent).
- → The leader chooses incentives and follower gives the best response to these incentives.
- → The leader optimizes incentives by taking into account the best response reaction of the follower.
- → **Stackelberg Equilibrium[5] is different than Nash Equilibrium!**

In our setup: Instead of one follower, we have a population of large number of noncooperative agents.



[5] Başar (1984, 1989), Holmström-Milgrom (1987), Sannikov (2008, 2013), Cvitanić-Possamaï-Touzi (2018)

What is Stackelberg Equilibrium?

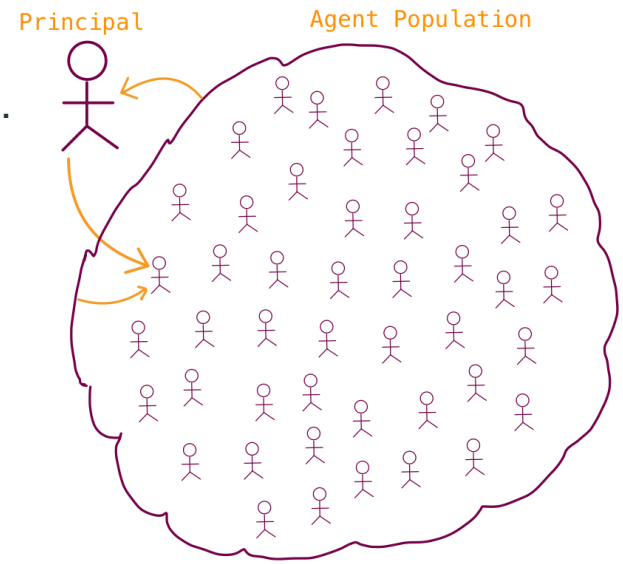
In Stackelberg equilibrium:

- → There is a leader (principal) and a follower (agent).
- → The leader chooses incentives and follower gives the best response to these incentives.
- → The leader optimizes incentives by taking into account the best response reaction of the follower.
- → **Stackelberg Equilibrium[5] is different than Nash Equilibrium!**

In our setup: Instead of one follower, we have a population of large number of noncooperative agents.

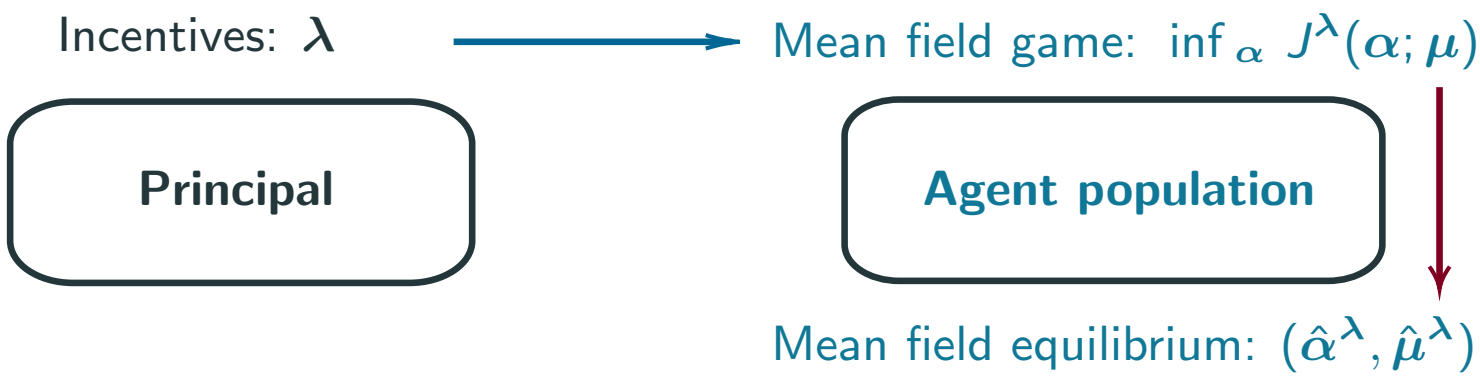
⇒ **Population will be in Nash Equilibrium given the incentives of the principal!**

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



[5] Başar (1984, 1989), Holmström-Milgrom (1987), Sannikov (2008, 2013), Cvitanić-Possamaï-Touzi (2018)

Stackelberg MFG: Agent Population



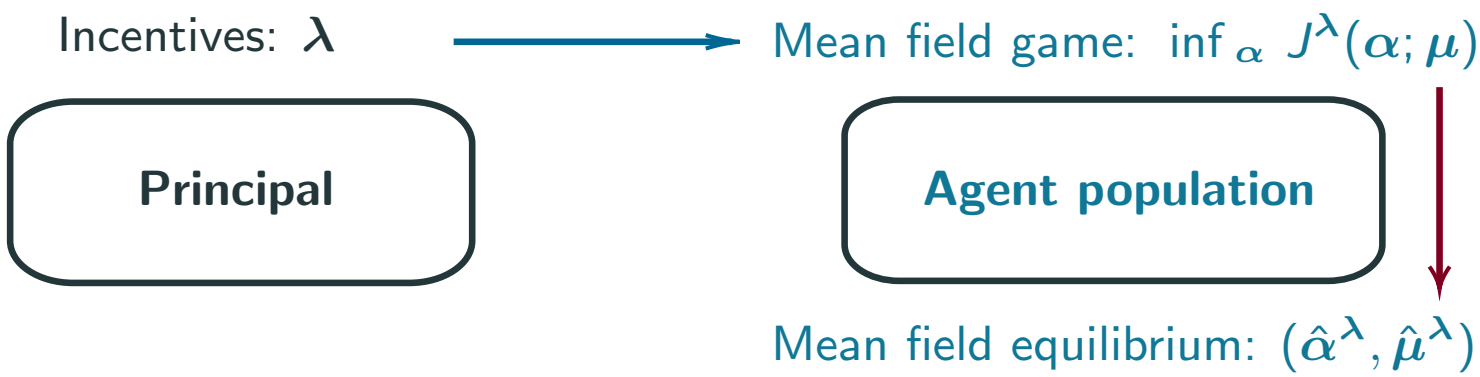
The **cost** for the **representative agent** using control $\boldsymbol{\alpha} \in \mathbb{A}$ when facing a population with state distribution $\boldsymbol{\mu}$ is

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

where $\boldsymbol{\lambda}$ is principal's incentive choice. The agent's state X_t has the following dynamics:

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

Stackelberg MFG: Agent Population



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

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

where λ is principal's incentive choice. The agent's state X_t has the following dynamics:

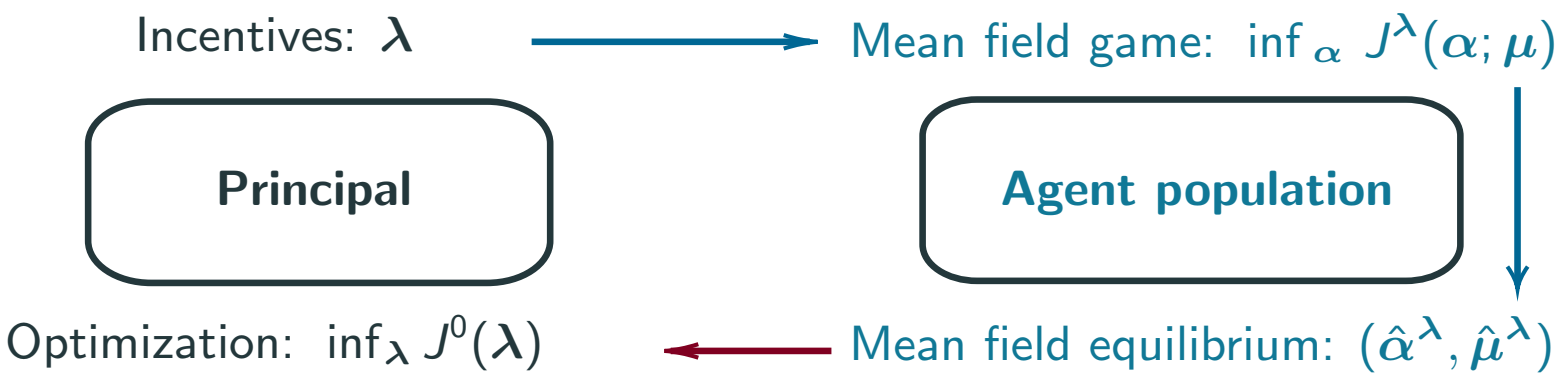
$$dX_t = b(t, X_t, \alpha_t, \mu_t; \lambda_t)dt + \sigma dW_t, \qquad X_0 = \zeta \sim \mu_0.$$

Different than before: We have the principal's incentives.

The Mean Field Nash Equilibrium can still be characterized with an **FBSDE** given the principal's incentives.

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

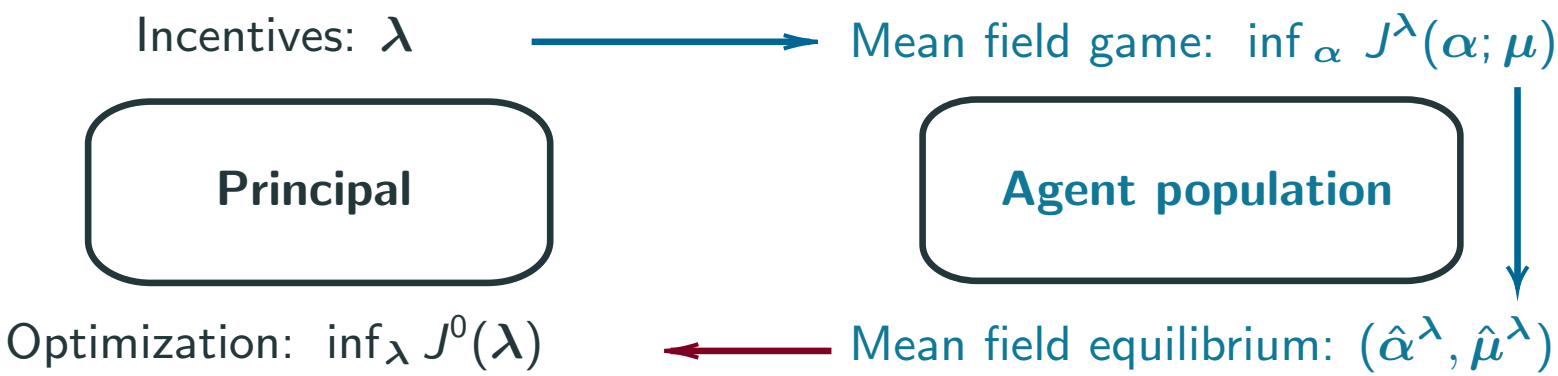
Stackelberg MFG: Principal



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

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

Stackelberg MFG: Principal



The principal's **cost** for 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:

$\rightarrow$ Given any incentive $\boldsymbol{\lambda}$, the agent population should be in mean field game Nash equilibrium: $(\hat{\boldsymbol{\alpha}}^{\boldsymbol{\lambda}}, \hat{\mu}^{\boldsymbol{\lambda}})$

Stackelberg Mean Field Game Problem

The full problem becomes:

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

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

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

This is a bi-level problem! One way to solve this can be an iterative process[6] or we need to rewrite the problem[7].

[6] Campbell, Chen, Shrivats and Jaimungal (2021)

[7] Aurell-Carmona-GD.-Laurière (2022), GD.-Laurière (2025)

Example: Systemic Risk Model with a Regulator

Extending the model of Carmona-Fouque-Sun (2013) to add a regulator.[8]

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

$$\inf_{\lambda} \int_0^T (\lambda_t - \lambda_t^{\text{aim}})^2 dt + \gamma \mathbb{E}\Big[\mathbb{1}_{X_T < D}\Big]$$

for exogenous $\underbrace{\lambda_t^{\text{aim}}}_{\text{aimed level}}$ and $\underbrace{D < 0}_{\text{Default value}}$.

Agent Population (Banks): Controls the lending and borrowing amounts α_t.

The objective of the representative bank is given as

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

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.

[8] GD.-Laurière (2025)

How to rewrite this model to end up with a single level 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 and Y_t are determined by the following forward backward stochastic differential equations:

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

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

Rewriting the Problem I: Rewriting Backward Equation

We have the following objective

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

where the trajectories of X_t and Y_t are determined by the following forward forward stochastic differential equations:

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

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

with the constraint

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

Controls of the problem are now λ, 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 solve it!

$\rightarrow$ Our constrained problem is:

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

with the constraint

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

and where the trajectories of X_t and Y_t are determined by the previously introduced forward forward stochastic differential equations.

Rewriting the Problem II: Introducing the Penalty

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

$\rightarrow$ Our constrained problem is:

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

with the constraint

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

and where the trajectories of X_t and Y_t are determined by the previously introduced forward forward stochastic differential equations.

$\rightarrow$ Introduce the penalty function and in this way the problem becomes:

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

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:

$$\inf_{\lambda, Z, Y_0} \underbrace{\int_0^T f_0(t, \mu_t^{\lambda,Z,Y_0}, \lambda_t)dt + g_0(\mu_T^{\lambda,Z,Y_0}, \lambda_T)}_{\text{Cost of the principal: } J^0}$$
$$+ \nu \underbrace{\mathbb{E}\left[\mathbf{P}(Y_T^{\lambda,Z,Y_0} - g(X_T^{\lambda,Z,Y_0}, \mu_T^{\lambda,Z,Y_0}; \lambda_T))\right]}_{\text{Penalty: } \bar{\mathbf{P}}},$$

where

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

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

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:

→ Parameterize the new controls $(\boldsymbol{\lambda}, \boldsymbol{Z}, Y_0)$ and implement NNs to approximate.

→ Monte Carlo simulate the trajectories of X_t and Y_t by using the time-discretized forward forward stochastic differential equations and for the distribution use the empirical distribution of state X_t.

→ Minimize over NN parameters the time-discretized (and penalized) cost of the principal.

Remark: These results hold even if policies are in more complicated forms such as $\lambda(t, X_t)$ or $\lambda(t, X_t, \mu_t)$.

Pseudo-Code for Stackelberg MFG

Algorithm 5

Input: number of particles N; time horizon T; time increment Δt; initial distribution μ_0
Output: Trajectories of X, Y and Z processes

Let $n = 0$, $t_0 = 0$; sample $X_0^i \sim \mu_0$ and set $Y_0^i = y_{0,\theta_1}(X_0^i)$, for all $i \in \{1, \dots, N\}$
while $n \times \Delta t = t_n < T$ **do**
  Set $Z_{t_n}^i = z_{\theta_2}(t_n, X_{t_n}^i)$, $\Lambda_{t_n} = \lambda_{\theta_3}(t_n)$, for all $i \in \{1, \dots, N\}$
  Approximate the mean field using the empirical distribution of $X_{t_n}^i$ and call it $\mu_{t_n}^N$
  Set $\hat{\alpha}_{t_n}^i = \hat{\alpha}(t_n, X_{t_n}^i, \mu_{t_n}^N, {\sigma^{-1}}^\top Z_{t_n}^i; \Lambda_{t_n})$, for all $i \in \{1, \dots, N\}$
  Sample $\Delta W_{t_n}^i \sim \mathcal{N}(0, \Delta t)$, for all $i \in \{1, \dots, N\}$
  Let $X_{t_{n+1}}^i = X_{t_n}^i + b(t_n, X_{t_n}^i, \hat{\alpha}_{t_n}^i, \mu_{t_n}^N; \Lambda_{t_n})\Delta t + \sigma \Delta W_{t_n}^i$, for all $i \in \{1, \dots, N\}$
  Let $Y_{t_{n+1}}^i = Y_{t_n}^i - f(t_n, X_{t_n}^i, \hat{\alpha}_{t_n}^i, \mu_{t_n}^N; \Lambda_{t_n})\Delta t + (Z_{t_n}^i)^\top \Delta W_{t_n}^i$, for all $i \in \{1, \dots, N\}$
  Set $t_n = t_n + \Delta t$ and $n = n + 1$
end while
Set $N_T = n$, $t_{N_T} = T$
return $(X_{t_n}^i, Y_{t_n}^i, Z_{t_n}^i, \Lambda_{t_n})_{n=0,\dots,N_T}$, $i \in [\![N]\!]$ and $(t_n)_{n=0,\dots,N_T}$

Remark: We update the parameters of neural networks by using stochastic gradient descent to minimize the penalized cost:

$$\sum_{n=0}^{N_T-1} f_0\left(t_n, \mu_{t_n}^N, \lambda_{\theta_1}(t_n)\right)\Delta t + g_0\left(\mu_T^N, \lambda_{\theta_1}(T)\right) + \frac{1}{N}\sum_{i=1}^{N} \nu\left(Y_T^i - g\left(X_T^i, \mu_T^N, \lambda_{\theta_1}(T)\right)\right)^2$$

Numerical Approach: Extensions

→ We can implement this numerical approach to 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*.

More examples

Example 1: Systemic Risk Model with a Regulator[9]

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

$$\inf_{\lambda} \int_0^T (\lambda_t - \lambda_t^{\text{aim}})^2 dt + \gamma \mathbb{E}\Big[\mathbb{1}_{X_T < D}\Big]$$

for exogenous $\underbrace{\lambda_t^{\text{aim}}}_{\text{aimed level}}$ and $\underbrace{D < 0}_{\text{Default value}}$.

Agent Population (Banks): Controls the lending and borrowing amounts α_t.

The objective of the representative bank is given as

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

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.

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

Solutions: Systemic Risk with a Regulator (I)

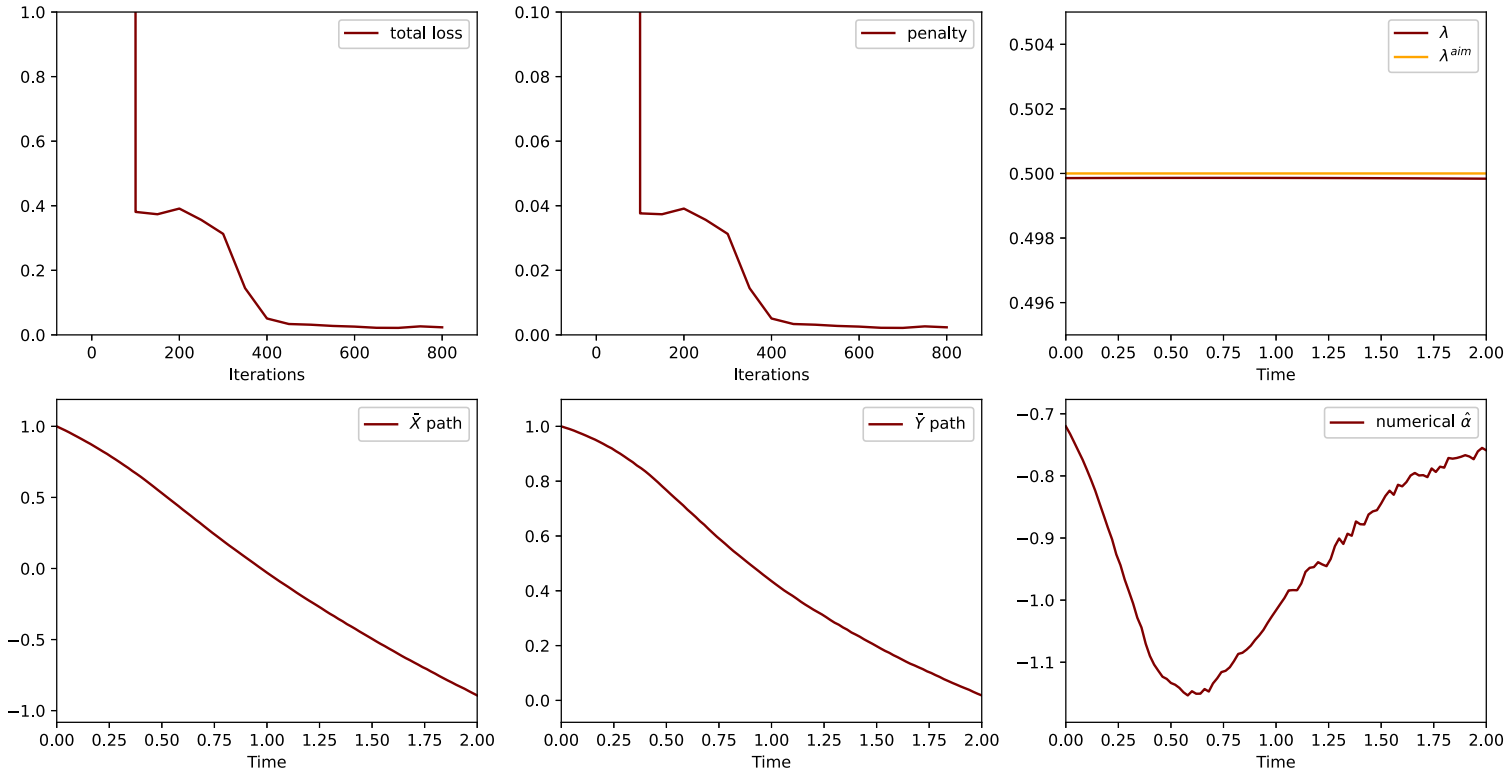


Figure 4: Systemic Risk Model with $\gamma = 0$.

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 (II)

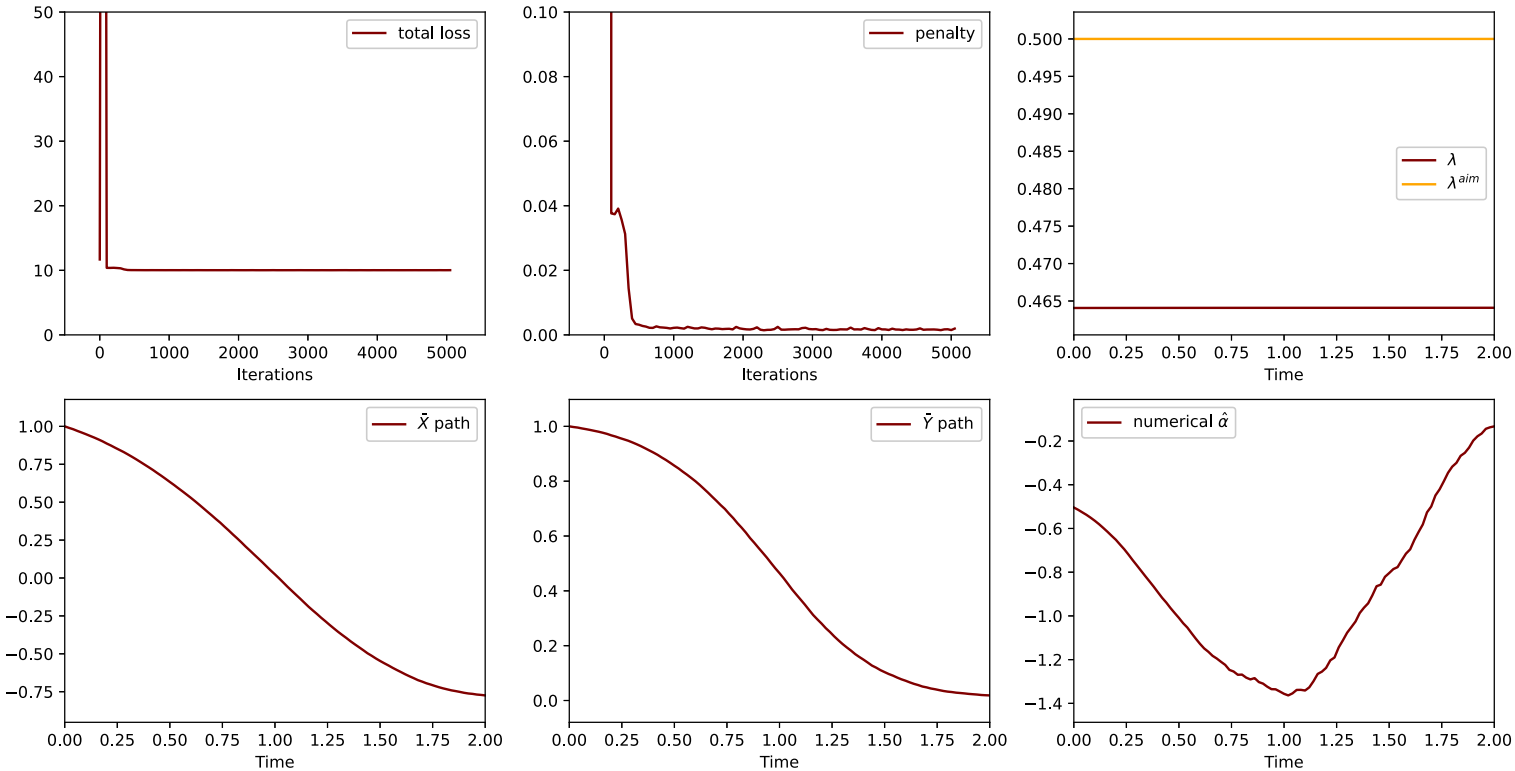


Figure 5: Systemic Risk Model with $\gamma = 10$.

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 (III: Higher Dim.)

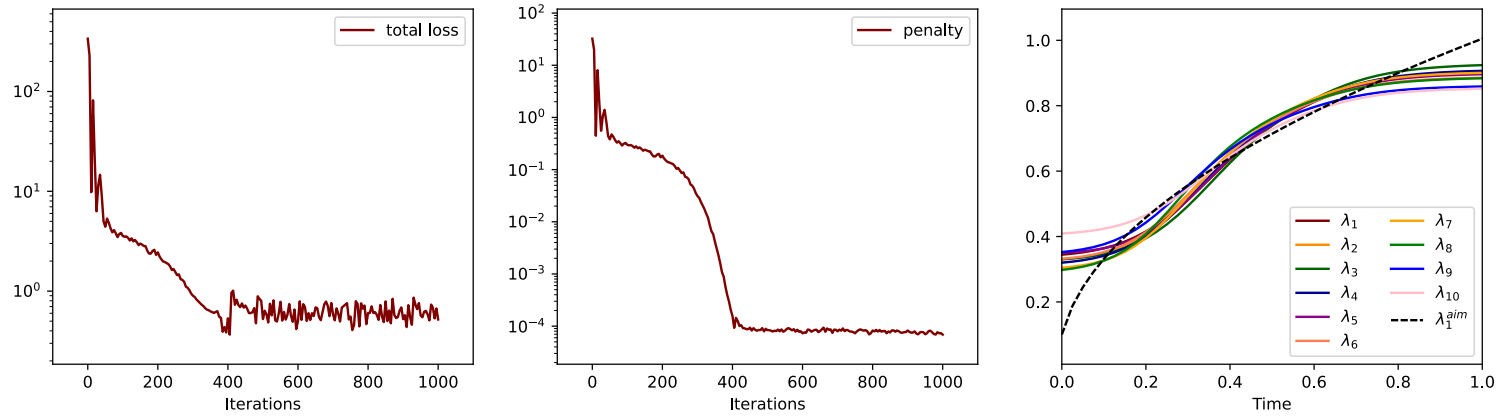


Figure 6: Systemic Risk Model with higher state and control dimensions and nonlinearity.

Example 2: Contract Theory Model with a Principal and Many Agents[10]

Principal: Proposes incentive: 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

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

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

Solutions: Interactions through the Mean of Controls

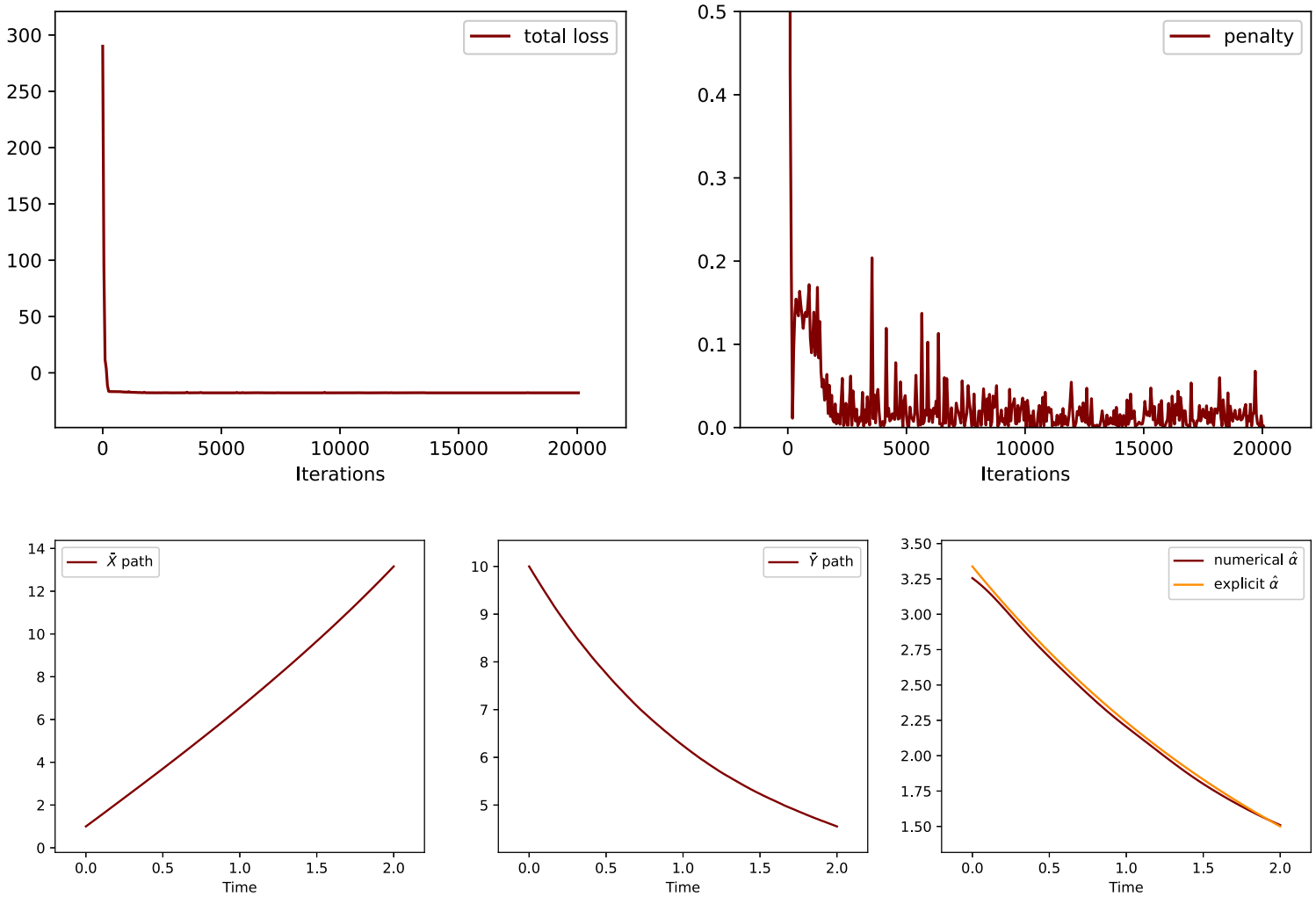


Figure 7: $dX_t = \left(\alpha_t + aX_t + \beta_2 \bar{\alpha} dt\right) dt + dW_t$

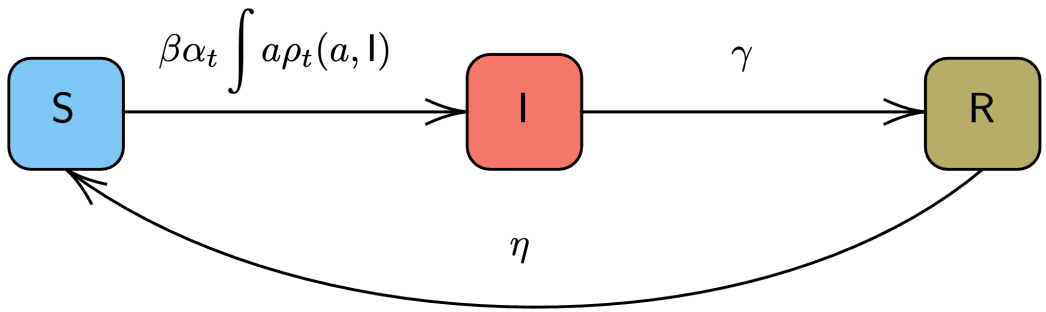
T	Δt	μ_0	σ	k	a	β_1	β_2	γ
2.0	0.02	δ_1	1.0	1.0	0.4	0.0	0.5	0

Example 3: Mitigating Epidemics (I): Intuitions[11]

(If time permits.)

→ **Agent Population:**

- → **Control:** Socialization levels
- → **Objectives:** Following the policies, minimizing the cost of treatment if infected



→ **Principal:**

- → **Control:** Social distancing measures, stimulus payment
- → **Objectives:** Following the recommendations from healthcare professionals and flattening the *curve*

[11] Aurell, Carmona, GD., Laurière (2022)

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

Control: Socialization Level: α_t

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

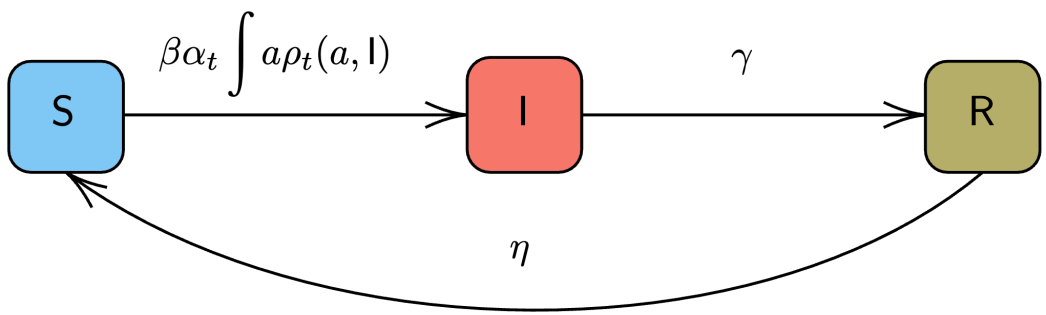
Objective:

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

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

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

State Dynamics:



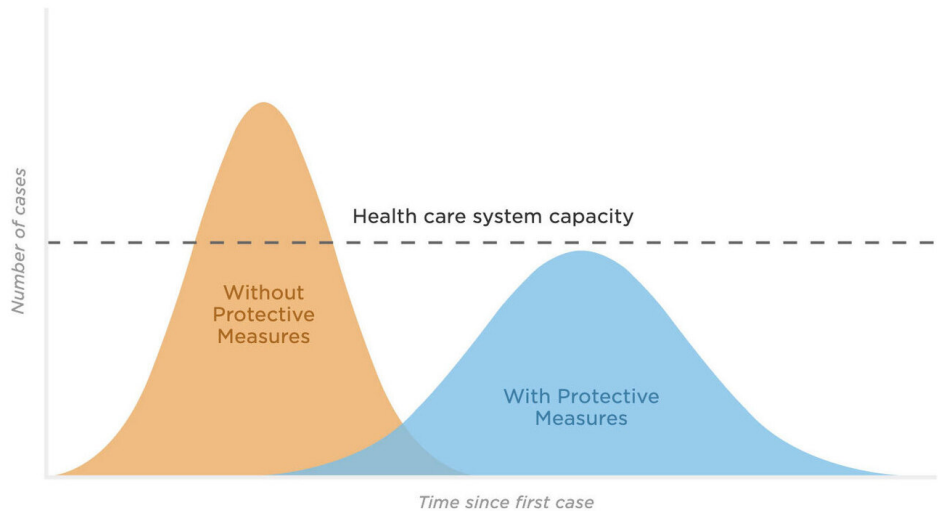
Example 3: Mitigating Epidemics (III): Principal's Model

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 $c_{\text{Inf}} > 0$.



[4]

[4] 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

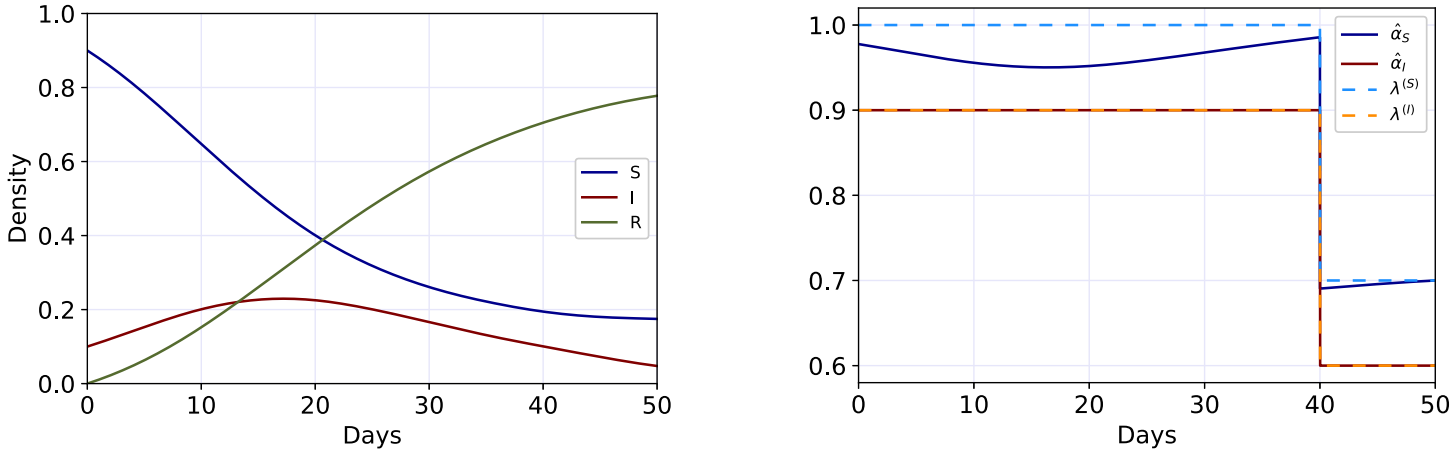


Figure 10: Late lockdown, **explicit** solution. Evolution of the population state distribution (left), evolution of the controls (right).

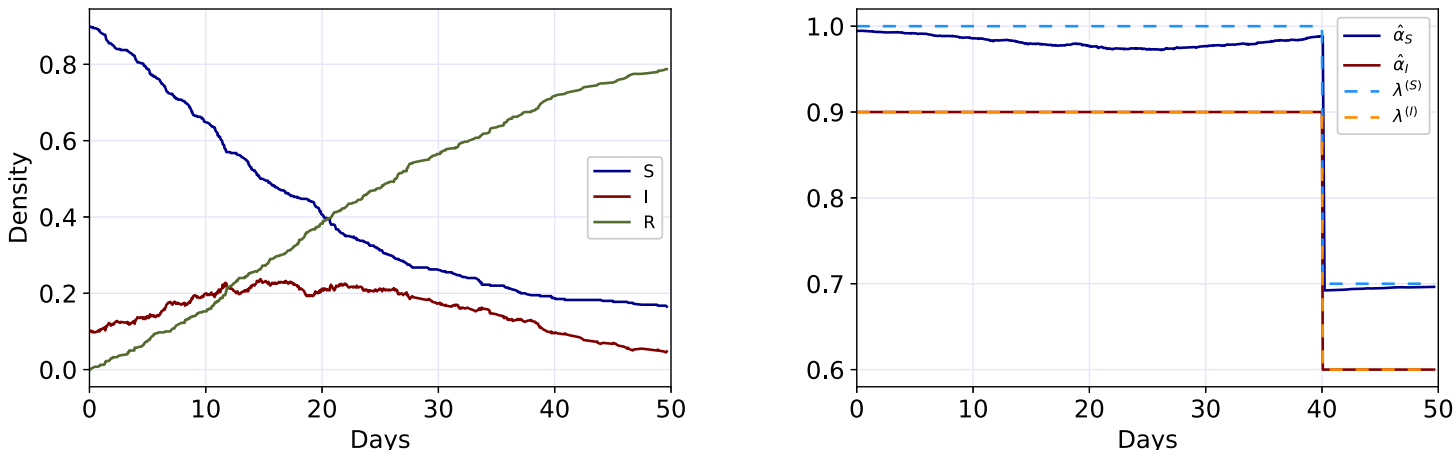


Figure 11: Late lockdown, **numerical** solution. Evolution of the population state distribution (left), evolution of the controls (right).

Solutions: SEIRD Stackelberg Mean Field Game

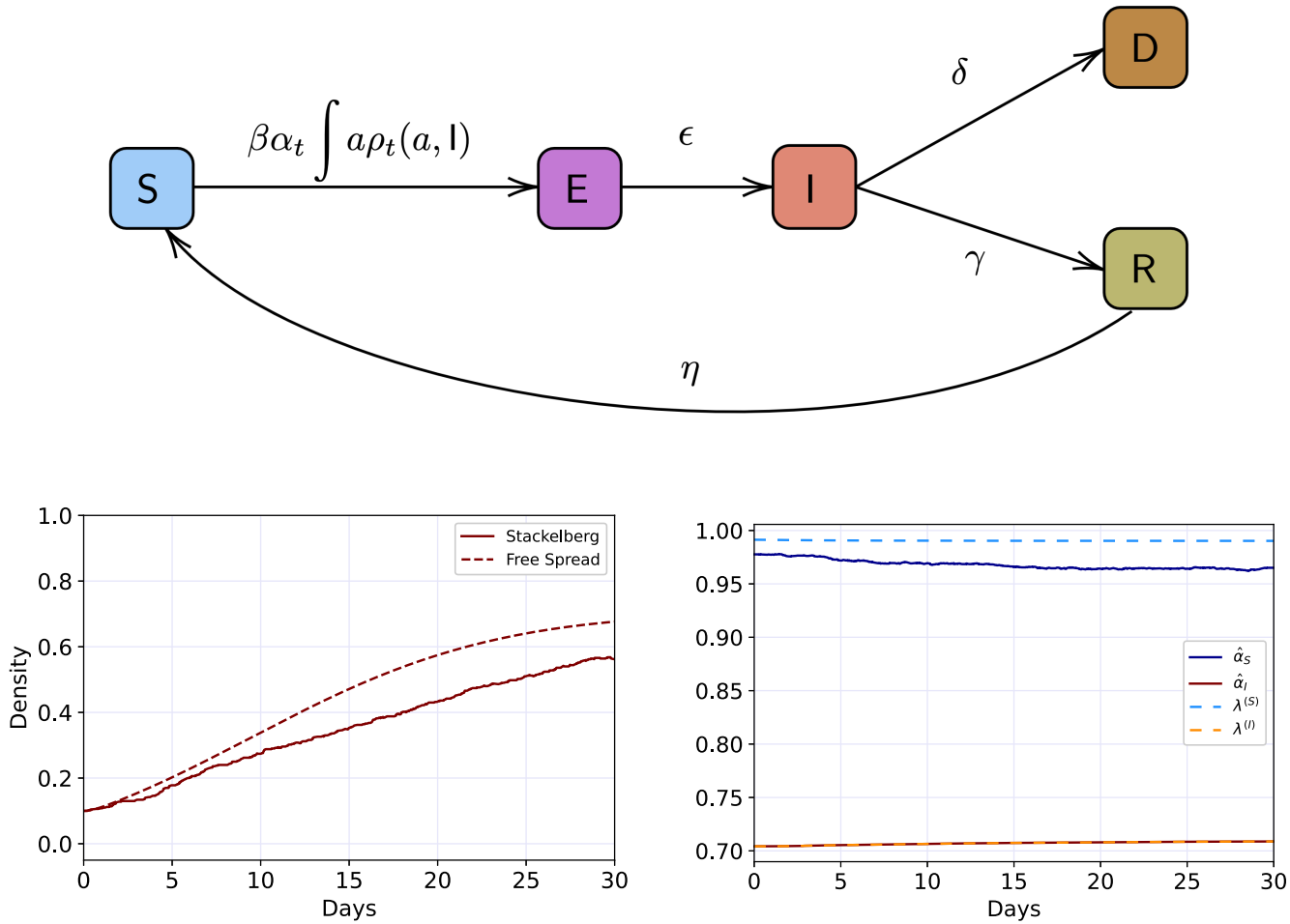


Figure 12: SEIRD Dynamics (top). SEIRD Stackelberg MFG in comparison with free spread SEIRD dynamics (bottom): Comparison of the Cumulative Density of Infected people (left); Evolution of the controls (right).

T	p^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

Solutions: SEIRD Power-Law Graphon Game

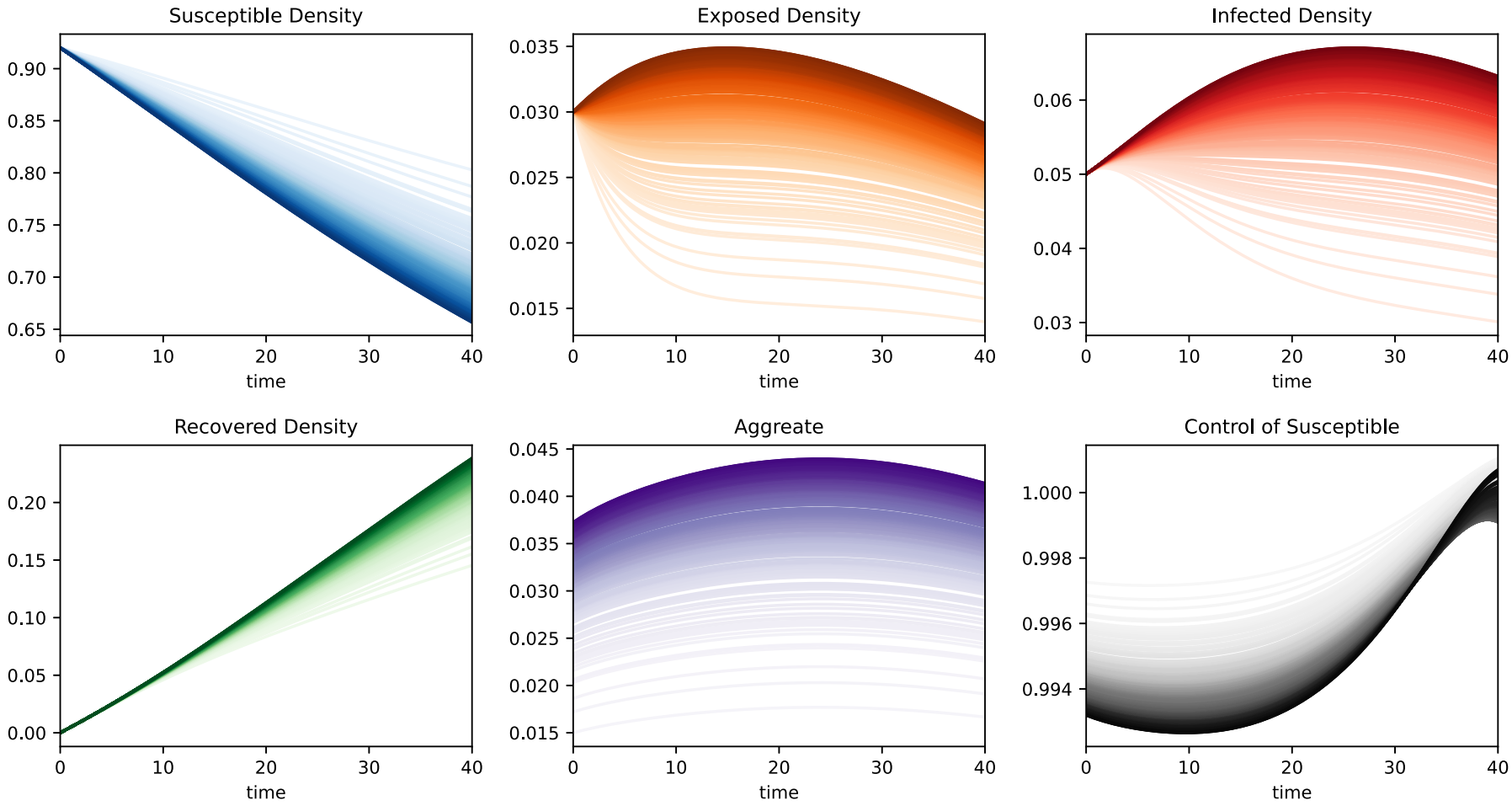


Figure 13: SEIRD & power-law graphon: $w(x, y) = (xy)^{-g}$ where $-\infty < g \leq 0$

T	β	γ	ϵ	ρ	c_λ	c_I	c_D	g	λ^S	λ^E	λ^I	λ^R
40	0.2	0.1	0.2	0.95	10	1	1	-0.2	1.0	1.0	0.9	1.0

Conclusions

This tutorial was about introducing the machine learning methods mean field games and related problems.

- → We gave an introduction to stochastic optimal problems and described how neural networks can be used to solve these problems.
- → We discussed extension to the large population setup with both cooperative and non-cooperative agents.
- → We introduced different extensions and how machine learning can be utilized to solve them.

Thank you!

gokced@illinois.edu
www.gokcedayanikli.com