APAC-Net: Alternating the Population and Control Neural Networks, for Mean-Field Games Problems

Alex Tong Lin (UCLA)
Joint with: Samy Wu Fung, Levon Nurbekyan, Wuchen Li, and Stanley Osher

March 19, 2021
1 Optimal Control

2 Mean-Field Games

3 Variational Mean-Field Games

4 Generative Adversarial Networks (GANs)

5 APAC-Net

6 Numerical Results
Optimal Control
The theory of optimal control seeks to find a control law for a given system such that a certain optimality criterion is reached.
Optimal Control in Words

- The theory of optimal control seeks to find a control law for a given system such that a certain optimality criterion is reached.
- A control problem includes a cost (payoff) functional and a path influenced by the controls. Optimal control seeks to find control policy that minimizes (maximizes) the cost functional.
Optimal Control in Words

- The theory of optimal control seeks to find a control law for a given system such that a certain optimality criterion is reached.

- A control problem includes a cost (payoff) functional and a path influenced by the controls. Optimal control seeks to find control policy that minimizes (maximizes) the cost functional.

- Example: Landing a spacecraft on the moon:

  ![Diagram of a spacecraft landing on the moon.](image)

  \[ \text{height} = h(t) \]

  \[ \text{moon's surface} \]
Optimal Control in Math

The optimal control problem consists of a state and a control. The state and control follows the differential equation (also called the dynamics)

\[
\begin{aligned}
\dot{x}(s) &= f(x(s), \alpha(s)), \quad 0 \leq s \leq T \\
x(0) &= x
\end{aligned}
\]

And we seek to minimize the cost functional

\[
C[x, t]\left[\alpha(\cdot)\right] = g(x(T)) + \int_{T}^{t} L(x(s), \alpha(s)) \, ds
\]

where \(x(\cdot) = x_{\alpha(\cdot)}(\cdot)\) solves the above ODE. Usually we want the control to lie in some admissable control set \(\alpha(\cdot) \in A\).

Then we put

\[
\phi(x, t) = \min_{\alpha(\cdot) \in A} C[x, t]\left[\alpha(\cdot)\right].
\]

which we call the value function. This value function satisfies a PDE, i.e. a Hamilton-Jacobi equation.
Optimal Control in Math

The optimal control problem consists of a state and a control. The state and control follows the differential equation (also called the dynamics)

\[
\begin{cases}
  \dot{x}(s) = f(x(s), \alpha(s)), & 0 \leq s \leq T \\
  x(0) = x
\end{cases}
\]

And we seek to minimize the cost functional

\[
C_{x,t}[\alpha(\cdot)] = g(x(T)) + \int_{t}^{T} L(x(s), \alpha(s)) \, ds
\]

where \( x(\cdot) = x^{\alpha(\cdot)}(\cdot) \) solves the above ODE. Usually we want the control to lie in some admissible control set \( \alpha(\cdot) \in \mathcal{A} \).
Optimal Control in Math

The optimal control problem consists of a state and a control. The state and control follows the differential equation (also called the dynamics)

\[
\begin{cases}
\dot{x}(s) = f(x(s), \alpha(s)), & 0 \leq s \leq T \\
x(0) = x
\end{cases}
\]

And we seek to minimize the cost functional

\[
C_{x,t}[\alpha(\cdot)] = g(x(T)) + \int_{t}^{T} L(x(s), \alpha(s)) \, ds
\]

where \( x(\cdot) = x^{\alpha(\cdot)}(\cdot) \) solves the above ODE. Usually we want the control to lie in some admissible control set \( \alpha(\cdot) \in \mathcal{A} \).

Then we put

\[
\varphi(x, t) = \min_{\alpha(\cdot) \in \mathcal{A}} C_{x,t}[\alpha(\cdot)].
\]

which we call the value function. This value function satisfies a PDE, i.e. a Hamilton-Jacobi equation.
Optimal Control to Hamilton-Jacobi Equations

So we have this

\[
\begin{align*}
\dot{x}(s) &= f(x(s), \alpha(s)), \quad 0 \leq s \leq T \\
x(0) &= x \\
\phi(x, t) &= \min_{\alpha(\cdot) \in \mathcal{A}} \left\{ g(x(T)) + \int_t^T L(x(s), \alpha(s)) \, ds \right\}
\end{align*}
\]

turned into this

\[
\begin{align*}
\partial_t \phi(x, t) + \max_{\alpha(\cdot) \in \mathcal{A}} \{ \nabla_x \phi(x, t) \cdot f(x, \alpha) - L(x, \alpha) \} &= 0 \\
\phi(x, T) &= g(x)
\end{align*}
\]
Optimal Control to Hamilton-Jacobi Equations

So we have this

\[
\begin{aligned}
\dot{x}(s) &= f(x(s), \alpha(s)), \quad 0 \leq s \leq T \\
x(0) &= x \\
\varphi(x, t) &= \min_{\alpha(\cdot) \in A} \left\{ g(x(T)) + \int_t^T L(x(s), \alpha(s)) \, ds \right\}
\end{aligned}
\]

turned into this

\[
\begin{aligned}
\partial_t \varphi(x, t) + \max_{\alpha \in A} \left\{ \nabla_x \varphi(x, t) \cdot f(x, \alpha) - L(x, \alpha) \right\} &= 0, \\
\varphi(x, T) &= g(x),
\end{aligned}
\]

where \( H(x, p) = \max_{\alpha \in A} \left\{ p \cdot f(x, \alpha) - L(x, \alpha) \right\} \).
Examples of optimal control problems

- The Linear-Quadratic-Regulator which has applications to robotics, also see Kalman Filter:

\[ \dot{x} = Ax + B\alpha \]

and

\[ C = x^T(T)F(T)x(T) + \int_t^T (x^T Qx + \alpha^T R\alpha + 2x^T N\alpha) \, ds \]

where all the matrices F, Q, R, and N are constant.

- This particular example has tremendous applications in robotics such as controlling aircraft, cars, electrical grids, etc.
Examples of optimal control problems

- Other examples include: optimal sterilization of canned foods, optimal management of waste-water treatment plants, and many more.
Examples of optimal control problems

- Other examples include: optimal sterilization of canned foods, optimal management of waste-water treatment plants, and many more.
- If one delves into stochastic optimal control, well, that’s pretty much the basis for Quantitative Finance. One of Nobel Prize winner Robert C. Merton’s (of Black-Scholes-Merton) main contributions to Mathematical Finance was his work with stochastic optimal control and mathematical finance.
- Just try to forget that he co-founded Long-Term Capital Management (which really only managed capital for about four years).
Mean-Field Games
Differential Games with $N$ players, the bridge

- Optimal control extended to more than one player is **differential games**.

- To get to mean-field games: Suppose we have $N$ agents $\{x_i\}_{i=1}^N$, and they are “homogeneous.” Namely, they all have the same dynamics,

$$\dot{x}_i(t) = f\left(x_i(t), \alpha_i(t), \mu^N(t)\right)$$

where $\mu^N(t) = \frac{1}{N} \sum_{k=1}^N \delta_{x_k(t)}$. These agents all want to minimize their own cost,

$$J_i^N(\alpha_i) = \int_0^T L(x_i(t), \alpha_i(t), \mu^N(t)) \, dt + g(x_i(T), \mu^N(T))$$

So the crucial assumption is that $(f, L, g)$ are all the same.

- If we take $N$ to infinity, then we get a model for a mean-field game.
Mean-Field Game Primer

- A mean-field game seeks to model the behavior of a very large number of small interacting agents that each seek to optimize their own value function.
The heuristic interpretation is that a typical agent has the following control:

$$\dot{x}(t) = f(x(t), \alpha(t), \rho(t))$$

and this agent aims to minimize,

$$\int_0^T L(x(t), \alpha(t), \rho(t)) \, dt + g(x(T), \rho(T))$$

Then the mean-field game is precisely the system of PDEs:

$$\begin{cases} 
-\partial_t \varphi + H(x, D\varphi, \rho) = 0 \\
\partial_t \rho - \text{div}(D\rho H(x, D\varphi, \rho)\rho) = 0 \\
\rho(x, 0) = \rho_0(x), \quad \varphi(x, T) = \varphi_T(x)
\end{cases}$$
Mean-Field Game Primer

Comparing optimal control and mean-field games,

<table>
<thead>
<tr>
<th>Optimal Control</th>
<th>Mean-Field Games</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\dot{x}(t) = f(x(t), \alpha(t), t)$</td>
<td>$\dot{x}(t) = f(x(t), \alpha(t), \rho(t), t)$</td>
</tr>
<tr>
<td>$g(x(T)) + \int_0^T L(x(t), \alpha(t)) , dt$</td>
<td>$g(x(T), \rho(T)) + \int_0^T L(x(t), \alpha(t), \rho(t)) , ds$</td>
</tr>
</tbody>
</table>
| $\left\{ \begin{array}{l}
-\partial_t \varphi + H(x, D\varphi) = 0 \\
\varphi(x, T) = \varphi_T(x)
\end{array} \right.$ | $\left\{ \begin{array}{l}
-\partial_t \varphi + H(x, D\varphi, \rho) = 0 \\
\partial_t \rho - \text{div}(D_pH(x, D\varphi, \rho)\rho) = 0 \\
\rho(x, 0) = \rho_0(x), \quad \varphi(x, T) = \varphi_T(x)
\end{array} \right.$ |
Comparing **stochastic** optimal control and **stochastic** mean-field games,

<table>
<thead>
<tr>
<th>Optimal Control</th>
<th>Mean-Field Games</th>
</tr>
</thead>
<tbody>
<tr>
<td>$dx(t) = f(x(t), \alpha(t), t)dt + \sqrt{2\nu}dB_t$</td>
<td>$dx(t) = f(x(t), \alpha(t), \rho(t), t)dt + \sqrt{2\nu}dB_t$</td>
</tr>
<tr>
<td>$\mathbb{E} \left[ g(x(T)) + \int_0^T L(x(t), \alpha(t)) , dt \right]$</td>
<td>$\mathbb{E} \left[ g(x(T), \rho(T)) + \int_0^T L(x(t), \alpha(t), \rho(t)) , ds \right]$</td>
</tr>
<tr>
<td>$\begin{cases} -\partial_t \varphi - \nu \Delta \varphi + H(x, D\varphi) = 0 \ \varphi(x, T) = \varphi_T(x) \end{cases}$</td>
<td>$\begin{cases} -\partial_t \varphi - \nu \Delta \varphi + H(x, D\varphi, \rho) = 0 \ \partial_t \rho - \nu \Delta \rho - \text{div}(D\rho H(x, D\varphi, \rho) \rho) = 0 \ \rho(x, 0) = \rho_0(x), \quad \varphi(x, T) = \varphi_T(x) \end{cases}$</td>
</tr>
</tbody>
</table>
Variational Mean-Field Games
The class of potential Mean-Field Games admits a variational formulation, namely the mean-field game equilibria can be found by minimizing a functional.
Namely for some mean-field games, the equilibrium solution can be found by minimizing an overall “energy” (e.g. multiply the value function for a single agent by $\rho$):

$$
\mathcal{A}(\rho, v) := \min_{\rho, v} \int_0^T \int_{\Omega} \rho(x, t) L(v) + F(x, \rho) \, dx \, dt \\
+ \int_{\Omega} g(x) \rho(x, T) \, dx
$$

where $x = x(t)$, $v = v(t)$ above, and where the optimization has the constraint,

$$
\partial_t \rho - \nu \Delta \rho + \text{div}(\rho v) = 0.
$$
Variational Mean-Field Games

We can elevate the constraints into the objective with Lagrange multipliers,

\[
\max_{\varphi} \min_{\rho, v} \int_{0}^{T} \int_{\Omega} \rho(x, t)L(v(t, x)) + F(x, \rho(x, t)) \, dx \, dt + \int_{\Omega} g(x)\rho(x, T) \, dx \\
+ \int_{0}^{T} \int_{\Omega} \varphi(x, t) \left( -\partial_{t} \rho(x, t) + \Delta \rho(x, t) - \nabla \cdot (\rho(x, t)v(x, t)) \right) \, dx \, dt
\]

where \( \varphi \) is the Lagrange multiplier.
We can perform integration by parts and push the minimization of $v$ inside to obtain,

$$
= \max_{\varphi} \min_{\rho} \int_0^T \int_\Omega F(x, \rho(x, t)) \, dx \, dt \\
+ \int_\Omega \left( g(x) - \varphi(x, T) \right) \rho(x, T) \, dx + \int_\Omega \varphi(x, 0) \rho(x, 0) \, dx \\
+ \int_0^T \int_\Omega \left( \partial_t \varphi(x, t) + \Delta \varphi(x, t) - L^* (-\nabla \varphi(x, t)) \right) \rho(x, t) \, dx \, dt
$$

which means we end up with a sampling problem – this is a preview of APAC-Net. This is in the spirit of Feynman-Kac.

Then the idea is to turn $\rho$ and $\varphi$ into neural networks and train as in GANs (Generative Adversarial Networks).
Feynman-Kac Quick Description

APAC-Net is like Feynman-Kac.

**Theorem**  [edit]

Consider the partial differential equation

\[
\frac{\partial u}{\partial t}(x, t) + \mu(x, t) \frac{\partial u}{\partial x}(x, t) + \frac{1}{2} \sigma^2(x, t) \frac{\partial^2 u}{\partial x^2}(x, t) - V(x, t) u(x, t) + f(x, t) = 0,
\]

defined for all \( x \in \mathbb{R} \) and \( t \in [0, T] \), subject to the terminal condition

\[ u(x, T) = \psi(x), \]

where \( \mu, \sigma, \psi, V, f \) are known functions, \( T \) is a parameter and \( u : \mathbb{R} \times [0, T] \to \mathbb{R} \) is the unknown. Then the Feynman–Kac formula tells us that the solution can be written as a conditional expectation

\[
u(x, t) = E^Q \left[ \int_t^T e^{-\int_t^r V(X_\tau, \tau) \, d\tau} f(X_r, r) \, dr + e^{-\int_t^T V(X_\tau, \tau) \, d\tau} \psi(X_T) \bigg| X_t = x \right]
\]

under the probability measure \( Q \) such that \( X \) is an \textit{Itô process} driven by the equation

\[ dX = \mu(X, t) \, dt + \sigma(X, t) \, dW^Q, \]

with \( W^Q(t) \) is a \textit{Wiener process} (also called Brownian motion) under \( Q \), and the initial condition for \( X(t) \) is \( X(t) = x \).
Generative Adversarial Networks (GANs)
GAN Training

- Generative-Adversarial Networks (GANs) is a method to produces samples from a desired distribution.
- For example, the distribution of celebrity faces:
Training has a discriminator and a generator. The generator produces samples (analogous to our $\rho$), and the discriminator evaluates the quality of those samples (analogous to our $\varphi$).
GAN Loss Functions

- Original Minimax Loss:

\[
\mathbb{E}_{x \sim \text{real}} \left[ \log(D(x)) \right] + \mathbb{E}_{z \sim \mathcal{N}} \left[ \log(1 - D(G(z))) \right]
\]

- Wasserstein Loss:

\[
\min_G \max_D \mathbb{E}_{x \sim \text{real}} \left[ D(x) \right] - \mathbb{E}_{z \sim \mathcal{N}} \left[ D(G(z)) \right] \quad \text{s.t.} \quad \|\nabla D\| \leq 1
\]

- Actually, one can show the Wasserstein loss is a special type of mean-field game.
GAN Training

Diagram of GAN Training:
- Random input
- Generator
- Sample
- Discriminator
- Discriminator loss
- Generator loss
- Backpropagation
APAC-Net
Why Neural Networks?

Because the usual way to do things – using a grid to discretize space – becomes impossible in high dimensions.
How Math Makes It Work

Recall the solution to the mean-field game turned into the min-max problem:

\[
\max_{\varphi} \min_{\rho} \int_{\Omega} \left( g(x) - \varphi(x, T) \right) \rho(x, T) \, dx + \int_{\Omega} \varphi(x, 0) \rho(x, 0) \, dx \\
+ \int_0^T \int_{\Omega} F(x, \rho(x, t)) \, dx \, dt \\
+ \int_0^T \int_{\Omega} \left( \partial_t \varphi(x, t) + \Delta \varphi(x, t) - L^*(-\nabla \varphi(x, t)) \right) \rho(x, t) \, dx \, dt
\]

And supposing a certain form for \( F \), then this can be expressed in expectation form,

\[
\max_{\varphi} \min_{\rho} \mathbb{E}_{y \sim \rho(\cdot, T)} \left[ g(y) - \varphi(y, T) \right] + \mathbb{E}_{y \sim \rho(\cdot, 0)} \left[ \varphi(y, 0) \right] \\
+ \int_0^T \mathbb{E}_{y \sim \rho(\cdot, t)} \left[ F(y) \right] \, dt \\
+ \int_0^T \mathbb{E}_{y \sim \rho(\cdot, t)} \left[ \partial_t \varphi(y, t) + \Delta \varphi(y, t) - L^*(-\nabla \varphi(y, t)) \right] \, dt
\]
Taking our inspiration from GANs, we produce samples from the distribution $\rho$ by using the neural network $G_\theta$.

This is done by first sampling $z \sim \rho_0(x)$ (the initial distribution, which is given), and then we train $G_\theta(z, t)$ to output samples from $\rho$ at time $t$.

We let $\phi_\omega$ act as the discriminator, i.e. the solution the Hamilton-Jacobi Bellman.
Automatic Boundary Conditions

- Letting $T = 1$,
- We also set $\phi_\omega$ to be,

$$\phi_\omega(x, t) = (1 - t)N_\omega(x, t) + tg(x)$$

and we set $G_\theta$ to be,

$$G_\theta(z, t) = (1 - t)z + tN_\theta(z, t), \quad z \sim \rho_0(x).$$

- This means the time-boundary conditions are actually satisfied automatically.
How We Train It

Training $\phi_\omega$:

1. Sample a batch $\{(z_b, t_b)\}_{b=1}^B$, where $z_b \sim \rho_0$ and $t_b \sim \text{Unif}(0, T)$.
2. Set $x_b \leftarrow G_\theta(z_b, t_b)$, $b = 1, \ldots, B$.
3. Compute loss:

$$\ell = \frac{1}{B} \sum_{b=1}^B \phi_\omega(x_b, 0) + \partial_t \phi_\omega(x_b, t_b) + \nu \Delta \phi_\omega(x_b, t_b) - H(\nabla_x \phi_x(x_b, t_b))$$

4. Compute regularization term:

$$\ell_{HJB} = \lambda \frac{1}{B} \sum_{b=1}^B \left\| \partial_t \phi_\omega(x_b, t_b) + \nu \Delta \phi_\omega(x_b, t_b) - H(\nabla_x \phi_\omega(x_b, t_b)) + f(x_b, t_b) \right\|$$

5. Backpropagate the loss $\ell + \ell_{HJB}$ to $\phi_\omega$ weights.
How We Train It

Training $G_\theta$:

1. Sample batch $\{(z_b, t_b)\}_{b=1}^B$, where $z_b \sim \rho_0$ and $t_b \sim \text{Unif}(0, T)$.
2. Compute loss:

   $\ell = \frac{1}{B} \sum_{b=1}^B \partial_{t_\omega}(G_\theta(z_b, t_b), t_b) + \nu \Delta \phi_\omega(G_\theta(z_b, t_b), t_b) - H(\nabla_x \phi_\omega(G_\theta(z_b, t_b), t_b)) + f(G_\theta(z_b, t_b), t_b)$

3. Backpropagate loss $\ell$ to $G_\theta$ weights.
How We Train It (For Reference)

We train just like in GANs, where we turn expectations into averages.

Algorithm 1 APAC-Net

Require: \( \nu \) diffusion parameter, \( G \) terminal cost, \( H \) Hamiltonian, \( f \) interaction term.
Require: Initialize neural networks \( N_\omega \) and \( N_\theta \), batch size \( B \)
Require: Set \( \phi_\omega \) and \( G_\theta \) as in (4.1)

while not converged do

train \( \phi_\omega \):
Sample batch \( \{(z_b, t_b)\}_{b=1}^{B} \) where \( z_b \sim \rho_0 \) and \( t_b \sim \text{Unif}(0, T) \)
\( x_b \leftarrow G_\theta(z_b, t_b) \) for \( b = 1, \ldots, B \).
\( \ell_0 \leftarrow \frac{1}{B} \sum_{b=1}^{B} \phi_\omega(x_b, 0) \)
\( \ell_t \leftarrow \frac{1}{B} \sum_{b=1}^{B} \partial_t \phi_\omega(x_b, t_b) + \nu \Delta \phi_\omega(x_b, t_b) - H(\nabla_x \phi_\omega(x_b, t_b)) \)
\( \ell_{\text{HJB}} \leftarrow \lambda \frac{1}{B} \sum_{b=1}^{B} \| \partial_t \phi_\omega(x_b, t_b) + \nu \Delta \phi_\omega(x_b, t_b) - H(\nabla_x \phi_\omega(x_b, t_b)) + f(x_b, t_b) \| \)
Backpropagate the loss \( \ell_{\text{total}} = \ell_0 + \ell_t + \ell_{\text{HJB}} \) to \( \omega \) weights.

train \( G_\theta \):
Sample batch \( \{(z_b, t_b)\}_{b=1}^{B} \) where \( z_b \sim \rho_0 \) and \( t_b \sim \text{Unif}(0, T) \)
\( \ell_t \leftarrow \frac{1}{B} \sum_{b=1}^{B} \partial_t \phi_\omega(G_\theta(z_b, t_b), t_b) + \nu \Delta \phi_\omega(G_\theta(z_b, t_b), t_b) - H(\nabla_x \phi_\omega(G_\theta(z_b, t_b), t_b)) + f(G_\theta(z_b, t_b), t_b) \)
Backpropagate the loss \( \ell_{\text{total}} = \ell_t \) to \( \theta \) weights.

end while
Numerical Results
Numerical Results - Obstacles

- $H(p) = \|p\|_2$
- Left is $\nu = 0$ (deterministic), and right is $\nu = 0.4$ (stochastic)
- 100 dimensional obstacle problem with an obstacle in the first two dimensions ("cylindrical" obstacle).
Numerical Results - Obstacles & Congestion

- $H(p) = \|p\|_2$
- 100 dimensional congestion problem with congestion in the first two dimensions. $\nu = 0$ (left), and $\nu = 0.1$ (right)
- Congestion term $G(x, \rho(x, t)) = \int_{\Omega} \int_{\Omega} \frac{1}{\|x-y\|^2+1} \rho(x, T) \rho(y, T) \, dx \, dy$
Numerical Results: A realistic example, the Quadcopter

The dynamics of a quadcopter are:

\[
\begin{align*}
\ddot{x} &= \frac{u}{m} \left( \sin(\phi) \sin(\psi) + \cos(\phi) \cos(\psi) \sin(\theta) \right) \\
\ddot{y} &= \frac{u}{m} \left( -\cos(\psi) \sin(\phi) + \cos(\phi) \sin(\theta) \sin(\psi) \right) \\
\ddot{z} &= \frac{u}{m} \cos(\theta) \cos(\phi) - g \\
\ddot{\psi} &= \tilde{\tau}_\psi \\
\ddot{\theta} &= \tilde{\tau}_\theta \\
\ddot{\phi} &= \tilde{\tau}_\phi
\end{align*}
\]

where \( u \) is the thrust, \( g \) is the gravitational acceleration (9.81 m/s^2), and \( x, y, z \) are the spatial coordinates, \( \psi, \theta, \phi \) are the angular coordinates, and \( \tilde{\tau}_\psi, \tilde{\tau}_\theta, \tilde{\tau}_\phi \).

Turns 12-dimensional when you transfer to first-order system.
Numerical Results: A realistic example, the Quadcopter

No Congestion

With Congestion

\[ \sigma = 0 \]

\[ \sigma = 0.1 \]
Thank you!