COLORADO SCHOOL OF MINES
EARTH • ENERGY • ENVIRONMENT

Jacobian Free Backpropagation for High-Dimensional Optimal Control with Implicit Hamiltonians

Samy Wu Fung

Joint work with Eric Gelphman, Deepanshu Verma, Nicole Yang, and Stanley Osher

High-Dimensional Optimal Control Background

Goal: Find the control that incurs minimal cost[1]

[1] Flemming and Soner. *Controller Markov Processes*

Optimal Control Formulation

Goal: Find the control that incurs minimal cost[1]

$$\min_{u \in U} \int_0^T L(s, z_x, u)ds + G(z_x(T))$$

$$\dot{z}_x = f(t, z_x, u), \quad z_x(0) = x,$$

[1] Flemming and Soner. *Controller Markov Processes*

Optimal Control Formulation

Goal: Find the control that incurs minimal cost[1]

$$\min_{u \in U} \int_0^T L(s, z_x, u) ds + G(z_x(T))$$

$$\dot{z}_x = f(t, z_x, u), \quad z_x(0) = x,$$

- L is the running cost and G is the terminal cost
- z_x is the state, u is the controller
- f are the dynamics

[1] Flemming and Soner. *Controller Markov Processes*

Pontryagin Maximum Principle

Let $\mathcal{H}(t, z_x, p_x, u) = -\langle p_x, f(t, z_x, u)\rangle - L(t, z_x, u)$ be the *Generalized* Hamiltonian

[2] Pontryagin et al. The Mathematical Theory of Optimal Processes. 1962.

Pontryagin Maximum Principle

Let $\mathcal{H}(t, z_x, p_x, u) = -\langle p_x, f(t, z_x, u)\rangle - L(t, z_x, u)$ be the *Generalized* Hamiltonian

By the Pontryagin Maximum Principle (PMP)[2] we have that at the **optimal controller** $u^\star$:

$$\dot{z}_x = -\nabla_p \mathcal{H}(t, z_x, p_x, u^\star), \quad z_x(0) = x$$

$$\dot{p}_x = \nabla_z \mathcal{H}(t, z_x, p_x, u^\star), \quad p_x(T) = \nabla G(z_x(T)),$$

$$u^\star \in \arg\max_u \mathcal{H}(t, z_x, p_x, u)$$

[2]Pontryagin et al. The Mathematical Theory of Optimal Processes. 1962.

Pontryagin Maximum Principle

Let $\mathcal{H}(t, z_x, p_x, u) = -\langle p_x, f(t, z_x, u)\rangle - L(t, z_x, u)$ be the *Generalized* Hamiltonian

By the Pontryagin Maximum Principle (PMP)[2] we have that at the **optimal controller** $u^\star$:

$$\dot{z}_x = -\nabla_p \mathcal{H}(t, z_x, p_x, u^\star), \quad z_x(0) = x$$

$$\dot{p}_x = \nabla_z \mathcal{H}(t, z_x, p_x, u^\star), \quad p_x(T) = \nabla G(z_x(T)),$$

$$u^\star \in \arg\max_u \mathcal{H}(t, z_x, p_x, u)$$

- p_x is the dual/adjoint variable

[2]Pontryagin et al. The Mathematical Theory of Optimal Processes. 1962.

Pontryagin Maximum Principle

Let $\mathcal{H}(t, z_x, p_x, u) = -\langle p_x, f(t, z_x, u)\rangle - L(t, z_x, u)$ be the *Generalized* Hamiltonian

By the Pontryagin Maximum Principle (PMP)[2] we have that at the **optimal controller** $u^\star$:

$$\dot{z}_x = -\nabla_p \mathcal{H}(t, z_x, p_x, u^\star), \quad z_x(0) = x$$

$$\dot{p}_x = \nabla_z \mathcal{H}(t, z_x, p_x, u^\star), \quad p_x(T) = \nabla G(z_x(T)),$$

$$u^\star \in \arg\max_u \mathcal{H}(t, z_x, p_x, u)$$

- p_x is the dual/adjoint variable
- $u^\star$ assumed to have explicit formula

[2] Pontryagin et al. The Mathematical Theory of Optimal Processes. 1962.

Pontryagin Maximum Principle

Let $\mathcal{H}(t, z_x, p_x, u) = -\langle p_x, f(t, z_x, u)\rangle - L(t, z_x, u)$ be the *Generalized* Hamiltonian

By the Pontryagin Maximum Principle (PMP)[2] we have that at the **optimal controller** $u^\star$:

$$\dot{z}_x = -\nabla_p \mathcal{H}(t, z_x, p_x, u^\star), \quad z_x(0) = x$$

$$\dot{p}_x = \nabla_z \mathcal{H}(t, z_x, p_x, u^\star), \quad p_x(T) = \nabla G(z_x(T)),$$

$$u^\star \in \arg\max_u \mathcal{H}(t, z_x, p_x, u)$$

- p_x is the dual/adjoint variable
- $u^\star$ assumed to have explicit formula
- PMP-based approaches are typically *local* solution methods (open loop)
  - can solve high-dimensional problems for a single initial x
  - need to resolve for new x

[2] Pontryagin et al. The Mathematical Theory of Optimal Processes. 1962.

By Pontryagin, we also know that the dual variable is the gradient of a value function:

$$p_x(t) = \nabla_z \phi(t, z_x^\star(t)), \qquad \text{where}$$

$$-\partial_t \phi(t,z) + \sup_u \mathcal{H}(t, z, \nabla\phi, u) = 0, \quad \phi(T,z) = G(z) \tag{HJB}$$

Hamilton-Jacobi-Bellman Equations

By Pontryagin, we also know that the dual variable is the gradient of a value function:

$$p_x(t) = \nabla_z \phi(t, z_x^\star(t)), \qquad \text{where}$$

$$-\partial_t \phi(t,z) + \sup_u \mathcal{H}(t, z, \nabla\phi, u) = 0, \quad \phi(T,z) = G(z) \tag{HJB}$$

- Solving OC problems via the HJB is a *global* solution method (feedback form) $\implies$ solved for all initial conditions x
- for new initial condition x, no need for re-computation
- grid-based method $\implies$ curse of dimensionality

Recent approaches leverage Pontryagin and parameterize the value function ϕ with a neural network to obtain feedback controller for high-dimensional OC[3].

[3] **Onken et al.**, IEEE TAC, 2022, **Li et al**. SIAM SISC 2024, **Yang et al.**, IEEE TNN 2020

Neural Networks for High-Dimensional OC

Recent approaches leverage Pontryagin and parameterize the value function ϕ with a neural network to obtain feedback controller for high-dimensional OC[3]. The training problem is given by

$$\min_\theta \ \mathbb{E}_{x\sim\rho_0} \ J_x(\theta) = \int_0^T L(t, z_x, u_\theta^\star)dt + G(z_x(T)), \tag{1}$$

$$\text{subject to:} \quad \dot{z}_x = f(t, z_x, u_\theta^\star), \quad z_x(0) = x, \tag{2}$$

$$u_\theta^\star \in \arg\max_u \mathcal{H}(t, z, \nabla\phi_\theta, u), \tag{3}$$

for some initial distribution of states ρ_0.

[3] **Onken et al.**, IEEE TAC, 2022, **Li et al**. SIAM SISC 2024, **Yang et al.**, IEEE TNN 2020

Neural Networks for High-Dimensional OC

Recent approaches leverage Pontryagin and parameterize the value function ϕ with a neural network to obtain feedback controller for high-dimensional OC[3]. The training problem is given by

$$\min_\theta \; \mathbb{E}_{x\sim\rho_0} \; J_x(\theta) = \int_0^T L(t, z_x, u^\star_\theta)dt + G(z_x(T)), \tag{1}$$

$$\text{subject to:} \quad \dot{z}_x = f(t, z_x, u^\star_\theta), \quad z_x(0) = x, \tag{2}$$

$$u^\star_\theta \in \arg\max_u \mathcal{H}(t, z, \nabla\phi_\theta, u), \tag{3}$$

for some initial distribution of states ρ_0.

Prior works assume $u^\star$ in (3) admits an explicit formula.

[3] **Onken et al.**, IEEE TAC, 2022, **Li et al**. SIAM SISC 2024, **Yang et al.**, IEEE TNN 2020

Recent approaches leverage Pontryagin and parameterize the value function ϕ with a neural network to obtain feedback controller for high-dimensional OC[3]. The training problem is given by

$$\min_{\theta} \ \mathbb{E}_{x\sim\rho_0} \ J_x(\theta) = \int_0^T L(t, z_x, u_\theta^\star)dt + G(z_x(T)), \tag{1}$$

$$\text{subject to:} \quad \dot{z}_x = f(t, z_x, u_\theta^\star), \quad z_x(0) = x, \tag{2}$$

$$u_\theta^\star \in \arg\max_u \mathcal{H}(t, z, \nabla\phi_\theta, u), \tag{3}$$

for some initial distribution of states ρ_0.

Prior works assume $u^\star$ in (3) admits an explicit formula.

Today: Develop efficient algorithms for training (1)-(3) when $u_\theta^\star$ *does not* admit explicit formula, i.e., the Hamiltonian $H = \sup_u \mathcal{H}$ is implicitly defined.

[3] **Onken et al.**, IEEE TAC, 2022, **Li et al**. SIAM SISC 2024, **Yang et al.**, IEEE TNN 2020

Implicit Networks for Optimal Control

Implicit Neural Networks (INNs)

- The output of an INN is given by the fixed point of a parameterized operator [4]:

$$u_\theta^\star(t, z) = T_\theta(u_\theta^\star; t, z) \tag{4}$$

[4] **El Ghaoui et al.**, SIMODS, 2021, **Bai et al.**, NeurIPS, 2019

Implicit Neural Networks (INNs)

- The output of an INN is given by the fixed point of a parameterized operator [4]:

$$u_\theta^\star(t, z) = T_\theta(u_\theta^\star; t, z) \tag{4}$$

- Standard forward propagation of an INN can use a fixed point iteration:

$$u_\theta^{k+1} = T_\theta(u_\theta^k; t, z), \quad k = 0, 1, \ldots \tag{5}$$

[4] **El Ghaoui et al.**, SIMODS, 2021, **Bai et al.**, NeurIPS, 2019

Implicit Neural Networks (INNs)

- The output of an INN is given by the fixed point of a parameterized operator [4]:

$$u_\theta^\star(t, z) = T_\theta(u_\theta^\star; t, z) \tag{4}$$

- Standard forward propagation of an INN can use a fixed point iteration:

$$u_\theta^{k+1} = T_\theta(u_\theta^k; t, z), \quad k = 0, 1, \dots \tag{5}$$

- We want $u_\theta^\star \in \arg\max_u \mathcal{H}(s, z, \nabla\phi_\theta, u) \implies \nabla_u \mathcal{H}(t, z, u_\theta^\star) = 0.$

[4] **El Ghaoui et al.**, SIMODS, 2021, **Bai et al.**, NeurIPS, 2019

Implicit Neural Networks (INNs)

- The output of an INN is given by the fixed point of a parameterized operator [4]:

$$u_\theta^\star(t,z) = T_\theta(u_\theta^\star; t, z) \tag{4}$$

- Standard forward propagation of an INN can use a fixed point iteration:

$$u_\theta^{k+1} = T_\theta(u_\theta^k; t, z), \quad k = 0, 1, \dots \tag{5}$$

- We want $u_\theta^\star \in \arg\max_u \mathcal{H}(s, z, \nabla\phi_\theta, u) \implies \nabla_u \mathcal{H}(t, z, u_\theta^\star) = 0.$

  A natural choice: $T_\theta(u; t, z) = u + \alpha \nabla_u \mathcal{H}(t, z, \nabla\phi_\theta, u).$

[4] **El Ghaoui et al.**, SIMODS, 2021, **Bai et al.**, NeurIPS, 2019

Training INNs

To compute the gradient of the objective $J_x(\theta)$, we need to compute $\dfrac{du_\theta^\star}{d\theta}$.

[5] Wu Fung, Heaton, McKenzie, Li, Yin, Osher. AAAI 2022

To compute the gradient of the objective $J_x(\theta)$, we need to compute $\dfrac{du^\star_\theta}{d\theta}$. General approaches:

[5] Wu Fung, Heaton, McKenzie, Li, Yin, Osher. AAAI 2022

Training INNs

To compute the gradient of the objective $J_x(\theta)$, we need to compute $\dfrac{du_\theta^\star}{d\theta}$. General approaches:

- Automatic Differentiation (AD) – backpropagate through each fixed point iteration
  - memory grows linearly per fixed point iteration ✖

[5] Wu Fung, Heaton, McKenzie, Li, Yin, Osher. AAAI 2022

Training INNs

To compute the gradient of the objective $J_x(\theta)$, we need to compute $\frac{du_\theta^\star}{d\theta}$. General approaches:

- Automatic Differentiation (AD) – backpropagate through each fixed point iteration
  - memory grows linearly per fixed point iteration ✖
- Implicit Differentiation – differentiate both sides of fixed pt. equation (and isolate $\frac{du_\theta^\star}{d\theta}$):

$$\frac{du_\theta^\star}{d\theta} = \left(I - \frac{\partial T_\theta(u_\theta^\star; t, z)}{\partial u}\right)^{-1} \frac{\partial T_\theta(u_\theta^\star; t, z)}{\partial \theta} \tag{6}$$

[5] Wu Fung, Heaton, McKenzie, Li, Yin, Osher. AAAI 2022

Training INNs

To compute the gradient of the objective $J_x(\theta)$, we need to compute $\dfrac{du^\star_\theta}{d\theta}$. General approaches:

- Automatic Differentiation (AD) – backpropagate through each fixed point iteration
  - memory grows linearly per fixed point iteration ✖
- Implicit Differentiation – differentiate both sides of fixed pt. equation (and isolate $\frac{du^\star_\theta}{d\theta}$):

$$\frac{du^\star_\theta}{d\theta} = \left(I - \frac{\partial T_\theta(u^\star_\theta; t, z)}{\partial u}\right)^{-1} \frac{\partial T_\theta(u^\star_\theta; t, z)}{\partial \theta} \tag{6}$$

  - constant in memory ✓
  - solve a linear system for each (t, z) ✖

[5] Wu Fung, Heaton, McKenzie, Li, Yin, Osher. AAAI 2022

Training INNs

To compute the gradient of the objective $J_x(\theta)$, we need to compute $\frac{du_\theta^\star}{d\theta}$. General approaches:

- Automatic Differentiation (AD) – backpropagate through each fixed point iteration
  - memory grows linearly per fixed point iteration ✖
- Implicit Differentiation – differentiate both sides of fixed pt. equation (and isolate $\frac{du_\theta^\star}{d\theta}$):

$$\frac{du_\theta^\star}{d\theta} = \left(I - \frac{\partial T_\theta(u_\theta^\star; t, z)}{\partial u}\right)^{-1} \frac{\partial T_\theta(u_\theta^\star; t, z)}{\partial \theta} \tag{6}$$

  - constant in memory ✓
  - solve a linear system for each (t, z) ✖
- Jacobian-Free Backpropagation (JFB)[5]: $\frac{du_\theta^\star}{d\theta} \approx \frac{\partial T_\theta(u_\theta^\star; t, z)}{\partial \theta}$

[5]Wu Fung, Heaton, McKenzie, Li, Yin, Osher. AAAI 2022

Training INNs

To compute the gradient of the objective $J_x(\theta)$, we need to compute $\frac{du_\theta^\star}{d\theta}$. General approaches:

- Automatic Differentiation (AD) – backpropagate through each fixed point iteration
  - memory grows linearly per fixed point iteration ✖
- Implicit Differentiation – differentiate both sides of fixed pt. equation (and isolate $\frac{du_\theta^\star}{d\theta}$):

$$\frac{du_\theta^\star}{d\theta} = \left(I - \frac{\partial T_\theta(u_\theta^\star; t, z)}{\partial u}\right)^{-1} \frac{\partial T_\theta(u_\theta^\star; t, z)}{\partial \theta} \tag{6}$$

  - constant in memory ✓
  - solve a linear system for each (t, z) ✖
- Jacobian-Free Backpropagation (JFB)[5]: $\frac{du_\theta^\star}{d\theta} \approx \frac{\partial T_\theta(u_\theta^\star; t, z)}{\partial \theta}$
  - **No system solve *and* constant memory**

[5]Wu Fung, Heaton, McKenzie, Li, Yin, Osher. AAAI 2022

Jacobian-Free Backpropagation Analysis

Assumptions from original JFB work[6]

Assumption 1: T_θ is γ-contractive in u for all $t \in [0, T], z \in \mathbb{R}^n$ and $\theta \in \mathbb{R}^p$.

[6] Wu Fung, Heaton, McKenzie, Li, Yin, Osher. AAAI 2022

Assumptions from original JFB work[6]

Assumption 1: T_θ is γ-contractive in u for all $t \in [0, T], z \in \mathbb{R}^n$ and $\theta \in \mathbb{R}^p$.

Assumption 2: For any θ, t, u, z:

- The matrix $M_\theta = \dfrac{\partial T_\theta}{\partial \theta}(u; t, z)$ has full row rank.
- $\exists \beta > 0$ such that the smallest eigenvalue of $(M_\theta M_\theta^\top)^{-1}$ satisfies $\lambda_{\min}\left((M_\theta M_\theta^\top)^{-1}\right) \geq \beta$.
- Condition number satisfies $\kappa\left((M_\theta M_\theta^\top)^{-1}\right) < \frac{1}{\gamma}$.

[6] Wu Fung, Heaton, McKenzie, Li, Yin, Osher. AAAI 2022

Assumptions from original JFB work[6]

Assumption 1: T_θ is γ-contractive in u for all $t \in [0, T], z \in \mathbb{R}^n$ and $\theta \in \mathbb{R}^p$.

Assumption 2: For any θ, t, u, z:

- The matrix $M_\theta = \dfrac{\partial T_\theta}{\partial \theta}(u; t, z)$ has full row rank.
- $\exists \beta > 0$ such that the smallest eigenvalue of $(M_\theta M_\theta^\top)^{-1}$ satisfies $\lambda_{\min}\left((M_\theta M_\theta^\top)^{-1}\right) \geq \beta$.
- Condition number satisfies $\kappa\left((M_\theta M_\theta^\top)^{-1}\right) < \frac{1}{\gamma}$.

Remark: These assumptions not enough to show descent (and convergence) in the OC setting because we have a *continuum/integral* of fixed point subproblems.

[6] Wu Fung, Heaton, McKenzie, Li, Yin, Osher. AAAI 2022

The true derivative of control objective, J_x, and its JFB approximation are given by

$$\frac{dJ_x(\theta)}{d\theta} = \int_0^T v_\theta(t)dt, \quad \text{and} \quad d_x^{JFB} = \int_0^T w_\theta(t)dt \tag{7}$$

respectively, where

$$\begin{aligned} v_\theta(t) &= \frac{du_\theta^\star}{d\theta}^\top (\nabla_u L(t, z_x, u_\theta^\star) + \nabla_u f^\top p_x), \\ w_\theta(t) &= \frac{\partial T_\theta}{\partial\theta}^\top (\nabla_u L(t, z_x, u_\theta^\star) + \nabla_u f^\top p_x), \end{aligned} \tag{8}$$

The true derivative of control objective, J_x, and its JFB approximation are given by

$$\frac{dJ_x(\theta)}{d\theta} = \int_0^T v_\theta(t)dt, \quad \text{and} \quad d_x^{JFB} = \int_0^T w_\theta(t)dt \tag{7}$$

respectively, where

$$\begin{aligned} v_\theta(t) &= \frac{du_\theta^\star}{d\theta}^\top (\nabla_u L(t, z_x, u_\theta^\star) + \nabla_u f^\top p_x), \\ w_\theta(t) &= \frac{\partial T_\theta}{\partial \theta}^\top (\nabla_u L(t, z_x, u_\theta^\star) + \nabla_u f^\top p_x), \end{aligned} \tag{8}$$

Assumption: Let $C_v = \frac{1}{T}\int_0^T v_\theta(t)dt$ and $C_w = \frac{1}{T}\int_0^T w_\theta(t)dt$. Assume that $\|v_\theta(t) - C_v\| \leq \|M_\theta w_\theta\| \sqrt{\lambda_- - \gamma\lambda_+}$ and $\|w_\theta(t) - C_w\| \leq \|M_\theta v_\theta\| \sqrt{\lambda_- - \gamma\lambda_+}$.

The true derivative of control objective, J_x, and its JFB approximation are given by

$$\frac{dJ_x(\theta)}{d\theta} = \int_0^T v_\theta(t)dt, \quad \text{and} \quad d_x^{JFB} = \int_0^T w_\theta(t)dt \tag{7}$$

respectively, where

$$\begin{aligned} v_\theta(t) &= \frac{du_\theta^\star}{d\theta}^\top (\nabla_u L(t, z_x, u_\theta^\star) + \nabla_u f^\top p_x), \\ w_\theta(t) &= \frac{\partial T_\theta}{\partial \theta}^\top (\nabla_u L(t, z_x, u_\theta^\star) + \nabla_u f^\top p_x), \end{aligned} \tag{8}$$

Assumption: Let $C_v = \frac{1}{T}\int_0^T v_\theta(t)dt$ and $C_w = \frac{1}{T}\int_0^T w_\theta(t)dt$. Assume that $\|v_\theta(t) - C_v\| \leq \|M_\theta w_\theta\|\sqrt{\lambda_- - \gamma\lambda_+}$ and $\|w_\theta(t) - C_w\| \leq \|M_\theta v_\theta\|\sqrt{\lambda_- - \gamma\lambda_+}$.

Theorem (Descent)

Under previous assumptions, $-d_x^{JFB}$ is a descent direction for J_x.

Jacobian-Free Backpropagation Analysis

Assumption: Let $E_1 = \mathbb{E}_x[\nabla_\theta J_x]$ and $E_2 = \mathbb{E}_x[d_x^{JFB}]$. Assume $\forall \theta, u^*, z$,

$$\mathbb{E}_x[\|\nabla_\theta J_x - E_1\|^2] \leq \mu_2 \|\mathbb{E}_x[\nabla_\theta J_x]\|^2, \tag{9}$$

$$\mathbb{E}_x[\|d_x^{JFB} - E_2\|^2] \leq \mu_2 \|\mathbb{E}_x[\nabla_\theta J_x]\|^2, \tag{10}$$

where μ_2 is a constant that depends on spectrum of $\dfrac{\partial T_\theta}{\partial \theta}$. That is, assume the variance of the true gradient and JFB are sufficiently bounded.

Jacobian-Free Backpropagation Analysis

Assumption: Let $E_1 = \mathbb{E}_x[\nabla_\theta J_x]$ and $E_2 = \mathbb{E}_x[d_x^{JFB}]$. Assume $\forall \theta, u^*, z$,

$$\mathbb{E}_x[||\nabla_\theta J_x - E_1||^2] \leq \mu_2 ||\mathbb{E}_x[\nabla_\theta J_x]||^2, \tag{9}$$

$$\mathbb{E}_x[||d_x^{JFB} - E_2||^2] \leq \mu_2 ||\mathbb{E}_x[\nabla_\theta J_x]||^2, \tag{10}$$

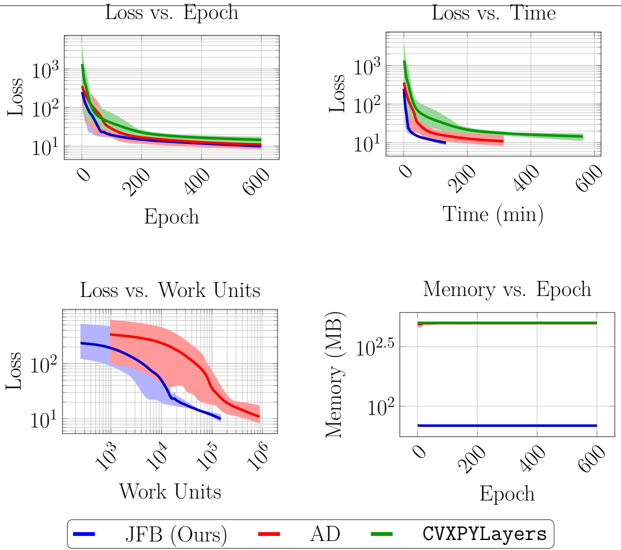
where μ_2 is a constant that depends on spectrum of $\dfrac{\partial T_\theta}{\partial \theta}$. That is, assume the variance of the true gradient and JFB are sufficiently bounded.

Theorem (Convergence)

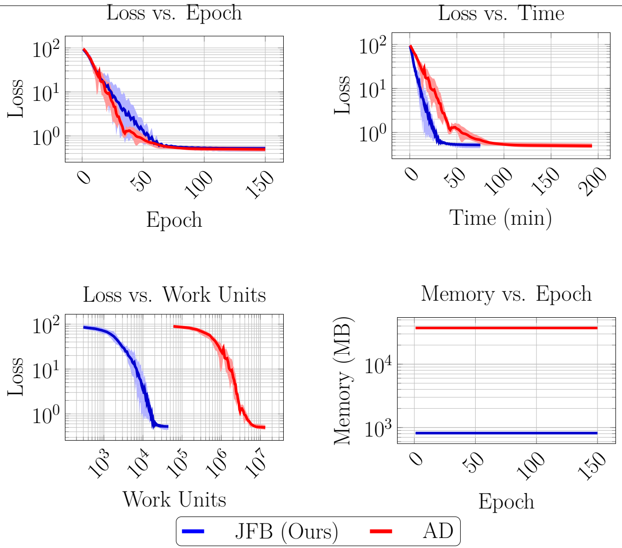
Under previous assumptions, SGD using the JFB gradient surrogate converges (in probability) to a stationary point.

Experiments

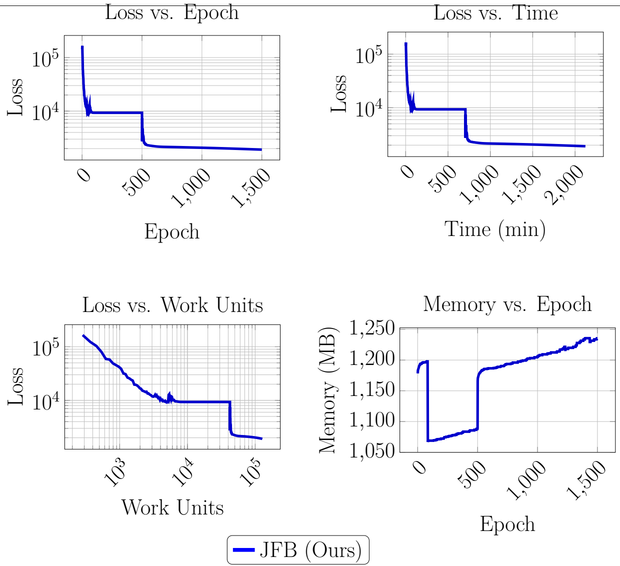
Quadrotor Dynamics with $L = \exp(\|u\|^2)$



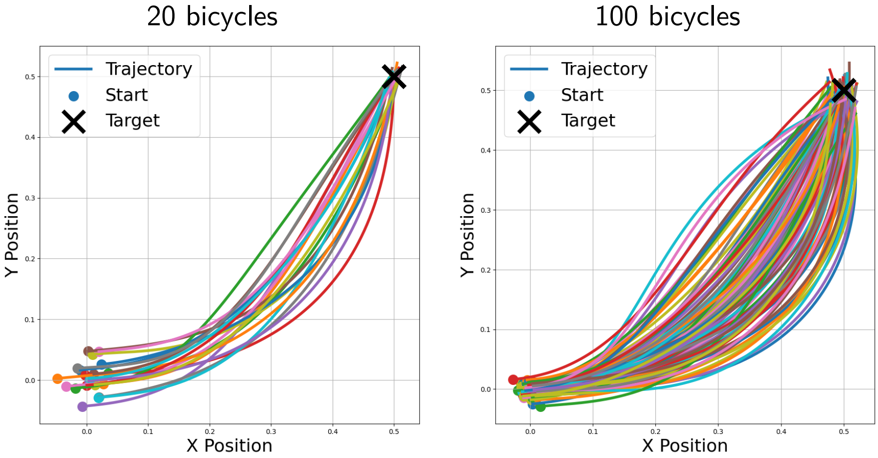
5 Interacting Bicycles (Nonlinear Control Dynamics)



100 Interacting Bicycles (Nonlinear Control Dynamics)



Bicycle Trajectories



Conclusion

- Introduce end-to-end approach for high-dimensional implicit control

Conclusion

- Introduce end-to-end approach for high-dimensional implicit control

- Implicit differentiation and AD computationally taxing even for moderately-sized problems.

Conclusion

- Introduce end-to-end approach for high-dimensional implicit control
- Implicit differentiation and AD computationally taxing even for moderately-sized problems.
- Jacobian-Free Backpropagation (JFB) allows for fast and efficient training with guarantees

Conclusion

- Introduce end-to-end approach for high-dimensional implicit control
- Implicit differentiation and AD computationally taxing even for moderately-sized problems.
- Jacobian-Free Backpropagation (JFB) allows for fast and efficient training with guarantees
- Visit poster #3 (presented by Eric Gelphman) for more details about this work!

Conclusion

- Introduce end-to-end approach for high-dimensional implicit control
- Implicit differentiation and AD computationally taxing even for moderately-sized problems.
- Jacobian-Free Backpropagation (JFB) allows for fast and efficient training with guarantees
- Visit poster #3 (presented by Eric Gelphman) for more details about this work!
- References:

  - Wu Fung, Heaton, McKenzie, Li, Yin, Osher. JFB: Jacobian-Free Backpropagation for Implicit Networks. AAAI 2022
  - Gelphman, Verma, Yang, Osher, Wu Fung. End-to-End Training of High-Dimensional Optimal Control with Implicit Hamiltonians via Jacobian-Free Backpropagation. arXiv

- Thanks to National Science Foundation Award DMS-2309810.