# Infinite Horizon Differentiable MPC

Sebastian East

**University of Oxford & NNAISENSE**

March 18, 2020

# Neural Network Background

Machine learning method for classification and regression

$$y = \hat{f}(x)$$

Neural Networks are a *function approximator*

$$\hat{f}(x) \approx f(x, \{W\}) = W_1 \phi(W_0 x)$$

Typically simple nonlinear activation functions, e.g.

$$\phi_i(x_i) = \mathrm{ReLU}(x_i) = \max\{0, x_i\}$$

'Trained' by minimising error $\ell$

$$\{W\}^\star = \underset{\{W\}}{\mathrm{argmin}}\ \ell(y, f(x, \{W\}))$$

Solution approximated using gradient-based optimization

- Backpropagation (chain rule)

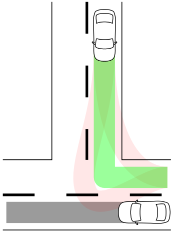
# Motivation

Significant interest in Deep Learning since the publication of AlexNet[1]

Neural networks now used for end-to-end learning based control[2]

Black box method - no guarantees of safety

- Stability
- Hard constraint satisfaction



**Goal: Introduce structure to NN architecture to provide guarantees of hard constraint satisfaction and stability.**

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012.

[2] M. Bojarski, D. D. Testa, D. Dworakowski, *et al.*, *End to end learning for self-driving cars*, 2016. [Online]. Available: `arXiv:1604.07316`.

# Overview

**Differentiable Optimization & MPC**

**Infinite Horizon Differentiable MPC**

**Numerical Experiments**

**Future Outlook & Conclusion**

# Differentiable Optimization

Solution of optimization problem as layer in neural network[3][4]

$$
\begin{aligned}
x_{l+1} = \underset{x}{\operatorname{argmin}} \; & x^\top H(x_l) x + q(x_l)^\top x \\
\text{s.t. } & l(x_l) \leq M(x_l) x \leq u(x_l)
\end{aligned}
$$

Output can be considered solution of the implicit equation $x_{l+1} = \{\text{KKT} = 0\}$, which can then be differentiated.

$$
\begin{aligned}
\mathrm{d}Hx^\star + H\mathrm{d}x + \mathrm{d}x + \mathrm{d}M^\top y^\star + \ldots &= 0 \\
\mathrm{d}Mx^\star + M\mathrm{d}x - \mathrm{d}q &= 0 \\
D(Mz^\star - u)\mathrm{d}\lambda + \ldots &= 0
\end{aligned}
$$

Can then be trained using backpropogation - no need to unroll.

---

[3] B. Amos and J. Z. Kolter, "OptNet: Differentiable Optimization as a Layer in Neural Networks," *arXiv:1703.00443 [cs, math, stat]*, Mar. 2017, arXiv: 1703.00443.

[4] A. Agrawal, B. Amos, S. Barratt, *et al.*, *Differentiable convex optimization layers*, 2019. arXiv: `1910.12430 [cs.LG]`.

# Differentiable Control

This idea allows *any* (convex) optimization-based controller to be embedded as a layer in a neural network[5]

Imitation Learning: 'expert' control behaviour, $(u, x)$, is available

$$x_{t+1} = g(x_t, u_t, d_t), \quad u_t = \hat{f}(x_t)$$

Expert controller is approximated with convex optimization policy

$$\hat{f}(x_t) \approx f(x_t, \{W\}) = \underset{u_t \in \mathcal{U}_t}{\operatorname{argmin}} \; J(x_t, u_t, \{W\})$$

Can be used to learn {system dynamics, cost function}.

[5] A. Agrawal, S. Barratt, S. Boyd, *et al.*, *Learning convex optimization control policies*, 2019. arXiv: 1912.09529 [math.OC].

# Differentiable Model Predictive Control

**Hard Constraints** dealt with systematically using MPC. Differentiable model predictive control proposed as end-to-end learning framework[6]

$$f(x_t, \{W\}) = \hat{u}_t, \quad \hat{u}_{t:t+N} = \underset{\substack{u_t \in \mathcal{U}_t, \\ \hat{x}_{t+1} = g(\hat{x}_t, u_t) \, \forall t}}{\text{argmin}} \sum_{i=t}^{t+N-1} J_t(\hat{x}_{t+1}, u_t, \{W\})$$

Can be used to learn {system dynamics, cost, constraints}

Limitations:

- Did not consider state constraints
- No guarantees of closed loop stability
- Considered very general case of MPC
  - 'solved' using box DDP: may not be convergent

[6] B. Amos, I. D. J. Rodriguez, J. Sacks, *et al.*, "Differentiable mpc for end-to-end planning and control," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, ser. NIPS'18, 2018.

# Overview

Differentiable Optimization & MPC

**Infinite Horizon Differentiable MPC**

Numerical Experiments

Future Outlook & Conclusion

# Linear Quadratic MPC

Consider only linear time invariant systems $x_{t+dt} = Ax_t + Bu_t$ with quadratic cost and box constraints, then finite horizon model predictive controller is given by

$$
\begin{aligned}
u^\star = \underset{u}{\operatorname{argmin}}\ & \frac{1}{2}\sum_{k=0}^{N-1} u_k^\top R u_k + \frac{1}{2}\sum_{k=1}^{N} x_k^\top Q x_k \\
\text{s.t. } & x_0 = x_t, \\
& x_{k+1} = Ax_k + Bu_k, \quad k \in \{0, \ldots, N-1\}, \\
& \underline{u} \le u_k \le \overline{u}, \quad k \in \{0, \ldots, N-1\}, \\
& \underline{x} \le x_k \le \overline{x}, \quad k \in \{1, \ldots, N\},
\end{aligned}
$$

Reduces to quadratic program form

+ Fast, accurate open-source solvers (e.g. OSQP)
– Badly conditioned in general

# Pre-stabilizing controller

Control input is decomposed into $u_t = Kx_t + \delta u_t$ where $\rho(A + BK) < 1$ so that

$$
\begin{aligned}
\delta u^\star = \underset{\delta u}{\operatorname{argmin}}\ & \frac{1}{2}\sum_{k=0}^{N-1} (Kx_k + \delta u_k)^\top R (Kx_k + \delta u_k) + \frac{1}{2}\sum_{k=1}^{N} x_k^\top Q x_k \\
\text{s.t. } & x_0 = x_t, \\
& x_{k+1} = (A + BK)x_k + B\delta u_k, \quad k \in \{0, \dots, N-1\}, \\
& \underline{u} \leq Kx_k + \delta u_k \leq \overline{u}, \quad k \in \{0, \dots, N-1\}, \\
& \underline{x} \leq x_k \leq \overline{x}, \quad k \in \{1, \dots, N\},
\end{aligned}
$$

**Exact same controller**, still QP, and now well conditioned in general

- Not feasible in general

# Soft Constraints

Augmented Lagrangian

$$
\begin{aligned}
\delta u^\star = \operatorname*{argmin}_{\delta u} \ & \frac{1}{2}\sum_{k=0}^{N-1}(Kx_k + \delta u_k)^\top R(Kx_k + \delta u_k) + \frac{1}{2}\sum_{k=1}^{N} x_k^\top Q x_k \\
& + k_x \sum_{k=1}^{N} 1_m^\top r_k + k_u \sum_{k=0}^{N-1} 1_n^\top s_k \\
\text{s.t. } & x_0 = x_t, \\
& x_{k+1} = (A + BK)x_k + B\delta u_k, \quad k \in \{0, \dots, N-1\}, \\
& \underline{u} - r_k \le Kx_k + \delta u_k \le \overline{u} + r_k, \quad k \in \{0, \dots, N-1\}, \\
& r \ge 0 \\
& \underline{x} - s_k \le x_k \le \overline{x} + s_k, \quad k \in \{1, \dots, N\}, \\
& s \ge 0
\end{aligned}
$$

Hard constrains **guaranteed** for sufficient cost

- No stability guarantees

# Terminal Cost

$Q_N$ can be used to provide inifnite-horizon cost

$$
\begin{aligned}
\delta u^\star = \underset{\delta u}{\operatorname{argmin}} \ & \frac{1}{2}\sum_{k=0}^{N-1}(Kx_k + \delta u_k)^\top R(Kx_k + \delta u_k) + \frac{1}{2}\sum_{k=1}^{N-1} x_k^\top Q x_k \\
& + k_x \sum_{k=1}^{N} 1_m^\top r_k + k_u \sum_{k=0}^{N-1} 1_n^\top s_k + x_N^\top Q_N x_N \\
\text{s.t. } & x_0 = x_t, \\
& x_{k+1} = (A + BK)x_k + B\delta u_k, \quad k \in \{0, \dots, N-1\}, \\
& \underline{u} - r_k \le Kx_k + \delta u_k \le \overline{u} + r_k, \quad k \in \{0, \dots, N-1\}, \\
& r \ge 0 \\
& \underline{x} - s_k \le x_k \le \overline{x} + s_k, \quad k \in \{1, \dots, N\}, \\
& s \ge 0
\end{aligned}
$$

- How do we determine $K$ and $Q_N$?

# Algebraic Riccati Equation

The infinite-horizon discrete-time linear quadratic regulator is

$$K = -(R + B^\top PB)^{-1} B^\top PA$$

where $P$ is solution of discrete time algebraic Riccati equation

$$P = A^\top PA - A^\top PB(R + B^\top PB)^{-1} B^\top PA + Q.$$

- Implement $K$ and terminal cost $Q_N = P$.
  - **For sufficient horizon,** $N$**,** $P$ **defines the infinite-horizon cost**
  - $\implies$ System is stable in closed loop, and robust to model mismatch
- K and P need to be differentiable

# Algebraic Riccati Equation Derivative

**Proposition 2.** *Let $P$ be the stabilizing solution of (8), and assume that $Z_1^{-1}$ and $(R + B^\top PB)^{-1}$ exist, then the Jacobians of the implicit function defined by (8) are given by*

$$\frac{\partial \mathrm{vec} P}{\partial \mathrm{vec} A} = Z_1^{-1} Z_2, \quad \frac{\partial \mathrm{vec} P}{\partial \mathrm{vec} B} = Z_1^{-1} Z_3, \quad \frac{\partial \mathrm{vec} P}{\partial \mathrm{vec} Q} = Z_1^{-1} Z_4, \quad \frac{\partial \mathrm{vec} P}{\partial \mathrm{vec} R} = Z_1^{-1} Z_5,$$

*where $Z_1, \ldots, Z_5$ are defined by*

$$\begin{aligned}
Z_1 &:= I_{n^2} - (A^\top \otimes A^\top)\big[I_{n^2} - (PBM_2B^\top \otimes I_n) - (I_n \otimes PBM_2B^\top) \\
&\qquad\qquad + (PB \otimes PB)(M_2 \otimes M_2)(B^\top \otimes B^\top)\big] \\
Z_2 &:= (V_{n,n} + I_{n^2})(I_n \otimes A^\top M_1) \\
Z_3 &:= (A^\top \otimes A^\top)\big[(PB \otimes PB)(M_2 \otimes M_2)(I_m^2 + V_{m,m})(I_m \otimes B^\top P) \\
&\qquad\qquad - (I_{n^2} + V_{n,n})(PBM_2 \otimes P)\big] \\
Z_4 &:= I_{n^2} \\
Z_5 &:= (A^\top \otimes A^\top)(PB \otimes PB)(M_2 \otimes M_2),
\end{aligned}$$

*and $M_1, M_2, M_3$ are defined by*

$$M_1 := P - PBM_2B^\top P, \quad M_2 := M_3^{-1}, \quad M_3 := R + B^\top PB.$$

## Proof in paper[7]

[7] S. East, M. Gallieri, J. Masci, *et al.*, "Infinite-horizon differentiable model predictive control," in *International Conference on Learning Representations*, 2020. [Online]. Available: https://openreview.net/forum?id=ryxC6kSYPr.

# Algorithm

---

**Algorithm 1** Infinite-horizon MPC Learning

---

**In:** $\mathcal{M} \setminus \mathcal{S}$, $N > 0$, $\beta \geq 0$, $N_{\text{epochs}} > 0$.
**Out:** $\mathcal{S}$
**for** $i = 0 ... N_{epochs}$ **do**
  **Forward Pass**
  $(K,\ P) \leftarrow$ DARE (7-8) solution
  $Q_T \leftarrow P$
  $\hat{u}^\star_{0:N} \leftarrow$ MPC QP (3-5) solution
  $L \leftarrow$ Imitation loss (6)
  **Backward Pass**
  Differentiate loss (6)
  Differentiate MPC QP solution, $\hat{u}^\star_{0:N}$, using Appendix B
  Differentiate DARE, $(P, K)$, using Proposition 2
  **Update step**
  $\mathcal{S} \leftarrow$ Gradient-based step

---

- Algorithm can be used to learn a subset $\mathcal{S}$ of $\mathcal{M} = \{A, B, Q, R, \underline{x}, \overline{x}, \underline{u}, \overline{u}, k_u, k_x\}$
- Learning entire set $\mathcal{M}$ simultaneously is hard in general
- $N$ is not differentiable

# Overview

Differentiable Optimization & MPC

Infinite Horizon Differentiable MPC

**Numerical Experiments**

Future Outlook & Conclusion

# Example 1: Mass Spring Damper

Nominal second order systems generated for a range of stability measures

| System | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| $c$ | 1 | 0.5 | 0.1 | -0.1 | -0.3 | -0.5 | -0.6 |

'Expert' data generated using infinite horizon MPC controller simulated in closed loop

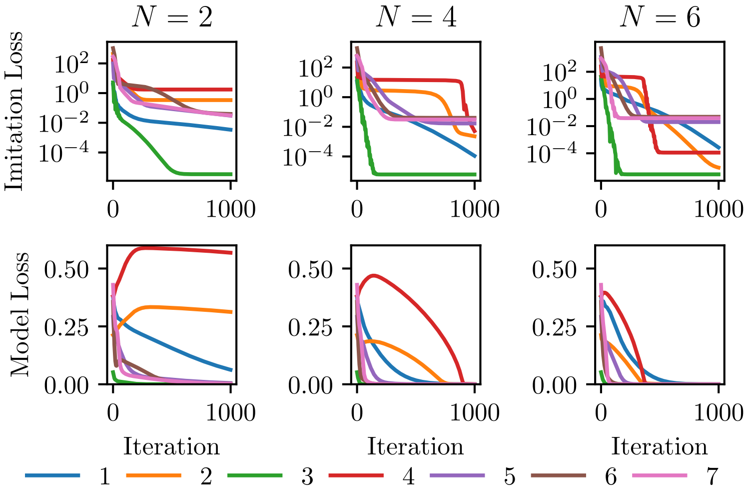
Learn system dynamics from initial random matrices $A$

Imitation loss - **control only**

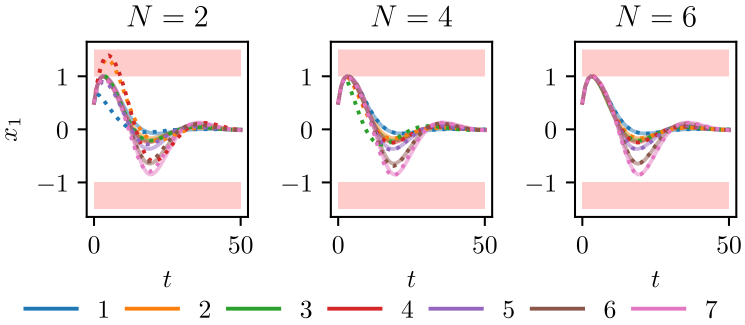
$$L = \frac{1}{T}\sum_{t=0}^{T} \|u_{t:t+Ndt} - \hat{u}^{\star}_{0:N}(x_t)\|_2^2$$

Trained with three horizons - $N \in \{2, 4, 6\}$

# Mass-Spring-Damper: Training



# Mass-Spring-Damper: Control



# Example 2: Vehicle Platooning

Higher-dimensional real world application: vehicle platooning.


Requirements

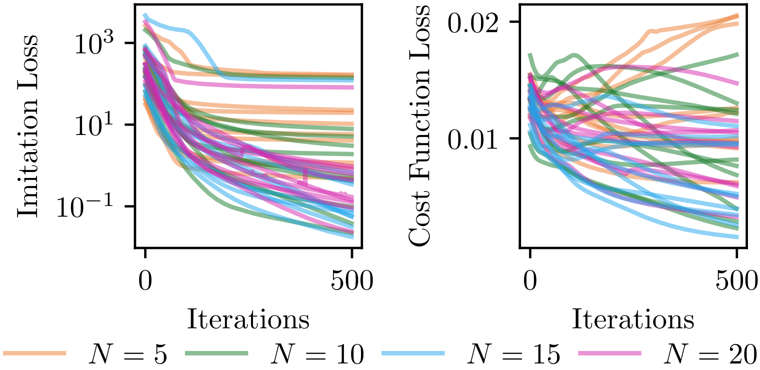
- Stabilize: $y_i - y_{i-1} \to y_{ss}$ and $\dot{y}_i - \dot{y}_{i-1} \to 0 \ \forall i$
- Safe minimum distance: $y_i - y_{i-1} \geq \underline{y} \ \forall i$
- Acceleration limits $b \leq \ddot{y}_i \leq a \ \forall i, \quad b \leq 0 \leq a$

Reduces to LTI regulation problem.

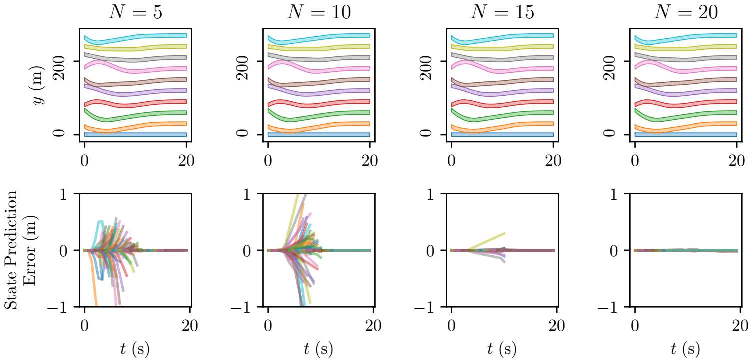
Systems generated for $y_n = 10, \implies x_t \in \mathbb{R}^{18}$ and $u_t \in \mathbb{R}^{10}$

Learned $Q$ and $R$ from random initial matrices, with $N \in \{5, 10, 15, 20\}$, in four experiments for each.

# Vehicle Platooning: Training



# Vehicle Platooning: Control



# Overview

Differentiable Optimization & MPC

Infinite Horizon Differentiable MPC

Numerical Experiments

**Future Outlook & Conclusion**

# Outlook

Main limitation is restriction to LTI systems

+ MPC solution still obtained from QP for LTV systems
– Stability becomes a significant problem over long prediction horizons

+ Can be addressed using LMI
– Challenging to enforce existence at each learning iteration

Other directions

- Deeper learning
- Reinforcement learning
- Dedicated solver(s)
- Adaptive/scenario MPC
- Scale experiments

# Conclusion

- Algorithmic advances in differentiable MPC
  - Inifnite-horizon cost obtained from solution of DARE (and differentiated)
  - Hard constraints on state and input considered
  - Solution guaranteed using augmented Lagrangian
  - QP conditioned using pre-stabilizing controller
- Algorithm demonstrated in simulation on MSD and vehicle platooning problem
- Work to be presented at ICLR 2020[8]

---

[8] S. East, M. Gallieri, J. Masci, *et al.*, "Infinite-horizon differentiable model predictive control," in *International Conference on Learning Representations*, 2020. [Online]. Available: https://openreview.net/forum?id=ryxC6kSYPr.