Deep Equilibrium Model Predictive Control

Swaminathan Gurumurthy, Khai Nguyen, Arun Bishop, Zachary Manchester, Zico J. Kolter
Carnegie Mellon University
{sgurumur, xuankhan, arunbish, zmanches, zkolter}@andrew.cmu.edu

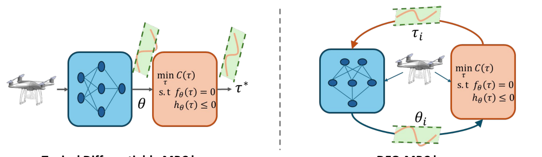

Figure 1: DEQ-MPC improves upon Diff-MPC by jointly solving both MPC parameter estimation (θ) and trajectory optimization (τ), resulting in richer feedback, consequently, increased representational power, nicer gradients, and improved amenability to warm-starting.

Differentiable Model Predictive Control (Diff-MPC) layers [1] integrate MPC as a differentiable layer within neural network architectures, embedding constraints and cost functions into the network and enabling true end-to-end training of control policies [2, 3]. While offering several advantages, standard differentiable MPC layers often treat the optimization solver as a black-box differentiable layer within the neural network (NN) architecture. This simplification, while convenient, overlooks the unique characteristics of MPC solvers that set them apart from typical NN layers. MPC solvers are implicit layers and hence inherently iterative as opposed to typical explicit layers. The outer problem often suffers from ill-conditioned, discontinuous and non-convex gradient landscapes [4, 5]. Additionally, MPC solvers frequently possess specialized structures that enable efficient warm-starting [6, 7] – a valuable property in recurrent control scenarios that is not fully leveraged in differentiable MPC frameworks.

To address these limitations, we propose Deep Equilibrium Model Predictive Control (DEQ-MPC), a novel approach that unifies the optimization solver and the neural network architecture. Instead of treating the optimization layer as just another layer within the network, we formulate a joint inference and optimization problem, where we treat the network inference and the optimization problem as a unified system and *jointly compute a fixed point* over the network outputs and the optimizer iterates. This approach, illustrated in Figure 1, allows us to condition the network outputs (optimization parameters, θ) on the optimizer state τ and vice versa. Interestingly, this can be expressed as a single constrained optimization problem:

$$\tau^*_{0:T}, \theta^* = \underset{\tau_{0:T}, \theta}{\arg\min} \quad \sum_t C_{\theta,t}(\tau_t) \tag{1}$$

$$\text{subject to} \quad x_0 = x_{\text{init}},\ x_{t+1} = f_\theta,\ h_\theta(\tau_t) \leq 0, \tag{2}$$

$$\theta = \text{NN}_\phi(x_{\text{init}}, o, \tau_{0:T}),\ t = 0, \dots, T, \tag{3}$$

where the last constraint expresses the neural network inference as an equality constraint. Typical non-linear solvers struggle with this due to the resulting constraint Jacobians of the neural network constraint. We solve this using the alternating direction method of multipliers (ADMM) algorithm [8], alternating between (i) solving the MPC optimization problem (with fixed θ), (1) and (2) using the augmented Lagrangian (AL) method and (ii) the constraint projection step, (3) (i.e, the standard neural net inference to compute θ with fixed τ). Specifically, we alternate between the following two operations for N iterations or until convergence,

$$\theta^i = \text{NN}_\phi(x_{\text{init}}, o, \tau^{i-1}), \tag{4}$$

$$\tau^i = \text{MPC-1}_{\theta^i}(x_{\text{init}}, \tau^{i-1}), \tag{5}$$

where MPC-1 performs one solver iteration of the AL solver, with the most recent parameter estimate θ^i from the network and warm-started using τ^{i-1} from the last MPC-1 iteration. The initial value τ^0 are initialized at x_{init} and zero controls across time steps. We refer to each alternating step as a DEQ-MPC-iteration, with the super-script, i, denoting the iteration count as illustrated in Figure 1. This iterative inference/optimization approach enables the network to provide an initial coarse parameter estimate and iteratively refine it based on the solver's progress.

This joint inference/optimization framework also allows us to explore several interesting aspects of the solver and architecture design. Specifically, for the optimization solver, we implement an augmented Lagrangian (AL) solver which works well with warm-starting and is robust at handling arbitrary non-linear constraints. This is important for the joint fixed point process as it allows us to change the optimization parameters (i.e, network outputs, θ_i) between successive optimization iterates. For the architecture, we experiment with parameterizing the network architecture itself as a Deep Equilibrium model (DEQ) [9], a type of implicit neural network that computes the outputs/latents as a fixed point of a non-linear transformation. It can be seen as an infinite depth network which applies the same layer an infinite number of times eventually reaching a fixed point in the outputs/latents. This iterative fixed point finding procedure blends nicely with the equilibrium/fixed point finding nature of the overall system. We observe nicer stability properties when using a DEQ as the network architecture when going to more complicated settings.

Our unified approach enables richer representations by allowing the network to adapt its features/outputs depending on the solver state. Furthermore, the inherently iterative nature of the network inference and solver optimization makes it very convenient to accommodates warm-starting on both the network outputs θ and the optimizer iterations τ, improving both computational efficiency and solution quality. DEQ-MPC thus offers a more robust and flexible framework for integrating optimization-based control with deep learning.

Results. We compare the performance of Diff-MPC and DEQ-MPC. Both approaches are evaluated with two variants: one using a DEQ network architecture and the other using a feedforward network. All experiments are conducted in a *Quadrotor-Pole* environment, where the quadrotor is initialized at a random position with an attached pole. The objective is for the quadrotor to reach the origin while swinging up the pole.

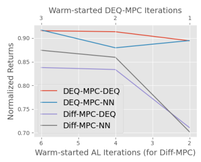

Figure 2: Warm-starting ablations

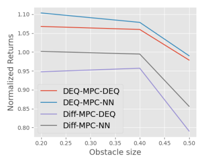

Figure 3: Constraints hardness

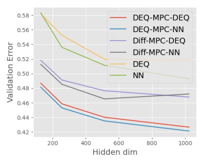

Figure 4: Network capacity ablations

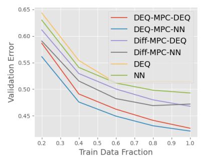

Figure 5: Generalization ablations

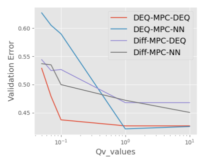

Figure 6: Cost parameter ablations

Warm-starting ablations (Figure 2): We observe that DEQ-MPC variants retain their performance even with few warm-started optimization iterations unlike Diff-MPC models. This is particularly advantageous in streaming settings where computational efficiency is crucial.

Constraint hardness (Figure 3): As the complexity of constraints increases, such as adding obstacles as inequality constraints and increasing obstacle size, DEQ-MPC's performance remains robust. Notably, DEQ-MPC variants outperform Diff-MPC models as task difficulty increases, demonstrating better scalability and adaptability to complex environments.

Scalability: network capacity and generalization (Figures 4 and 5): DEQ-MPC demonstrates clear advantages in scaling both in terms of network capacity and data availability. The results indicate that DEQ-MPC models effectively utilize higher network capacities, continuing to improve as model size increases, while Diff-MPC models show saturation beyond certain capacity limits.

Cost sensitivity ablations (Figure 3): As the problem sensitivity increases by varying cost parameters, DEQ-MPC models show more stability compared to Diff-MPC models, which become unstable with ill-conditioned cost matrices. This robustness is crucial for maintaining reliable performance in sensitive control problems.

Overall, DEQ-MPC demonstrates clear advantages in warm-starting, constraint handling, scalability, generalization and robustness. These results suggest that DEQ-MPC is well-suited for complex control tasks, especially when requiring efficient streaming and robust handling of hard constraints. Future work will explore broader applications of our approach beyond controls and robotics.

References

[1] B. Amos, I. Jimenez, J. Sacks, B. Boots, and J. Z. Kolter. Differentiable mpc for end-to-end planning and control. *Advances in neural information processing systems*, 31, 2018.

[2] J. Shrestha, S. Idoko, B. Sharma, and A. K. Singh. End-to-end learning of behavioural inputs for autonomous driving in dense traffic. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10020–10027. IEEE, 2023.

[3] X. Xiao, T. Zhang, K. Choromanski, E. Lee, A. Francis, J. Varley, S. Tu, S. Singh, P. Xu, F. Xia, et al. Learning model predictive controllers with real-time attention for real-world navigation. *arXiv preprint arXiv:2209.10780*, 2022.

[4] R. Antonova, J. Yang, K. M. Jatavallabhula, and J. Bohg. Rethinking optimization with differentiable simulation from a global perspective. In *Conference on Robot Learning*, pages 276–286. PMLR, 2023.

[5] H. J. Suh, M. Simchowitz, K. Zhang, and R. Tedrake. Do differentiable simulators give better policy gradients? In *International Conference on Machine Learning*, pages 20668–20696. PMLR, 2022.

[6] T. A. Howell, B. E. Jackson, and Z. Manchester. Altro: A fast solver for constrained trajectory optimization. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7674–7679. IEEE, 2019.

[7] K. Nguyen, S. Schoedel, A. Alavilli, B. Plancher, and Z. Manchester. Tinympc: Model-predictive control on resource-constrained microcontrollers. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–7. IEEE, 2024.

[8] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, July 2011. ISSN 1935-8237, 1935-8245. doi:10.1561/2200000016. URL `https://www.nowpublishers.com/article/Details/MAL-016`. Publisher: Now Publishers, Inc.

[9] S. Bai, J. Z. Kolter, and V. Koltun. Deep equilibrium models. *Advances in neural information processing systems*, 32, 2019.