Enforcing State Constraints in Dynamical Systems Modelled with Neural Networks

Namhoon Cho[1][0000−0001−9999−9501] and Davide Amato[2][0000−0002−9263−0485]

[1] Cranfield University, Cranfield, Bedfordshire, MK43 0AL, United Kingdom
n.cho@cranfield.ac.uk
[2] Imperial College London, South Kensington Campus, London, SW7 2AZ, United Kingdom
d.amato@imperial.ac.uk

Introduction

Deep neural networks (NNs) are usually trained with unconstrained optimisation algorithms [3]. With a reasoning similar to the constrained Kalman filter [4], incorporating known information in the form of equality constraints at certain checkpoints can potentially improve prediction accuracy. For continuous-time dynamical systems, the state constraints should be enforced in an ordinary differential equation (ODE) model which embeds NNs to represent a learned part of dynamics or a control policy. To this end, incremental correction methods are developed for post-processing of the dynamical systems modelled with NNs for which the parameters are determined by previous optimisation process. The proposed approach is to find a small amount of local correction needed to satisfy given state constraints with the updated solution. Algorithms for updating the neural network parameters and the control function are considered.

Incremental Correction Methods

Consider the continuous-time system dynamics given by

$$\dot{\mathbf{x}}(t) = \mathbf{f}(t, \mathbf{x}(t), \boldsymbol{\theta}) \tag{1}$$

where t, $\mathbf{x}$, and $\boldsymbol{\theta}$ denote the time, the state, and a vector of parameters, respectively. In control problems, the mathematical description for the dynamic system has the control input as its argument while the parameters enter Eq. (1) through the policy $\boldsymbol{\pi}$:

$$\dot{\mathbf{x}}(t) = \mathbf{f}(t, \mathbf{x}(t), \mathbf{u}(t)), \qquad \mathbf{u}(t) = \boldsymbol{\pi}(t, \mathbf{x}(t), \boldsymbol{\theta}) \tag{2}$$

The state constraints are expressed as

$$\mathbf{x}(t_i) = \mathbf{x}_i \tag{3}$$

for $i = 1, \cdots, N$. The NN policy can be obtained through gradient-descent-based unconstrained optimisation in order to minimise a certain cost/loss function with the gradients computed by using the adjoint method [2]. The resulting NN can provide sufficient performance in terms of state constraint violation, however, the focus of this study is to incorporate Eq. (3) as the hard constraints. The aim of this study is to correct $\boldsymbol{\theta}$ in Eq. (1) or $\mathbf{u}(t)$ in Eq. (2) to satisfy Eq. (3).

Parameter Correction. A NN embedded in the ODE function is implicit in Eq. (1). Suppose that the pre-trained NN has its optimised, nominal parameters denoted by $\boldsymbol{\theta}^*$. Given the nominal parameters, the corresponding nominal solution can be obtained by propagating the system model $\dot{\mathbf{x}}^*(t) = \mathbf{f}(t, \mathbf{x}^*(t), \boldsymbol{\theta}^*)$ with a given initial condition $\mathbf{x}^*(t_0) = \mathbf{x}_0$. The system dynamics can be linearised around the nominal trajectory and the nominal parameter, leading to a linear time-varying form of the perturbation dynamics:

$$\dot{\delta\mathbf{x}}(t) \approx \mathbf{A}(t)\,\delta\mathbf{x}(t) + \mathbf{B}_\theta(t)\,\delta\boldsymbol{\theta} \tag{4}$$

By using linear control design techniques, one can find the parameter correction $\delta\boldsymbol{\theta}$ to achieve $\delta\mathbf{x}(t_i) = \mathbf{x}_i - \mathbf{x}^*(t_i)$ so that Eq. (3) holds.

Control Function Correction. In controlled dynamical systems, one can consider perturbations in the control input instead of the neural network parameters, which brings about another form of linear time-varying perturbation dynamics:

$$\dot{\delta\mathbf{x}}(t) \approx \mathbf{A}(t)\,\delta\mathbf{x}(t) + \mathbf{B}_u(t)\,\delta\mathbf{u}(t) \tag{5}$$

Again, the control function correction $\delta\mathbf{u}(t)$ can be designed to achieve Eq. (3) by using various linear control methods.

Concluding Remarks

This study presents refinement methods for improving the prediction accuracy of continuous-time dynamic systems involving neural networks. Depending on the application, the proposed methods can be useful for dynamic model learning as well as finite-horizon control purposes. The control function correction approach can be useful for online implementation with its relative advantage in computational efficiency. We show practical applications of the methods developed in this work to spacecraft guidance problems, such as planetary entry, descent, and landing [1] and low-thrust transfers.

References

1. Amato, D., McMahon, J.W.: Deep learning method for martian atmosphere reconstruction. Journal of Aerospace Information Systems **18**(10), 728–738 (2021). https://doi.org/10.2514/1.I010922
2. Cho, N., Shin, H.S.: Optimisation of structured neural controller based on continuous-time policy gradient (2022), https://arxiv.org/abs/2201.06262, arXiv:2201.06262
3. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning, chap. 8, pp. 271–325. MIT Press (2016), http://www.deeplearningbook.org
4. Simon, D., Chia, T.L.: Kalman filtering with state equality constraints. IEEE Transactions on Aerospace and Electronic Systems **38**(1), 128–136 (2002). https://doi.org/10.1109/7.993234