Physics-Informed Neural Network

이동로봇 튜토리얼 #2

박수환

경희대학교
로봇 제어 및 지능 연구실 (Prof. 김상현)

경희대학교
KYUNG HEE UNIVERSITY

Optimizer

ADAM

- Adam은 Momentum과 Adaptive Learning Rate를 결합해 기울기의 1차·2차 모멘트를 동시에 추정
- 편향 보정을 수행하여 안정적이고 빠른 파라미터 업데이트를 가능

ADAM 수식

1. First Momentum

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$$

2. Second Momentum

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2$$

3. Bias Correction

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t} \qquad \hat{m}_t = \frac{m_t}{1-\beta_1^t}$$

4. Update

$$\theta_t = \theta_{t-1} - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

ADAM

- Adam은 Momentum과 Adaptive Learning Rate를 결합해 기울기의 1차·2차 모멘트를 동시에 추정
- 편향 보정을 수행하여 안정적이고 빠른 파라미터 업데이트를 가능

모멘텀이 왜 필요한가?

Function $f(p) = p^2$

Start Point $p_0 = 10$

Gradient $f'(p) = 2p$

Learning Rate $\eta = 0.1$

Update Function $p_{t+1} = p_t - \eta * f^{'}(p_t)$

Iteration	Current p	Gradient g=2 p	Update 0.1·g	New p
0	10.000	20.000	2.000	8.000
1	8.000	16.000	1.600	6.400
2	6.400	12.800	1.280	5.120
3	5.120	10.240	1.024	4.096
4	4.096	8.192	0.819	3.277

수렴에 시간이 오래 걸림

Optimizer

경희대학교 KYUNG HEE UNIVERSITY

ADAM

- Adam은 Momentum과 Adaptive Learning Rate를 결합해 기울기의 1차·2차 모멘트를 동시에 추정
- 편향 보정을 수행하여 안정적이고 빠른 파라미터 업데이트를 가능

모멘텀이 왜 필요한가?

Function $f(p) = p^2$

Gradient $f'(p) = 2p$

Start Point $p_0 = 10$

Learning Rate $\eta = 0.1$

Update Function

$$v_{t+1} = 0.5 * v_t + g$$
$$p_{t+1} = p_t - \eta * v_{t+1}$$

Iter	Current p	Gradient g	Velocity v = 0.5·v + g	New p
0	10.000	20.000	20.000	8.000
1	8.000	16.000	26.000	5.400
2	5.400	10.800	23.800	3.020
3	3.020	6.040	17.940	1.226
4	1.226	2.452	11.422	0.084

수렴이 빠름

ADAM

- Adam은 Momentum과 Adaptive Learning Rate를 결합해 기울기의 1차·2차 모멘트를 동시에 추정
- 편향 보정을 수행하여 안정적이고 빠른 파라미터 업데이트를 가능

파라미터 별 학습률의 필요성

Function $f(p) = 50p_0^2 + p_1^2$

Gradient $g = [\,100p_0,\ 2p_1\,]$

Start Point $p = (1.5,\ 10.0)$

Learning Rate $\eta = 0.01$

Update Function

$$p_{t+1} = p_t - \eta * g$$

Iter	Current p	Gradient g	Update 0.01·g	New p
0	(1.50, 10.00)	[150.0, 20.0]	[1.500, 0.200]	(0.00, 9.80)
1	(0.00, 9.80)	[0.0, 19.6]	[0.000, 0.196]	(0.00, 9.60)
2	(0.00, 9.60)	[0.0, 19.2]	[0.000, 0.192]	(0.00, 9.41)
3	(0.00, 9.41)	[0.0, 18.8]	[0.000, 0.188]	(0.00, 9.22)

각 파라미터별 속도가 다름

Optimizer

ADAM

- Adam은 Momentum과 Adaptive Learning Rate를 결합해 기울기의 1차·2차 모멘트를 동시에 추정
- 편향 보정을 수행하여 안정적이고 빠른 파라미터 업데이트를 가능

파라미터 별 학습률의 필요성

Function $f(p) = 50p_0^2 + p_1^2$

Gradient $g = [\,100p_0,\ 2p_1\,]$

Start Point $p = (1.5,\ 10.0)$

Learning Rate $\eta = 0.01$

Update Function

$$\eta_a = 1.5/(\sqrt{g^2} + \varepsilon)$$

$$p_{t+1} = p_t - \eta_a * g$$

Iter	p	g	g_squared	Effective LR	Update	New p
0	(1.5, 10.0)	[150, 20]	[22500, 400]	[0.01, 0.075]	[1.5, 1.5]	(0.0, 8.5)
1	(0.0, 8.5)	[0, 17]	[22500, 689]	[0.01, 0.057]	[0, 0.97]	(0.0, 7.53)
2	(0.0, 7.53)	[0, 15.1]	[22500, 916]	[0.01, 0.050]	[0, 0.75]	(0.0, 6.78)
3	(0.0, 6.78)	[0, 13.6]	[22500, 1100]	[0.01, 0.045]	[0, 0.61]	(0.0, 6.17)

ADAM

- Adam은 Momentum과 Adaptive Learning Rate를 결합해 기울기의 1차·2차 모멘트를 동시에 추정
- 편향 보정을 수행하여 안정적이고 빠른 파라미터 업데이트를 가능

ADAM이란?

방금 설명한 두가지 알고리즘의 장점만 적용시킴!

1. Momentum 업데이트 $$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$$

2. 학습률 업데이트 변수 $$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2$$

3. 초기값이 너무 작음 방지 $$\hat{m}_t = \frac{m_t}{1-\beta_1^t}, \qquad \hat{v}_t = \frac{v_t}{1-\beta_2^t}$$

3. 파라미터 업데이트 $$\theta_t = \theta_{t-1} - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon}$$

L-BFGS

- 대규모 문제에서는 Hessian 계산·저장 비용이 너무 커 Newton Method를 사용할 수 없음
- 따라서 적은 메모리로 곡률 정보를 근사할 수 있는 L-BFGS가 필요

문제 제기

Newton Method

$$p_{t+1} = p_t - H^{-1}(p_t)\,\nabla f(p_t)$$

Newton 방법은 Hessian의 역행렬을 사용해 2차 곡률 정보까지 활용하는 최적화 방법
하지만 Hessian 계산 비용이 매우 크고 역행렬 계산은 O(n3) 이라 **대규모 문제에 부적합**

L-BFGS

- 대규모 문제에서는 Hessian 계산·저장 비용이 너무 커 Newton Method를 사용할 수 없음
- 따라서 적은 메모리로 곡률 정보를 근사할 수 있는 L-BFGS가 필요

핵심 아이디어

L-BFGS는 최근 m개의 변화량(s, y)만 저장하여 Hessian의 역행렬을 직접 계산하지 않고도 Newton의 곡률 정보를 근사하는 방법

BFGS

$$\nabla f(p_{t+1}) \approx \nabla f(p_t) + H\,(p_{t+1} - p_t)$$

$$H_{t+1} s_t = y_t \qquad s_t = p_{t+1} - p_t, \qquad y_t = \nabla f(p_{t+1}) - \nabla f(p_t)$$

$$H_{t+1}^{-1} = \left(I - \rho_t s_t y_t^\top\right) H_t^{-1} \left(I - \rho_t y_t s_t^\top\right) + \rho_t\, s_t s_t^\top, \qquad \rho_t = \frac{1}{y_t^\top s_t}$$

최종적인 Hessian Inverse 업데이트 방식

L-BFGS

- 대규모 문제에서는 Hessian 계산·저장 비용이 너무 커 Newton Method를 사용할 수 없음
- 따라서 적은 메모리로 곡률 정보를 근사할 수 있는 L-BFGS가 필요

핵심 아이디어

L-BFGS는 최근 m개의 변화량(s, y)만 저장하여 Hessian의 역행렬을 직접 계산하지 않고도 Newton의 곡률 정보를 근사하는 방법

L-BFGS

$$H_{k+1} = \left(I - \rho_k s_k y_k^T\right) H_k \left(I - \rho_k y_k s_k^T\right) + \rho_k s_k s_k^T \quad s_i = x_{i+1} - x_i, \qquad y_i = g_{i+1} - g_i, \qquad \rho_i = \frac{1}{y_i^T s_i}$$

$$H_k g_k = \left(I - \rho_{k-1} s_{k-1} y_{k-1}^T\right) H_{k-1} \left(I - \rho_{k-1} y_{k-1} s_{k-1}^T\right) g_k + \rho_{k-1} s_{k-1} s_{k-1}^T g_k$$

H_k는 H_{k-1}, H_{k-2}...에 의존하는 점화식 형태

$$\alpha_i = \rho_i s_i^T q, \qquad q \leftarrow q - \alpha_i y_i$$
$$\beta_i = \rho_i y_i^T r, \qquad r \leftarrow r + s_i(\alpha_i - \beta_i)$$
$$H_k g_k = r$$

Optimizer

경희대학교 KYUNG HEE UNIVERSITY

ADAM

- 비용이 매우 저렴하고 실시간 학습에 강함
- 하이퍼파라미터 튜닝 필요성이 낮고 기본값이 잘 작동
- 대규모 신경망 / 비선형 / online-learning에서 효과적

L-BFGS

- 2차 곡률 정보를 근사해 매우 정밀한 최적화가 가능
- 손실 함수의 미세한 굴곡까지 반영할 수 있어 고정밀 문제(PINN, PDE)에 특히 강함
- 적은 메모리로 Newton 수준의 수렴 특성을 재현함

따라서 PINN은 ADAM으로 1차 최적화 후 L-BFGS를 활용한 2차 최적법을 많이 활용함

PINN MPC

- 정확한 동역학 모델을 알기 어려운 시스템에 대해 PINN이 Dynamics를 학습하여 MPC의 예측 모델로 사용
- PINN은 미분 가능하고 계산이 빠르기 때문에 NMPC의 반복 최적화에 적합
- 비선형·고차원 시스템에서도 더 정확한 예측과 안정적인 제어 성능을 제공

Available online at www.sciencedirect.com
ScienceDirect
IFAC Papers Online
CONFERENCE PAPER ARCHIVE
IFAC PapersOnLine 55-20 (2022) 331–336

Physics-informed Neural Networks-based Model Predictive Control for Multi-link Manipulators ⋆

Jonas Nicodemus * Jonas Kneifl ** Jörg Fehr **
Benjamin Unger *
* Stuttgart Center for Simulation Science (SC SimTech), University of Stuttgart, Universitätsstr. 32, 70569 Stuttgart, Germany (e-mail: {jonas.nicodemus,benjamin.unger}@simtech.uni-stuttgart.de)
** Institute of Engineering and Computational Mechanics, University of Stuttgart, Pfaffenwaldring 9, 70569 Stuttgart, Germany (e-mail: {joerg.fehr,jonas.kneifl}@itm.uni-stuttgart.de).

Abstract: We discuss nonlinear model predictive control (MPC) for multi-body dynamics via physics-informed machine learning methods. In more detail, we use a physics-informed neural networks (PINNs)-based MPC to solve a tracking problem for a complex mechanical system, a multi-link manipulator. PINNs are a promising tool to approximate (partial) differential equations but are not suited for control tasks in their original form since they are not designed to handle variable control actions or variable initial values. We thus follow the strategy of Antonelo et al. (arXiv:2104.02556, 2021) by enhancing PINNs with adding control actions and initial conditions as additional network inputs. Subsequently, the high-dimensional input space is reduced via a sampling strategy and a zero-hold assumption. This strategy enables the controller design based on a PINN as an approximation of the underlying system dynamics. The additional benefit is that the sensitivities are easily computed via automatic differentiation, thus leading to efficient gradient-based algorithms for the underlying optimal control problem.

Keywords: Physics-informed Machine Learning, Model Predictive Control, Surrogate Model, Mechanical System, Real-time Control

1. INTRODUCTION

Model predictive control (MPC) is a flexible and intuitive control scheme that lets us impose constraints and helps to operate complex systems optimally. The major challenge for MPC is the repetitive solution of an optimal control problem. Even with today's computing power, efficient model representation to be real time capable remains the bottleneck. This issue is especially prominent in systems with a fast dynamic, like robotic manipulators, where operation speed relates to increased productivity.

In this work, we study a tracking problem for a multi-link manipulator, see section 2 for further details, with an a-priori unknown tracking trajectory. In this scenario, standard linearization strategies for the nonlinear dynamics, which are used to speed up the computation, cannot be implemented without further challenges. Instead, a repetitive numerical evaluation of the underlying nonlinear dynamics is required. Since the time required by standard time integration schemes may pose a critical constraint during the solution of the optimal control problem, we propose replacing the time-integration with a *machine learning* (ML) approach. Since standard ML techniques typically require an extensive training data set and cannot compete with state-of-the-art time-integrators (Otness et al., 2021), we propose a physics-informed approach, see Karniadakis et al. (2021) for a recent overview, thus exploiting the underlying known physical law during the training process. More precisely, we approximate the solution of the nonlinear dynamics with a *physics-informed neural network* (PINN), initially introduced by Raissi et al. (2019). Other approaches to tackle this issue are Deep Lagrangian Networks or Hamiltonian Neural Network (cf. Lutter and Peters (2021)), which are based on Neural Ordinary Differential Equations; see Chen et al. (2019). In contrast to our approach, these approaches incorporate Lagrangian or Hamiltonian mechanics into model learning rather than using the explicit differential equation, and thus rely on data.

Our main results are the following:

(1) Following ideas presented by Antonelo et al. (2021), we detail in section 3.3 how to replace the nonlinear dynamics with a PINN approximation in the context of *nonlinear model predictive control* (NMPC). Due to the efficient computation of the partial derivative of the PINN with respect to the controller, the optimal control problem can be solved efficiently with a gradient-based method, without any adjoint computations (for gradient-based methods) or acceleration strategies (for nonlinear programming methods).

⋆ The authors acknowledge funding from the DFG under Germany's Excellence Strategy – EXC 2075 – 390740016 and are thankful for support by the Stuttgart Center for Simulation Science (SimTech).

10.1016/j.ifacol.2022.09.117

$$\min_{u_0,\dots,u_{N-1}} \quad J = \sum_{k=0}^{N-1} \ell\left(x_k^{\mathrm{ref}}, x_k, u_k\right)$$

$$\text{s.t.} \quad x_{k+1} = \varphi(\tau, u_k, x_k), \qquad k = 0, \dots, N-1,$$

$$x_0 = x(t_0),$$

$$x_k \in \mathcal{X}, \quad u_k \in \mathcal{U}, \qquad k = 0, \dots, N-1.$$

- 기존 MPC는 정확한 모델 φ가 필요하지만, 실제 시스템의 비선형·고차원 Dynamics는 모델링이 어려움
- PINN은 실제 Dynamics를 물리 제약과 데이터로 학습하여 $\varphi(\tau,u_k,x_k)$를 대체하는 고정밀 예측 모델을 제공
미분 가능하고 빠르게 계산되므로 NMPC 최적화에 직접 활용 가능
(자세한 문의는 민형씨에게)

[1] J. Nicodemus, J. Kneifl, J. Fehr, and B. Unger,"Physics-informed neural networks-based model predictive control for multi-link manipulators,"IFAC-PapersOnLine, vol. 55, no. 20, pp. 331–336, 2022

감사합니다