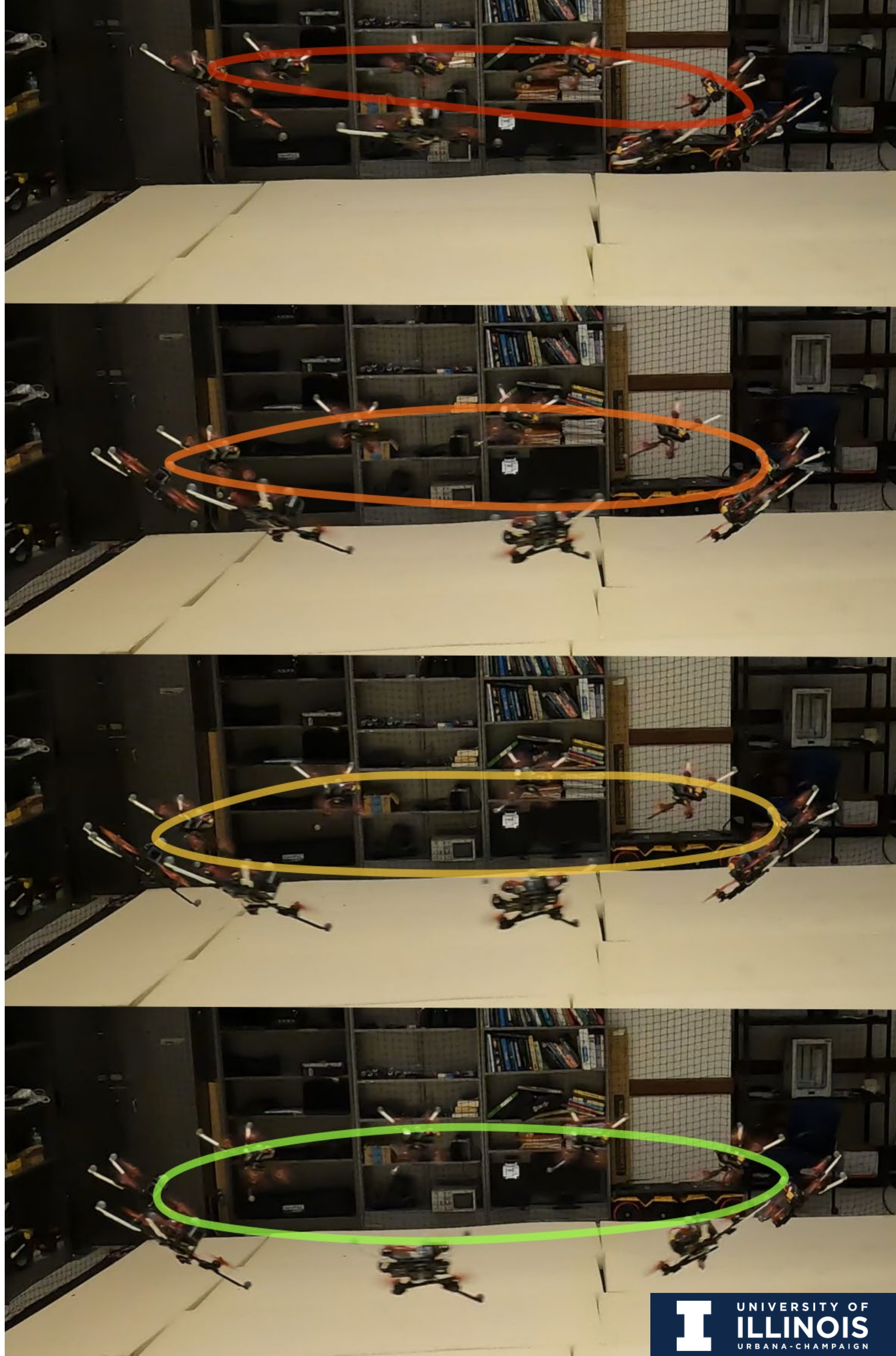
DiffTune: Auto-Tuning through Auto-Differentiation

Sheng Cheng

Postdoctoral Research Associate

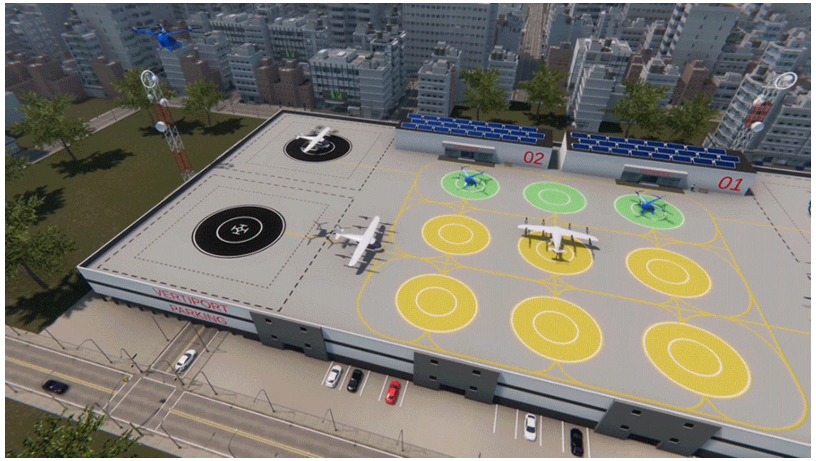
Joint work with Minkyung Kim, Lin Song, Chengyu Yang, Yiquan Jin, Shenlong Wang, and Naira Hovakimyan

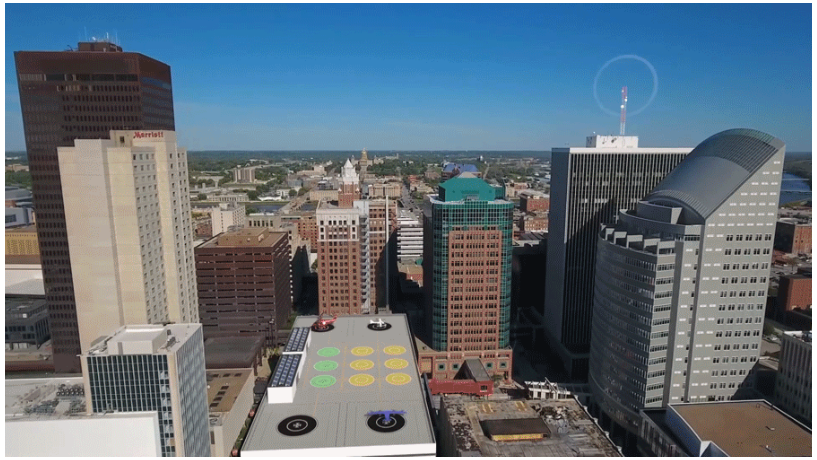
AVIATE Seminar
September 8, 2023



UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

Advanced Air Mobility





Credits: NASA Graphics/Kyle Jenkins

Motivation and Background

- Controller design -> **Controller tuning** -> Performance delivery
- Tuning is often done by hand: tedious and time-consuming.
  - Nonlinear relationship between parameters and performance metric;
  - Tuning decisions scale exponentially to number of parameters (in the worst case).
- Solution: auto-tuning

	Model-based	**Model-free**
Pros	Leveraging system's knowledge Stability by design (e.g., Lyapunov)	Data-driven with many tools (GPR, DNN, etc) Generally, no assumptions on the system
Cons		
Literature		

Combine the best of the two worlds?

Problem Setup of Controller Tuning

Goal: tune parameter $\boldsymbol{\theta}$ to achieve smaller loss L, e.g., tracking error $L=\sum_{k=0}^{N}\|\boldsymbol{x}_k - \bar{\boldsymbol{x}}_k\|^2$

dynamical system:

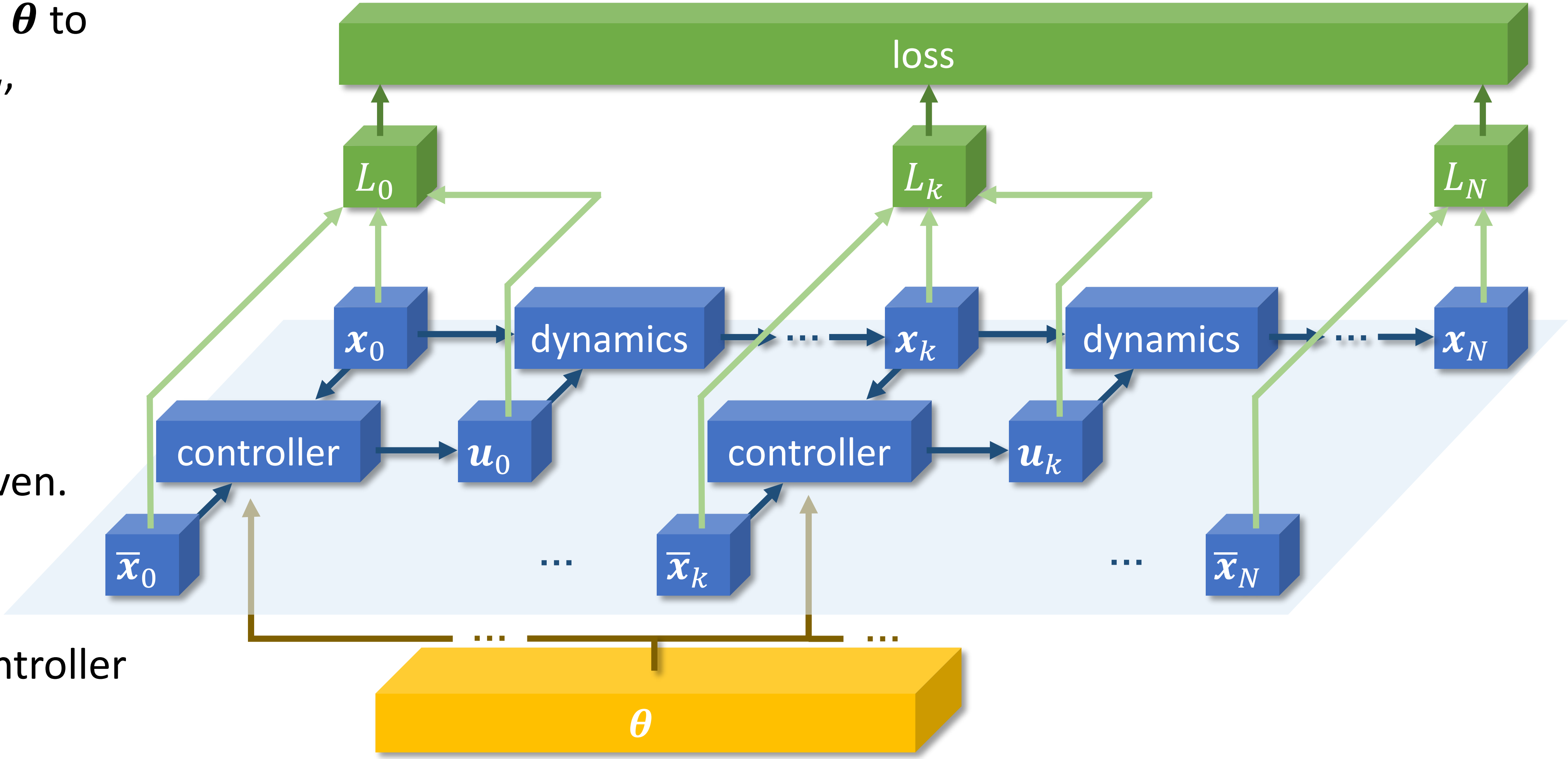
$$\boldsymbol{x}_{k+1} = f(\boldsymbol{x}_k, \boldsymbol{u}_k)$$

with initial state $\boldsymbol{x}_0$ given.

feedback tracking controller

$$\boldsymbol{u}_k = h(\boldsymbol{x}_k, \bar{\boldsymbol{x}}_k, \boldsymbol{\theta})$$

parameterized by $\boldsymbol{\theta}$



S. Cheng, M. Kim, L. Song, C. Yang, Y. Jin, S. Wang, N. Hovakimyan, "DiffTune: Auto-Tuning through Auto-Differentiation," under review. https://arxiv.org/abs/2209.10021

DiffTune for Controller Tuning

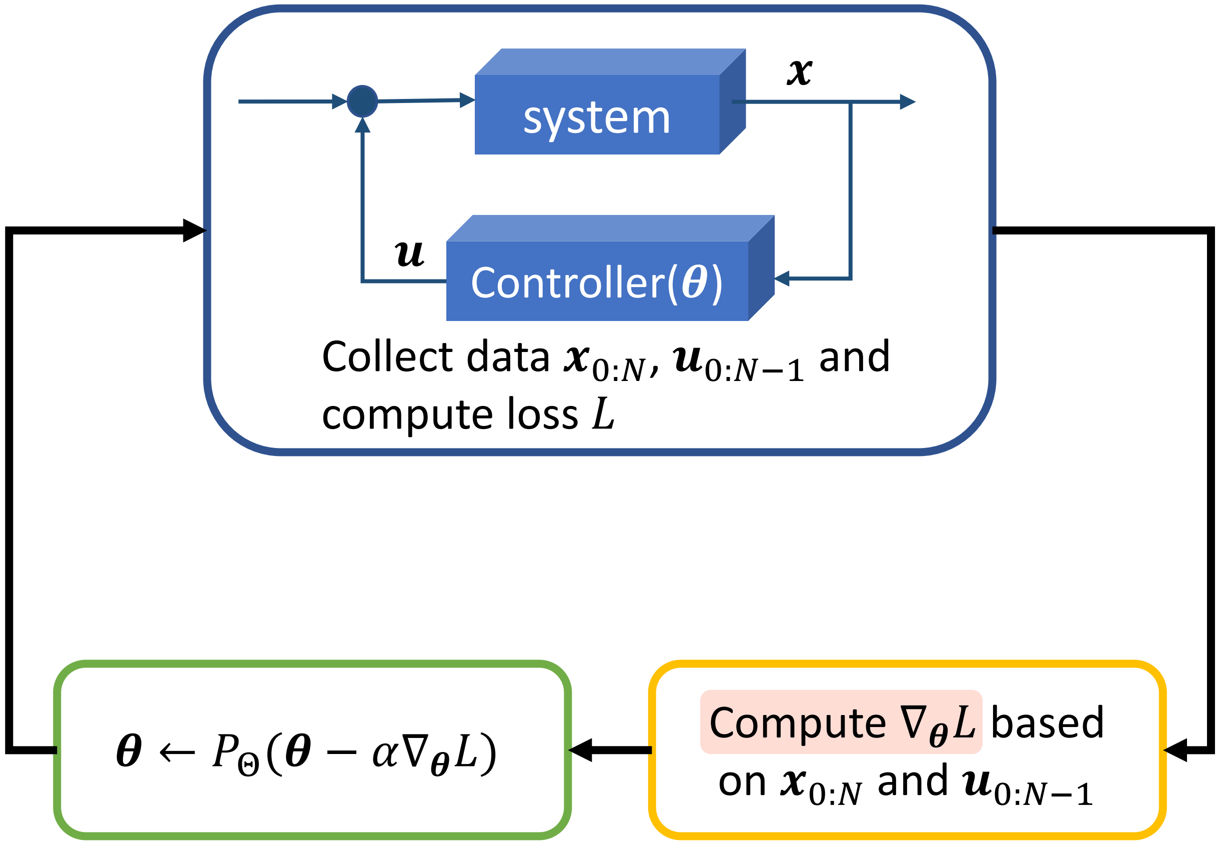
- Optimization formulation

$$\begin{aligned} &minimize_{\boldsymbol{\theta}\in\Theta} \quad L(\boldsymbol{\theta}) \\ &subject\ to \quad \boldsymbol{x}_{k+1} = f(\boldsymbol{x}_k, \boldsymbol{u}_k) \\ &\qquad\qquad\quad\ \ \boldsymbol{u}_k = h(\boldsymbol{x}_k, \overline{\boldsymbol{x}}_k, \boldsymbol{\theta}) \end{aligned}$$

- Requiring differentiable dynamics and controller
  - Many controllers are differentiable, e.g., PID [Kumar, 2021], MPC [Amos, 2018;East, 2020], optimal control [Jin, 2020, 2022], CBF [Parwana, 2021; Vien, 2021].
- Solution method: ***DiffTune***[1]
  - Gradient-based method;
  - Stability, real-data compatibility, and efficiency.

1. S. Cheng, M. Kim, L. Song, C. Yang, Y. Jin, S. Wang, N. Hovakimyan, “DiffTune: Auto-Tuning through Auto-Differentiation,” under review. https://arxiv.org/abs/2209.10021

How does DiffTune work?

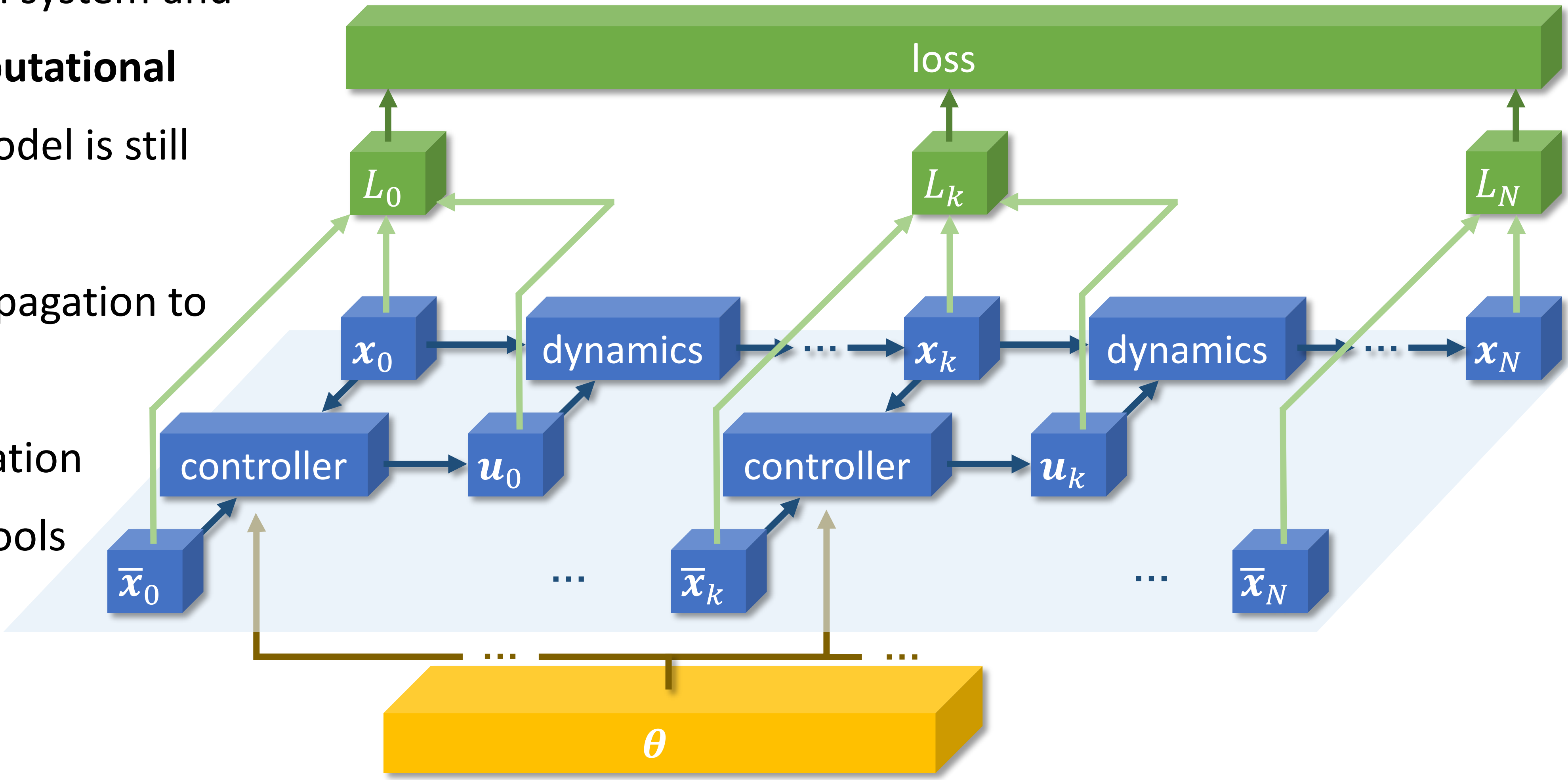


Example: tuning a geometric controller[1] for quadrotor tracking control

1. T. Lee, M. Leok, and N. H. McClamroch, "Geometric tracking control of a quadrotor UAV on SE(3)," in IEEE Conference on Decision and Control (CDC), 2010, pp. 5420–5425.

How to obtain the gradient $\nabla_{\boldsymbol{\theta}} L$?

- Unroll the dynamical system and controller as a **computational graph**, where the model is still interpretable;
- Apply backward propagation to obtain $\nabla_{\boldsymbol{\theta}} L$;
- Efficient implementation using off-the-shelf tools from the learning community



PyTorch TensorFlow

Pros and Cons of Backward Propagation

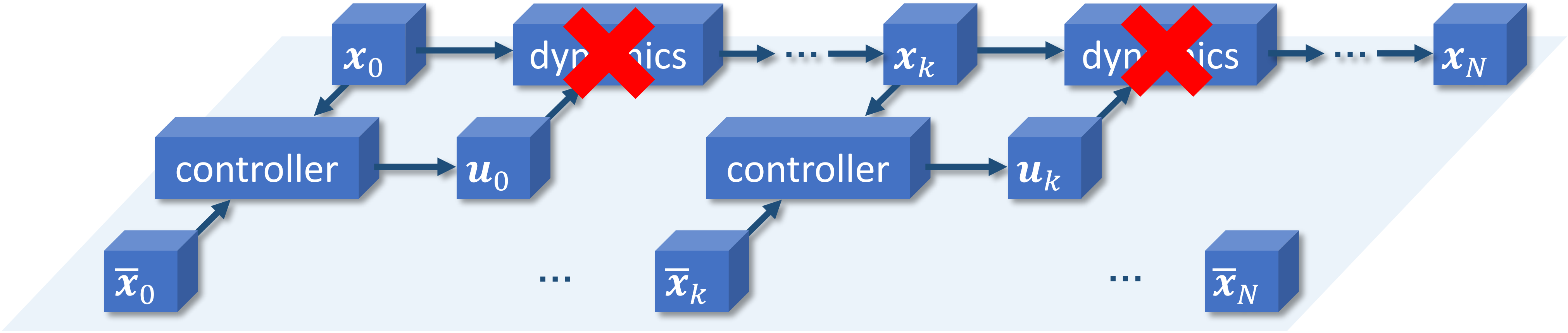
Pros

- Leveraging the **efficient computational tools** developed in ML community.
- The gradient $\nabla_{\boldsymbol{\theta}} L$ is **analytical**
  - Not computed by approximations (e.g., finite difference) or symbolic approaches
  - Analytical gradients yield better informed descent direction than numerical gradients.

Cons

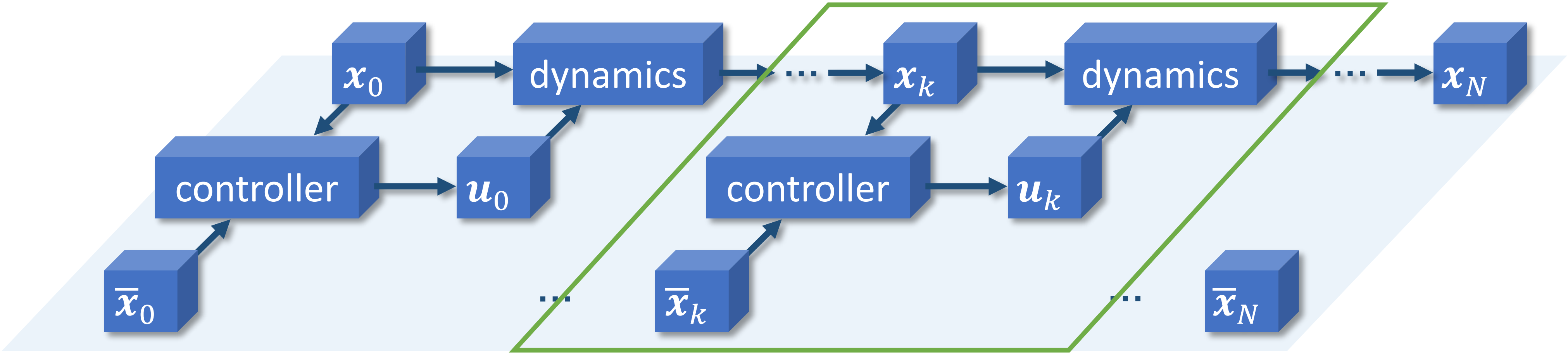
- **Incompatible** with data from a physical system

Why Backward Propagation is Real-Data Incompatible?



- Backward propagation relies on the computational graph:
  - The graph is **traversed forward first** and then **backpropagated** to compute $\nabla_{\boldsymbol{\theta}} L$.
  - The new states are obtained by **evaluating the "dynamics,"** only feasible in simulations.
- For a real system:
  - The new states are obtained by **sensor measurement** or **state estimation**.
  - The computational graph is **broken**.

Gradient Computation using Sensitivity Propagation



- Decomposition by chain rule:

$$\nabla_{\boldsymbol{\theta}} L = \sum_{k=0}^{N} \frac{\partial L}{\partial \boldsymbol{x}_k} \frac{\partial \boldsymbol{x}_k}{\partial \boldsymbol{\theta}} + \sum_{k=0}^{N-1} \frac{\partial L}{\partial \boldsymbol{u}_k} \frac{\partial \boldsymbol{u}_k}{\partial \boldsymbol{\theta}}$$

- Iteratively updating the sensitivities:

Sensitivity propagation:

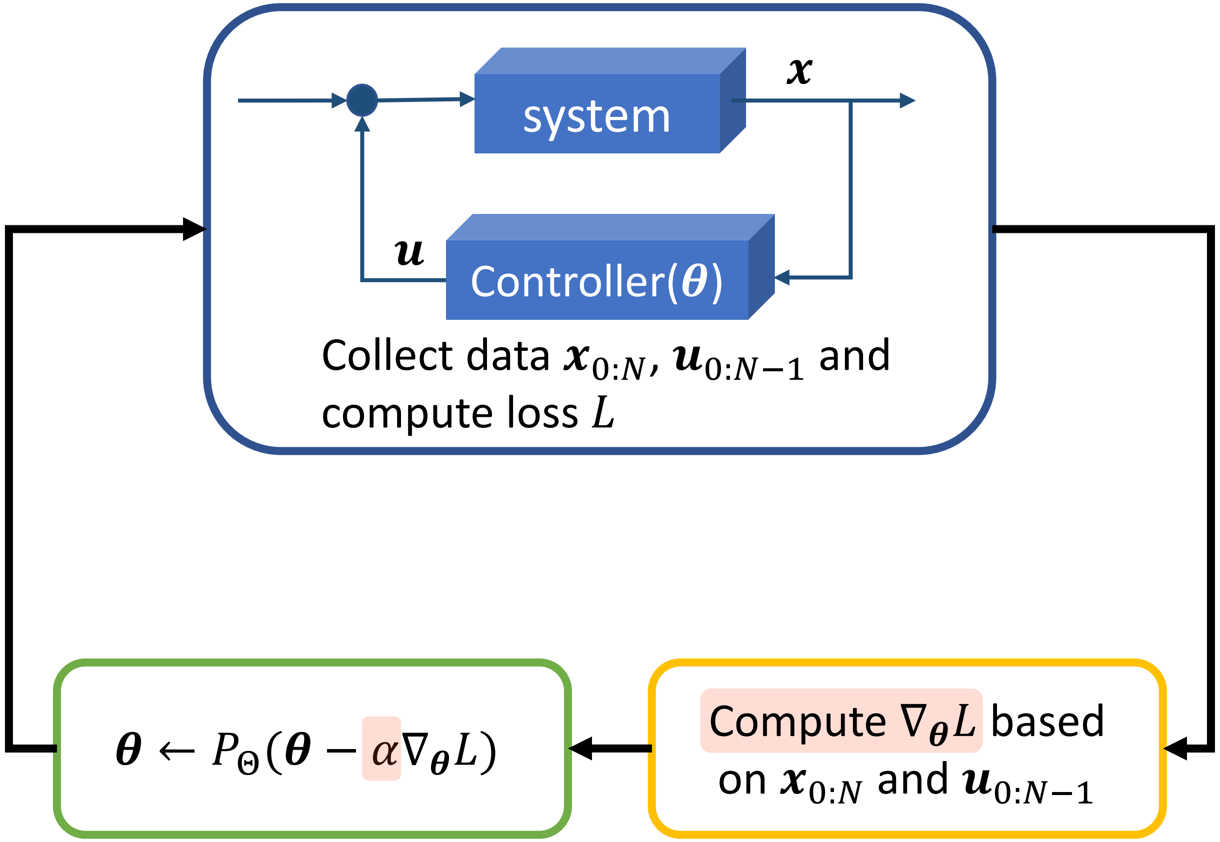
capable of running **in parallel** to the system's evolution

$$\frac{\partial \boldsymbol{x}_{k+1}}{\partial \boldsymbol{\theta}} = \left(\nabla_{\boldsymbol{x}_k} f + \nabla_{\boldsymbol{u}_k} f \nabla_{\boldsymbol{x}_k} h\right) \frac{\partial \boldsymbol{x}_k}{\partial \boldsymbol{\theta}} + \nabla_{\boldsymbol{u}_k} f \nabla_{\boldsymbol{\theta}} h$$

$$\frac{\partial \boldsymbol{u}_k}{\partial \boldsymbol{\theta}} = \nabla_{\boldsymbol{x}_k} h \frac{\partial \boldsymbol{x}_k}{\partial \boldsymbol{\theta}} + \nabla_{\boldsymbol{\theta}} h$$

S. Cheng, M. Kim, L. Song, C. Yang, Y. Jin, S. Wang, N. Hovakimyan, "DiffTune: Auto-Tuning through Auto-Differentiation," under review. https://arxiv.org/abs/2209.10021

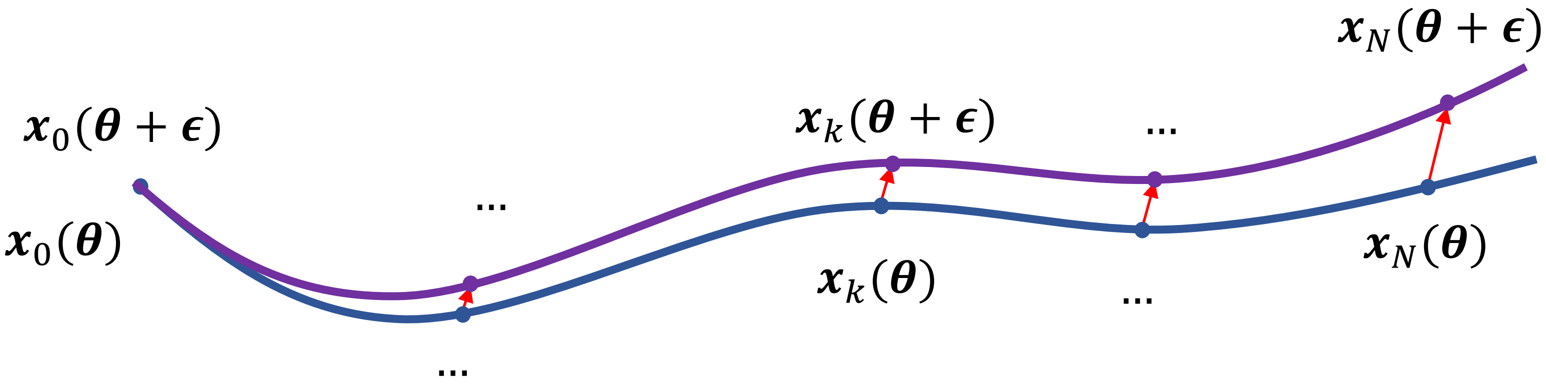
Auto-Tuning: You Still Need to Tune Hyperparameters



Can we eliminate hyperparameter tuning?

Using Sensitivity States for Hyperparameter-free Tuning

- Can we predict states subject to parameter change?



$$x_k(\theta + \epsilon) \approx x_k(\theta) + \boxed{\frac{\partial x_k(\theta)}{\partial \theta}} \epsilon$$

❯ $L(\theta + \epsilon)$

- First-order approximation of the states once the parameter θ changes by ϵ
- With the approximate state, we can approximate the loss $L(\theta + \epsilon)$.
- Pick ϵ^* to maximize the loss reduction $L(\theta) - L(\theta + \epsilon)$. [*Hyperparameter-free*]

S. Cheng, L. Song, M. Kim, S. Wang, and N. Hovakimyan. "DiffTune⁺: Hyperparameter-Free Auto-Tuning using Auto-Differentiation."*5th L4DC Conference [**Oral Presentation**]* (2023).

Hyperparameter-free Tuning: Two Methods

Optimizing over a scalar (Line search):

Find the optimal learning rate (scalar) α^* such that $L(\boldsymbol{\theta}) - L(\boldsymbol{\theta} + \alpha^* \nabla_{\boldsymbol{\theta}} L)$ is maximized.

Optimizing over a vector (Gauss-Newton):

Find the optimal parameter update (vector) $\boldsymbol{\epsilon}^*$ such that $L(\boldsymbol{\theta}) - L(\boldsymbol{\theta} + \boldsymbol{\epsilon}^*)$ is maximized.

If the loss function is quadratic, then

- explicit formula of α^* and $\boldsymbol{\epsilon}^*$ can be derived;
- $\boldsymbol{\epsilon}^*$ has the identical form to the **Gauss-Newton** method.

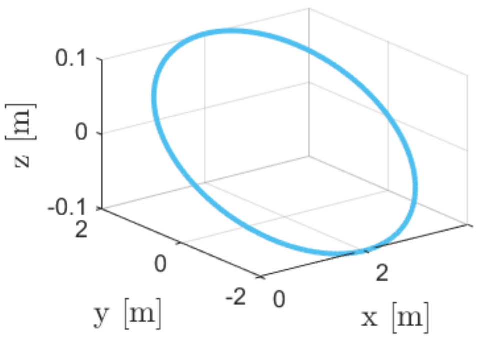
S. Cheng, L. Song, M. Kim, S. Wang, and N. Hovakimyan. "DiffTune⁺: Hyperparameter-Free Auto-Tuning using Auto-Differentiation." *5th L4DC Conference [**Oral Presentation**]* (2023).

Tuning: Hyperparameter-based vs Hyperparameter-free

	Hyperparameter-based	Hyperparameter-free
1st-order		
2nd-order		

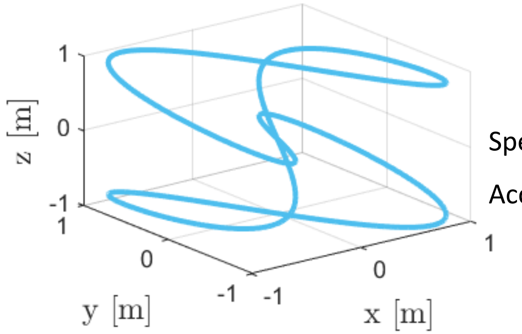
S. Cheng, L. Song, M. Kim, S. Wang, and N. Hovakimyan. "DiffTune$^+$: Hyperparameter-Free Auto-Tuning using Auto-Differentiation." *5th L4DC Conference [**Oral Presentation**]* (2023).

Simulation Results



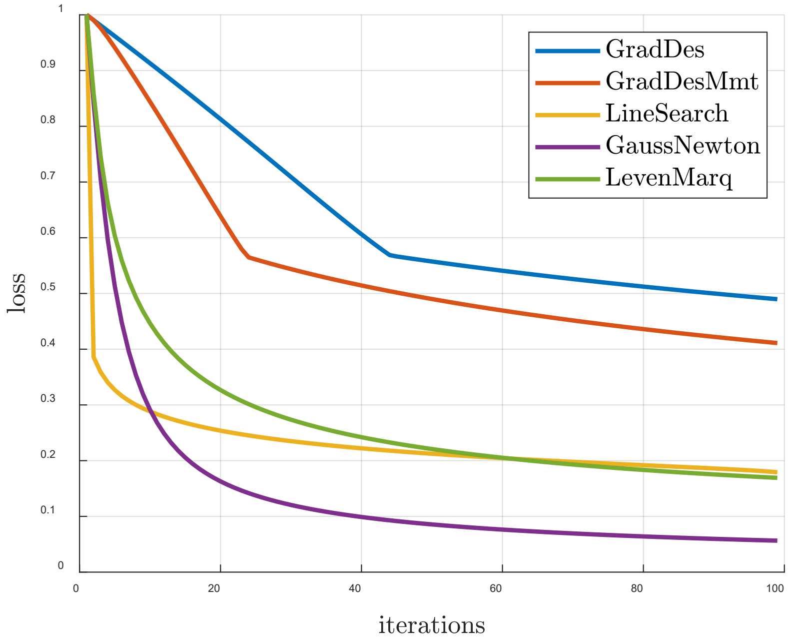
Speed: 2 [m/s]

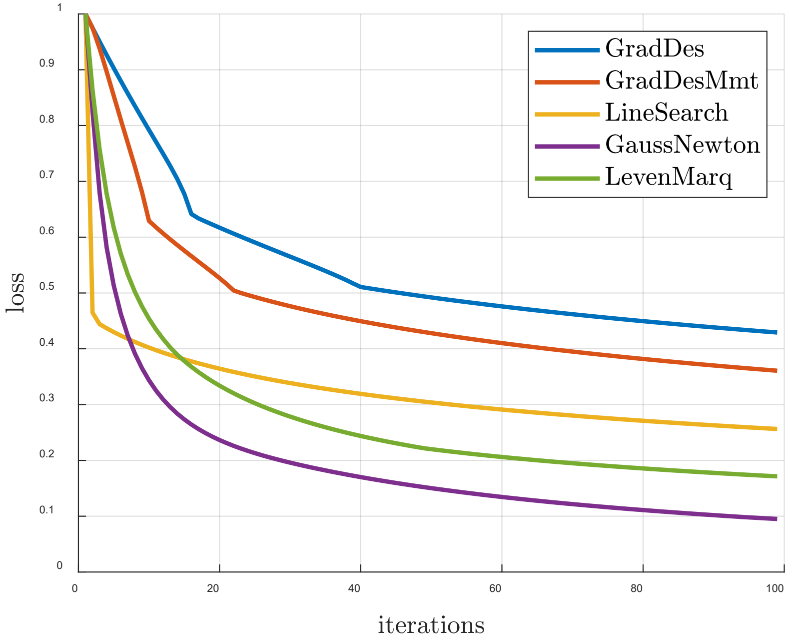
Accel.: 2 [m/s²]



Speed: 1.44—4.58 [m/s]

Accel.: 0.21—16.27 [m/s²]

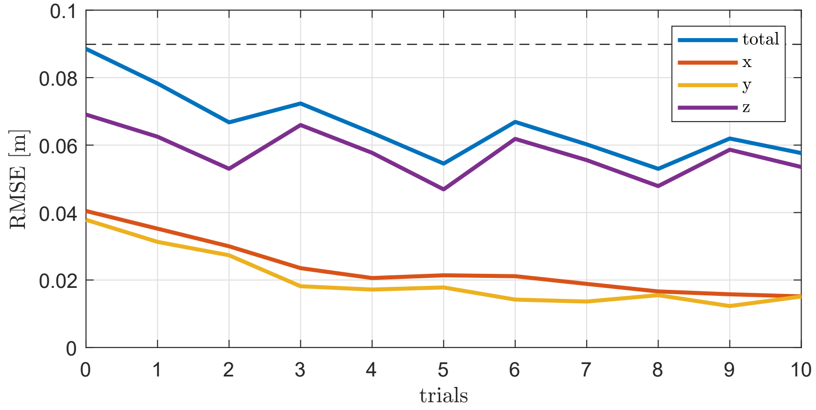




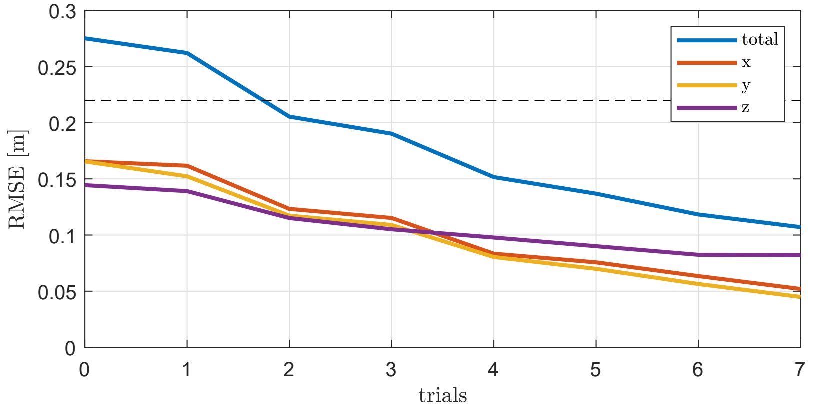
- Two trajectories with different characteristics
- HP-based:
  - Performance depends on the hyperparameter tuning.
- HP-free:
  - Gauss-Newton has the minimum loss albeit the parameters are tuned to be very aggressive;
  - Line-search produces acceptable gains with second-best loss reduction.

S. Cheng, L. Song, M. Kim, S. Wang, and N. Hovakimyan. "DiffTune⁺: Hyperparameter-Free Auto-Tuning using Auto-Differentiation." *5th L4DC Conference [**Oral Presentation**]* (2023).

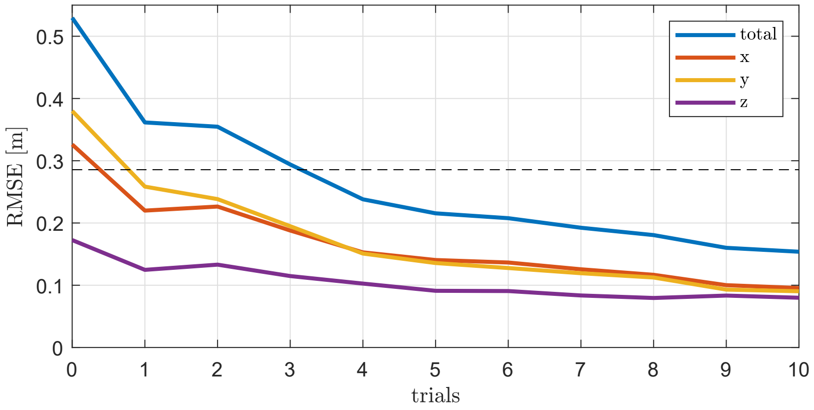
Experiments on a Real Quadrotor



1 m/s

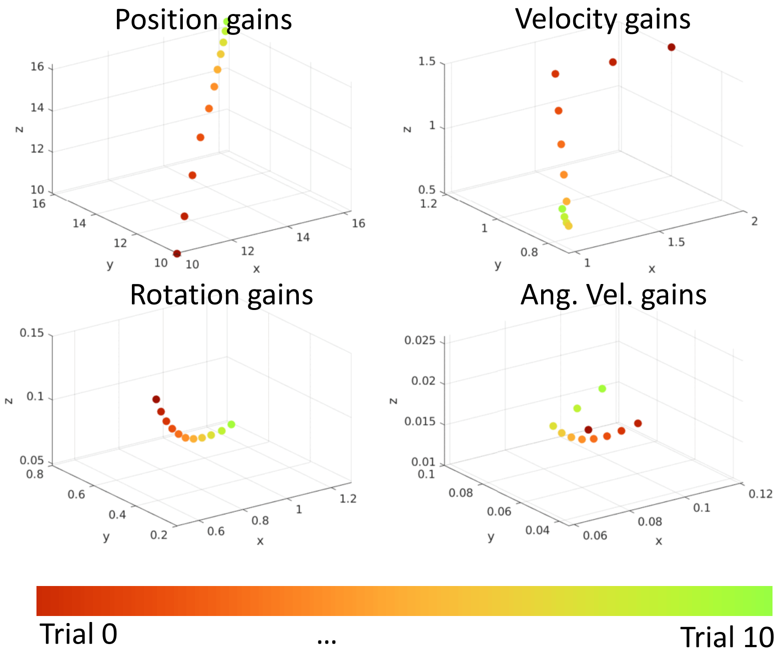


2 m/s



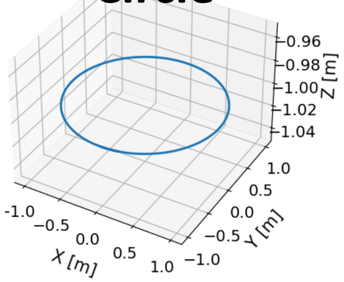
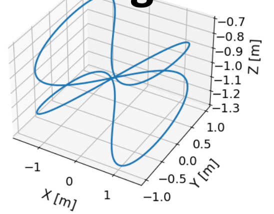
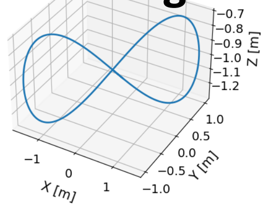
3 m/s

- 18-dimensional **nonlinear** dynamics, 12 controller parameters[1]
- Tracking circular trajectories with 1, 2, 3 m/s speed
- RMSE reduction in all three speeds and superseding manual tuning
  - **3.5x** reduction in 3 m/s case within **10 trials**
- Gradient helps tuning for handling both the **nonlinear mapping** and **high-dimensional parameter space**



1. T. Lee, M. Leok, and N. H. McClamroch, “Geometric tracking control of a quadrotor UAV on SE(3),” in IEEE Conference on Decision and Control (CDC), 2010, pp. 5420–5425.

Generalization of Tuned Parameters

Trajectory	Speed [m/s]	Acc. [m/s²]	Tuned parameters			Hand-tuned
			T1	T2	T3	
Circle 	1	1	**0.057**	**0.057**	0.055*	0.090
	2	4	0.272	**0.107**	0.166	0.180
	3	9	0.728	0.317	**0.154**	0.286
3D figure 8 	0.59-1.43	0-1.35	0.055	**0.049**	0.048*	0.053
	1.18–2.86	0-5.41	0.106	0.088	**0.083**	0.092
Vertical figure 8 	0.48-1.52	0-1.2	0.125	0.105	0.097*	**0.096**
	0.96-3.04	0-2.4	0.178	0.130	**0.098**	0.127

* minor oscillations (dissipated in finite time)

- The parameters work the best with the trajectories they are tuned on.
- When generalizing to variants of the figure 8 trajectory, high-speed parameters work better due to better adaptation to the agile angular maneuvers.

1. S. Cheng, M. Kim, L. Song, C. Yang, Y. Jin, S. Wang, N. Hovakimyan, "DiffTune: Auto-Tuning through Auto-Differentiation," under review. https://arxiv.org/abs/2209.10021

How to Handle Uncertainties?

- For real systems, the gradient can be contaminated by
  - Noise
  - Uncertainties
- Noise can be addressed by filtering or state estimation.
- How to handle the uncertainties?
  - **Compensate for the uncertainties** to reduce their impact.
  - Use $\mathcal{L}_1$ adaptive augmentation: forcing the closed-loop system to **behave like the nominal model** by uncertainty compensation.
  - Preserving the validity of the gradient.

Ideal system

$$\dot{\boldsymbol{x}}^\star = f(\boldsymbol{x}^\star) + B(\boldsymbol{x}^\star)\boldsymbol{u}$$

Physical system
(with uncertainties $\boldsymbol{\sigma}$)

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}) + B(\boldsymbol{x})(\boldsymbol{u} + \boldsymbol{\sigma})$$

Physical system with
$\mathcal{L}_1$ adaptive augmentation ($\boldsymbol{u}_{\text{ad}}$)

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}) + B(\boldsymbol{x})(\boldsymbol{u} + \underbrace{\boldsymbol{u}_{\text{ad}} + \boldsymbol{\sigma}}_{\|\boldsymbol{u}_{\text{ad}} + \boldsymbol{\sigma}\| \approx \boldsymbol{0}})$$

S. Cheng, M. Kim, L. Song, C. Yang, Y. Jin, S. Wang, N. Hovakimyan, “DiffTune: Auto-Tuning through Auto-Differentiation,” under review. https://arxiv.org/abs/2209.10021

Ablation Study on DiffTune and $\mathcal{L}_1$AC

Circle 1 m/s RMSE [m]		$\mathcal{L}_1$ off	$\mathcal{L}_1$ on
DiffTune	off	0.089	0.075
	on	0.057	**0.030**

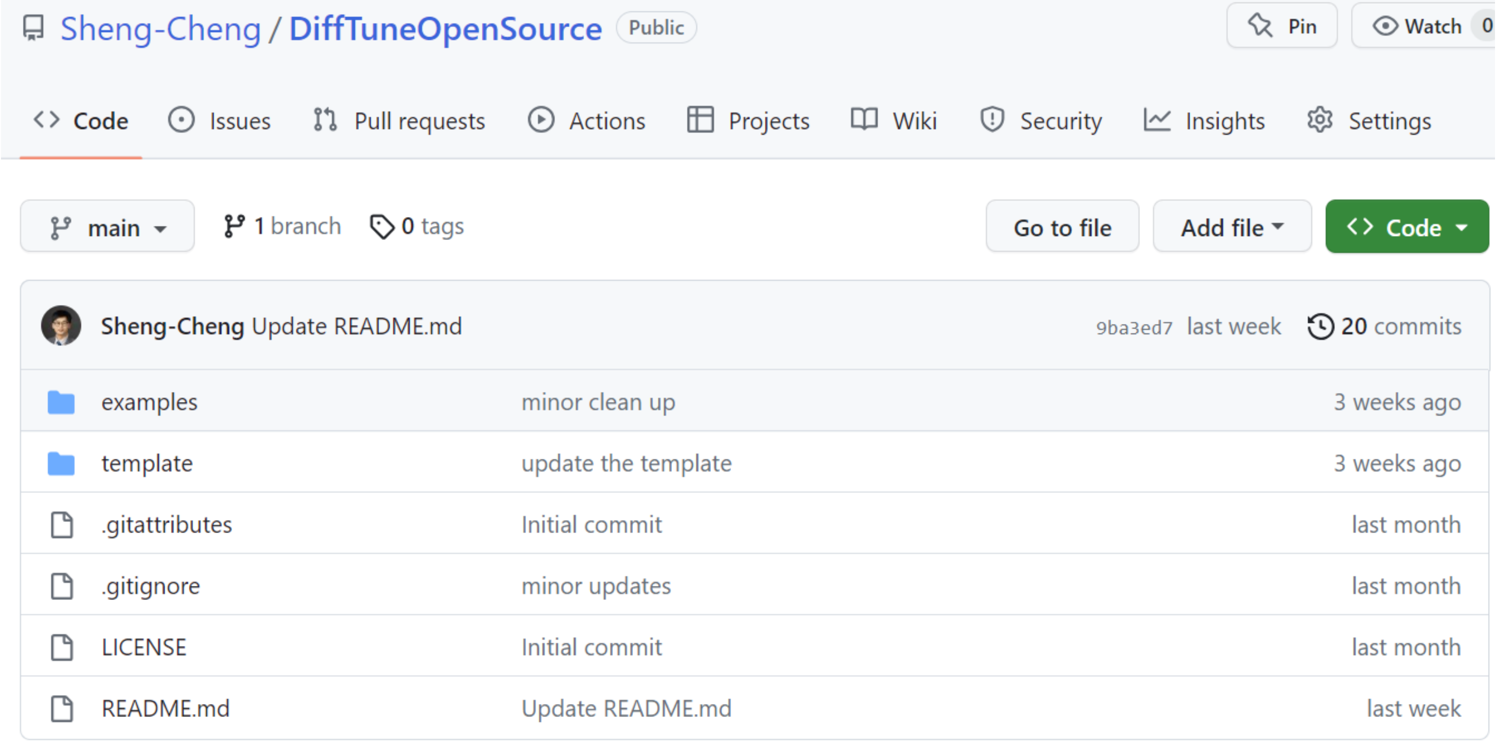
Circle 2 m/s RMSE [m]		$\mathcal{L}_1$ off	$\mathcal{L}_1$ on
DiffTune	off	0.275	0.251
	on	0.107	**0.069**

Circle 3 m/s RMSE [m]		$\mathcal{L}_1$ off	$\mathcal{L}_1$ on
DiffTune	off	0.618	0.466
	on	0.177	**0.162**

- Implementation of $\mathcal{L}_1$AC follows $\mathcal{L}_1$Quad[1]
- DiffTune and $\mathcal{L}_1$AC can individually improve performance:
  - DiffTune utilizes the first-order information to iteratively update the parameters;
  - $\mathcal{L}_1$AC compensates for the uncertainties.
- Best performance obtained when DiffTune and $\mathcal{L}_1$AC are used together.

1. Wu, Zhuohuan*, Sheng Cheng*, Pan Zhao, Aditya Gahlawat, Kasey A. Ackerman, Arun Lakshmanan, Chengyu Yang, Jiahao Yu, and Naira Hovakimyan. "$\mathcal{L}_1$ Quad: $\mathcal{L}_1$ Adaptive Augmentation of Geometric Control for Agile Quadrotors with Performance Guarantees," under review, arXiv preprint arXiv:2302.07208 (2023).

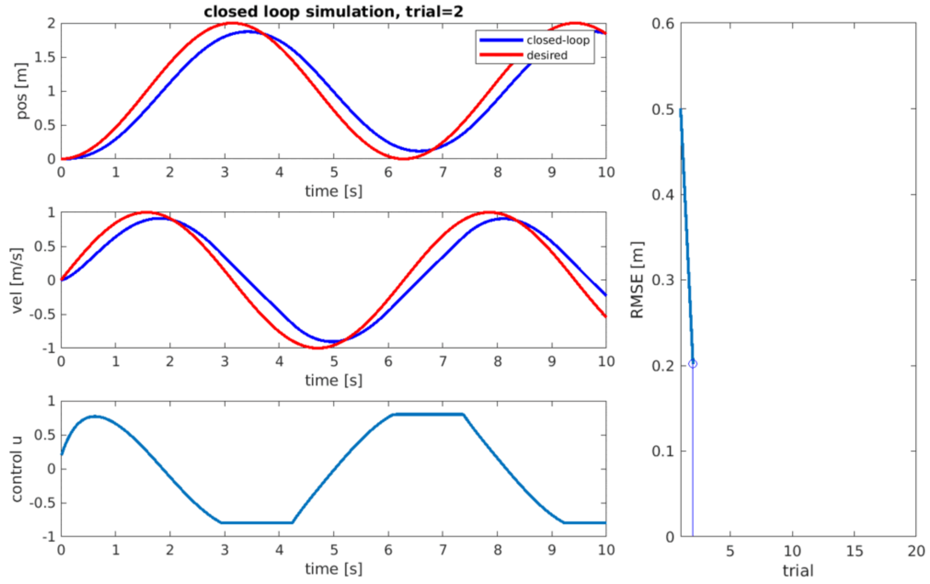
Open-Source Tool Set: DiffTuneOpenSource



- Enabling the **automatic generation** of the partial derivatives required for sensitivity propagation.
- Offering one **template** and two **examples** for users' custom tuning scenarios.
- Neural network controllers are supported!

Ongoing and Future Work

- DiffTune-MPC
- (Online) system identification and auto-tuning
- Nondifferentiable dynamics/controllers
- Tradeoff between performance and robustness



Many thanks to my advisor, collaborator, and students:

Naira Hovakimyan, Shenlong Wang, Minkyung Kim, Lin Song, Chengyu Yang, Yiquan Jin, John Bullock, Yuliang Gu.