Learning Algorithm Hyperparameters for Fast Parametric Convex Optimization

INFORMS 2026

Assistant Professor at Texas A&M

Rajiv Sambharya



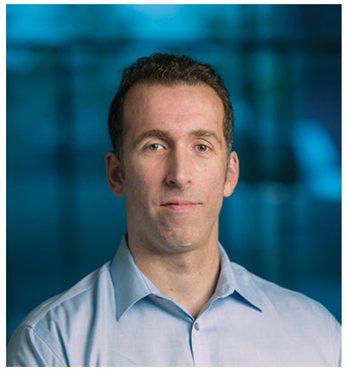
Bartolomeo Stellato

PRINCETON UNIVERSITY



Jinho Bok

Penn
University of Pennsylvania



Nikolai Matni

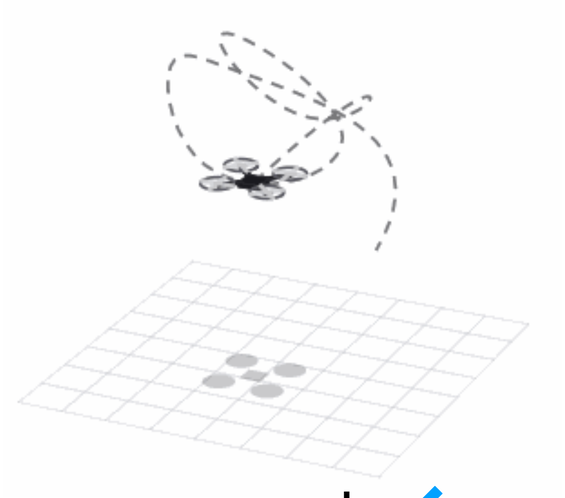
Penn
University of Pennsylvania



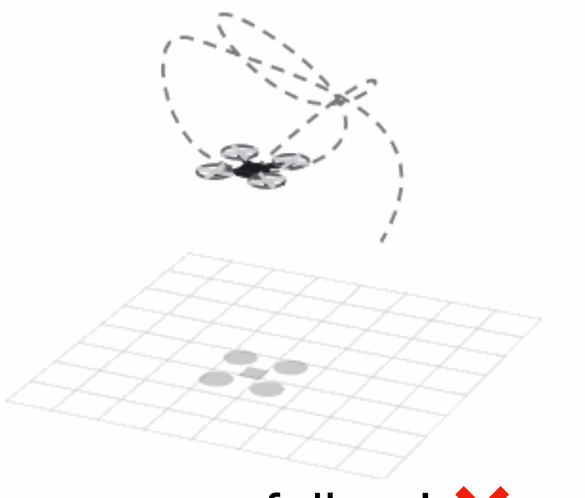
George Pappas

Penn
University of Pennsylvania

Quadcopter trajectory-tracking with optimization



success! ✓
(if given enough time)



failure! ✗
not enough time to solve

Model predictive control

optimize over a smaller horizon (T steps),
implement first action,
repeat

current state,
reference trajectory →

optimization

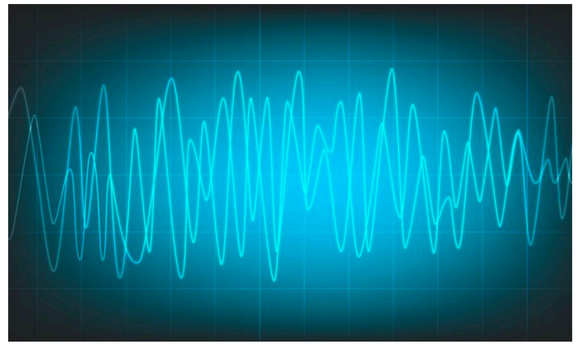
$$\begin{aligned} \text{minimize} \quad & \textstyle\sum_{t=1}^{T} \|s_t - s_t^{\text{ref}}\|_2^2 \\ \text{subject to} \quad & s_{t+1} = As_t + Bu_t \\ & s_t \in \mathcal{S}, \quad u_t \in \mathcal{U} \\ & s_0 = s_{\text{init}} \end{aligned}$$

→ actions

we need good algorithms to enable real-time optimization!

Real-world optimization is parametric

problem parameter x → $\begin{array}{ll}\text{minimize} & f(z,x) \\ \text{subject to} & z \in \Omega(x)\end{array}$ → optimal solution $z^\star(x)$

	robotics + control	energy	signal processing
			
problem parameter x	initial state	demand, renewable generation	noisy measurement
decision variable z	thrusts, torques	power flow	underlying signal

First-order methods can be too slow for real-time applications

$$\begin{array}{ll} \text{minimize} & f(z, x) \\ \text{subject to} & z \in \Omega(x) \end{array}$$

First-order methods are popular…

…but can be slow to converge!

Example: projected gradient descent

$$z^{k+1}(x) = \Pi_{\Omega(x)}\underbrace{(z^k - \theta^k \nabla f(z^k(x), x))}_{\text{gradient step}}$$

projection

Problem

In many applications, we have a **budget** of iterations

e.g., in robotics, we may only have 10ms to solve each problem



Can machine learning speed up convex parametric optimization?

Goal: Do mapping quickly and accurately

Parameter x → $\begin{aligned} \text{minimize} \quad & f(z, x) \\ \text{subject to} \quad & z \in \Omega(x) \end{aligned}$ → Optimal solution $z^\star(x)$

x → Only Optimization → $\hat{z}^{\text{Opt}}(x)$



x → Only Machine Learning → $\hat{z}^{\text{ML}}(x)$



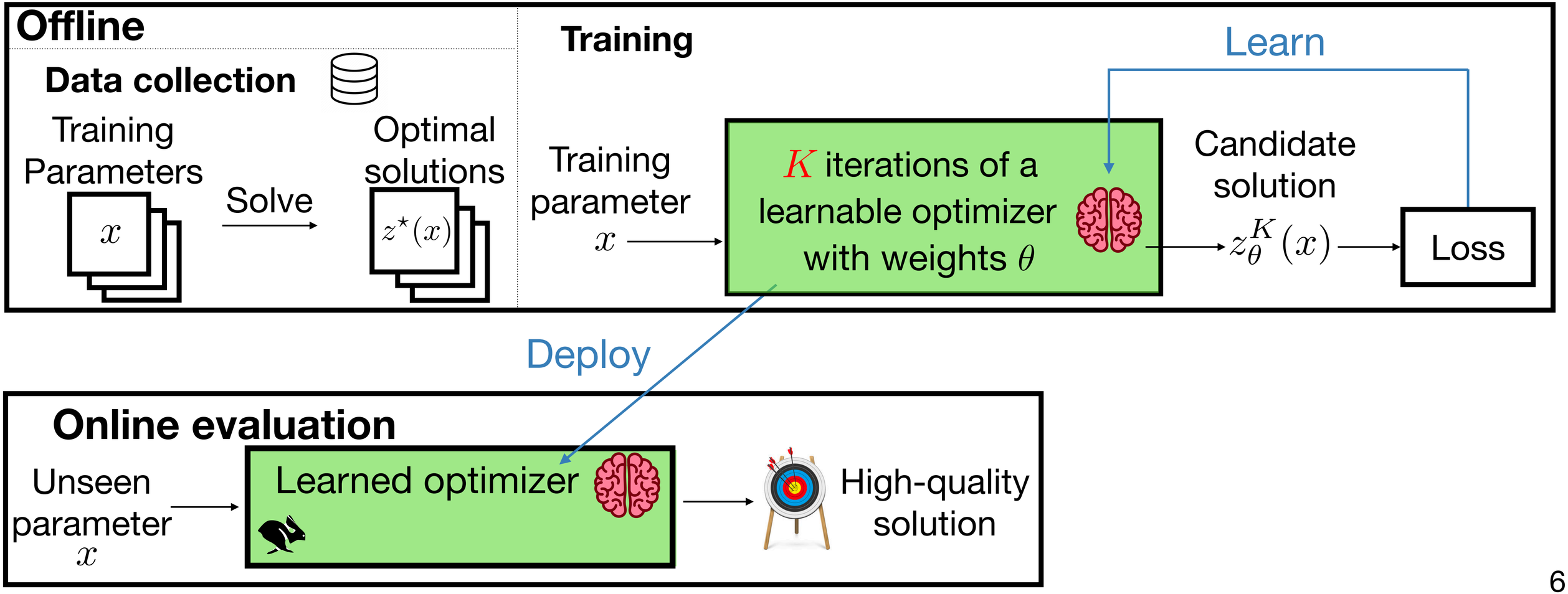
x → Optimization Machine Learning → $\hat{z}^{\text{Opt+ML}}(x)$



Learning to optimize

Goal: solve the problem within a **budget** of K iterations

$$\begin{aligned} &\text{minimize} && f(z, x) \\ &\text{subject to} && z \in \Omega(x) \end{aligned}$$



Can we learn powerful optimizers with limited training data?

data expensive to get (e.g., robotics) • problems often high-dimensional (10000+ variables)

existing methods: learning to optimize

learning warm starts

learn a high-quality initial point from data

Baker 2019; Finn et al. 2017; Chen et al. 2018; Chen et al. 2022; Sharony et al. 2024; **Sambharya et al. 2024**

learning algorithm steps

learn the update function from data

Gregor and LeCun 2010; Ichnowski 2021; Venkataraman and Amos 2021; Banert et al. 2024; Saravanos et al. 2024; King et al. 2024

limitation: these methods are data-hungry since they require learning many weights



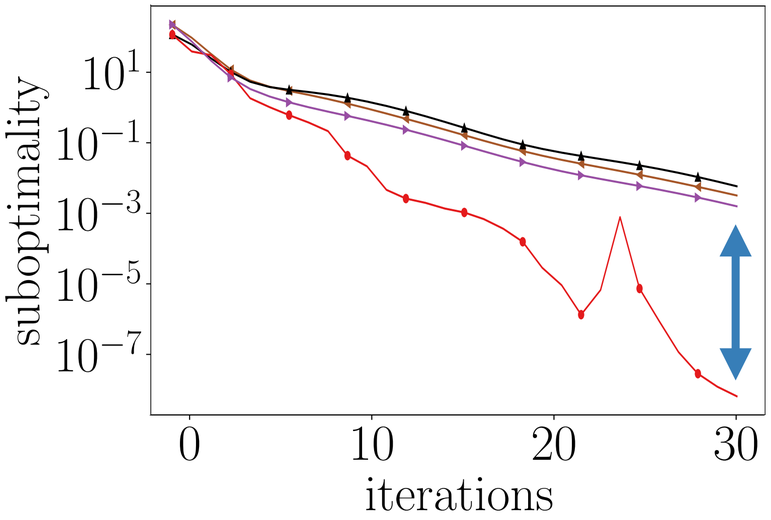
approach: **learn** the step sizes!

$\theta = \theta_0, \ldots, \theta_{K-1}$

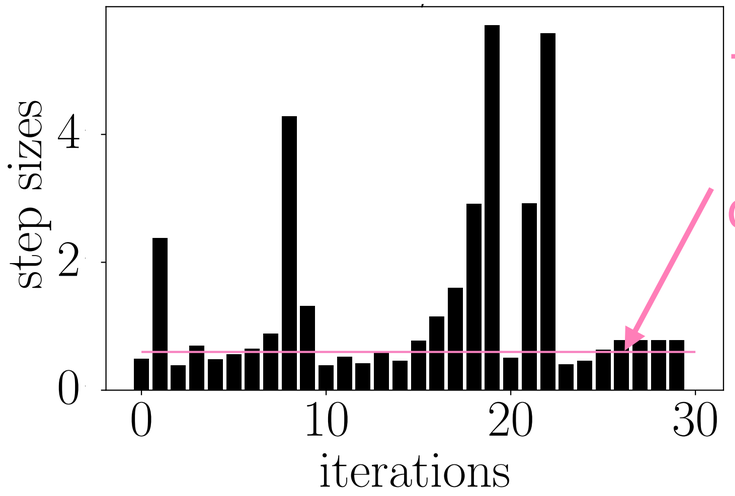
very low number of weights

Learned step sizes converge faster by many orders of magnitude

noisy measurement $x \in \mathbf{R}^{1000}$ → **signal reconstruction** minimize $(1/2)\|Az - x\|_2^2 + \lambda\|z\|_1$ → reconstructed signal $z^\star(x) \in \mathbf{R}^{2000}$



5 orders of magnitude



Threshold to guarantee convergence

We learn spiky step sizes!

Nesterov

Nearest neighbor warm start

Learned warm starts

Sambharya et. al 2024

10000 training instances

Learned step sizes

10 training instances

Learning step sizes can be powerful

This is a highly data-efficient approach

Model predictive control of a quadcopter

$x \in \mathbf{R}^{44}$

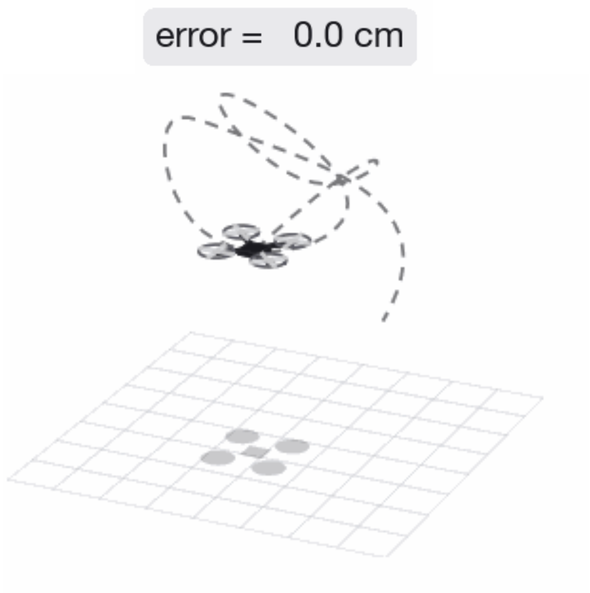
current state,
reference trajectory

quadratic program

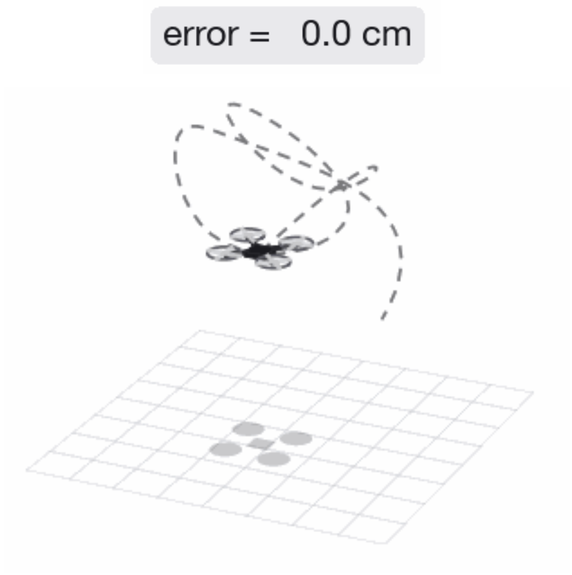
$$\begin{array}{ll} \text{minimize} & \sum_{t=1}^{T} \|s_t - s_t^{\text{ref}}\|_2^2 \\ \text{subject to} & s_{t+1} = As_t + Bu_t \\ & s_t \in \mathcal{S}, \quad u_t \in \mathcal{U} \\ & s_0 = s_{\text{init}} \end{array}$$

$z^\star(x) \in \mathbf{R}^{1750}$

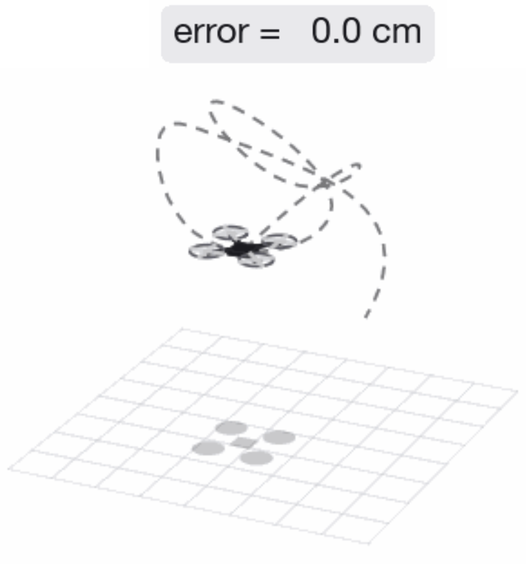
actions



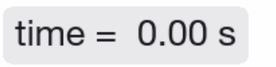
nearest neighbor
80 iterations



previous solution
80 iterations



learned
20 iterations



with learning, we can track the trajectory well

We adapt our method to obtain worst-case guarantees

gap: learned optimizers lack worst-case guarantees



we adapt our training loss to obtain them



Train for a strong form of robustness: worst-case guarantees for all parameters in a set $\mathcal{X}$

$$r(z_\theta^K(x), x) \leq \gamma(\theta)\|z^0(x) - z^\star(x)\|^2 \quad \forall x \in \mathcal{X}$$

Level of robustness

Performance metric: *e.g.*,

$$r(z_\theta^K(x), x) = \|z_\theta^K(x) - z^\star(x)\|_2^2$$

Regularized training to control $\gamma(\theta)$

- Performance estimation problem (PEP)
- Differentiable optimization

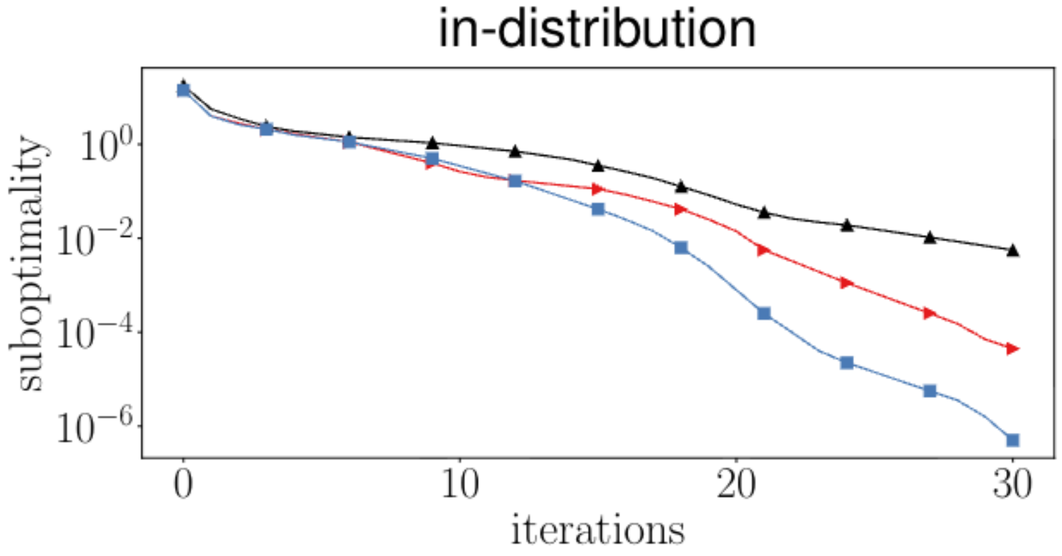
PEP
Drori, Teboulle, Hendrickx, Glineur, Taylor, Ryu, Grimmer, and many more

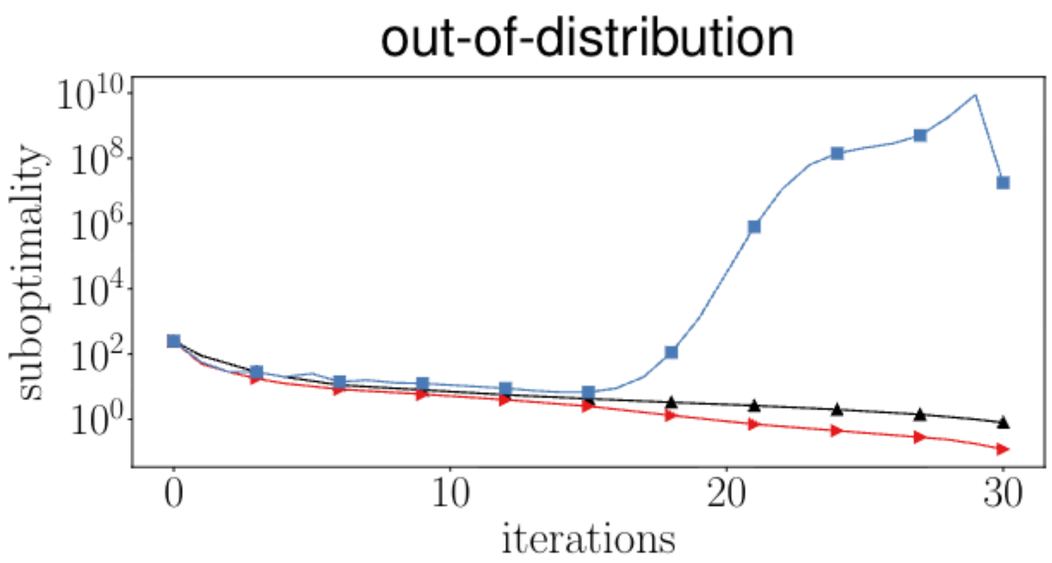
Differentiable optimization
Amos, Agrawal, Kolter, Boyd

Learning robust hyperparameters for sparse coding

noisy measurement $x \in \mathbf{R}^{1000}$ ⟶ **signal reconstruction** ⟶ reconstructed signal $z^\star(x) \in \mathbf{R}^{2000}$

$$\text{minimize} \quad (1/2)\|Az - x\|_2^2 + \lambda\|z\|_1$$





	Nesterov	Learned step sizes - robust	Learned step sizes
Robustness Level	$\gamma = 0.01$ (most robust)	$\gamma = 0.10$	$\gamma = \infty$ (least robust)

We can train while maintaining robustness

Conclusion



Traditional view

- One-size fits all

Learning to optimize



New view

- Task-specific + trainable

Takeaways from this talk specifically

- Only **learning** the **hyperparameter sequence** dramatically improves performance
- Very **low amount of training data** needed
- We evaluate and train for **robustness** using PEP



Learning Algorithm Hyperparameters for Fast Parametric Convex Optimization

R. Sambharya, B. Stellato

SIAM Journal on Mathematics of Data Science, 2026



INFORMS Computing Society Prize 2026: (honorable mention)



Learning Acceleration Algorithms for Fast Parametric Convex Optimization with Certified Robustness

R. Sambharya, J. Bok, N. Matni, G. Pappas

Arxiv, 2025



rajivsambharya.github.io