Learning to Warm-Start Fixed-Point Optimization Algorithms

Rajiv Sambharya
INFORMS 2023

PRINCETON
UNIVERSITY

ORFE

Collaborators

Georgina
Hall

INSEAD

Brandon
Amos

Meta

Bartolomeo
Stellato

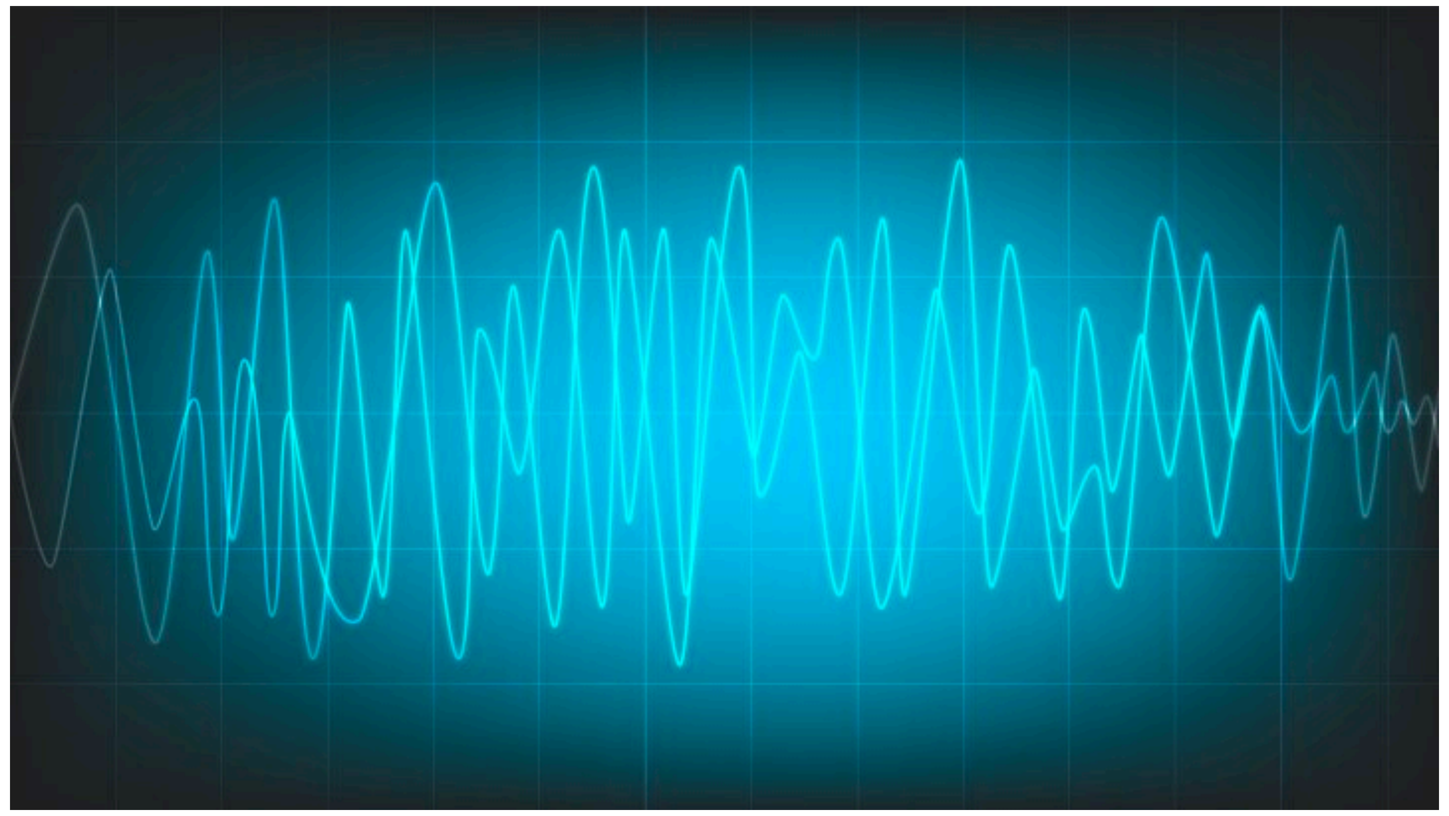
PRINCETON
UNIVERSITY

Fixed-point problems need solutions in real-time

Fixed-point problem: find z such that $z = T(z)$

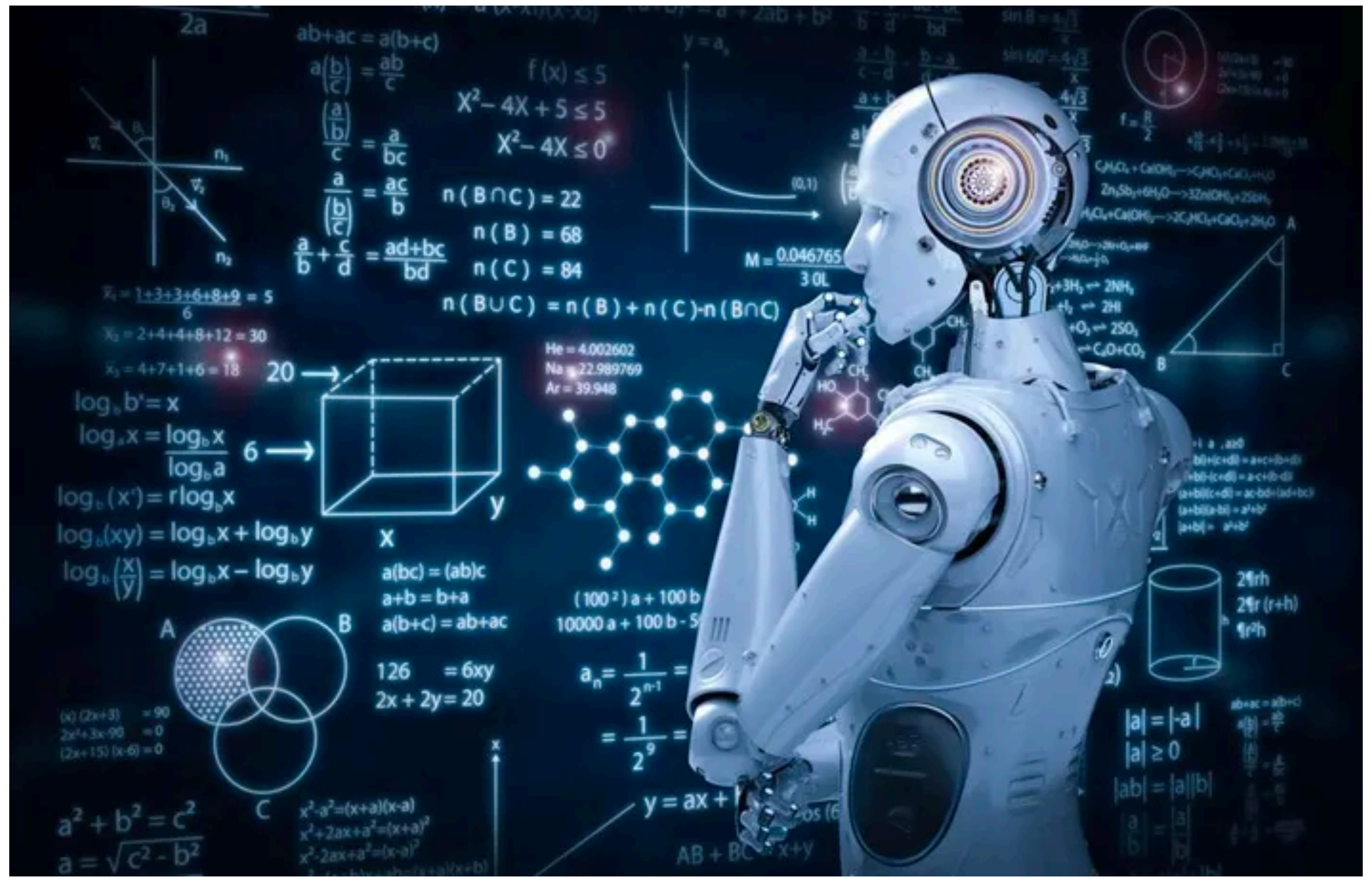
Robotics and control

Signal processing



Energy

Machine learning



Can machine learning speed up parametric optimization?

Often, we solve **parametric** fixed-point problems from the same family

Goal: Do mapping efficiently

Parameter

Optimal solution

$\theta \longrightarrow$ find z such that $z = T_\theta(z)$ $\longrightarrow z^\star(\theta)$

$\theta \longrightarrow$ Only Optimization $\longrightarrow \hat{z}(\theta)$

Accurate
Slow to compute

$\theta \longrightarrow$ Only Machine Learning $\longrightarrow \hat{z}(\theta)$

Inaccurate
Fast to compute

$\theta \longrightarrow$ Optimization + Machine Learning $\longrightarrow \hat{z}(\theta)$

Goals: Accurate
Fast to compute

Many optimization algorithms are fixed-point iterations

Fixed-point iterations: $z^{i+1} = T_\theta(z^i)$

Initialize with z^0 (a warm-start)

Terminate when $f_\theta(z^i) = \|T_\theta(z^i) - z^i\|_2$ is small **Fixed-point residual**

Example: Proximal gradient descent

minimize $g_\theta(z) + h_\theta(z)$

Convex
Smooth

Convex
Non-smooth

Iterates $z^{i+1} = \textbf{prox}_{\alpha h_\theta}(z^i - \alpha \nabla g_\theta(z^i))$

$$\textbf{prox}_s(v) = \arg\min_x \left(s(x) + \frac{1}{2}\|x - v\|_2^2 \right)$$

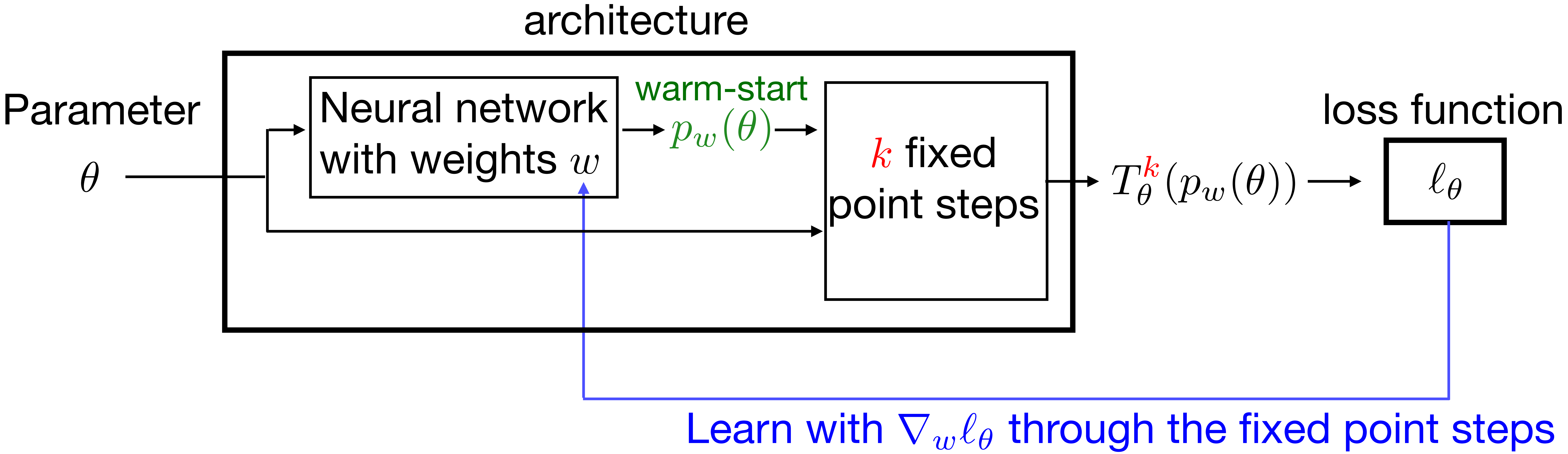
Operator $T_\theta(z) = \textbf{prox}_{\alpha h_\theta}(z - \alpha \nabla g_\theta(z))$

Problem: limited iteration budget

Solution: learn the warm-start to improve the solution within budget

Learning Framework

End-to-end learning architecture



architecture

Parameter

θ

Neural network with weights w

warm-start $p_w(\theta)$

k fixed point steps

$T_\theta^k(p_w(\theta))$

loss function

ℓ_θ

Learn with $\nabla_w \ell_\theta$ through the fixed point steps

Loss function: $\ell_\theta(z) = \|z - z^\star(\theta)\|_2$ Ground truth solution

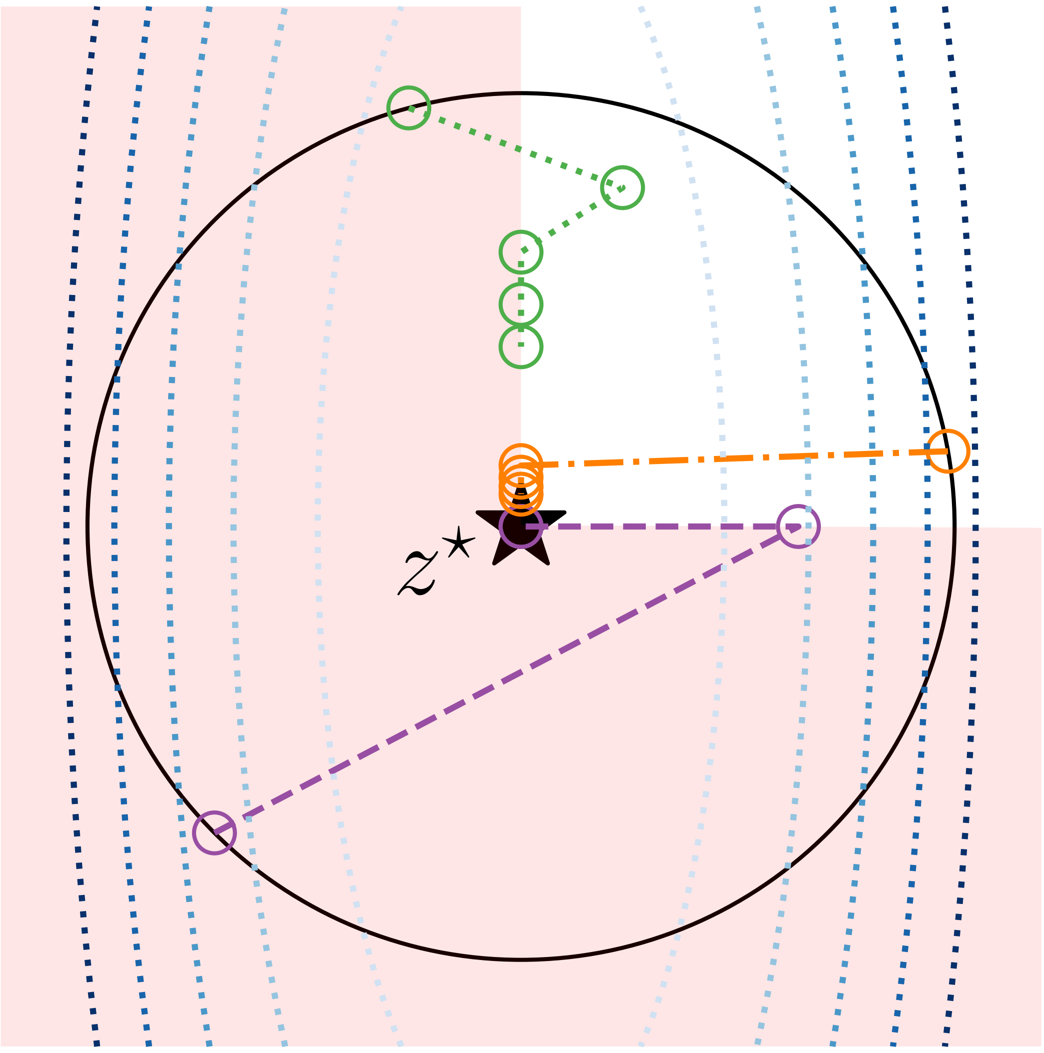
End-to-end learning scheme

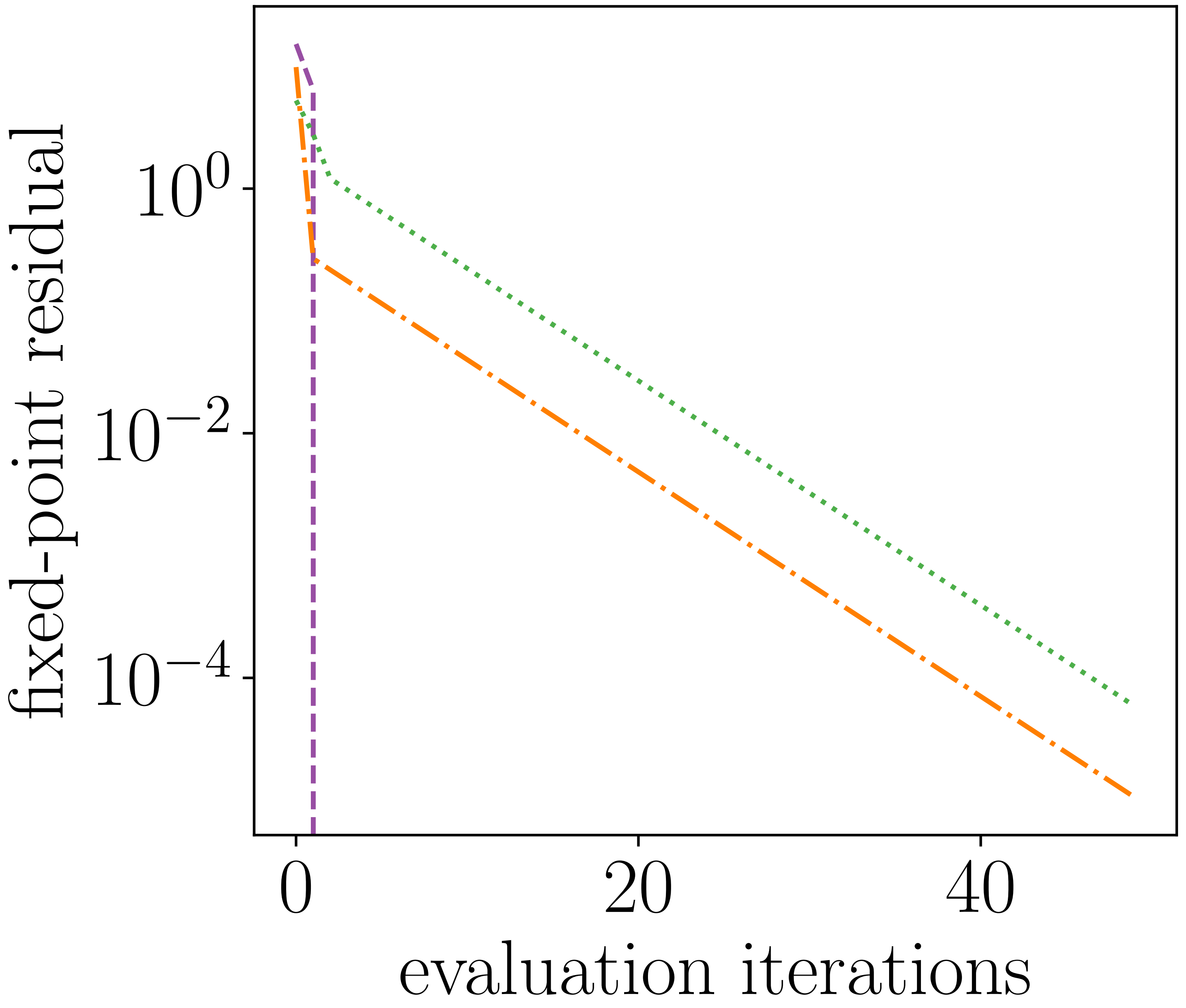
Some warm-starts are better than others

$$\begin{array}{ll} \text{minimize} & 10z_1^2 + z_2^2 \\ \text{subject to} & z \geq 0 \end{array}$$

Optimal solution at the origin

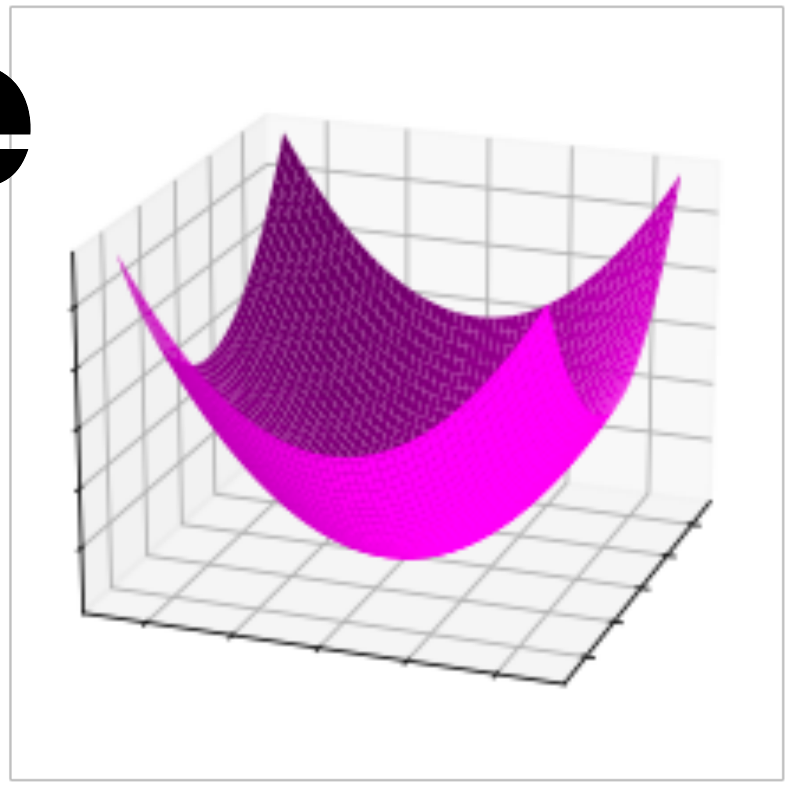
Run proximal gradient descent to solve





All three warm starts appear to be equally suboptimal but converge at very different rates

Theoretical advantages of architecture



Major benefit of learned warm-starts: fixed-point iterations always converge

Flexibility: # of evaluation steps can differ from # of train steps

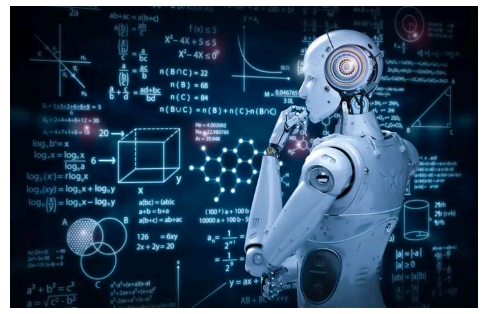
Number of fixed-point steps

Train with k steps

Loss: $\ell_\theta(z) = \|z - z^\star(\theta)\|_2$

Budget is t steps

Goal: min $f_\theta(z) = \|T_\theta(z) - z\|_2$



Guarantees from k training steps to t evaluation steps

β**-contractive case** $f_\theta(T_\theta^t(z)) \leq 2\beta^{t-k}\ell_\theta(T_\theta^k(z))$

Generalization bounds to unseen data

β-contractive case

Theorem 1. *With high probability over a training set of size N, for any γ,*

$$\underbrace{\mathbf{E} f_\theta(T^t_\theta(p_w(\theta)))}_{\text{Risk}} \le \underbrace{\frac{1}{N}\sum_{i=1}^{N} f_{\theta_i}(T^t_{\theta_i}(p_w(\theta_i)))}_{\text{Empirical risk}} + \underbrace{2\beta^t\gamma + \mathcal{O}\left(c_1(t)\sqrt{\frac{c_2(w)+\log(\frac{LN}{\delta})}{\gamma^2 N}}\right)}_{\text{Penalty term}}$$

$c_1(t)$: worst-case fixed-point residual after t steps

As $N \to \infty$, the **penalty term** decreases

As $t \to \infty$, the **penalty term** goes to zero

Derived from the PAC-Bayes framework

Non-contractive case: we provide similar bounds

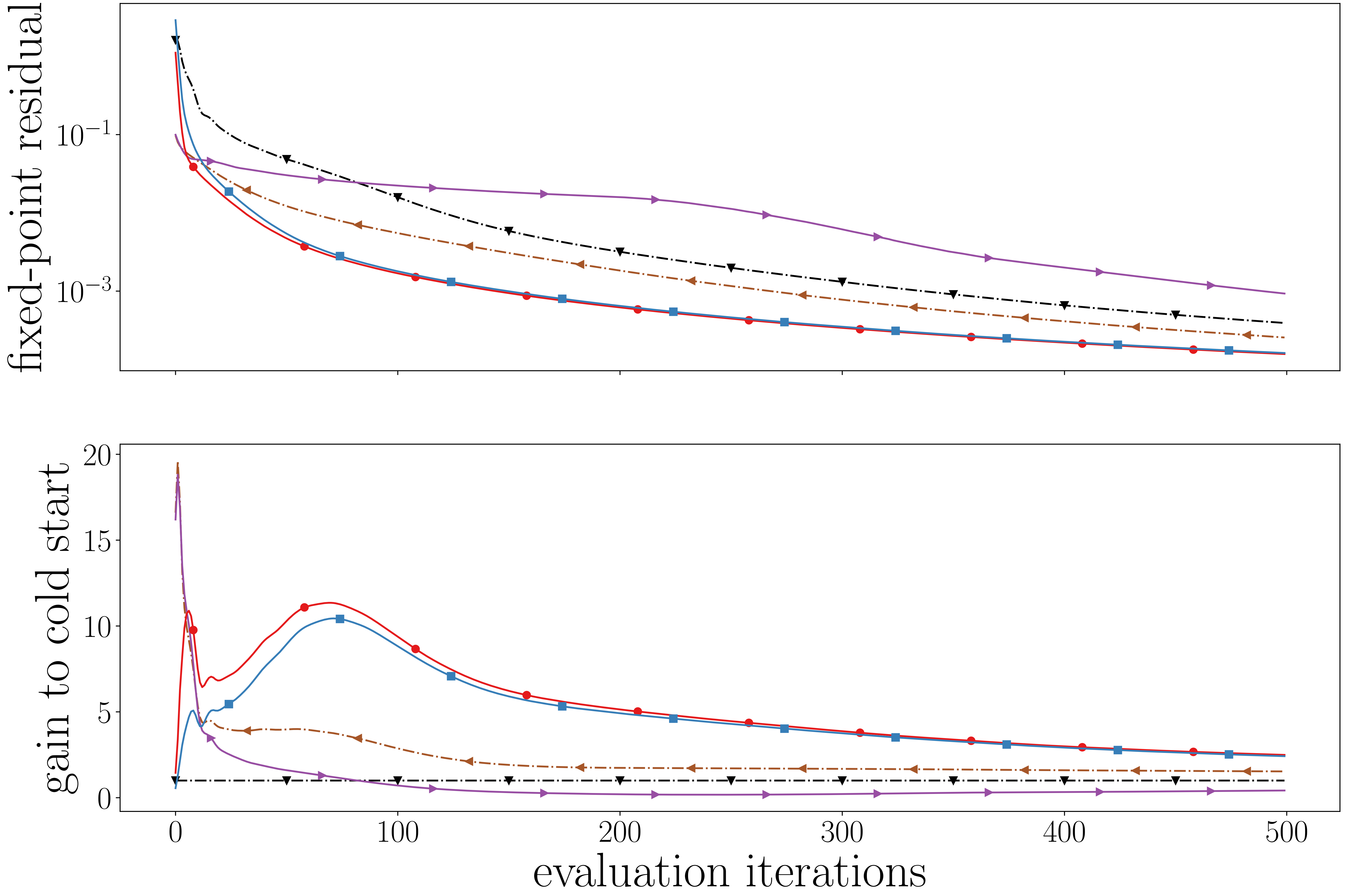
Numerical Experiments

Sparse PCA

Semidefinite relaxation

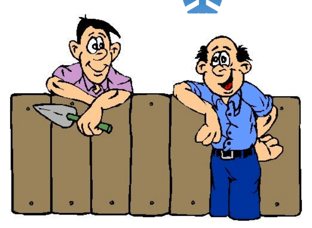
$$\begin{array}{ll} \text{maximize} & \mathbf{Tr}(AX) \\ \text{subject to} & \mathbf{Tr}(X) = 1 \\ & \mathbf{1}^T|X|\mathbf{1} \le c \\ & X \succeq 0 \end{array}$$

$$\theta = \text{vec}(A)$$



Different initializations

Baselines



▼ Cold-start

◀ Nearest neighbor

Learned

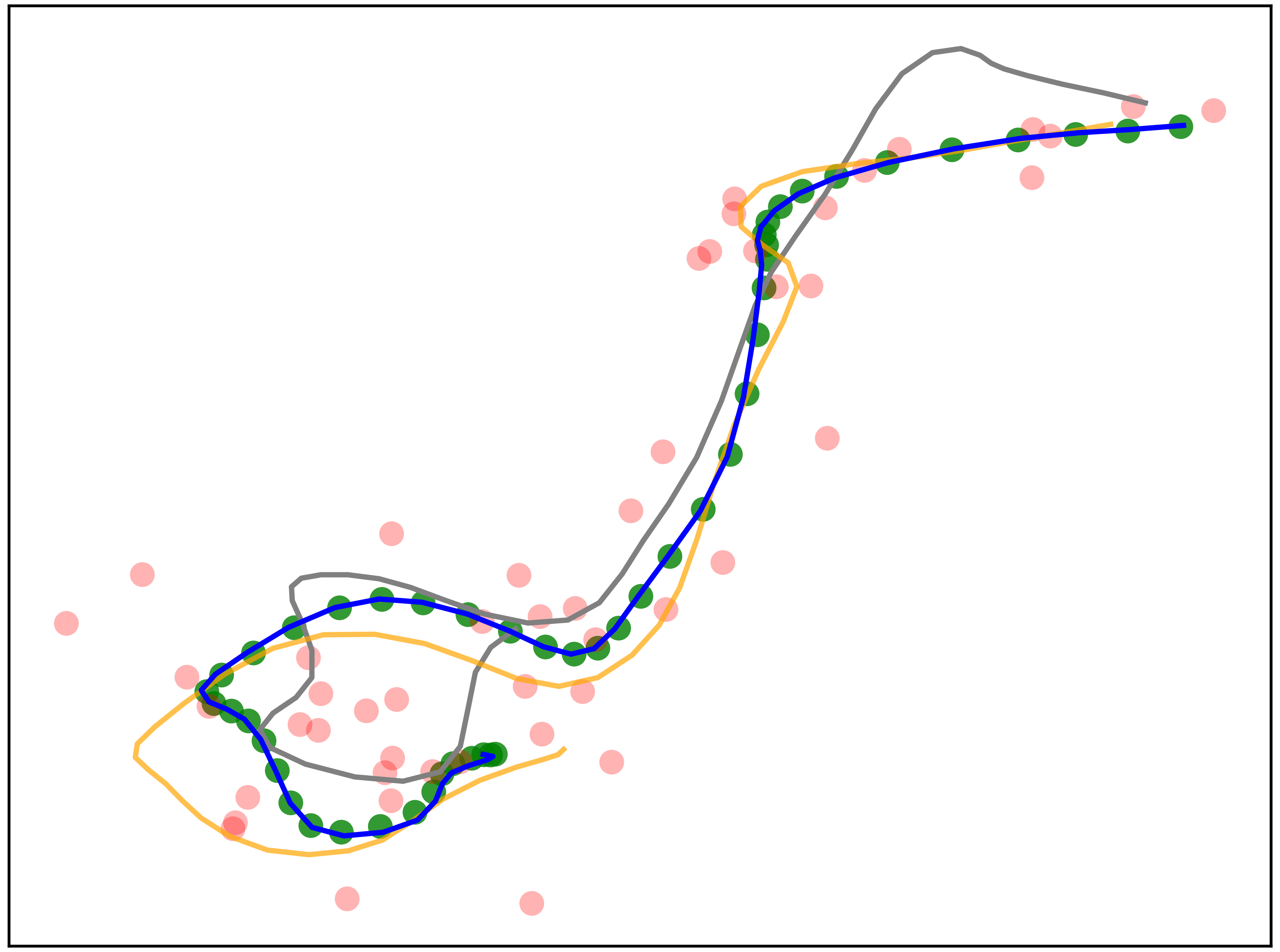
▶ $k = 0$

● $k = 5$

■ $k = 15$

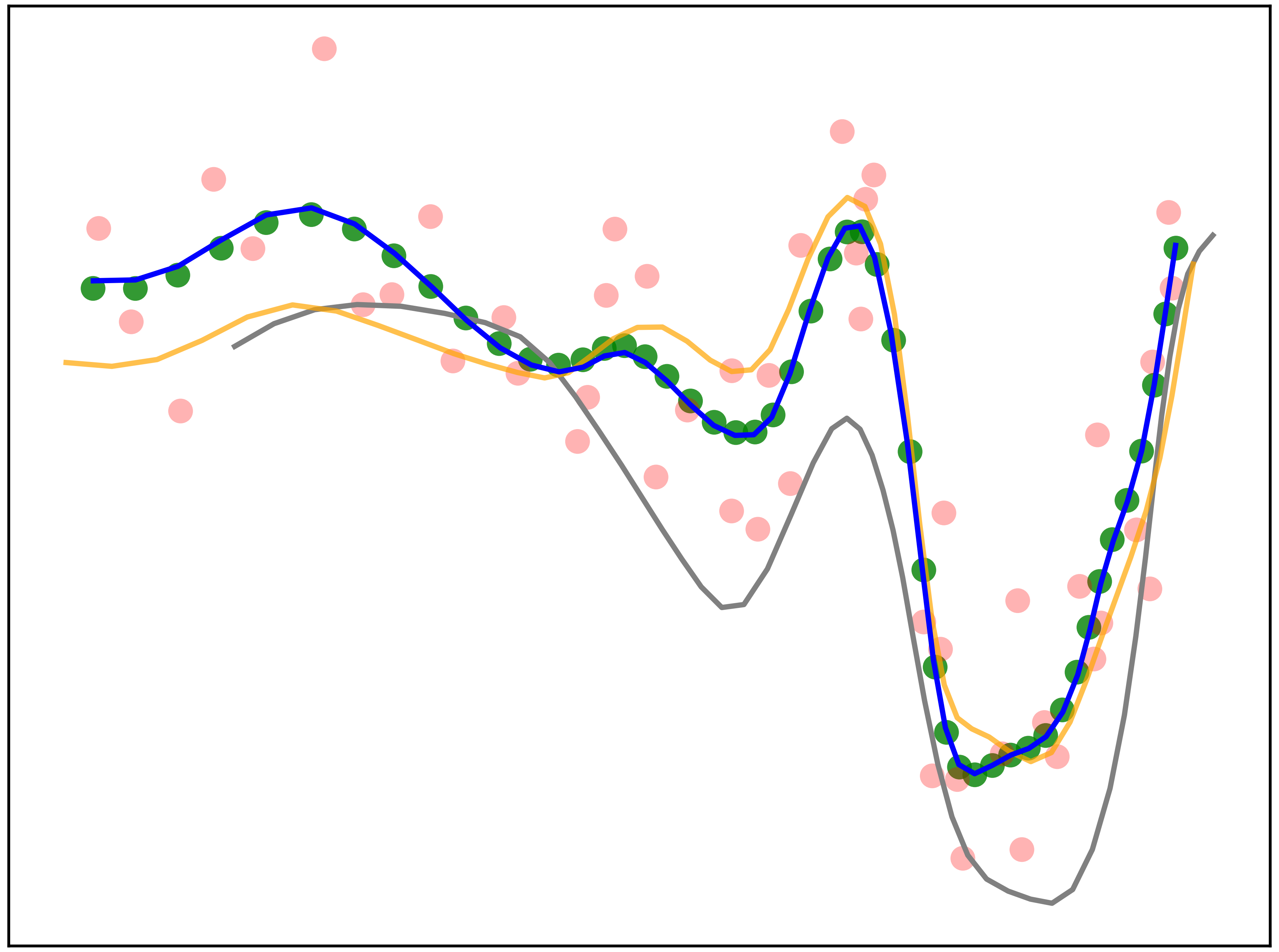
Picking k > 0 is essential to improve convergence

Robust Kalman filtering



🟥 Noisy trajectory

🟩 Optimal solution



Solution after 5 fixed-point steps
with different initializations

🟨 Nearest neighbor

⬛ Previous solution

🟦 Learned: $k = 5$

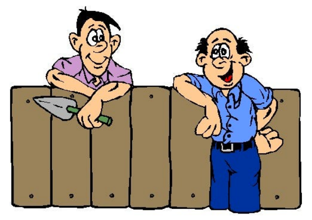
Can be formulated as an SOCP

With learning, we can estimate the state well

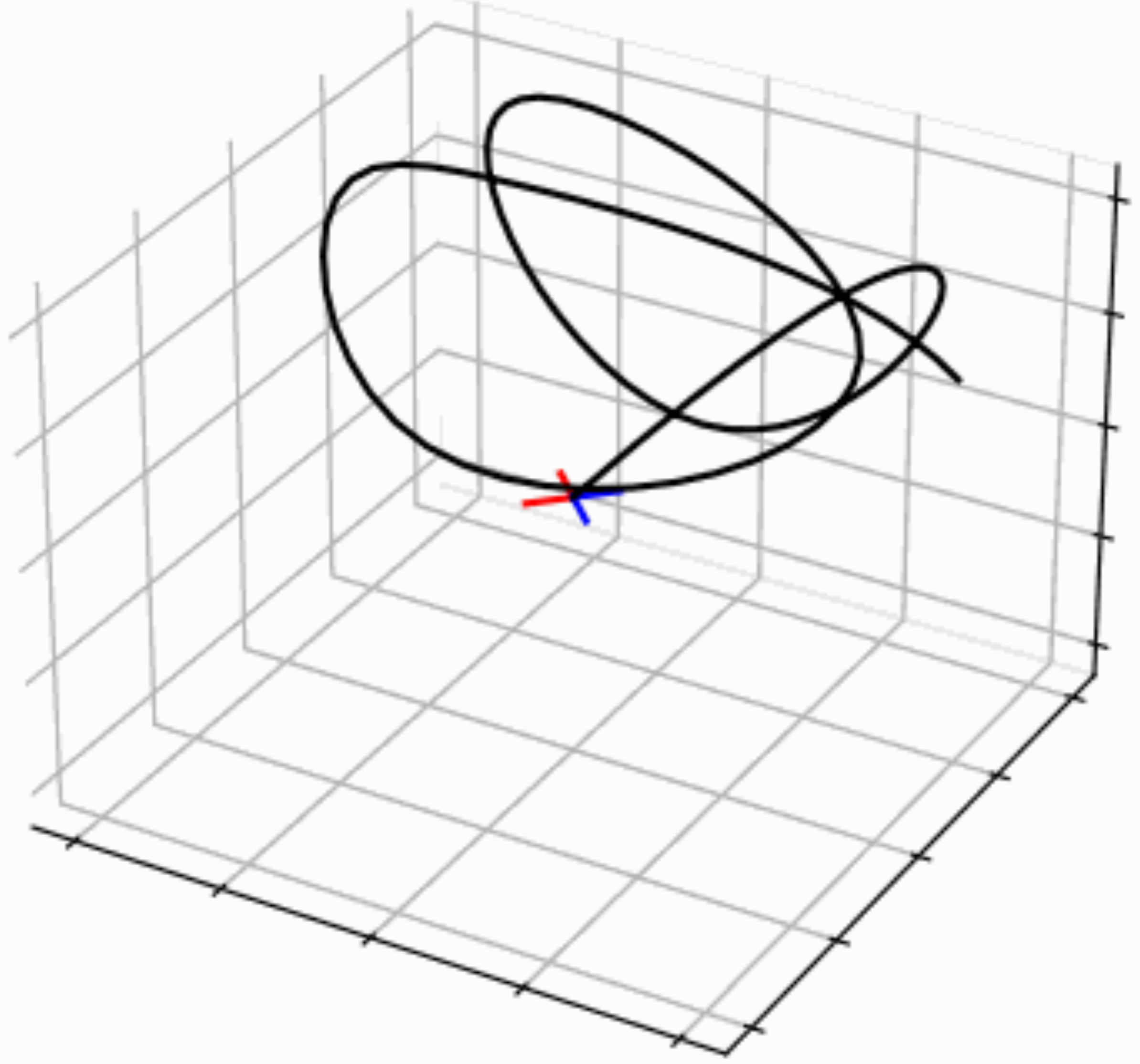
Model Predictive Control of a quadcopter in closed loop

Problem parameters: Initial state, linearized dynamics, reference trajectory

Budget of 15 fixed-point steps to solve each QP

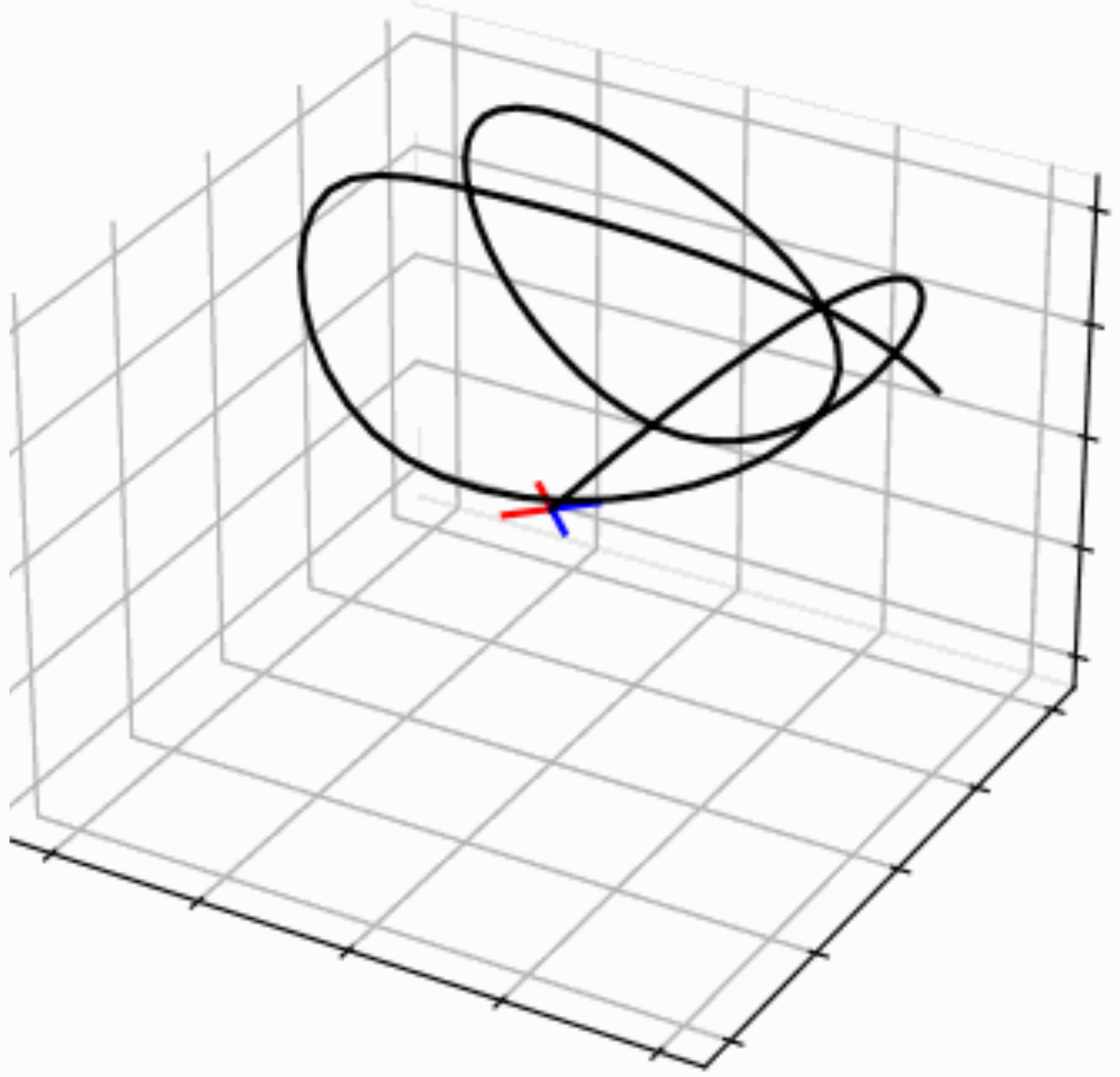


Nearest neighbor



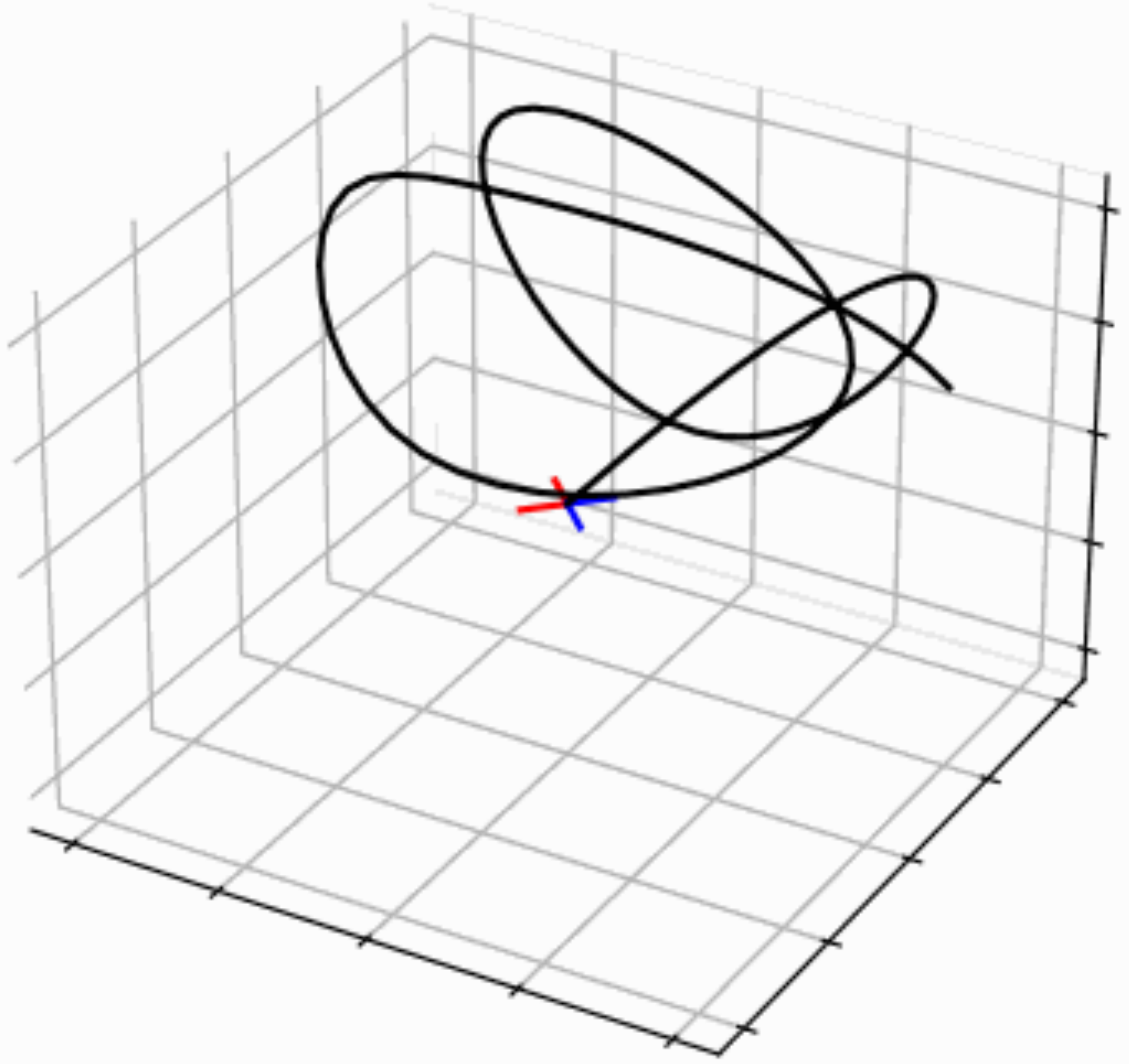
🔥

Previous solution



🚀

Learned: $k = 5$



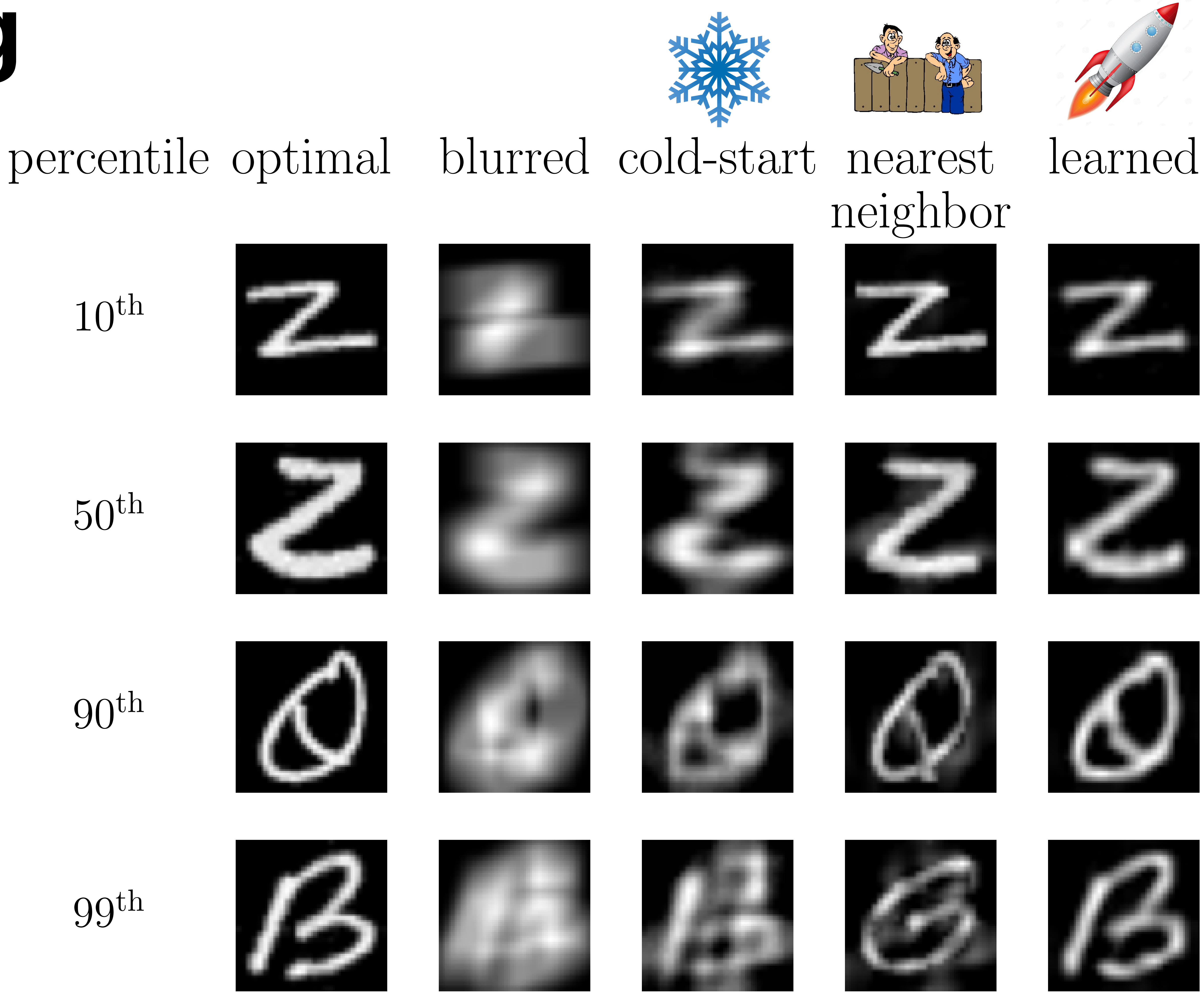
With learning, we can track the trajectory well

Image deblurring

Can be formulated as a QP

50 fixed-point steps

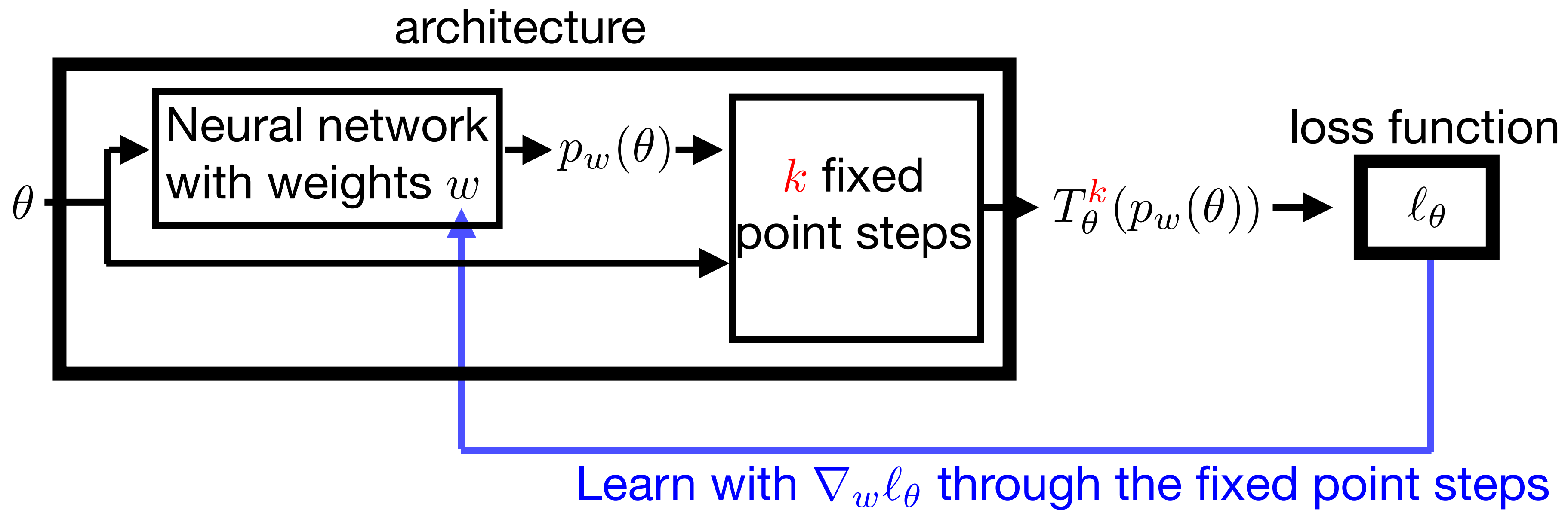
Distance to nearest neighbor increases



With learning, we can deblur all of the images quickly

Benefits of our learning framework

End-to-end learning: warm-start predictions tailored to downstream algorithm



architecture

θ

Neural network with weights w

$p_w(\theta)$

k fixed point steps

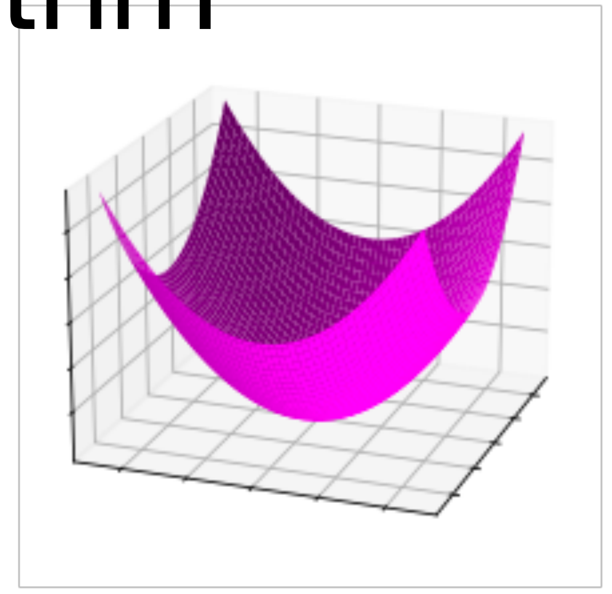
$T_\theta^k(p_w(\theta))$

loss function

ℓ_θ

Learn with $\nabla_w \ell_\theta$ through the fixed point steps

Guaranteed convergence



Can interface with state-of-the-art solvers

OSQP
UNIVERSITY OF OXFORD

SCS
SPLITTING CONIC SOLVER

Quadratic programs Conic programs

Generalization to

Future iterations
Unseen data

rajivs@princeton.edu

rajivsambharya.github.io

arXiv

https://arxiv.org/pdf/2309.07835.pdf