Learning to Warm-Start Fixed-Point Optimization Algorithms

MOPTA 2023

Rajiv Sambharya

Collaborators

Vinit
Ranjan

PRINCETON UNIVERSITY

Georgina
Hall

INSEAD

Brandon
Amos

Meta

Bartolomeo
Stellato

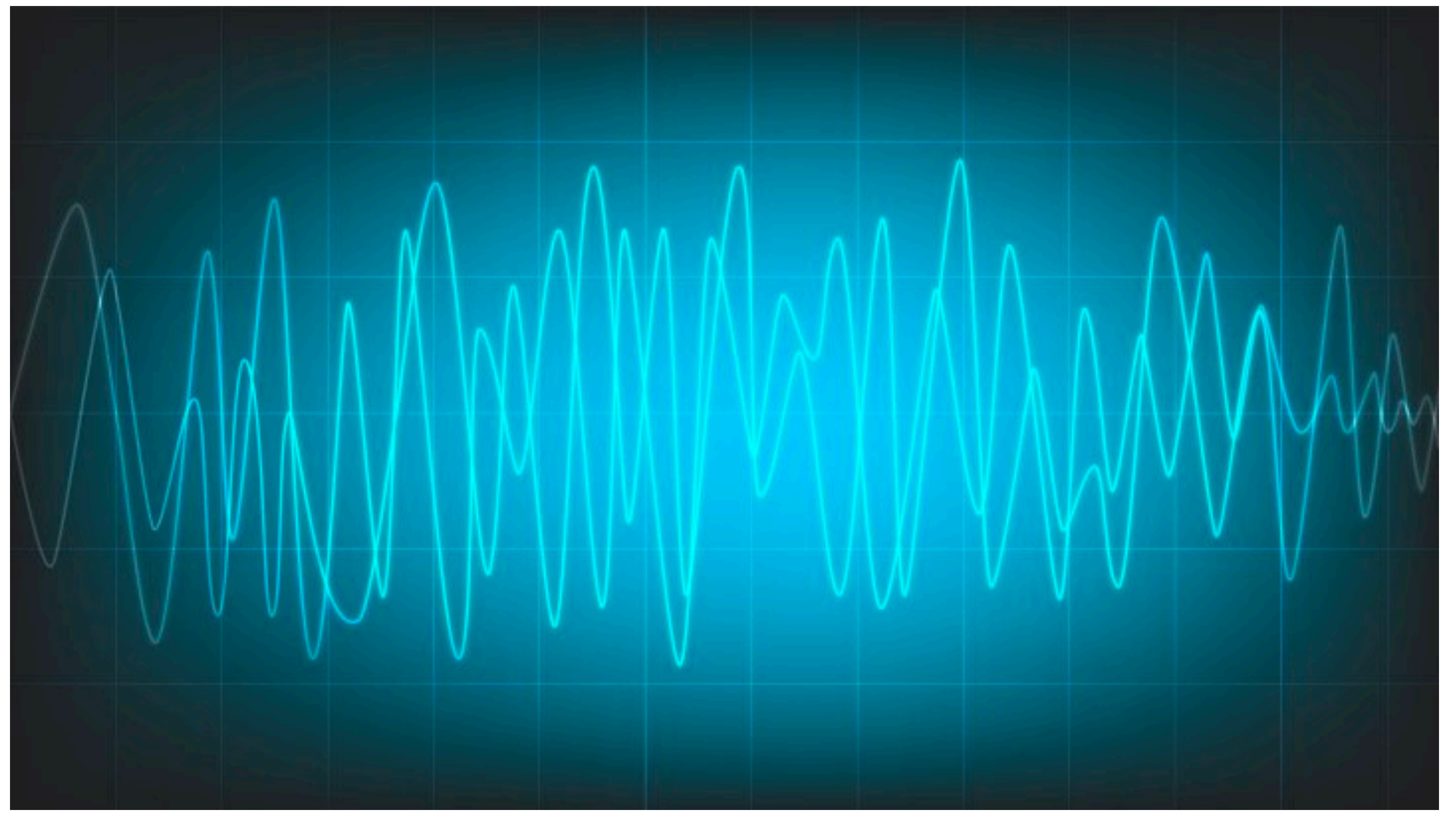
PRINCETON UNIVERSITY

Fixed-point problems need solutions in real-time

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

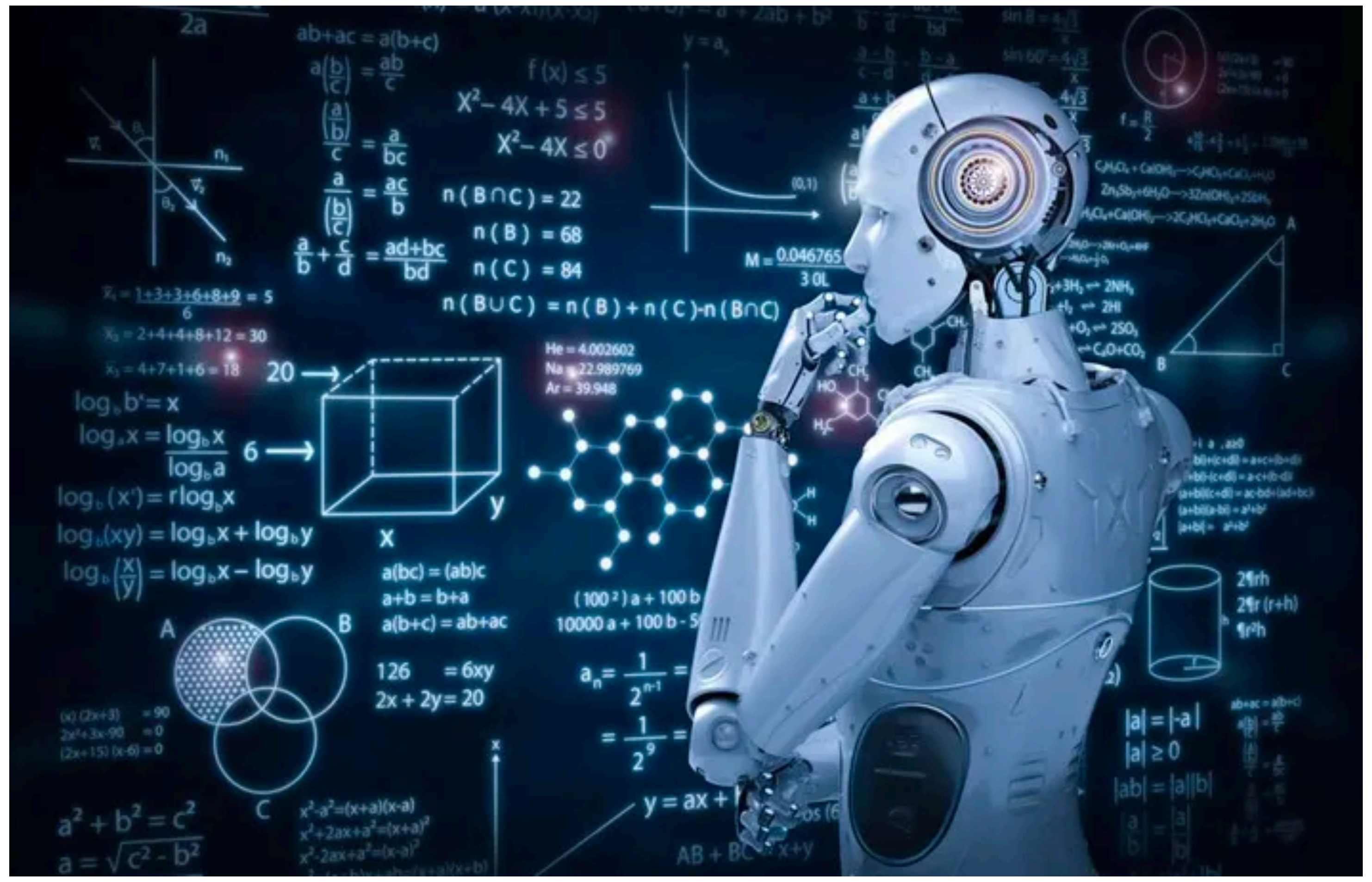
Robotics and control

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

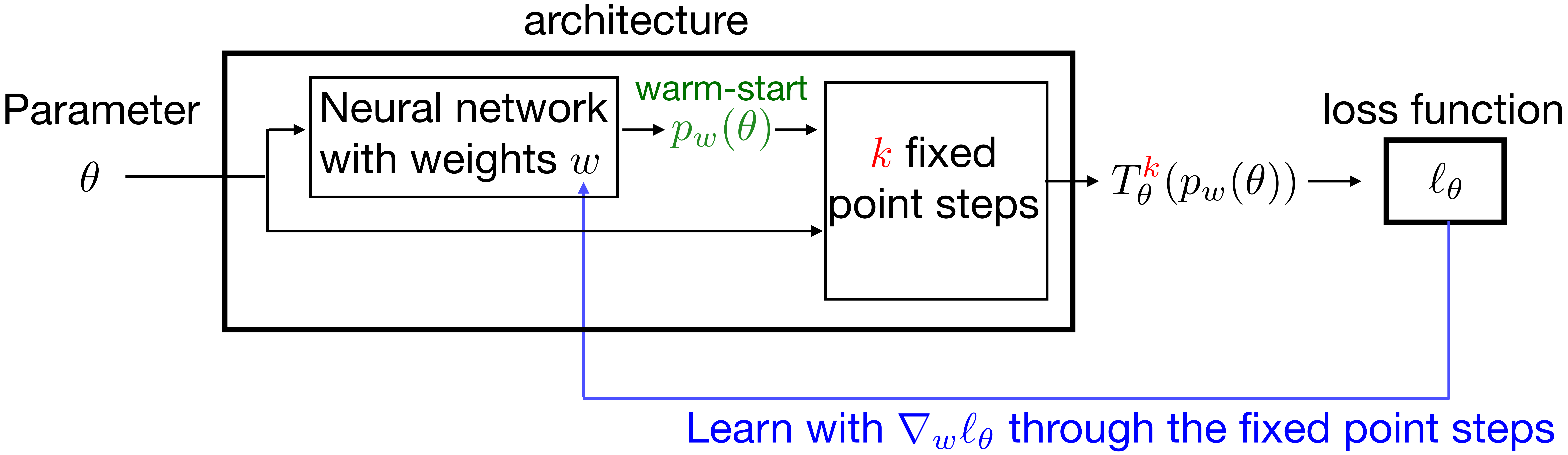
Used to solve: Lasso

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

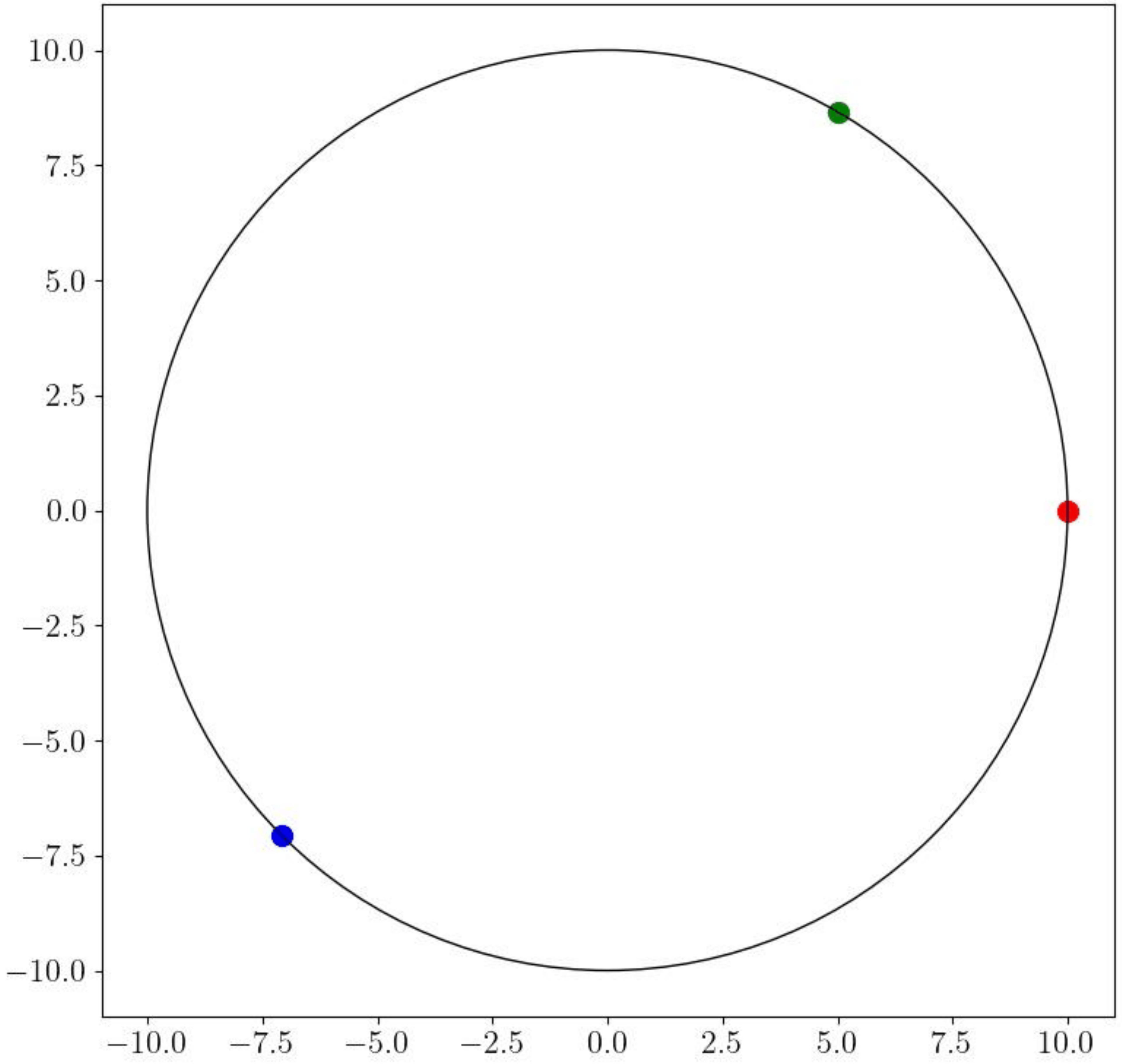
Learn with $\nabla_w \ell_\theta$ through the fixed point steps

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

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



Run proximal gradient descent to solve

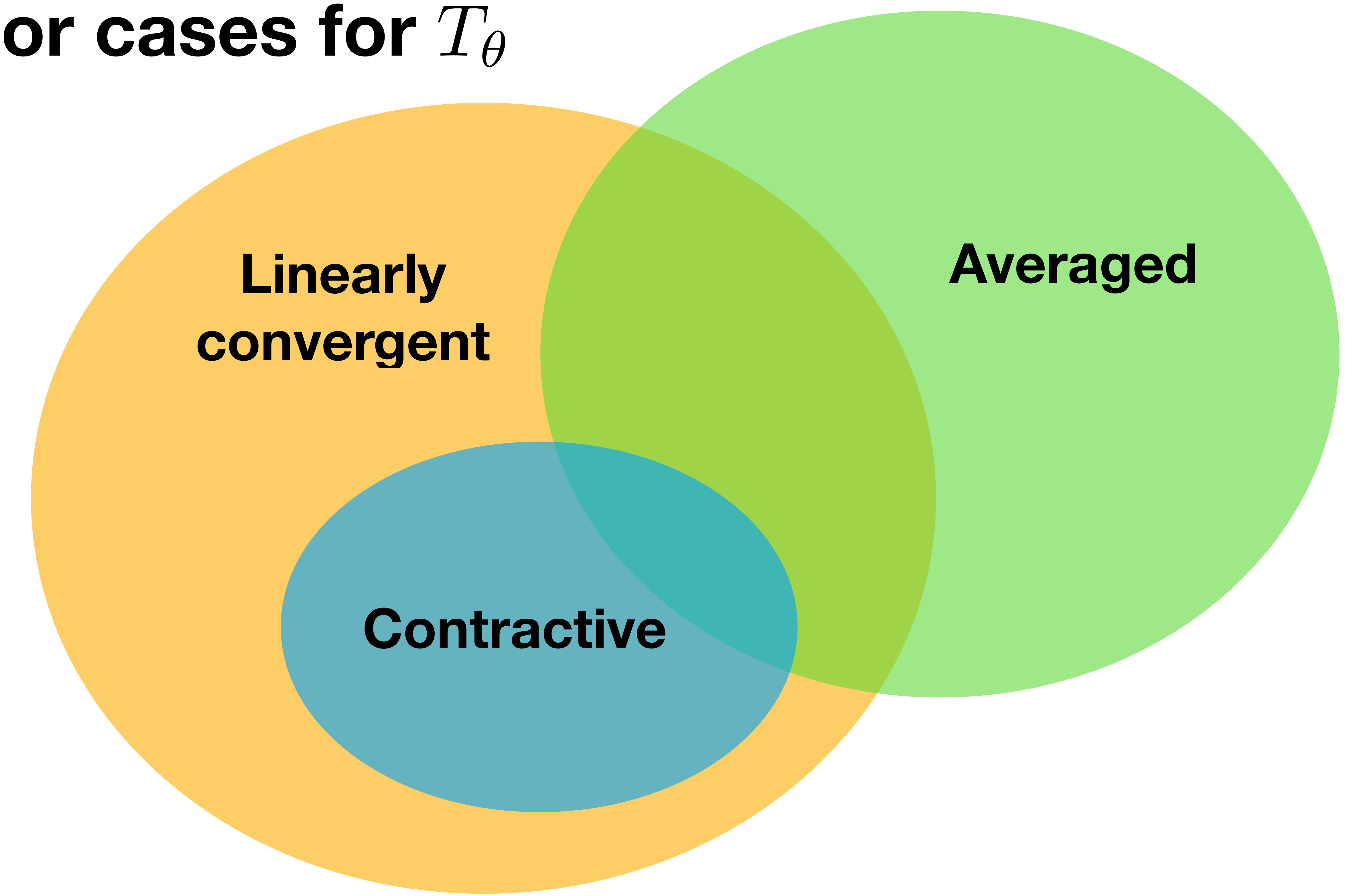
Optimal solution at the origin

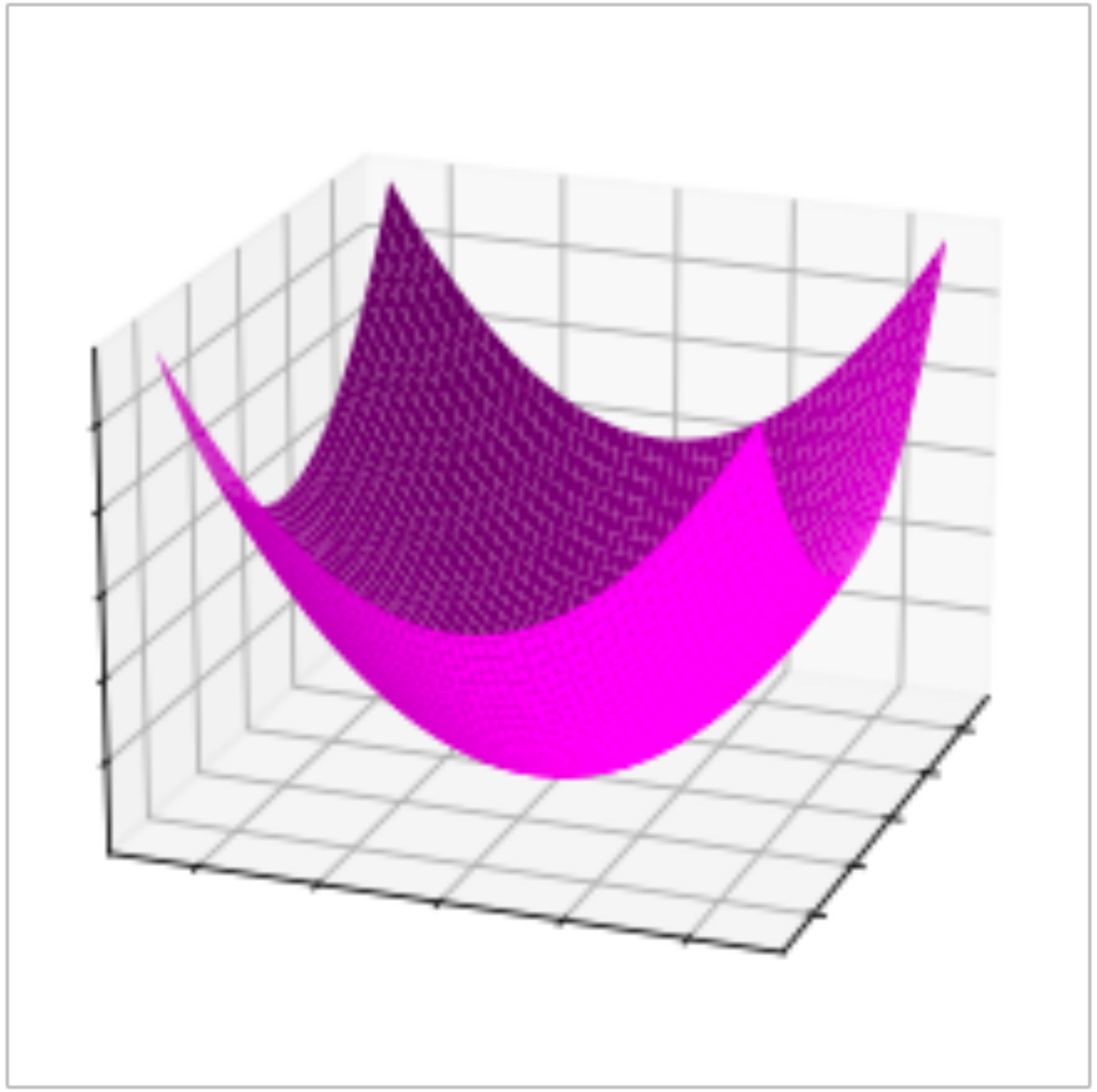
All three warm-starts are equally suboptimal but converge at very different rates

Convergence and Generalization Bounds

Guaranteed convergence independent of warm-start

Operator cases for T_θ





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

Upcoming generalization guarantees depend on the case

Generalization bounds: train for k, evaluate for t

Number of fixed-point steps

Train with k steps

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

Budget is t steps

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

Can we bound the fixed point residual after t fixed-point steps?

Yes

Operator	$\frac{f_\theta(T_\theta^t(z))}{\ell_\theta(T_\theta^k(z))}$ bound
β-contractive	$2\beta^{t-k}$
β-linearly convergent	$2\beta^{t-k}$
α-averaged	$\sqrt{\frac{\alpha}{(1-\alpha)(t-k+1)}}$

e.g., Contractive case: $f_\theta(T_\theta^t(z)) \leq 2\beta^{t-k}\ell_\theta(T_\theta^k(z))$

We can get guarantees on future iterations

Generalization bounds: unseen data

β-contractive case

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

$$\mathbf{E} f_\theta(T_\theta^t(p_w(\theta))) \le \frac{1}{N}\sum_{i=1}^{N} f_{\theta_i}(T_{\theta_i}^t(p_w(\theta_i))) + 2\beta^t\gamma + \mathcal{O}\left(c_1(t)\sqrt{\frac{c_2(w)+\log(\frac{LN}{\delta})}{\gamma^2 N}}\right)$$

Risk

Empirical risk

Penalty term

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

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

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

We evaluate the gain over a cold-start

Baseline initializations

1. Cold-start: initialize at zero ❄️
2. Nearest neighbor: initialize with solution of nearest training problem

Metrics plotted

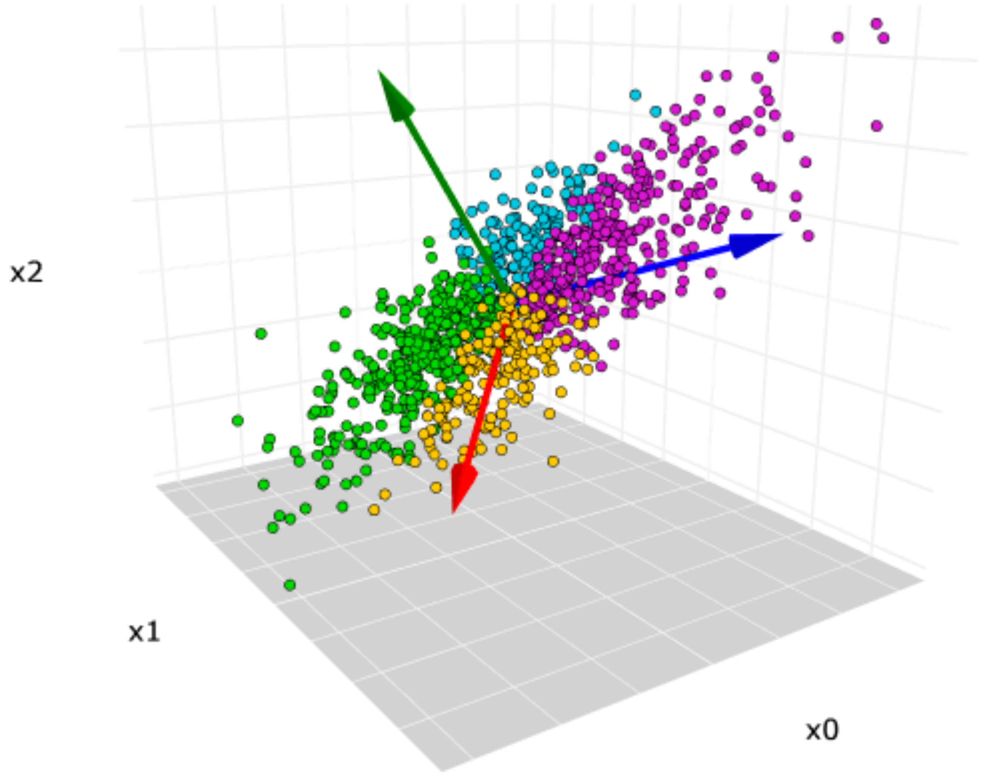
1. Fixed-point residual
2. Gain over the cold-start

$$\text{gain} = \frac{f_\theta(T_\theta^t(0))}{f_\theta(T_\theta^t(p_w(\theta)))}$$

Cold-start ❄️

Learned warm-start 🚀

Sparse PCA



Non-convex problem

$$\begin{array}{ll} \text{maximize} & x^T A x \\ \text{subject to} & \|x\|_2 \leq 1 \\ & \textbf{Card}(x) \leq c \end{array}$$

$$\longrightarrow$$

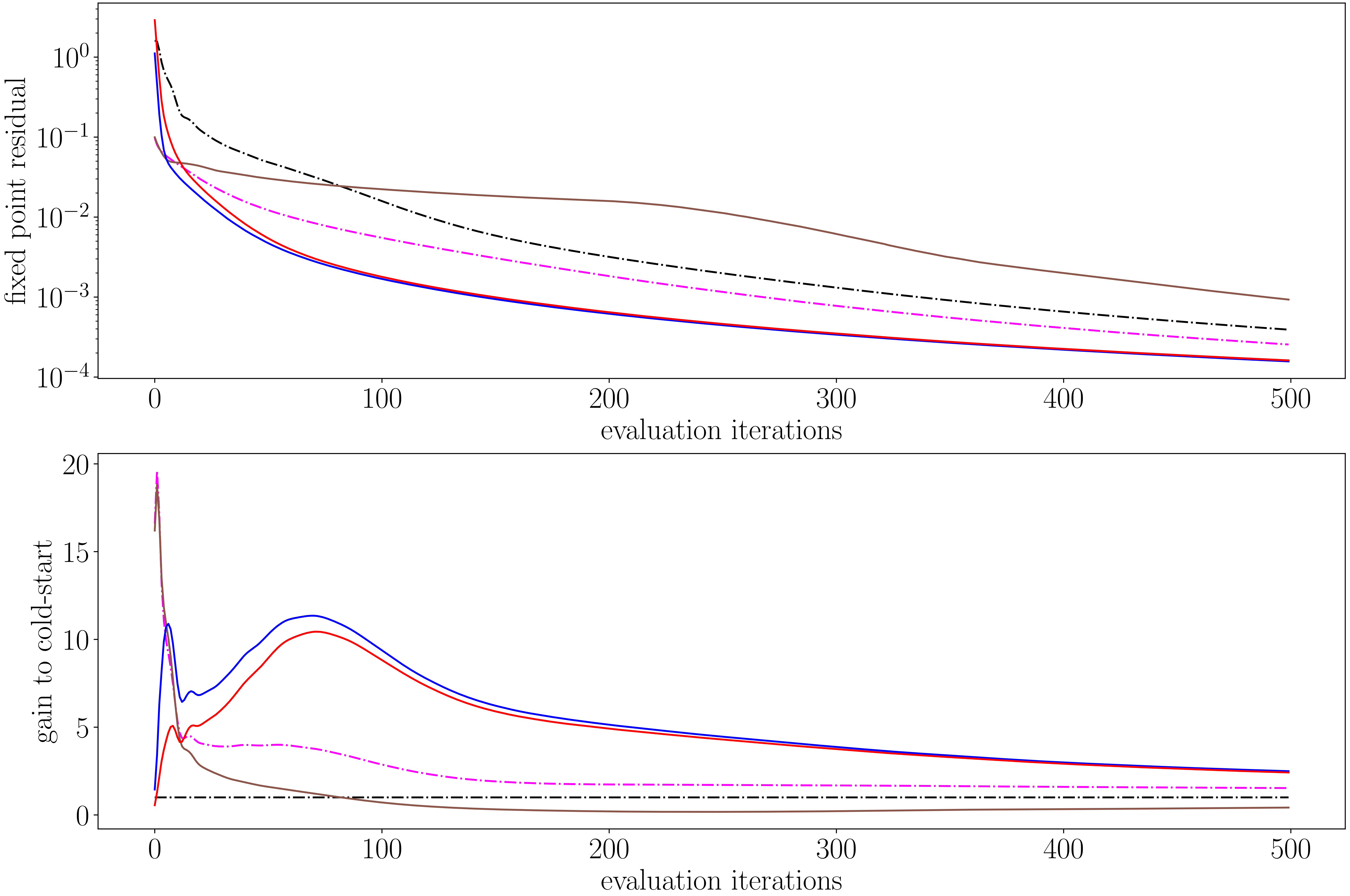
Semidefinite relaxation

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

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

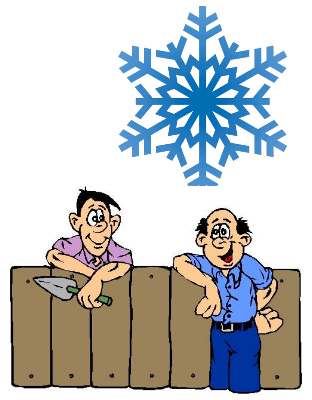
Applications such as streaming data-analysis need quick solutions

Sparse PCA results



Different initializations

Baselines



⬛ Cold-start

🟪 Nearest neighbor

Learned

🟫 $k = 0$

🟦 $k = 5$

🟥 $k = 15$

Picking k > 0 is essential to improve convergence

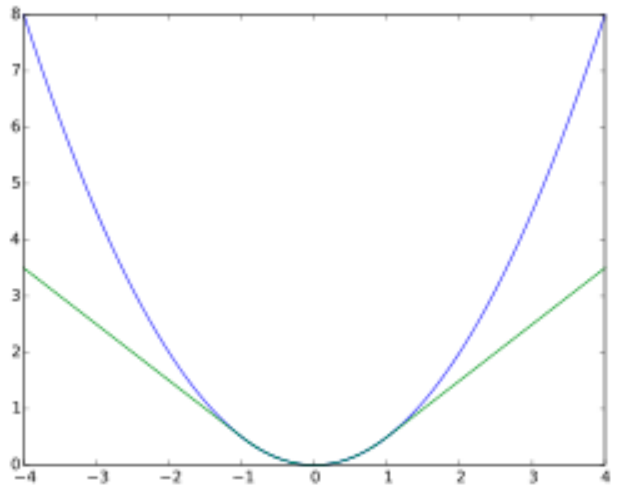
Robust Kalman filtering

Second-order cone program

$$\begin{array}{ll} \text{minimize} & \sum_{t=1}^{T-1} \|w_t\|_2^2 + \mu\psi_\rho(v_t) \\ \text{subject to} & x_{t+1} = Ax_t + Bw_t \quad t = 0, \ldots, T-1 \\ & {\color{red} y_t} = Cx_t + v_t \quad t = 0, \ldots, T-1 \end{array}$$

Robustifying against outliers

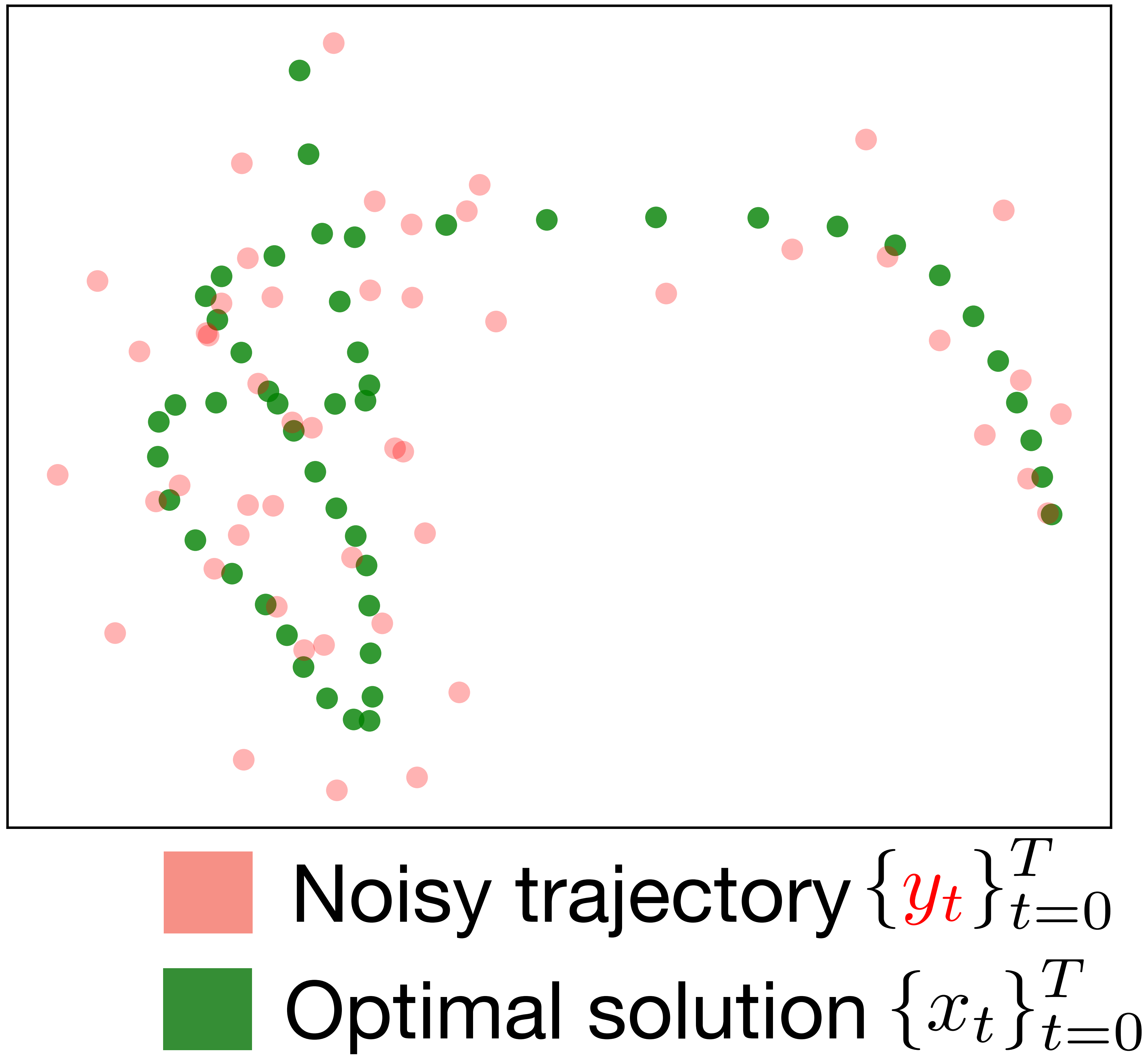
$$\psi_\rho(a) = \begin{cases} \|a\|_2 & \|a\|_2 \leq \rho \\ 2\rho\|a\|_2 - \rho^2 & \|a\|_2 \geq \rho \end{cases}$$

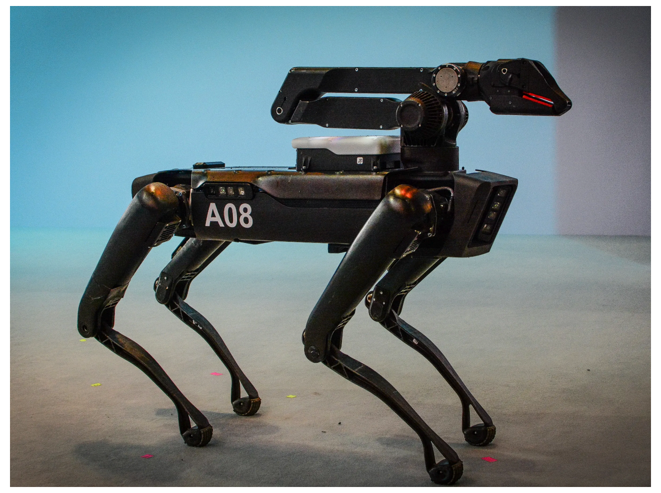


$${\color{red} \theta = \{y_t\}_{t=0}^{T-1}}$$

Dynamics matrices: A, B

Observation matrix: C

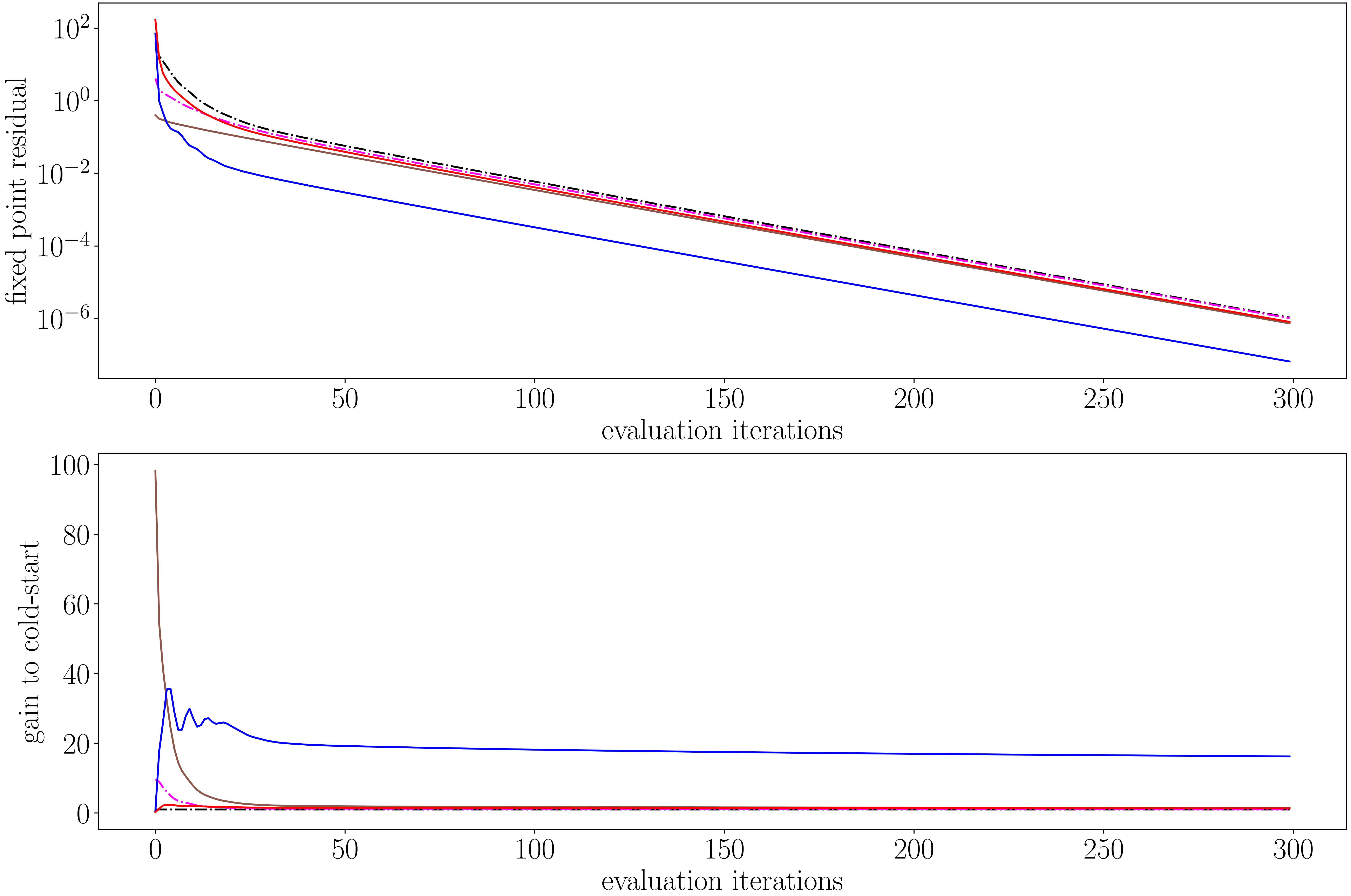




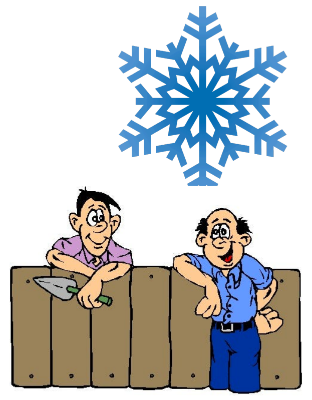
Applications such as GPS and robot tracking need real-time solutions

Robust Kalman filtering results

Different initializations



Baselines



⬛ Cold-start

🟪 Nearest neighbor

Learned

🟫 $k = 0$

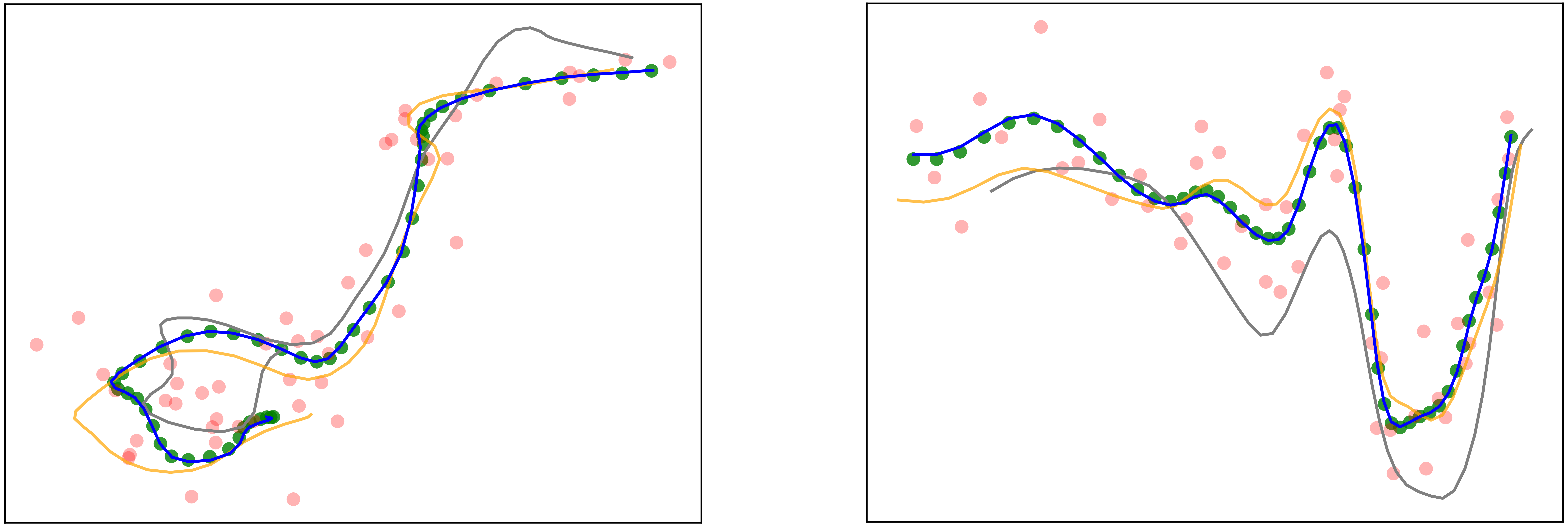
🟦 $k = 5$

🟥 $k = 15$

Small k: fails to generalize to more steps
Large k: fails to generalize to unseen data

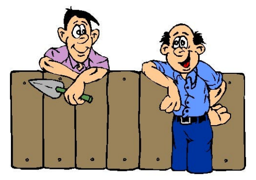
Sweet spot in the middle

Robust Kalman filtering visuals



Solution after 5 fixed-point steps
with different initializations

- 🟥 Noisy trajectory
- 🟩 Optimal solution

- 🟨 Nearest neighbor 
- ⬛ Previous solution 🔥
- 🟦 Learned: $k = 5$ 🚀

With learning, we can estimate the state well

Model predictive control (MPC) of a quadcopter

**MPC main idea: solve problem over finite horizon,
implement first control, repeat**

Quadratic program

minimize $(x_T - x_T^{\text{ref}})^T Q_T (x_T - x_T^{\text{ref}}) + \sum_{t=1}^{T-1} (x_t - x_t^{\text{ref}})^T Q (x_t - x_t^{\text{ref}}) + \sum_{t=0}^{T-1} u_t^T R u_t$

subject to

$$x_{t+1} = Ax_t + Bu_t$$
$$u_{\min} \leq u_t \leq u_{\min}$$
$$x_{\min} \leq x_t \leq x_{\max}$$
$$|u_{t+1} - u_t| \leq \Delta u$$
$$x_0 = x_{\text{init}}$$
$$u_{-1} = u_{\text{prev}}$$

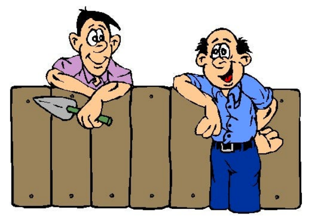
Linearized dynamics

$$\theta = (x_{\text{init}}, u_{\text{prev}}, x_1^{\text{ref}}, \ldots, x_T^{\text{ref}})$$

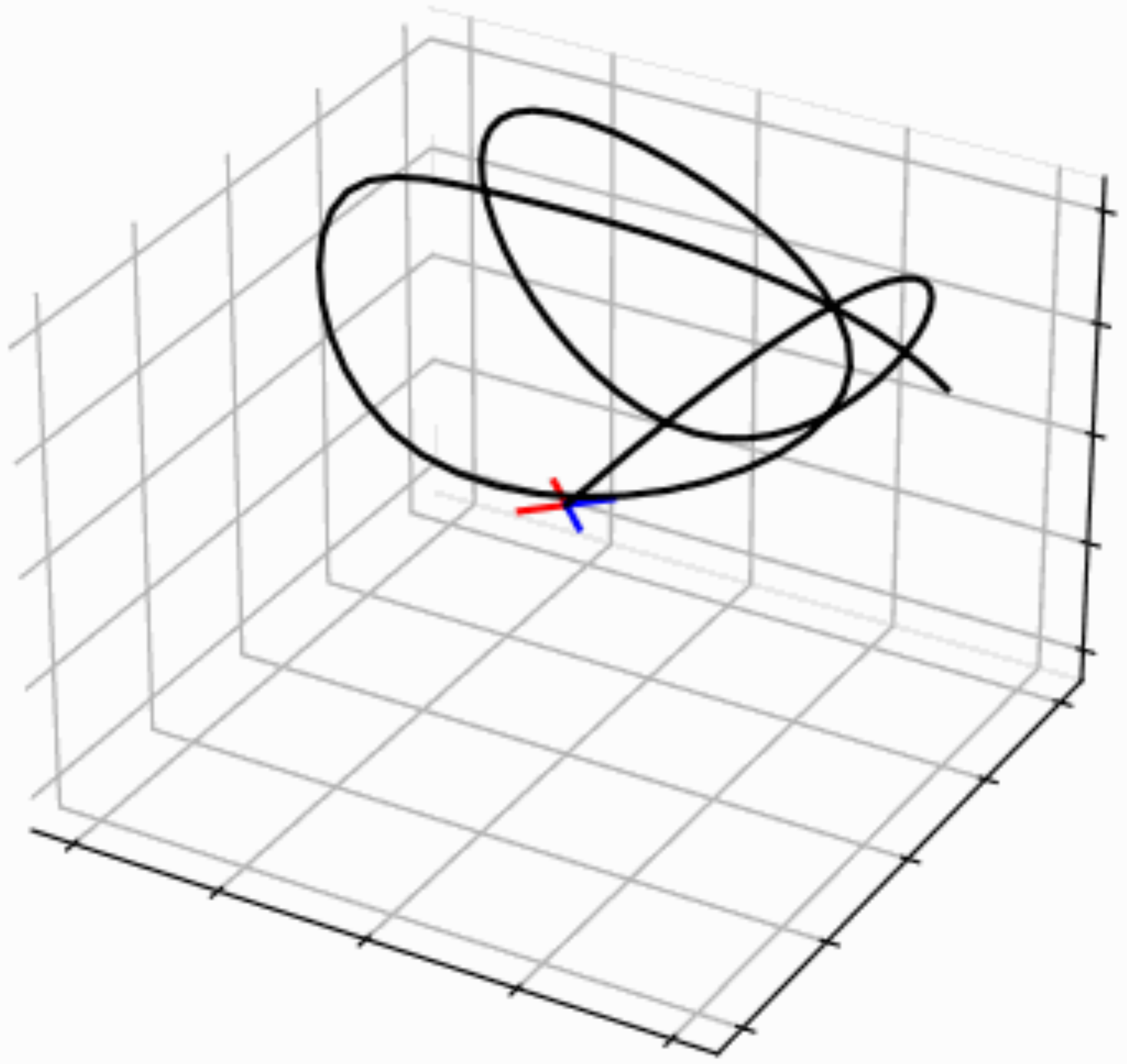
Flying safely requires real-time solutions

MPC of a quadcopter in a closed loop

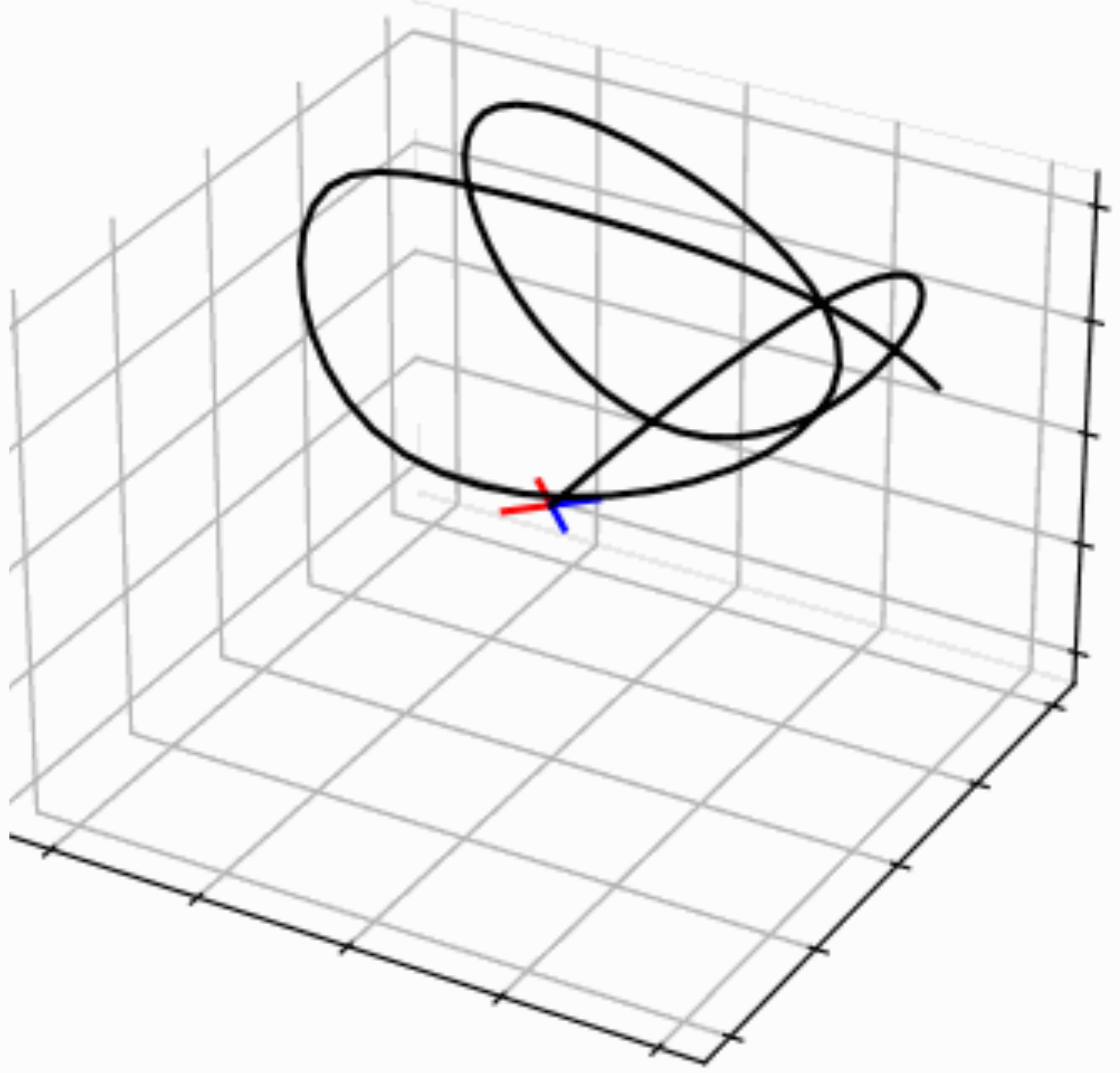
Budget of 5 fixed-point steps



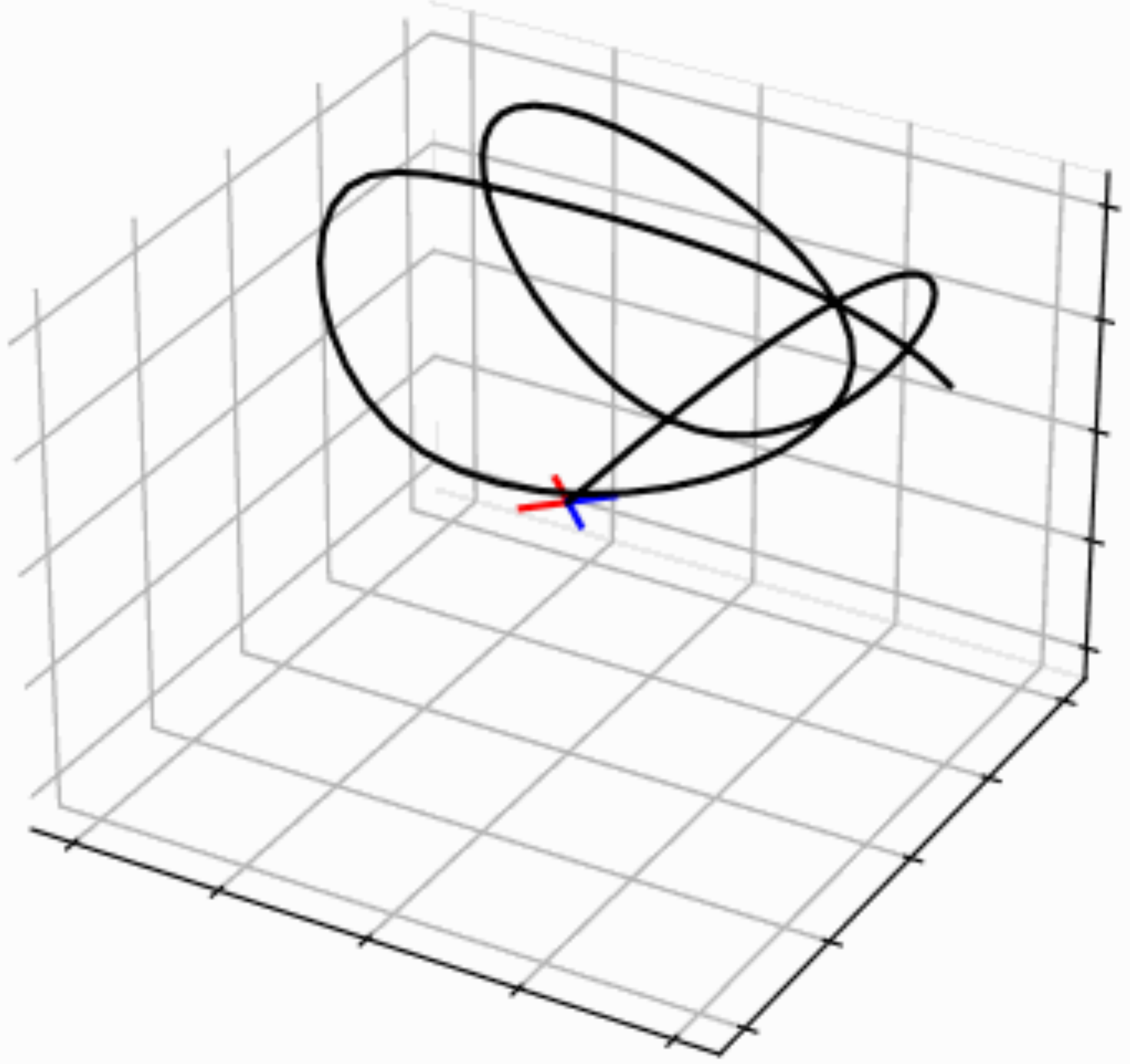
Nearest neighbor



Previous solution



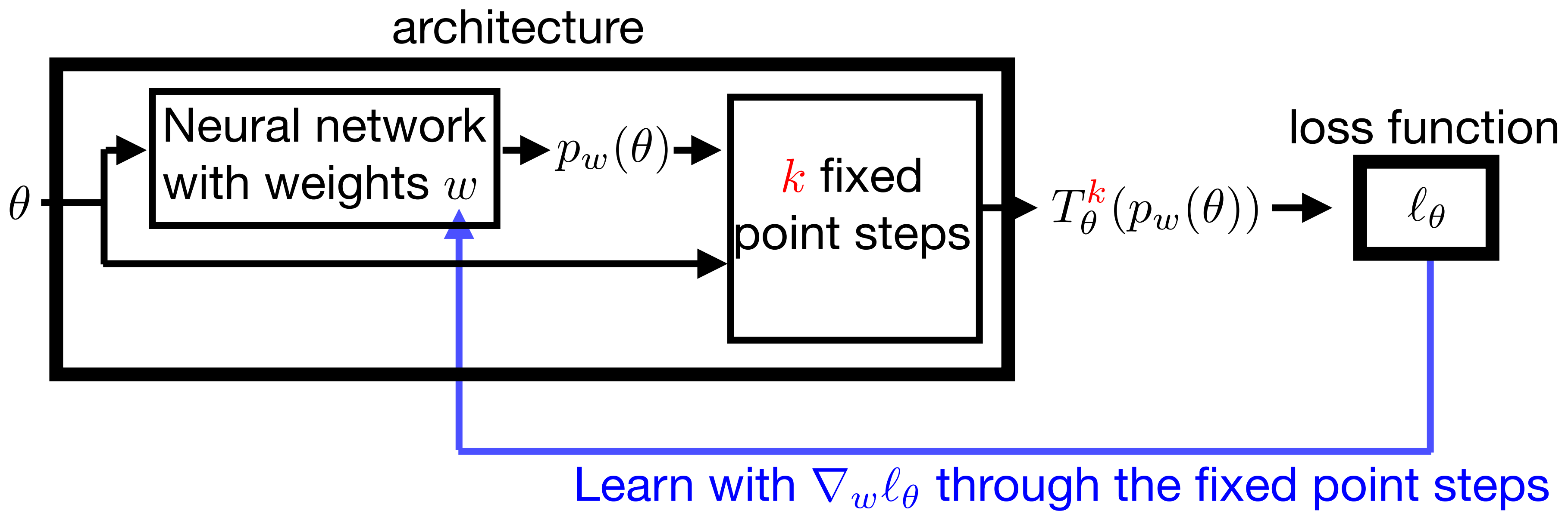
Learned: $k = 5$



With learning, we can track the trajectory well

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

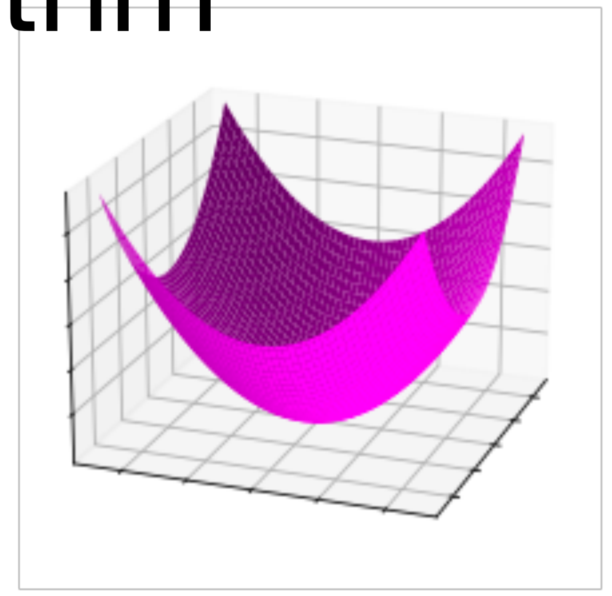
$T_\theta^k(p_w(\theta))$

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

Paper coming out in August!

Earlier paper on quadratic programs

5th Conference on Learning for Dynamics and Control, 2023

rajivs@princeton.edu

rajivsambharya.github.io