Practical Performance Guarantees for Classical and Learned Optimizers

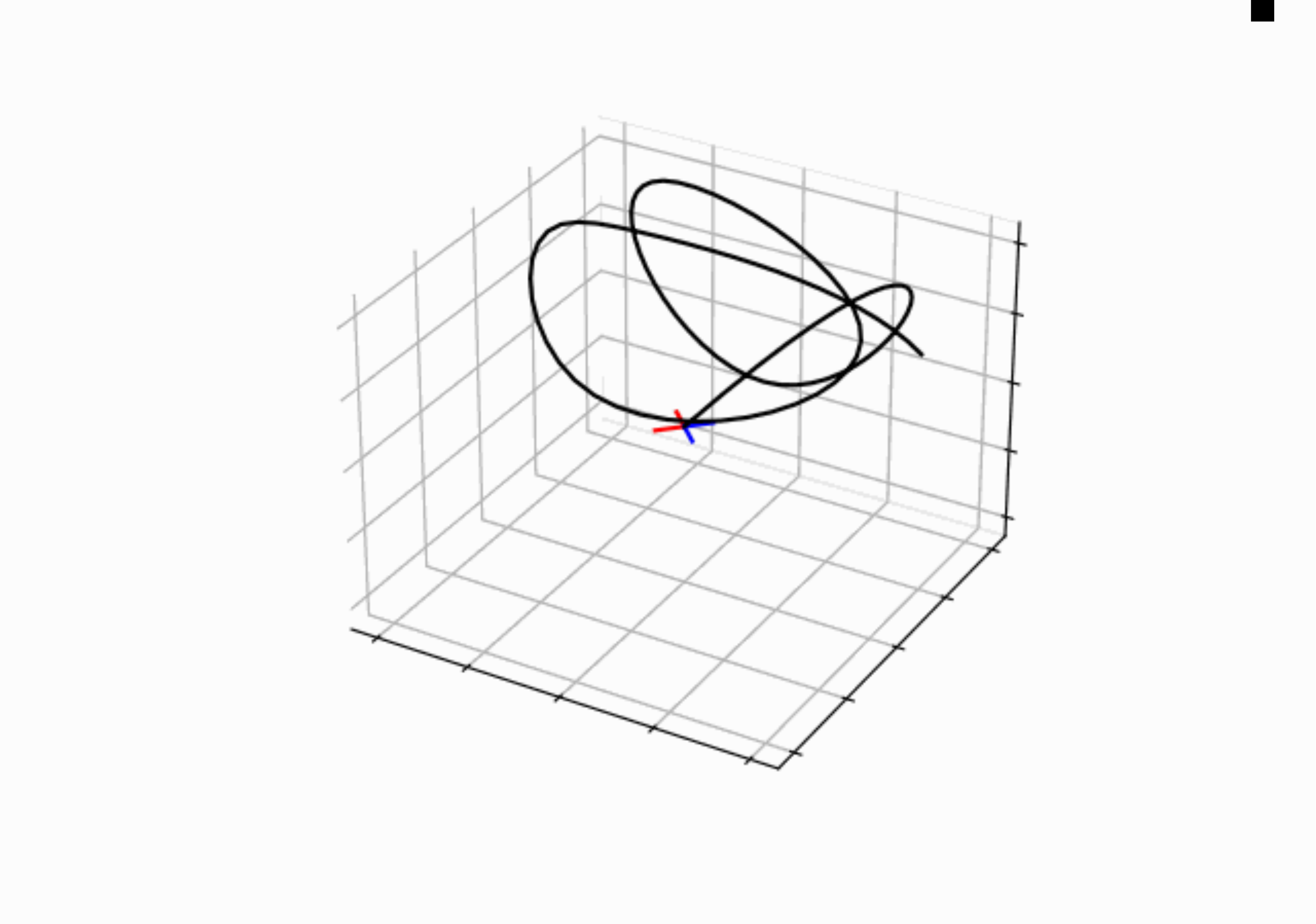
CISS Talk 2024

Rajiv Sambharya

PRINCETON UNIVERSITY

ORFE

Claim: real-world optimization is parametric



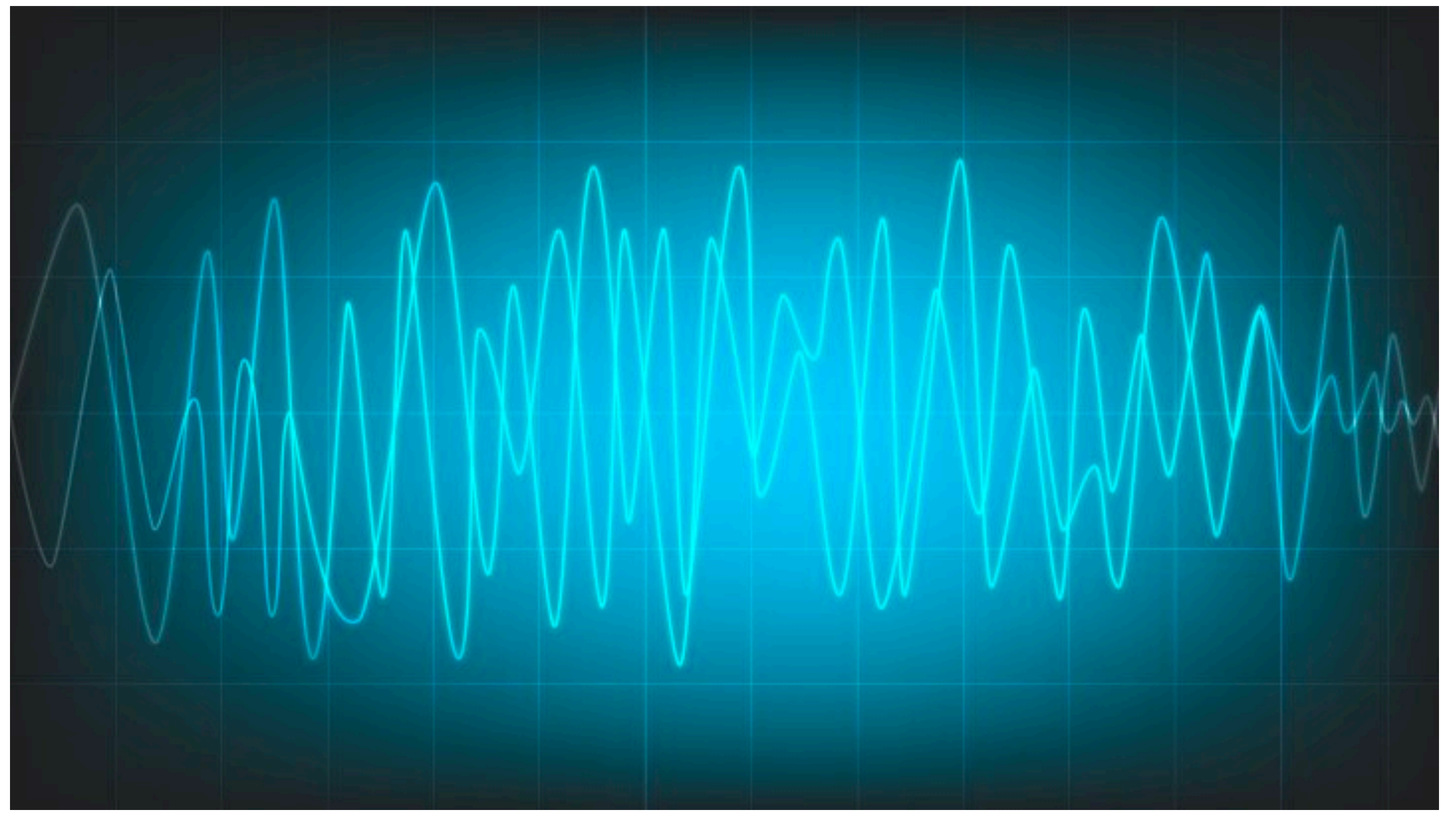
Model predictive control

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

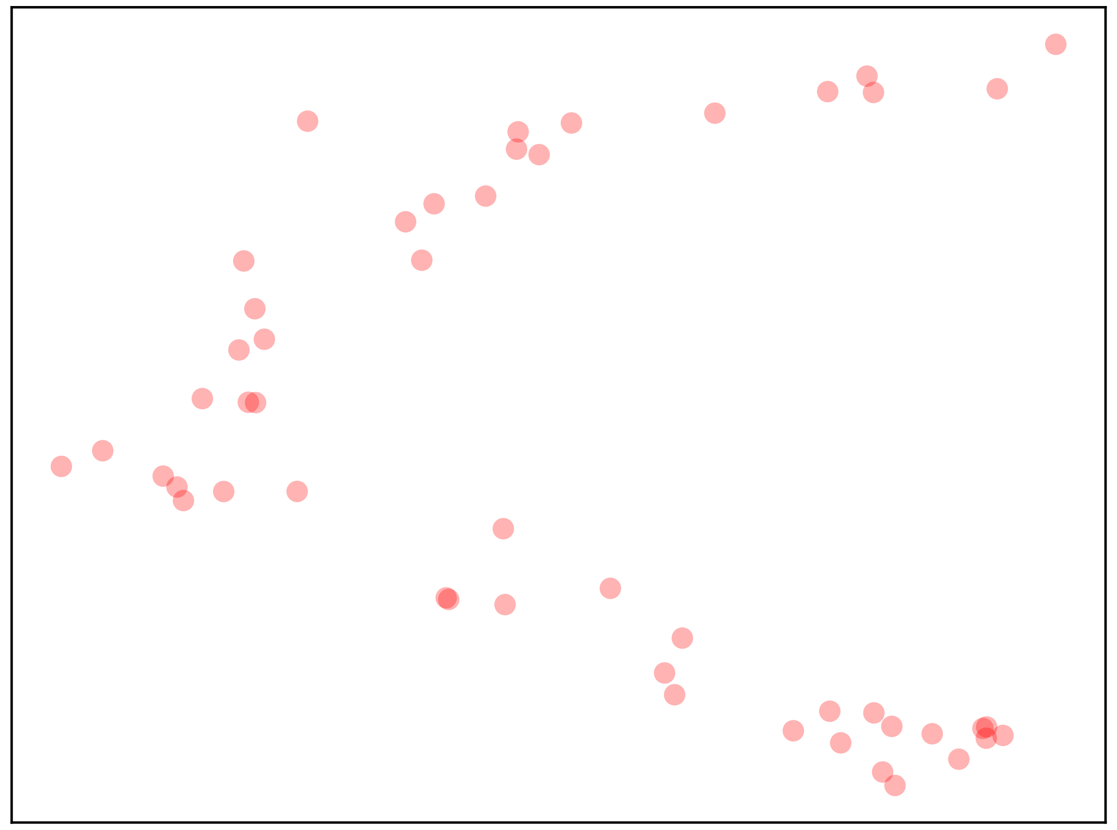
Robotics and control

Energy

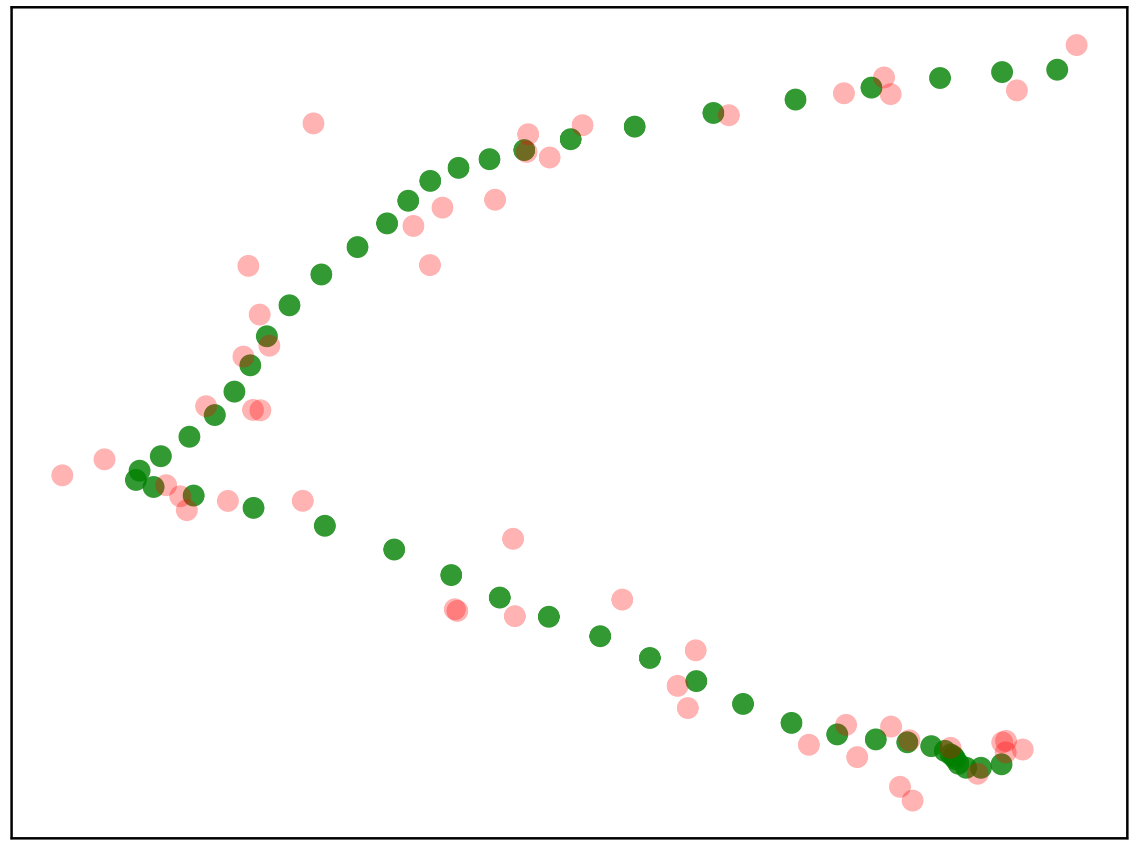
Signal processing



Robust Kalman filtering



→ **Robust Kalman filtering** →



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

Noisy trajectory

→

Second-order cone program

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

→

$\{x_t^\star, w_t^\star, v_t^\star\}_{t=0}^{T-1}$

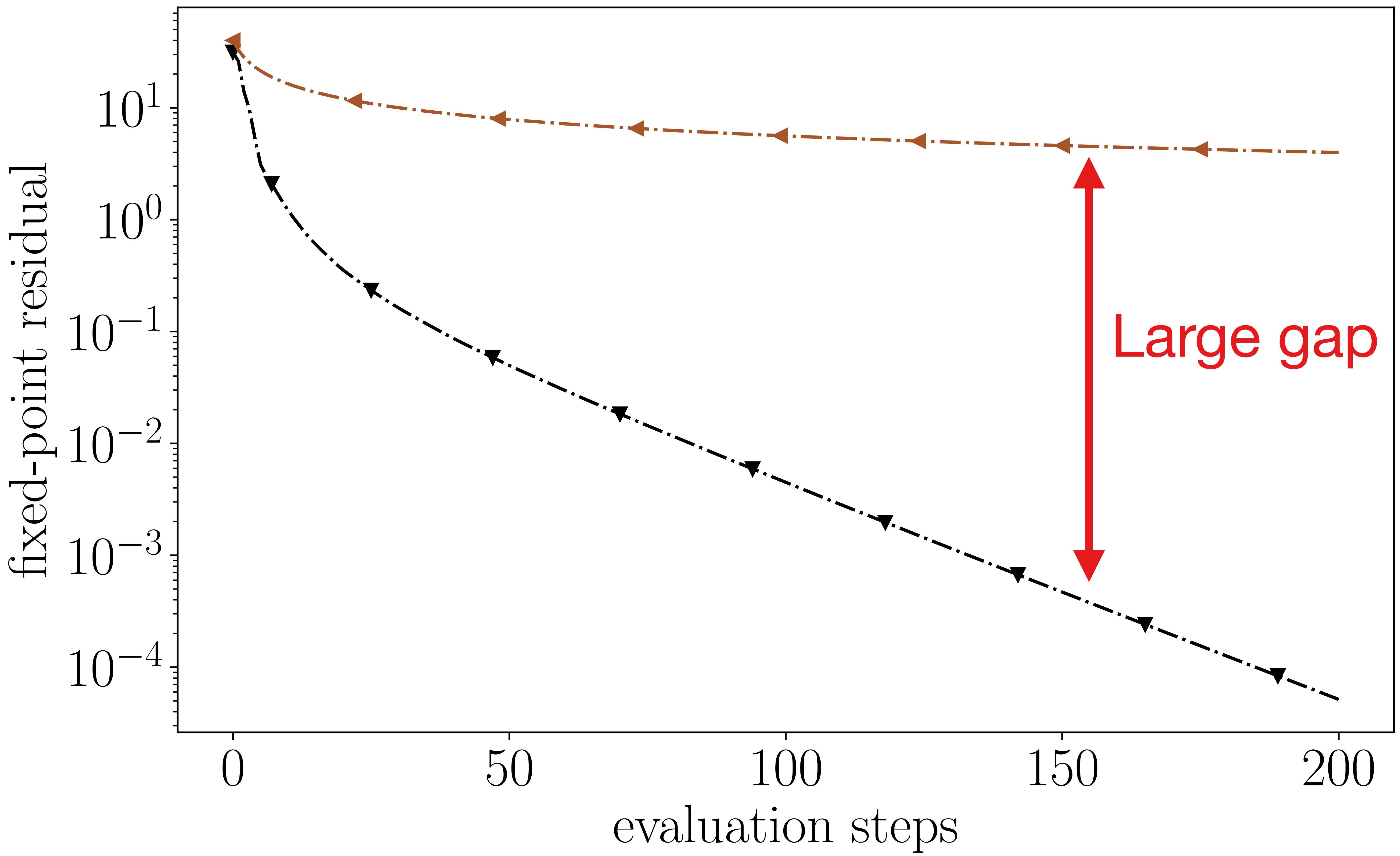
Recovered trajectory

Dynamics matrices: A, B

Observation matrix: C

Huber loss: ψ_ρ

Worst-case bounds can be very loose



Example: robust Kalman filtering

Second-order cone program

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

▼ SCS empirical average performance over 1000 parametric problems

◀ Worst-case bound

In practice: **linear** convergence over the parametric family

Worst-case analysis: **sublinear** convergence

Worst-case bounds do not consider the **parametric** structure

Practical Performance Guarantees for Classical and Learned Optimizers

We will bound 0-1 error metrics

We will provide guarantees for any measured quantity

$$e(\theta) = \mathbf{1}(\ell^{\overset{\text{algorithm steps}}{k}}(\theta) > \overset{\text{tolerance}}{\epsilon})$$

Standard metrics

e.g., fixed-point residual

Task-specific metrics:

e.g., quality of extracted states in robust Kalman filtering

Background: Kullback-Liebler Divergence

KL divergence: measures distance between distributions

$$\mathrm{KL}(q \parallel p) = \sum_{i=1}^{m} q_i \log\left(\frac{q_i}{p_i}\right)$$

Our bounds on the risk will take the form

$\mathrm{KL}($empirical risk $\parallel$ risk$) \leq$ regularizer

Invert these bounds by solving

risk $\leq \mathrm{KL}^{-1}$ (empirical risk $|$ regularizer)

$$\mathrm{KL}^{-1}(q \mid c) = \begin{array}{ll} \text{maximize} & p \\ \text{subject to} & q \log \frac{q}{p} + (1-q)\log\frac{1-q}{1-p} \leq c \\ & 0 \leq p \leq 1 \end{array}$$

1D convex optimization problem

Statistical learning theory can provide probabilistic guarantees

algorithm steps

tolerance

$$e(\theta) = \mathbf{1}(\ell^{k}(\theta) > \epsilon)$$

Sample convergence bound: with probability $1 - \delta$ [Langford et. al 2001]

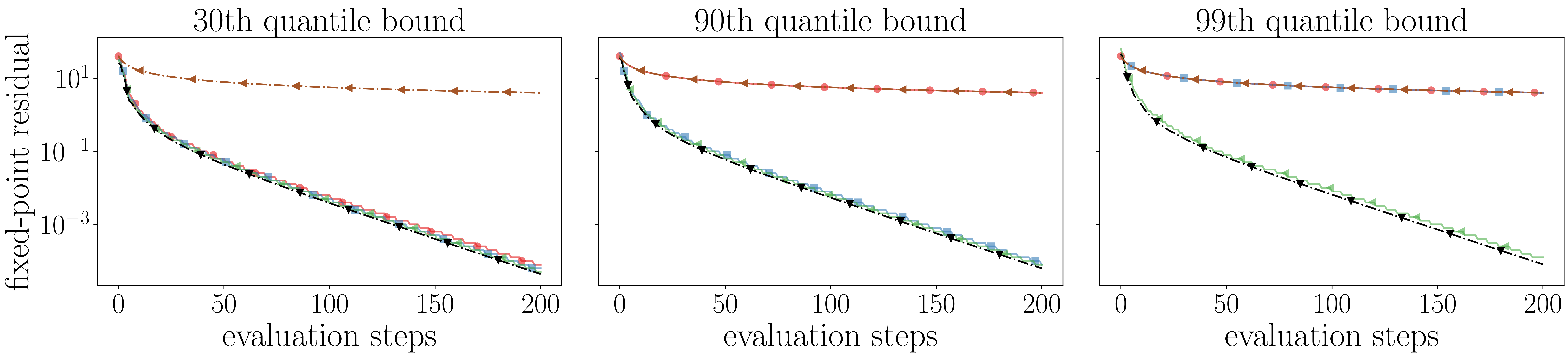
$$\mathbf{E}_{\theta \sim \mathcal{X}} e(\theta) \leq \mathrm{KL}^{-1}\left(\frac{1}{N}\sum_{i=1}^{N} e(\theta_i) \middle| \frac{\log(2/\delta)}{N}\right)$$

Number of problems

$$\mathbf{P}(\ell^{k}(\theta) > \epsilon) = \text{risk} \leq \mathrm{KL}^{-1}\,(\text{empirical risk} \mid \text{regularizer})$$

”With probability $1-\delta$, 90% of the time the fixed-point residual is below $\epsilon = 0.01$ after $k = 20$ steps”

Robust Kalman filtering guarantees



▼ Empirical average

◀ Worst-case bound

Probabilistic bound with

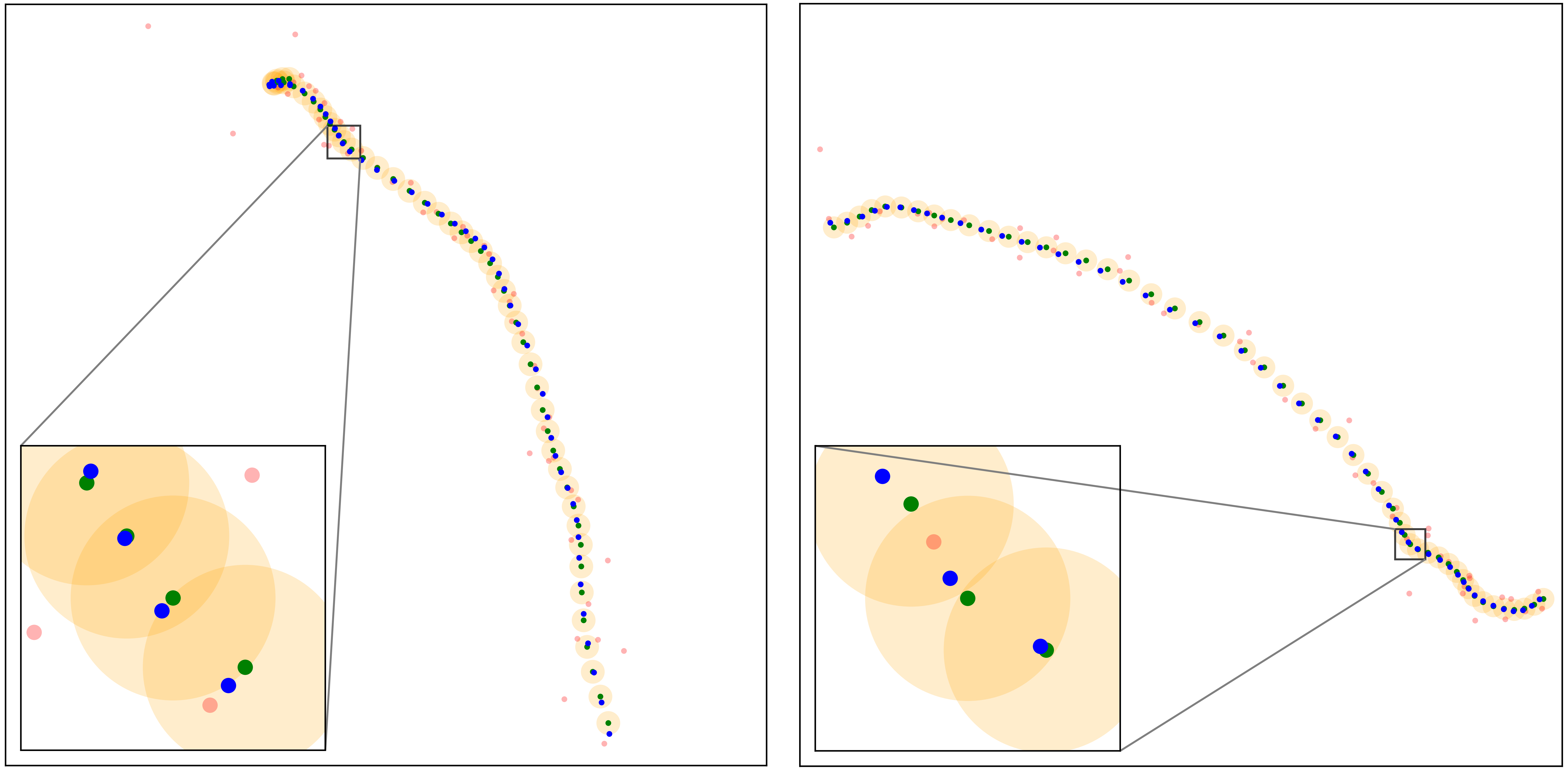
● 10 samples

■ 100 samples

▶ 1000 samples

With 1000 samples, we provide strong probabilistic guarantees on the 99th quantile

Visualizing Robust Kalman filtering guarantees



Task-specific error metric

$$e(\theta) = \mathbf{1}\left(\max_{t=1,\ldots,T} \|x_t - x_t^\star\|_2 > \epsilon\right)$$

🟥 Noisy trajectory

🟩 Optimal solution

🟦 Solution after 15 steps

🟨 Region with guarantee

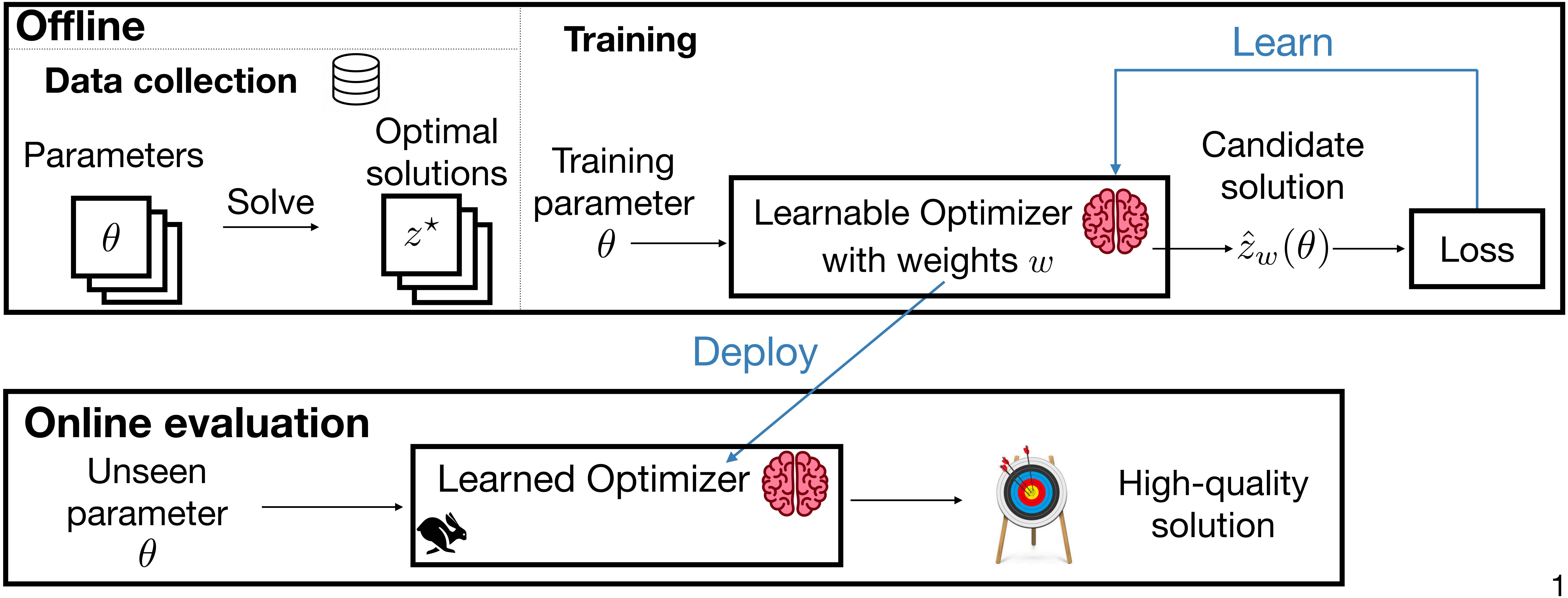
"With high probability, 90% of the time, all of the recovered states after 15 steps of problems drawn from the distribution will be within the correct ball with radius 0.1"

Practical Performance Guarantees for Classical and Learned Optimizers

The learning to optimize paradigm

Goal: solve the parametric optimization problem fast

$$\begin{array}{ll} \text{minimize} & f_\theta(z) \\ \text{subject to} & g_\theta(z) \leq 0 \end{array}$$



Offline

Data collection

Parameters

θ

Solve

Optimal solutions

$z^\star$

Training

Learn

Training parameter θ

Learnable Optimizer with weights w

Candidate solution

$\hat{z}_w(\theta)$

Loss

Deploy

Online evaluation

Unseen parameter θ

Learned Optimizer

High-quality solution

PAC-Bayes guarantees for learned optimizers

algorithm steps

tolerance

$$e_w(\theta) = \mathbf{1}(\ell_w^k(\theta) > \epsilon)$$

learnable weights

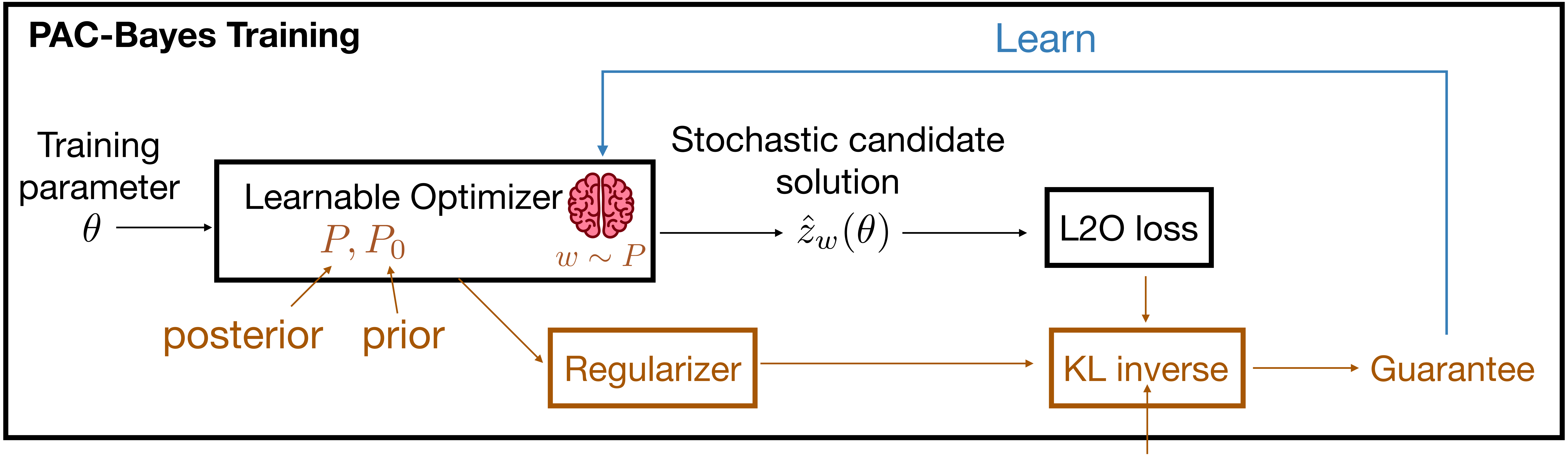
McAllester bound: given posterior and prior distributions [McAllester et. al 2003]
P and P_0, with probability $1-\delta$

$$\mathbf{E}_{\theta\sim\mathcal{X}}\mathbf{E}_{w\sim P}e_w(\theta) \leq \mathrm{KL}^{-1}\left(\frac{1}{N}\sum_{i=1}^{N}\mathbf{E}_{w\sim P}e_w(\theta_i)\middle|\frac{1}{N}\left(\mathrm{KL}(\mathrm{P}\parallel\mathrm{P}_0)+\log(\mathrm{N}/\delta)\right)\right)$$

risk $\leq \mathrm{KL}^{-1}$ (empirical risk | regularizer)

Optimize the bounds directly

PAC-Bayes training architecture to optimize the guarantees

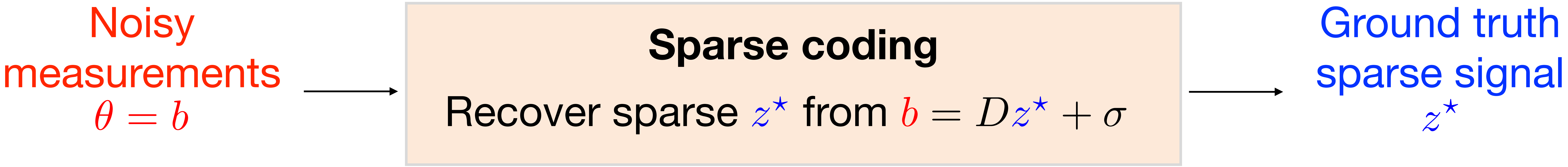


Use differentiable optimization

We show that the derivative always exists

We implement the learnable optimizer and train with this architecture

Learned algorithms for sparse coding



Noisy measurements

$\theta = b$

Sparse coding

Recover sparse $z^\star$ from $b = Dz^\star + \sigma$

Ground truth sparse signal

$z^\star$

D: dictionary, σ: noise

Standard technique

minimize $\|Dz - b\|_2^2 + \lambda \|z\|_1$

ISTA (iterative shrinkage thresholding algorithm)

(Classical optimizer)

$$z^{j+1} = \text{soft threshold}_{\frac{\lambda}{L}} \left(z^j - \frac{1}{L} D^T (Dz^j - b) \right)$$

Learned ISTA

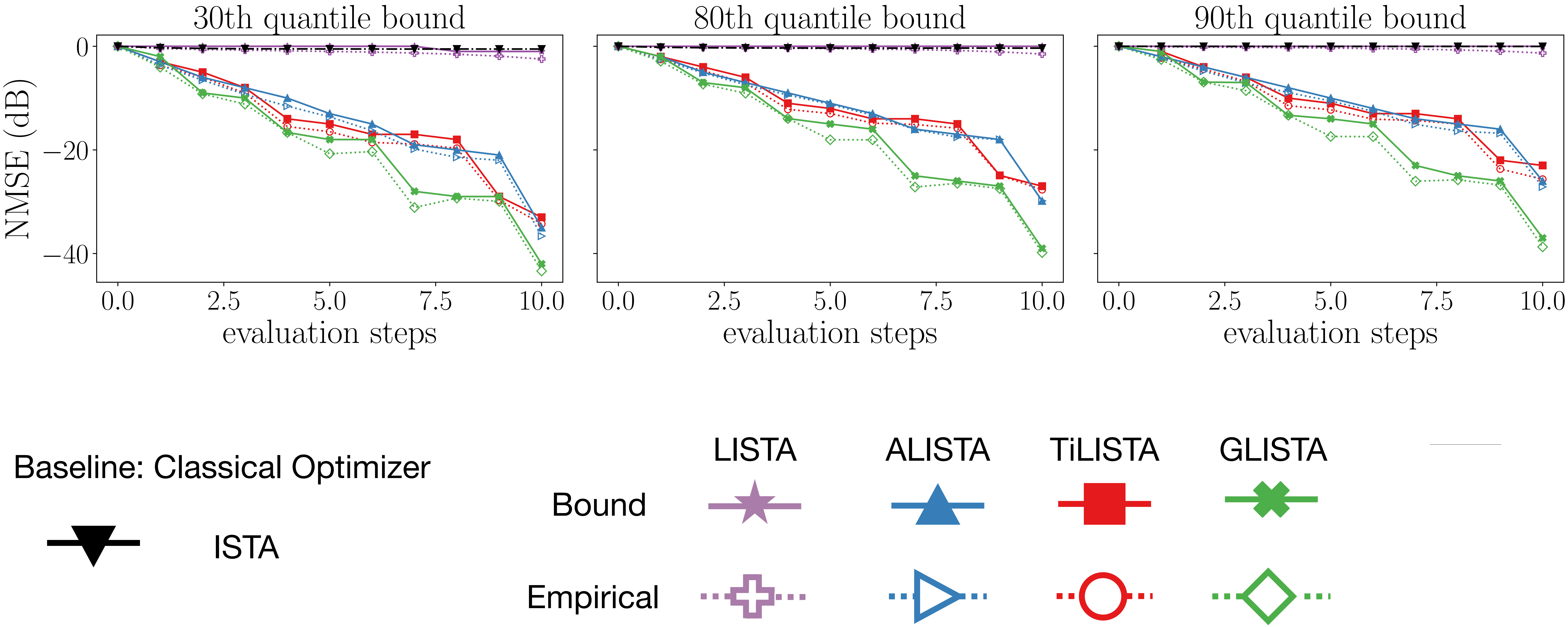
(Learned optimizer)

$$z^{j+1} = \text{soft threshold}_{\psi^j} \left(W_1^j z^j + W_2^j b \right)$$

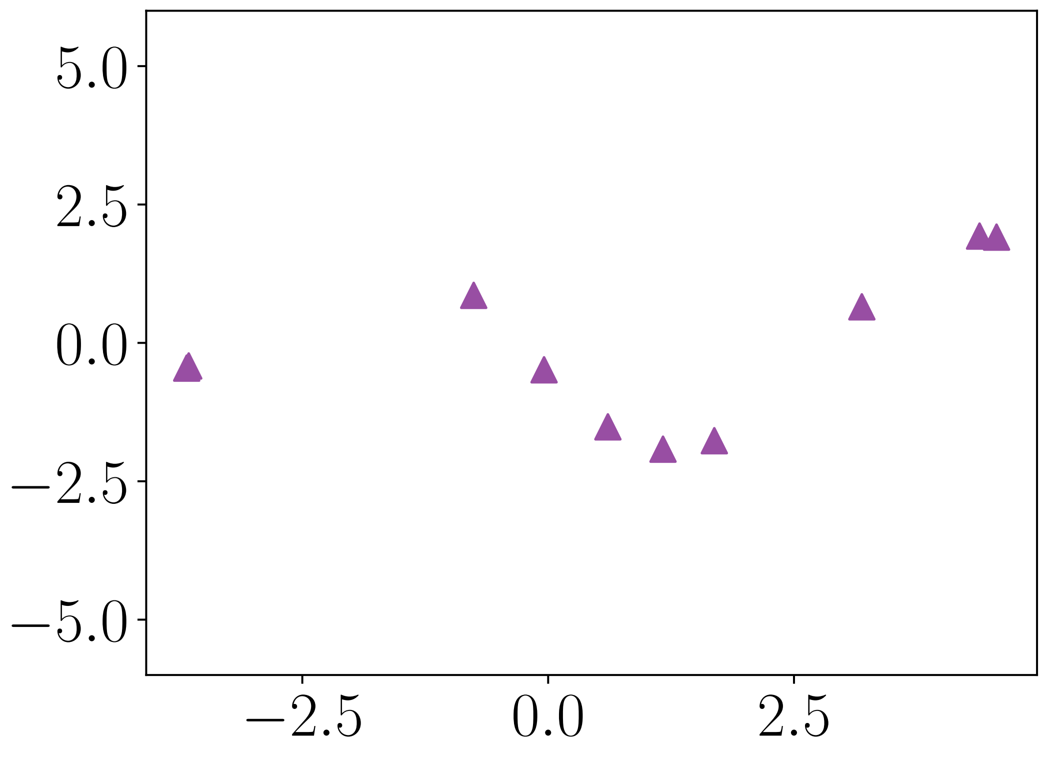
+ variants [Gregor and LeCun 2010, Liu et. al 2019]

$\text{soft threshold}_\psi(z) = \textbf{sign}(z) \max(0, |z| - \psi)$

Learned ISTA results for sparse coding



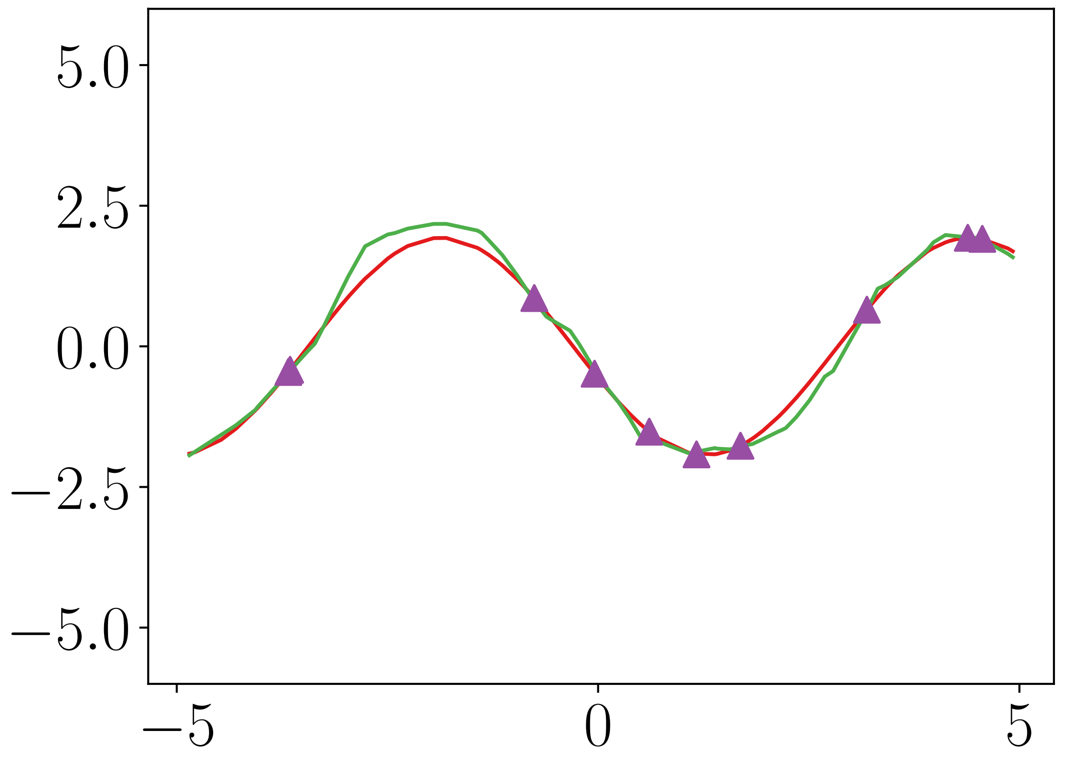
K-shot Meta-Learning for Sine Curves



Neural network learning

find weights z so that $g_z(x_i) \approx y_i$

predictor with weights z



Training dataset with K points

$\mathcal{D}^{\text{train}}$ →

Gradient step

$$\hat{z} = z - \alpha \nabla_z \mathcal{L}(z, \mathcal{D}^{\text{train}})$$

→ Weights that generalize to new points quickly

$\hat{z}$

Model-Agnostic Meta-Learning (MAML) [Finn et. al 2017]

MAML learns a shared initialization z so that $\hat{z}$ performs well on test data

Visualizing Guarantees: K-shot Meta-Learning for Sine Curves

5.0
2.5
0.0
−2.5
−5.0
−5
0
5
▲ Used for gradients
🟥 Ground truth
After 10 grad steps
🟩 Stochastic MAML
🟦 Pretrained
🟧 Region with MAML guarantee

With high probability, 90% of the time stochastic MAML after 10 steps will stay within the band

The pretrained baseline only stays within the band 30% of the time

Conclusions

Statistical learning theory can provide **bounds for parametric optimization**

We do not need to sacrifice **generalization guarantees for learned optimizers**

Practical Performance Guarantees
for Classical and Learned Optimizers

To be on Arxiv soon!

✉ rajivs@princeton.edu

🌐 rajivsambharya.github.io