Data-Driven Algorithm Design and Verification for Parametric Convex Optimization

Bartolomeo Stellato

Department of Operations Research and Financial Engineering

Department of Electrical and Computer Engineering

Department of Computer Science

PRINCETON UNIVERSITY

Northwestern IEMS Seminar — Oct 15 2024

Most applications require fast and effective decisions in real-time

Real-time optimization can help us

decisions

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

parameters

objective f: energy consumption, costs
constraints C: dynamics, physical limits

re-planning in real-time is the key to effective decision-making

How do we solve such problems?

First-order methods are now widely popular…

use only first-order information (e.g., gradients)
to solve optimization problems

example
projected gradient descent

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

$$z^{k+1} = \underset{\substack{\uparrow\\ \text{projection}}}{\Pi_{C(x)}}\Big(\underset{\substack{\uparrow\\ \text{gradient step}}}{z^k - \theta\nabla f(z^k, x)}\Big)$$

benefits of first-order methods

✓cheap iterations

✓easy to warm-start

embedded
optimization

large-scale
optimization

…and they can solve many constrained convex problems!

Linear Programs

PDLP

Applegate, Díaz, Hinder, Lu, Lubin, O'Donoghue, Schudy (2021)

Quadratic Programs

OSQP

Stellato, Banjac, Goulart, Bemporad, Boyd (2020)

Conic Programs

SCS

SPLITTING CONIC SOLVER

O'Donoghue, Chu, Parikh, Boyd (2016)

But they can converge slowly

major issue in safety-critical applications with

real-time
requirements

limited
computing power

💡 main idea

in most applications we repeatedly solve the same problem with varying parameters

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

→

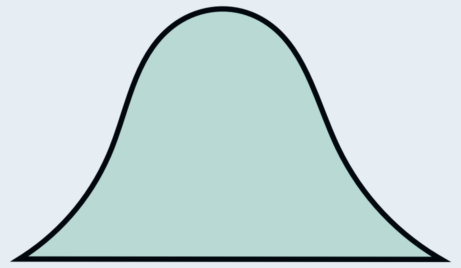
large amount of data
(e.g., instances, solutions)

First-order methods in parametric convex optimization

verification
analysis

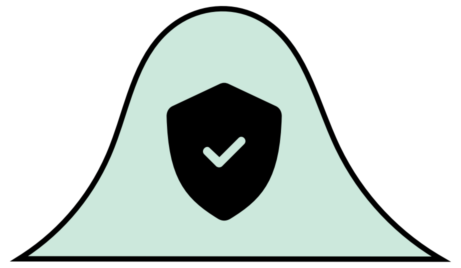
worst-case

probabilistic



design
learning

with probabilistic
guarantees



Performance verification

Convergence of first-order methods

iterations

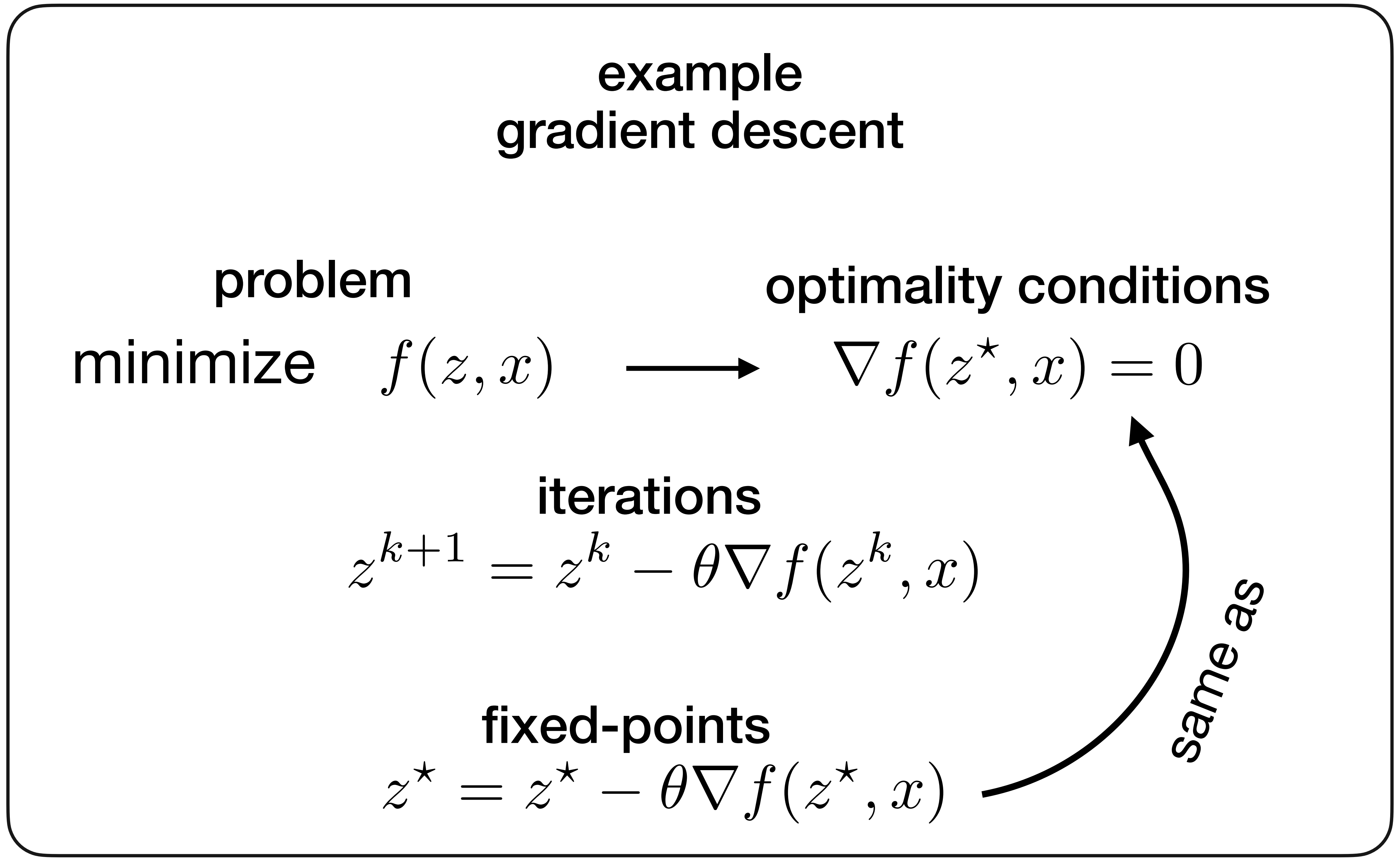
$$z^{k+1} = T(z^k, x) \quad \text{for } k = 0, 1, \ldots$$

operator

(e.g., contractive, averaged)

goal: find fixed-points

$$z^\star = T(z^\star, x)$$



example
gradient descent

problem

minimize $f(z, x)$ $\longrightarrow$

optimality conditions

$\nabla f(z^\star, x) = 0$

iterations

$z^{k+1} = z^k - \theta \nabla f(z^k, x)$

fixed-points

$z^\star = z^\star - \theta \nabla f(z^\star, x)$

performance metric

$$r^k(x) = \|T(z^{k-1}) - z^{k-1}\| = \|z^k - z^{k-1}\|$$

fixed-point residual

(converges to 0)

Classical convergence bounds via Performance Estimation

maximize $\|z^K - z^{K-1}\|$

subject to $f \in \mathcal{F}$

$z^0, z^1, \ldots, z^K$ iterates of algorithm T on f

$\|z^1 - z^0\| \leq r$

performance metric

write using interpolation conditions

function class
(e.g., strongly-convex, smooth functions)

initial condition

convex SDP
Gram matrix reformulation

$$G = \begin{bmatrix} \|z^1 - z^0\|_2^2 & (z^1 - z^0)^T g^1 & (z^1 - z^0)^T g^0 & \\ (z^1 - z^0)^T g^1 & \|g^1\|_2^2 & (g^1)^T g^0 & \\ (z^1 - z^0)^T g^0 & (g^1)^T g^0 & \|g^0\|_2^2 & \\ & & & \ddots \end{bmatrix}$$

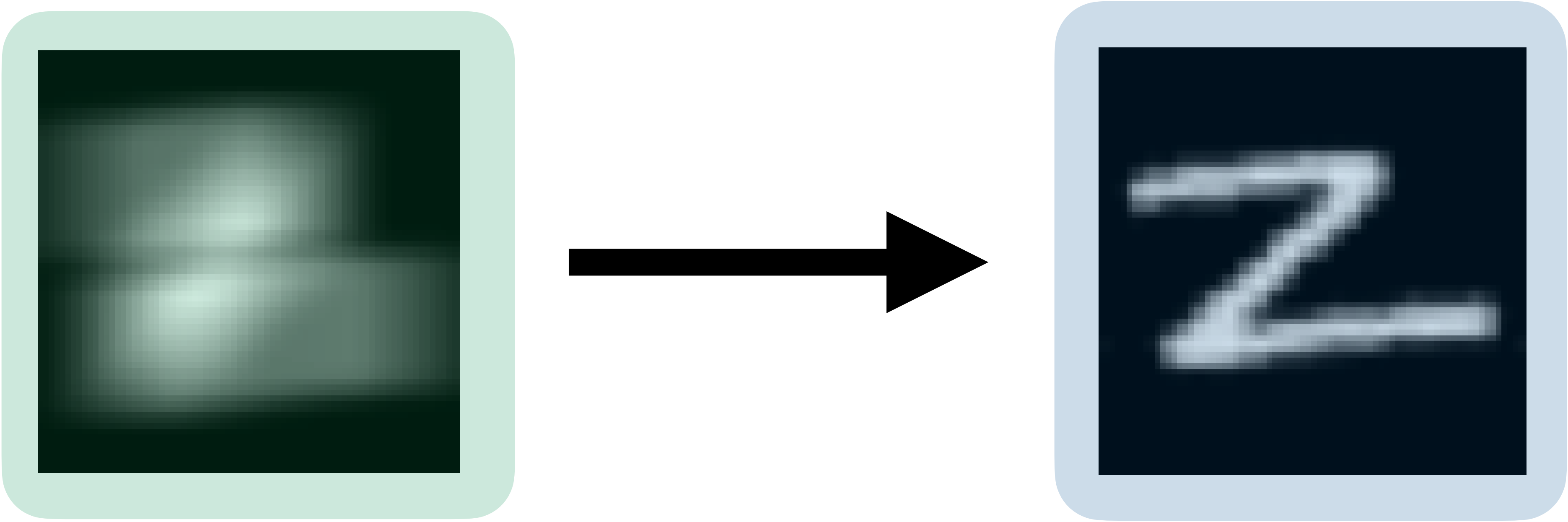
gradients

independent from iterate dimensions

Drori and Teboulle (2014), Taylor, Hendrickx, Gilneur (2017), and many others including Ryu, De Klerk, Grimmer…

Classical worst-case convergence bounds can be very loose

image deblurring problem
emnist dataset

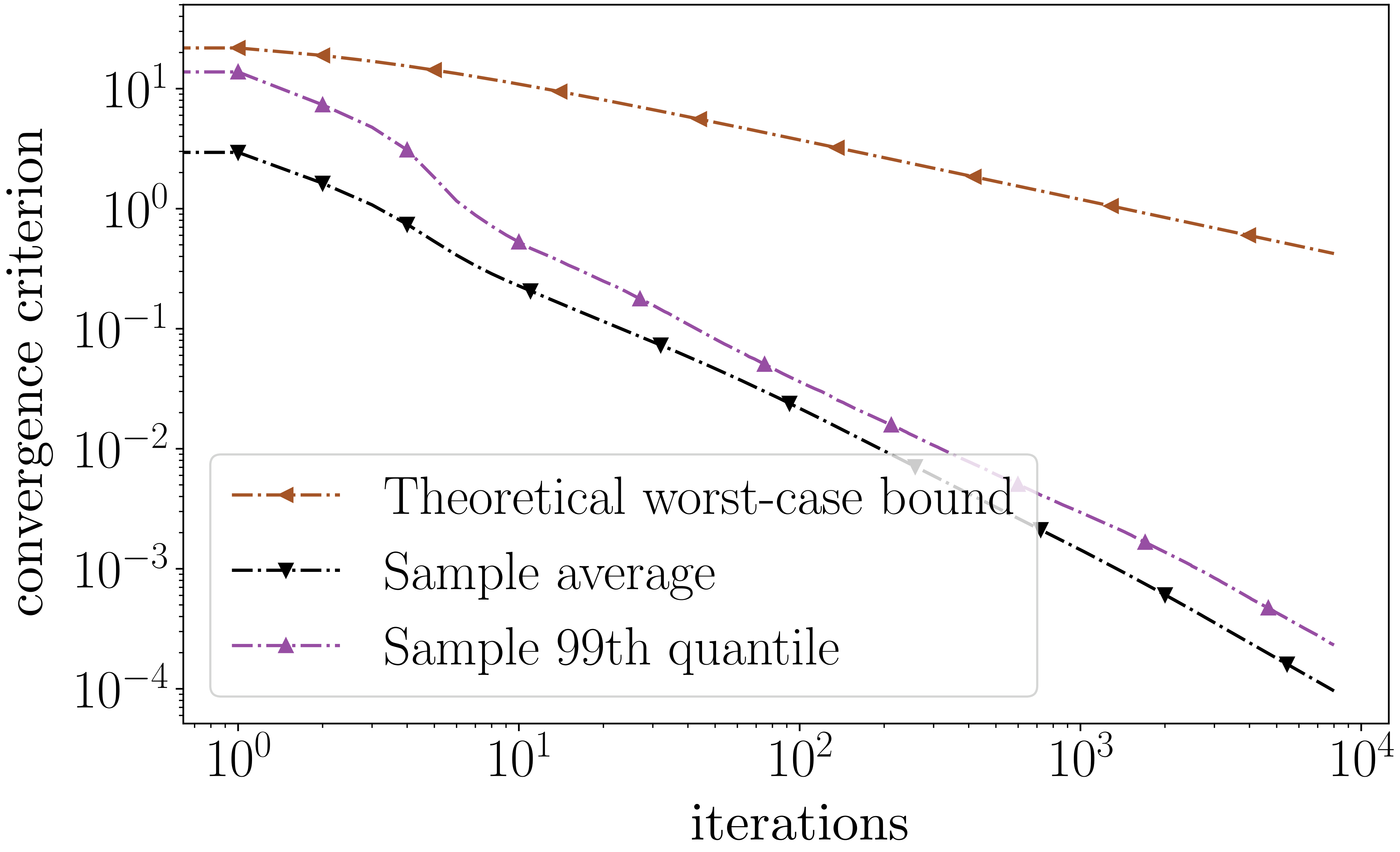


minimize $\|Az - x\|_2^2 + \lambda\|z\|_1$

subject to $0 \leq z \leq 1$

blurred image

deblurred image

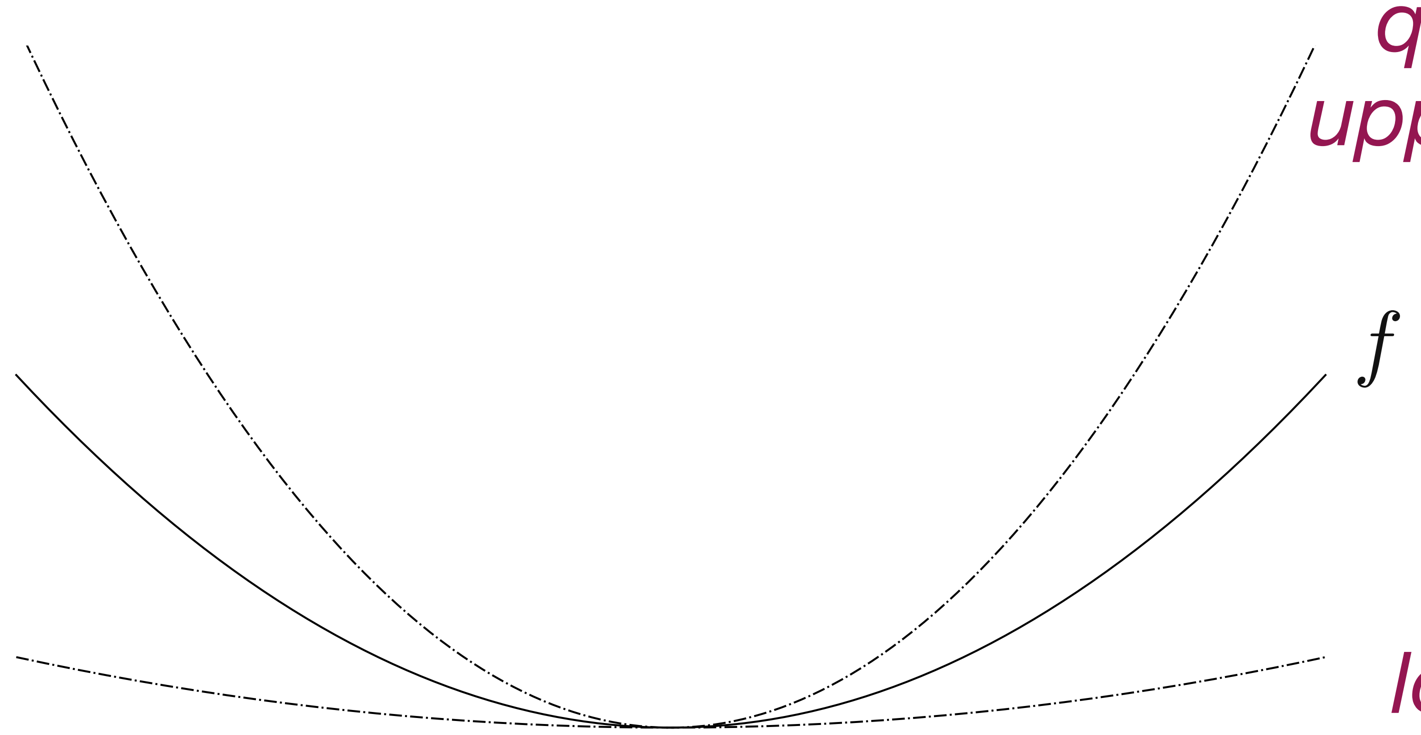


why are worst-case bounds pessimistic?

Issues with classical convergence analysis

general function classes

(f is strongly convex and smooth...)



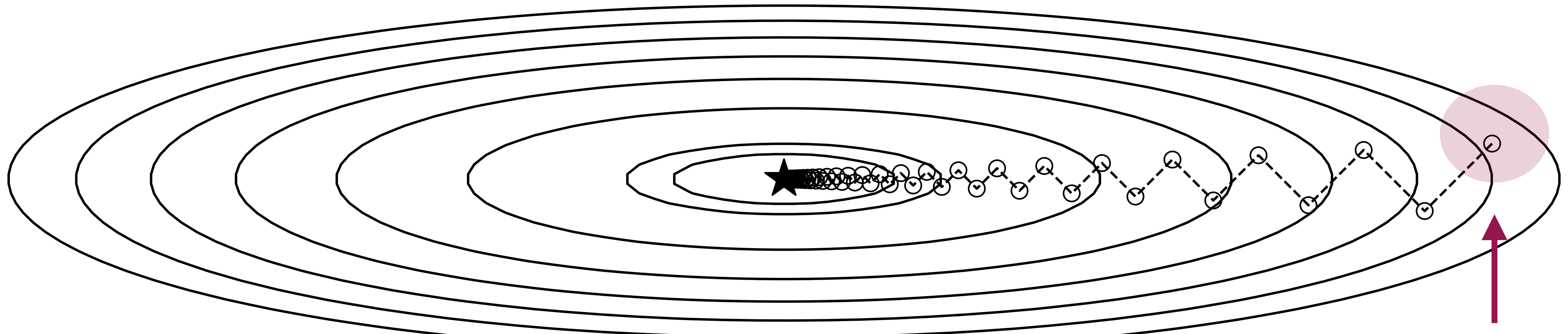
quadratic upper bound

f

quadratic lower bound

we may never encounter that function

pessimistic bounds



worst-case initial point

we may never start from that point

practical settings

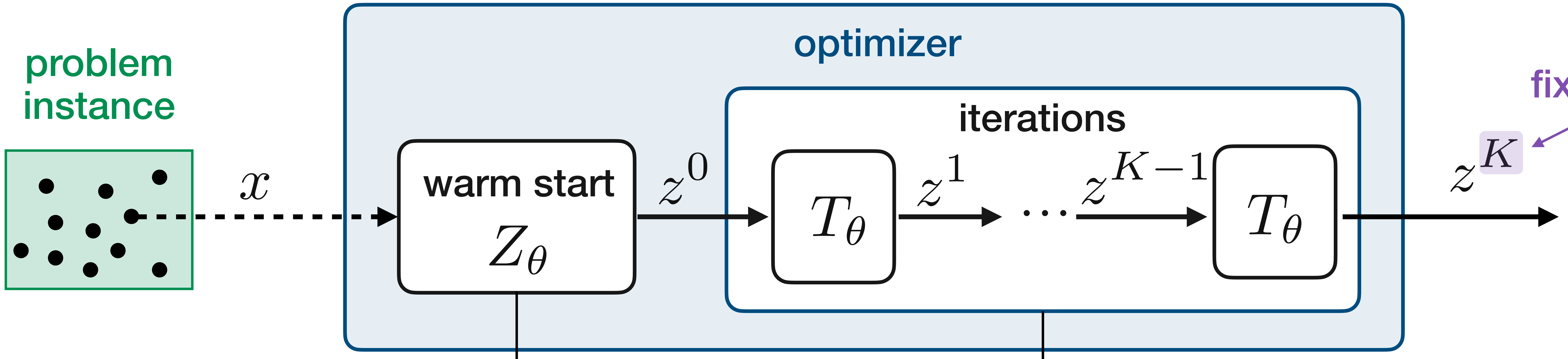
$$\begin{array}{ll} \text{minimize} & f(z, x) \\ \text{subject to} & z \in C(x) \end{array} \longrightarrow$$

same problem with varying parameters

$$x \sim \mathbf{P}$$

(unknown distribution)

Algorithms as fixed-length computational graphs



$$z^0 = Z_\theta(x) \qquad z^{k+1} = T_\theta(z^k, x)$$

algorithm parameters
(e.g., step-sizes, accelerations, warm-starts…)

example
projected gradient descent

$$z^{k+1} = \Pi_{C(x)}(z^k - \theta \nabla_z f(z^k, x))$$

Verifying the algorithm performance after K iterations

goal

estimate norm of fixed-point residual

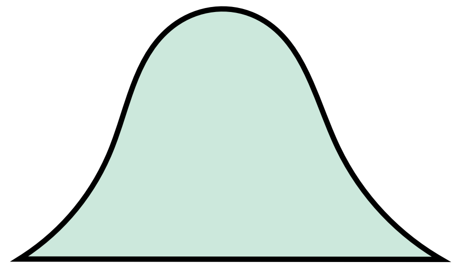
$$r^K(x) = \|z^K - z^{K-1}\|$$

worst-case

$$\max_{x \in \mathcal{X}} r^K(x) \le \epsilon$$

problem instances

convergence tolerance



probabilistic

$$\mathbf{P}\left(r^K(x) > \epsilon\right) \le \eta$$

probability bound

problem instances

convergence tolerance

Vinit Ranjan

Worst-case algorithm verification

Parametric quadratic optimization

performance metric

$$\max_{x \in \mathcal{X}} r^K(x) = \quad \text{maximize} \quad \|z^K - z^{K-1}\|$$

$$\text{subject to} \quad z^{k+1} = T_\theta(z^k, x), \quad k = 0, \dots, K-1$$

$$z^0 = Z_\theta(x), \quad x \in \mathcal{X}$$

problem instances

directly encode proximal algorithms
without interpolation inequalities

	step	verification constraint
affine *(e.g., gradient, restarts, linear system solves)*	$Dz^{k+1} = Az^k + Bx$	$Dz^{k+1} = Az^k + Bx$
elementwise maximum *(e.g., separable projections, soft-thresholding,…)*	$z^{k+1} = \max\{z^k, 0\}$	$z^{k+1} \geq 0, \quad z^{k+1} \geq z^k$ $(z^{k+1})^T(z^{k+1} - z^k) = 0$

similar to ReLU

similar constraints to neural network verification

Liu et al. (2021), Albarghouthi (2021)

Relaxing verification problem to an SDP

The verification problem is NP-hard
(by reduction from 0-1 integer programming) → convex semidefinite program relaxation

	step	verification constraint	relaxed constraint
elementwise maximum *(e.g., box projections, soft-thresholding,…)*	$z^{k+1} = \max\{z^k, 0\}$	$z^{k+1} \geq 0, \quad z^{k+1} \geq z^k$ $(z^{k+1})^T(z^{k+1} - z^k) = 0$	$z^{k+1} \geq 0, \quad z^{k+1} \geq z^k$ $\mathbf{tr}\left(\begin{bmatrix} I & -I/2 \\ -I/2 & 0 \end{bmatrix} M\right) = 0$ $M \succeq \begin{bmatrix} z^{k+1} \\ z^k \end{bmatrix}\begin{bmatrix} z^{k+1} \\ z^k \end{bmatrix}^T$

depends on iterate dimensions

Raghunathan et al. (2018), Dathathri et al. (2020),
Fazlyab et al. (2020), Chen et al. (2022), Brown et al. (2022)

Unconstrained QP

Exact SDP reformulation

minimize $\quad (1/2)z^TPz + x^Tz$

parameters

verification problem

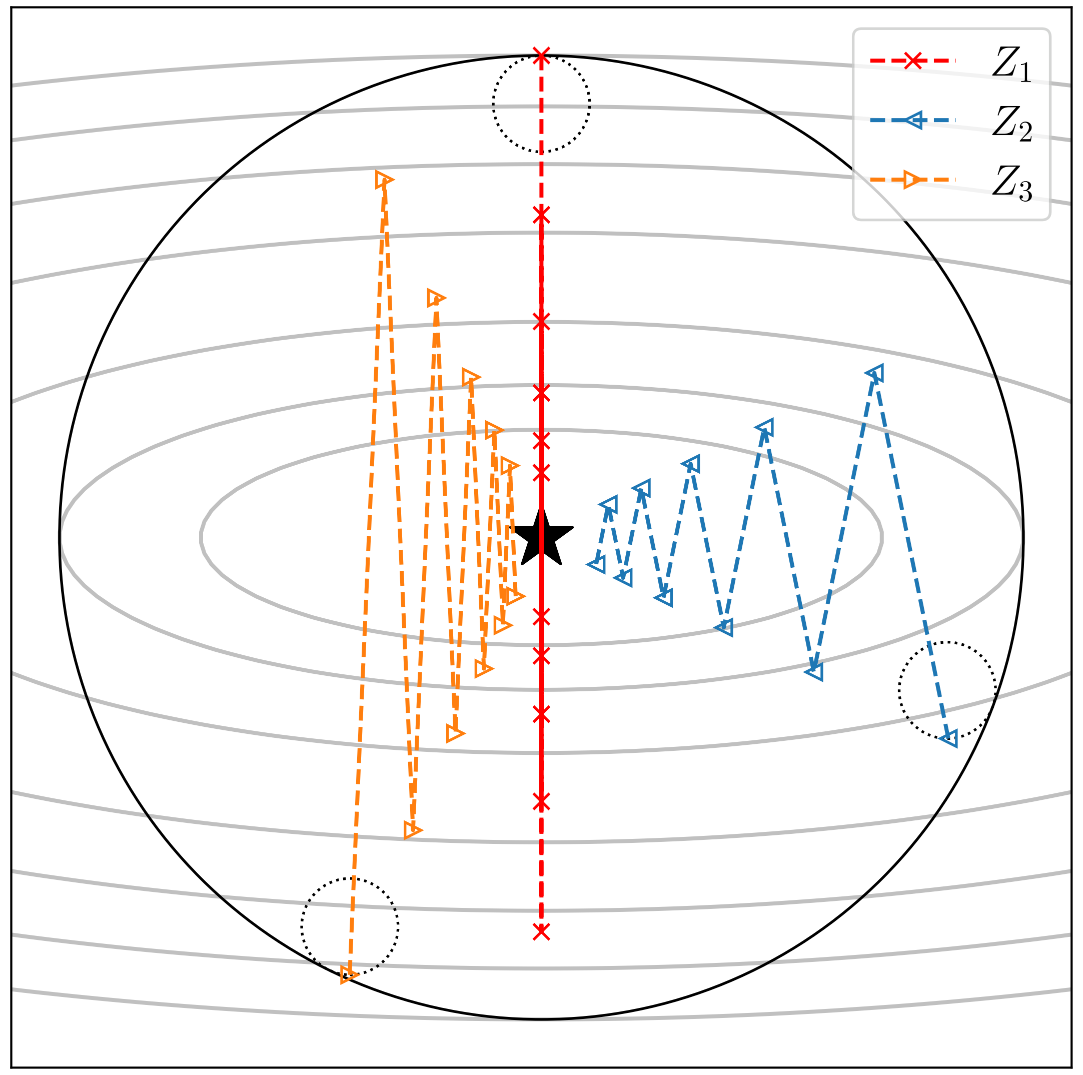
$$\begin{array}{ll} \text{maximize} & \|z^K - z^{K-1}\| \\ \text{subject to} & z^{k+1} = z^k - \theta(Pz^k + x), \quad k = 0, \dots, K-1 \\ & z^0 = Z_\theta(x), \quad x \in \mathcal{X} \end{array}$$

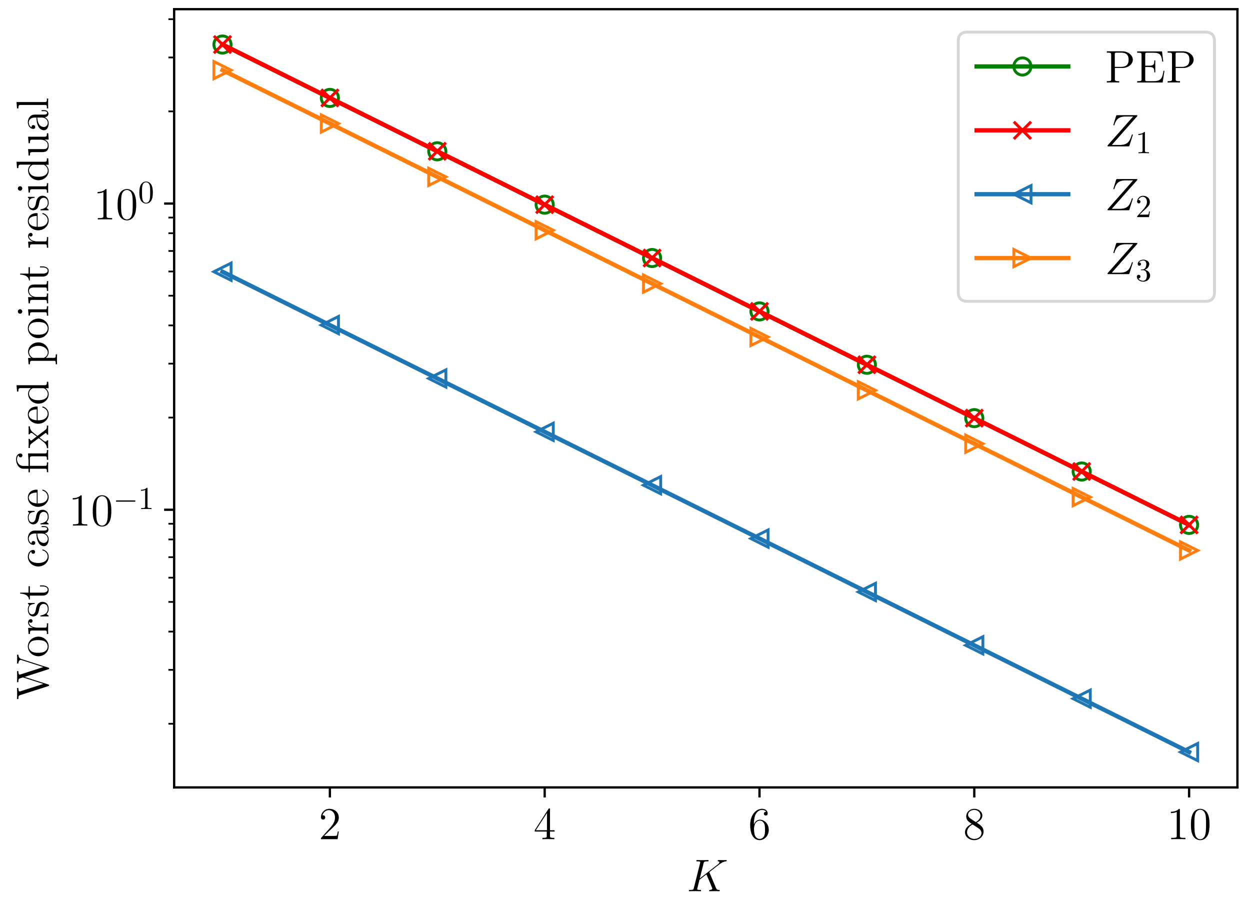
gradient descent

warm-starts

case I $\quad Z_\theta(x) = Z_1, Z_2,$ or Z_3

$x \in \mathcal{X} = \{0\}$





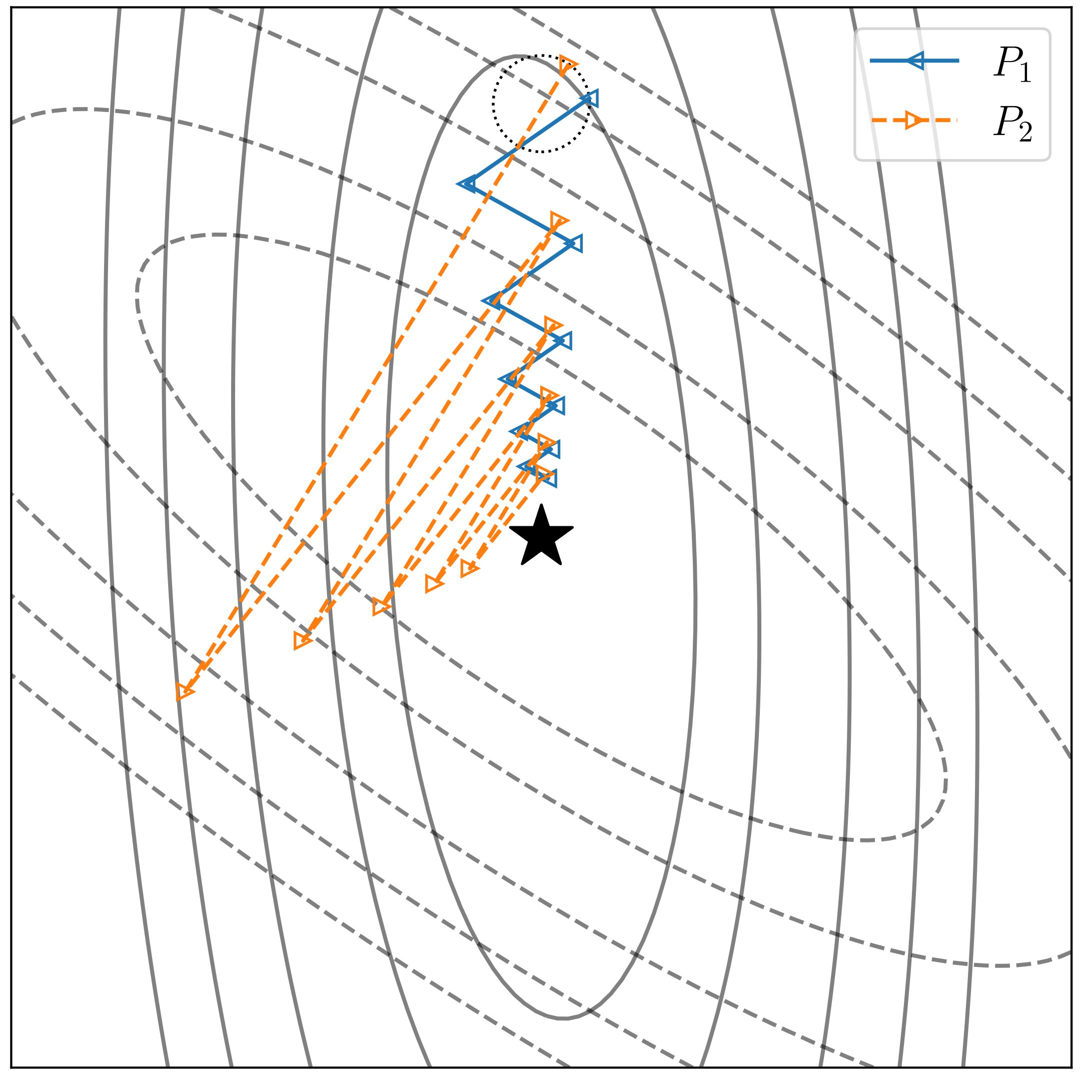
PEP cannot distinguish warm-starts

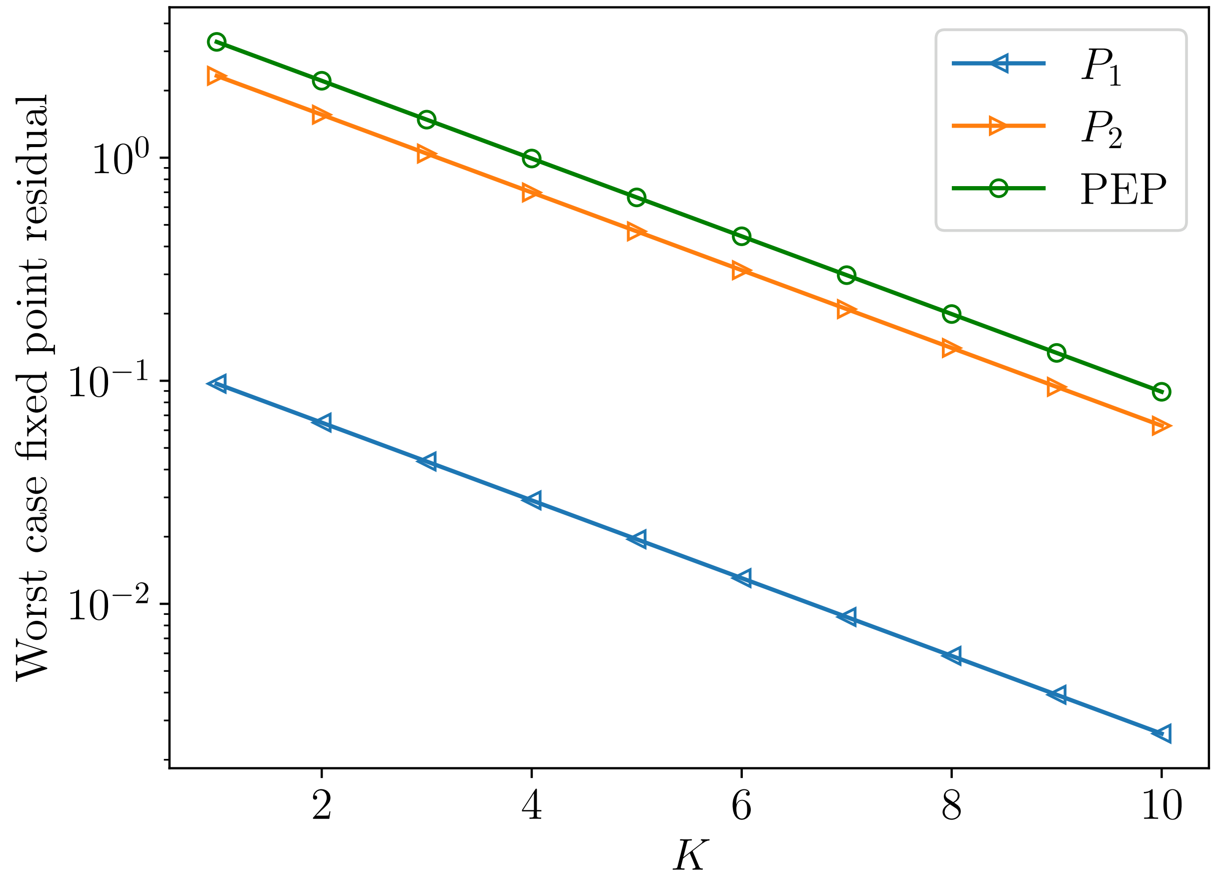
rotated functions

case II $\quad Z_\theta(x) = \{z \mid \|z - 0.9 \cdot \mathbf{1}\| \leq 0.1\}$

$x \in \mathcal{X} = \{0\}$

P_1, P_2 rotations of P





PEP cannot distinguish quadratic functions

Nonnegative least-squares verification

nonnegative least squares

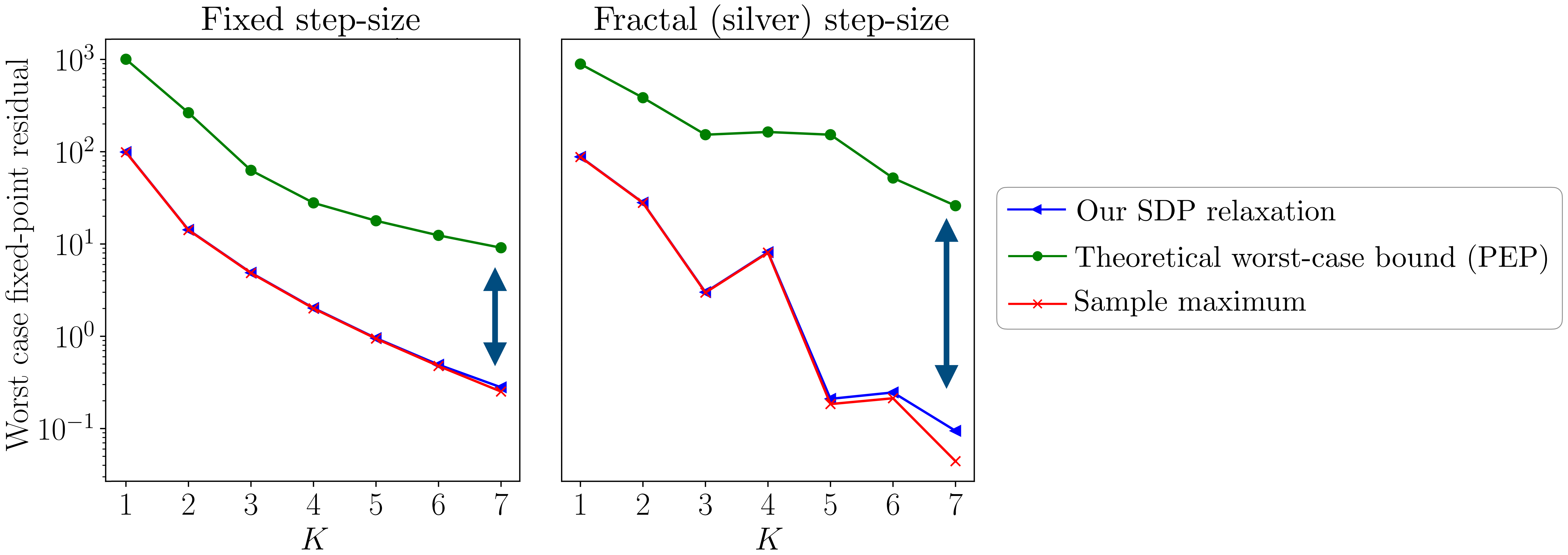
$$\begin{array}{ll} \text{minimize} & (1/2)\|Az - x\|_2^2 \\ \text{subject to} & z \geq 0 \end{array}$$

parameters

verification problem

projected gradient descent

$$\begin{array}{ll} \text{maximize} & \|z^K - z^{K-1}\| \\ \text{subject to} & z^{k+1} = \max\{(I - \theta A^T A)z^k + \theta(A^T x), 0\}, \quad k = 0, \dots, K-1 \\ & z^0 = \{0\}, \quad x \in \mathcal{X} = \{x \mid \|x - 30 \cdot \mathbf{1}\| \leq 0.5\} \end{array}$$



10x-1000x reduction
(exploiting parametric structure)

computationally more expensive than PEP
(up to 1000 seconds for these instances)

Verification of First-Order Methods for Parametric Quadratic Optimization
V. Ranjan and B. Stellato
arXiv e-prints:2403.03331 (2024)
github.com/stellatogrp/algorithm_verification

Verifying the algorithm performance after K iterations

goal

estimate norm of fixed-point residual

$$r^K(x) = \|z^K - z^{K-1}\|$$

worst-case

$$\max_{x \in \mathcal{X}} r^K(x) \leq \epsilon$$

problem instances

convergence tolerance

probabilistic

$$\mathbf{P}\left(r^K(x) > \epsilon\right) \leq \eta$$

problem instances

convergence tolerance

probability bound

Probabilistic analysis

goal

*estimate probability of
computing bad-quality solutions*

$$\mathbf{P}(r^K(x) > \epsilon)$$

any metric
(e.g., fixed-point residual)

data

issue

we don't know P! → $D = \{x^i\}_{i=1}^N$ → how can we bound the true probability?

Rajiv Sambharya

Our recipe to bound performance

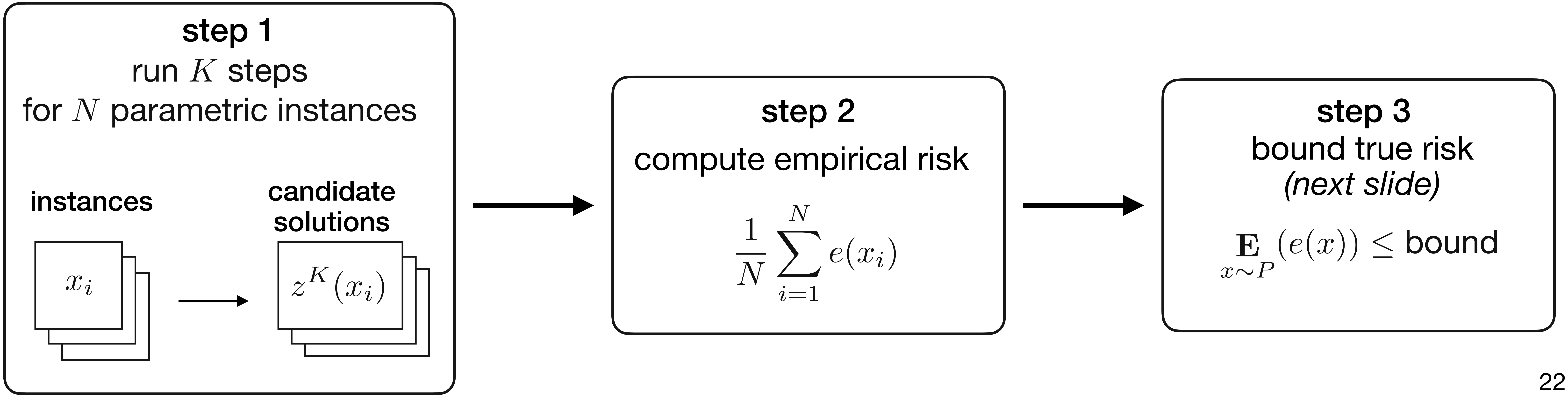
goal

estimate probability of computing bad-quality solutions

$$\mathbf{P}(r^K(x) > \epsilon) = \underset{x\sim P}{\mathbf{E}}(e(x))$$

any metric
(e.g., fixed-point residual)

error
$\mathbf{1}(r^K(x) > \epsilon)$



step 1
run K steps
for N parametric instances

instances

x_i

candidate solutions

$z^K(x_i)$

step 2
compute empirical risk

$$\frac{1}{N}\sum_{i=1}^{N} e(x_i)$$

step 3
bound true risk
(next slide)

$$\underset{x\sim P}{\mathbf{E}}(e(x)) \leq \text{bound}$$

Statistical learning gives us probabilistic guarantees

sample convergence bound

with probability $1-\delta$

$$\mathbf{P}(r^K(x) > \epsilon) = \mathop{\mathbf{E}}_{x\sim P}(e(x)) \leq \mathrm{kl}^{-1}\left(\frac{1}{N}\sum_{i=1}^{N} e(x_i) \,\middle|\, \frac{\log(2/\delta)}{N}\right)$$

true risk

inverse kl divergence *(1D convex problem)*

empirical risk

regularizer

number of instances

interpretation of bound equal to B

With probability $1-\delta$, the fixed-point residual is above ϵ after K steps

B fraction of times

Langford (2001)

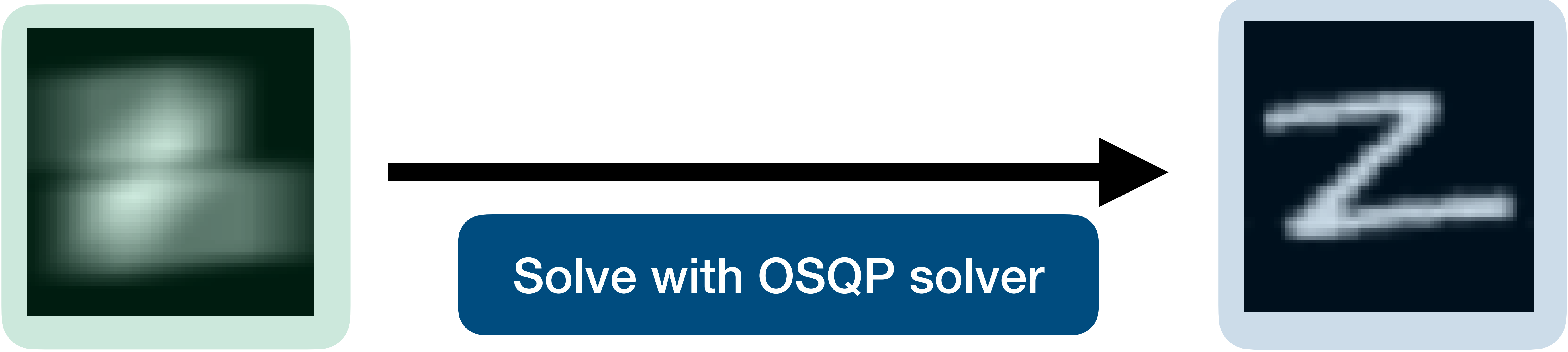
Success rates for OSQP in image deblurring

minimize $\quad \|Az - x\|_2^2 + \lambda\|z\|_1$

subject to $\quad 0 \leq z \leq 1$

blurred image

deblurred image

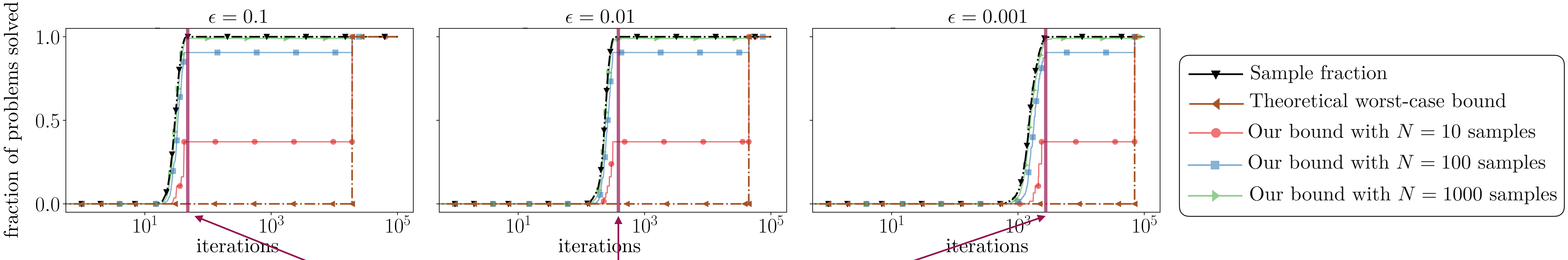


Solve with OSQP solver

fraction of problems solved

$$1 - \mathop{\mathbf{E}}_{x\sim P}(e(x)) \longleftarrow \mathbf{1}(r^K(x) > \epsilon)$$

error



iterations required
to solve all test instances

First-order methods in parametric convex optimization

verification
analysis

worst-case

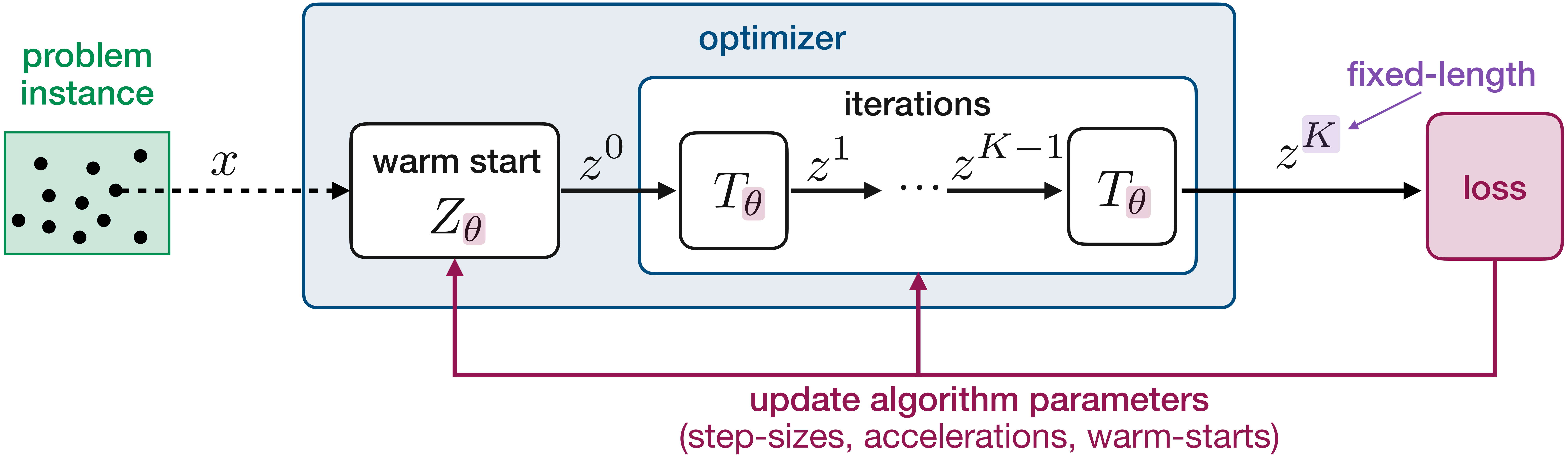
probabilistic

design
learning

with probabilistic guarantees

Algorithm design

Training algorithms as fixed-length computational graphs



example

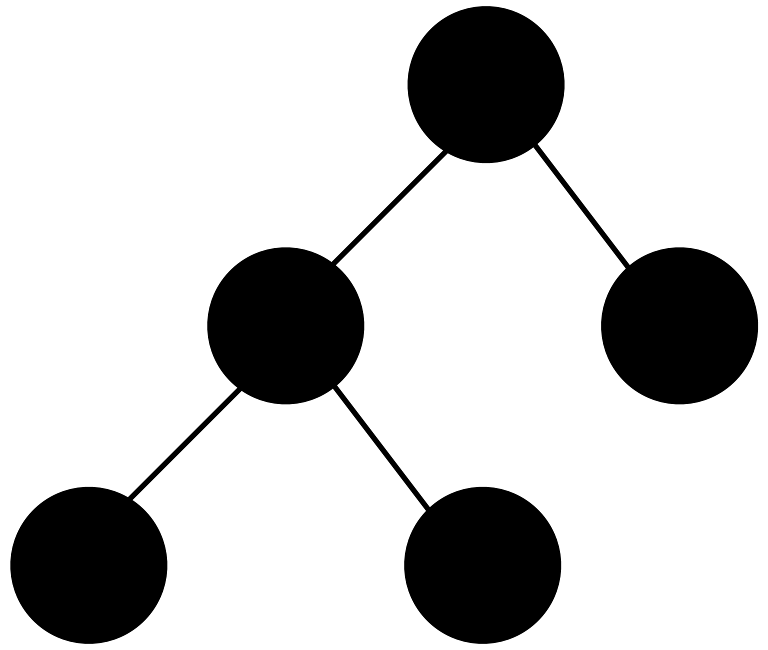
projected gradient descent

$$z^{k+1} = \Pi_{C(x)}(z^k - \theta \nabla_z f(z^k, x))$$

Learning can accelerate optimizers

Combinatorial optimization

B. Dilkina, E. Khalil, A. Lodi, P. Van Hentenryck, P. Bonami, S. Jegelka, …



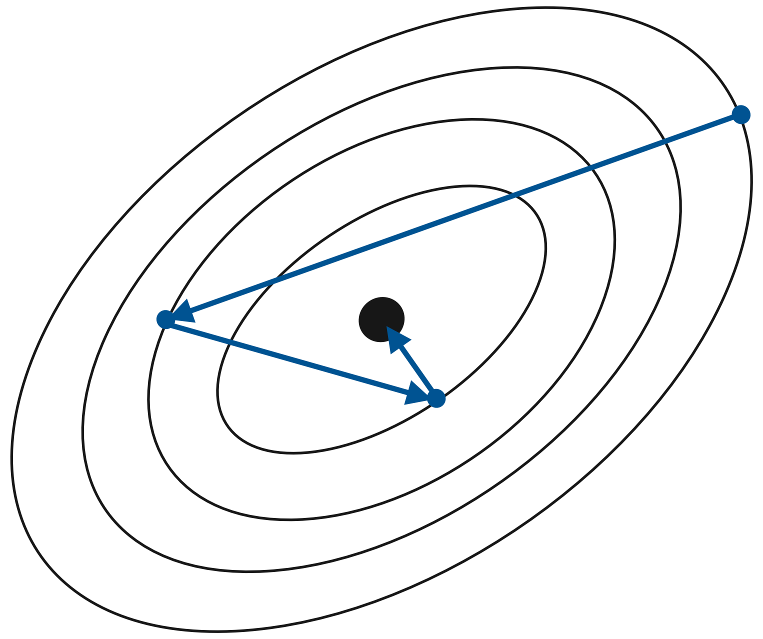
our previous contributions

The voice of optimization
D. Bertsimas, B. Stellato
Machine Learning (2021)

Online mixed-integer optimization in milliseconds
D. Bertsimas, B. Stellato
INFORMS Journal on Computing (2022)

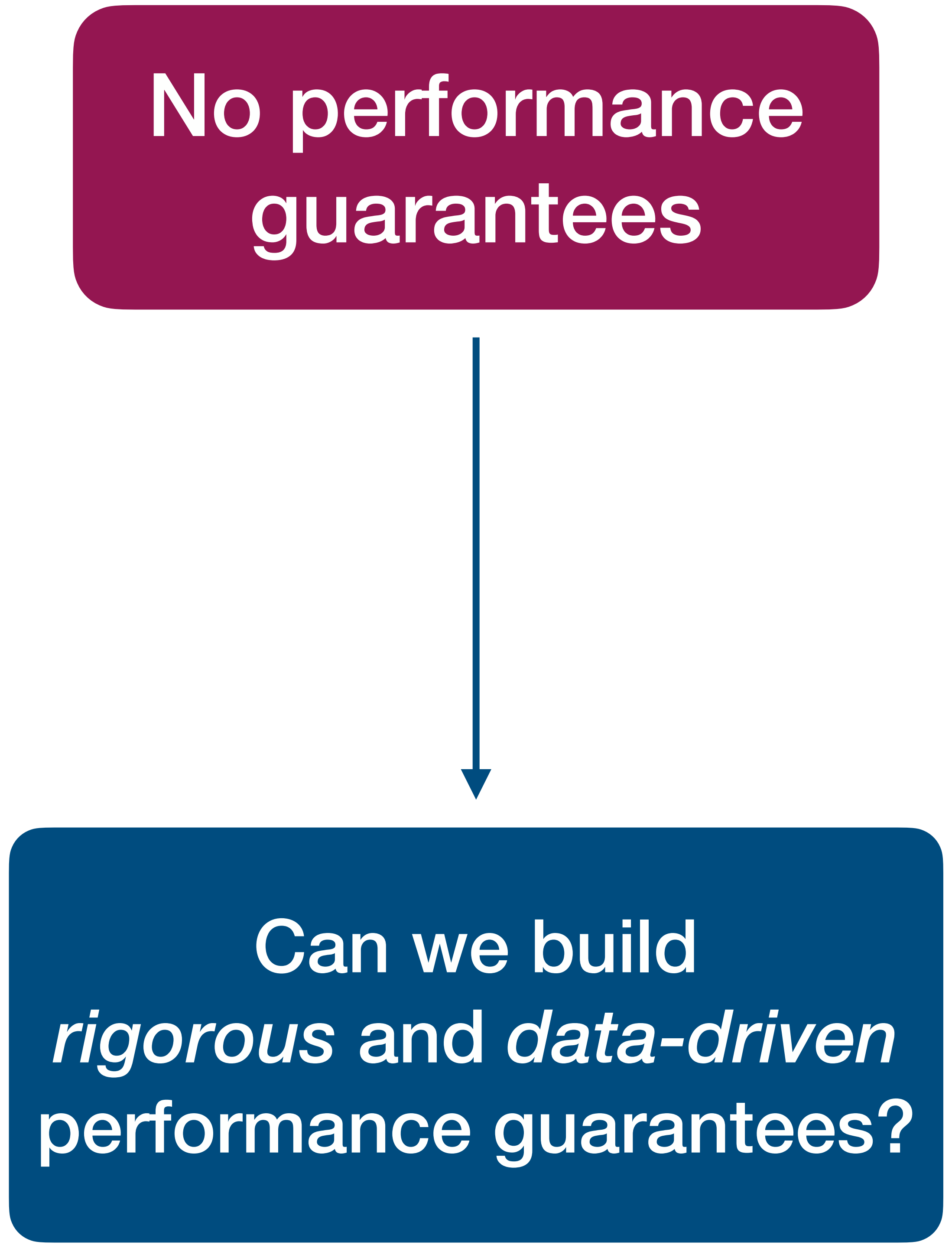
Continuous optimization

W. Yin, B. Amos, Z. Kolter, M. Andrychowicz, C. Finn, P. Van Hentenryck …



our previous contributions

Accelerating quadratic optimization with reinforcement learning
J. Ichnowski, P. Jain, B. Stellato, … et al.
NeurIPS (2021)



No performance guarantees

Can we build *rigorous* and *data-driven* performance guarantees?

Statistical learning theory for optimization algorithms

	supervised learning	learning to optimize
input		problem instance *(with parameter x)*
hypothesis	cat	residual $r_{\theta}^{K}(x)$
error	0 (1 if wrong)	$e_{\theta}(x) = \mathbf{1}(r_{\theta}^{K}(x) > \epsilon)$
guarantees	expected loss on **new data**	expected loss on **new problem instances**

algorithm parameters
(step-sizes,
accelerations,
warm-starts)

PAC-Bayes generalization bounds

learning task

$$\underset{\Theta}{\text{minimize}} \; \mathop{\mathbf{E}}_{\theta\sim\Theta} \mathop{\mathbf{E}}_{x\sim P}(e_\theta(x))$$

distribution of algorithm parameters
(step-sizes, accelerations, warm-starts)

can be anything

1. Pick *prior* Θ_0 before observing data
2. Observe data $D = \{x^i\}_{i=1}^N$
3. Learn *posterior* Θ: $\theta \sim \Theta$
4. Bound performance $\mathbf{P}^N\left(\mathop{\mathbf{E}}_{\theta\sim\Theta} \mathop{\mathbf{E}}_{x\sim P}(e_\theta(x)) \leq \hat{t}_N\right) \geq 1-\delta$

McAllester (1999), Maurer (2004)

data-driven bound

$$\hat{t}_N = \mathrm{kl}^{-1}\left(\underbrace{\frac{1}{N}\sum_{i=1}^N \mathop{\mathbf{E}}_{\theta\sim\Theta}(e_\theta(x_i))}_{\text{empirical risk}} \;\middle|\; \underbrace{\frac{\mathrm{KL}(\Theta||\Theta_0) + \log(2\sqrt{N}/\delta)}{2N}}_{\text{regularizer}}\right)$$

Rajiv Sambharya

Learning optimizers with guarantees

minimize data-driven upper bound

with stochastic gradient methods

$$\underset{\Theta}{\text{minimize}} \quad \mathrm{kl}^{-1}\left(\frac{1}{N}\sum_{i=1}^{N} \underset{\theta\sim\Theta}{\mathbf{E}}\left(e_\theta(x_i)\right) \,\middle|\, \frac{\mathrm{KL}(\Theta\|\Theta_0)+\log(2\sqrt{N}/\delta)}{2N}\right)$$

empirical risk

regularizer

derivative through
convex optimization problem
Reeb et al. (2018)

results

distribution over algorithm parameters

$$\theta \sim \Theta = \mathcal{N}(\mu, \lambda I)$$

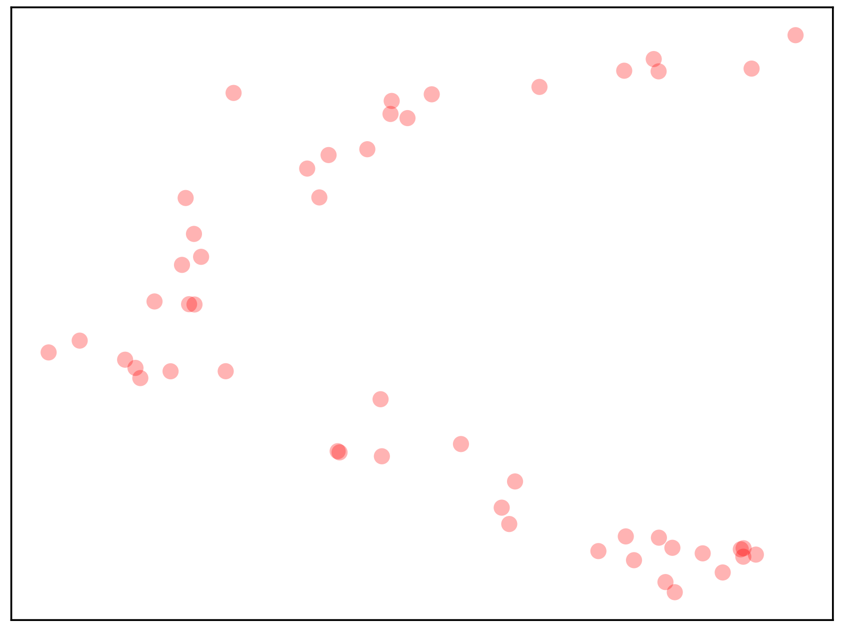
(e.g., sequence of step-sizes)

numerical performance bounds

Bottou et al (2018), Dziugaite et al (2017), Bartlett et al (2017, 2018), Jiang (2020), Majumdar et al (2021)

Robust Kalman Filtering with learned warm starts

noisy trajectory



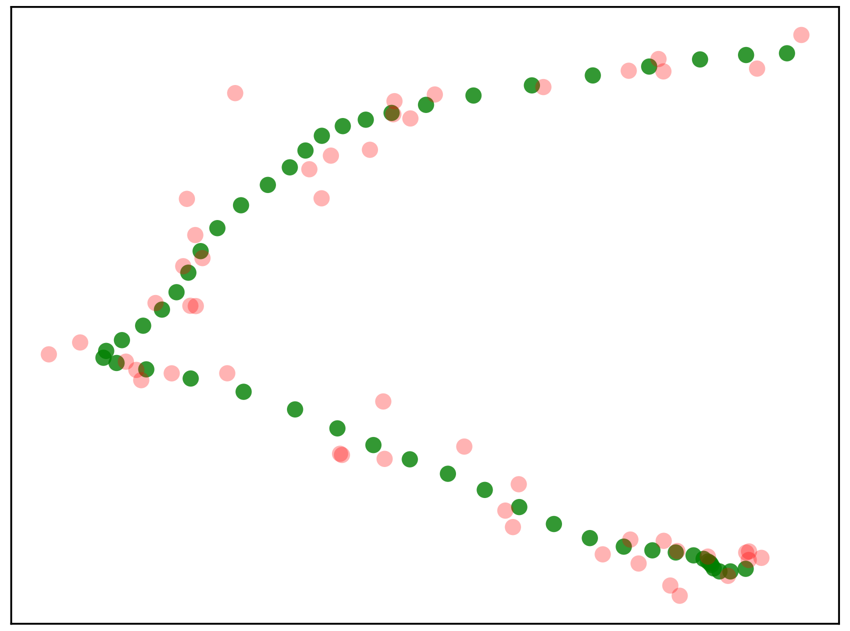
$x = \{y_t\}_{t=0}^{T-1}$

second-order cone program solver (SCS)

Huber loss

$$\begin{array}{ll} \text{minimize} & \sum_{t=0}^{T} \|w_t\|_2^2 + \psi(v_t) \\ \text{subject to} & s_{t+1} = As_t + Bw_t, \quad t = 0, \ldots, T-1 \\ & y_t = Cs_t + v_t, \qquad t = 0, \ldots, T \end{array}$$

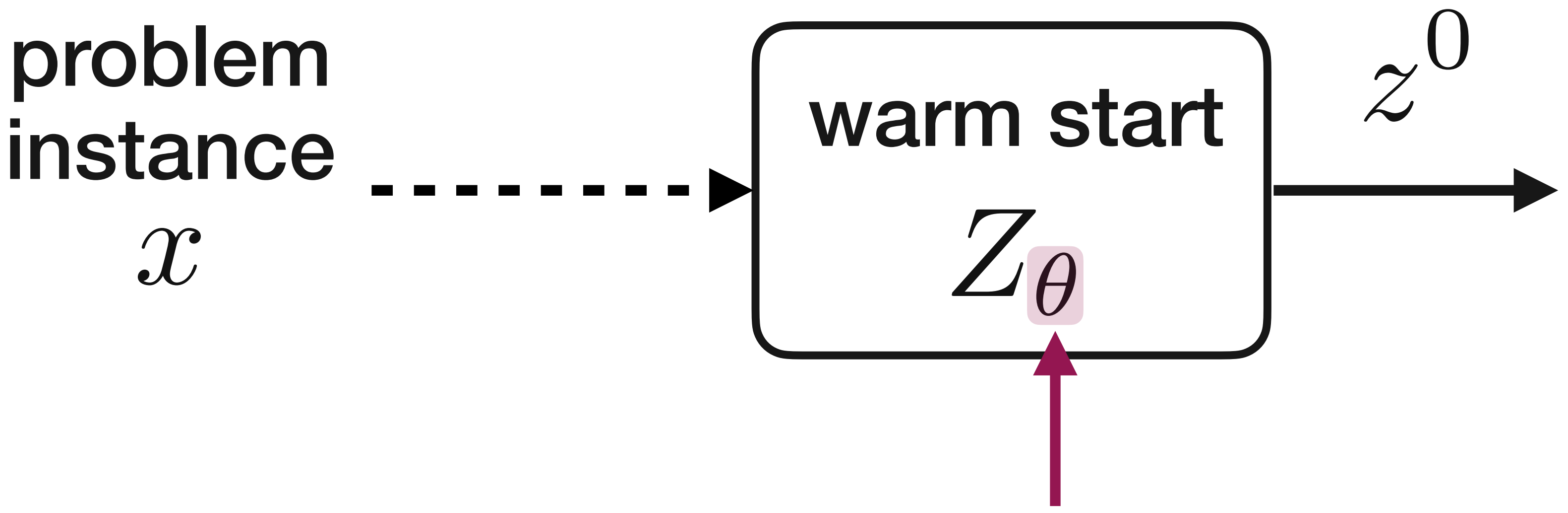
recovered trajectory



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

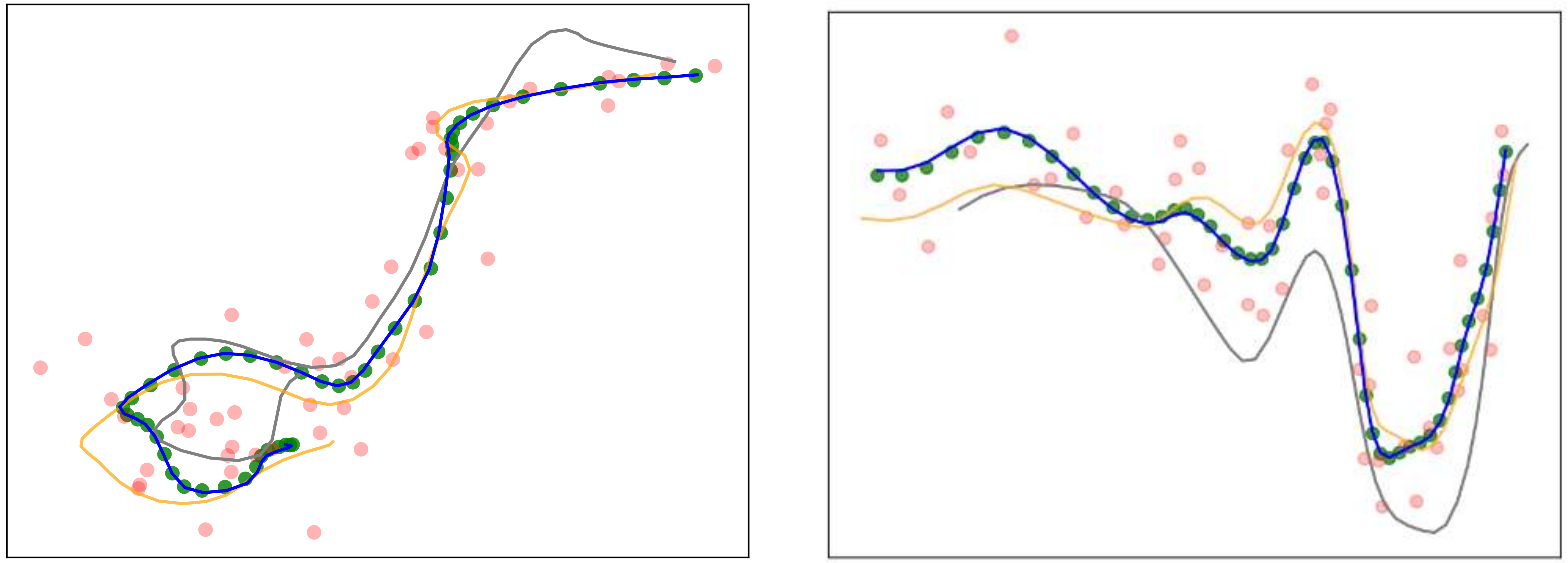
goal

learn warm-start mapping



Robust Kalman Filtering with learned warm starts

two example trajectories



points

● noisy trajectory

● optimal solution

Solution after 5 fixed-point iterations with different warm-starts

— nearest neighbor

— previous solution

— learned $K = 5$

with learning, we can estimate the state well

we also showed *warm-start specific* PAC Bayes generalization guarantees

Learning to Warm-Start Fixed-Point Optimization Algorithms
R. Sambharya, G. Hall, B. Amos, and B. Stellato
Journal of Machine Learning Research (2024)
github.com/stellatogrp/l2ws

Signal reconstruction with learned optimizer

reconstructed signal

$$\text{minimize} \quad \|Dz - x\|_2^2 + \lambda \|z\|_1$$

known dictionary

noisy signal

performance metric

normalized mean squared error

$$\mathrm{NMSE}_{\mathrm{dB}}(z) = 10 \log_{10} \left(\|z - \bar{z}\|^2 / \|\bar{z}\|^2 \right)$$

ground truth

classical algorithm (ISTA)

$$z^{k+1} = \phi_{\lambda t} \left(z^k - t2D^T (Dz^k - x) \right)$$

shrinkage operator

$$\phi_{\lambda t}(v) = \max\{v, \lambda t\} - \max\{-v, \lambda t\}$$

learned variants (e.g., ALISTA)

$$z^{k+1} = \phi_{\gamma^k} \left(z^k - \psi^k W^T (Dz^k - x) \right)$$

algorithm parameters

$$\theta = \{\gamma^k, \psi^k\}_{k=0}^{K-1}$$

Gregor and LeCun (2010), Liu et al. (2019)

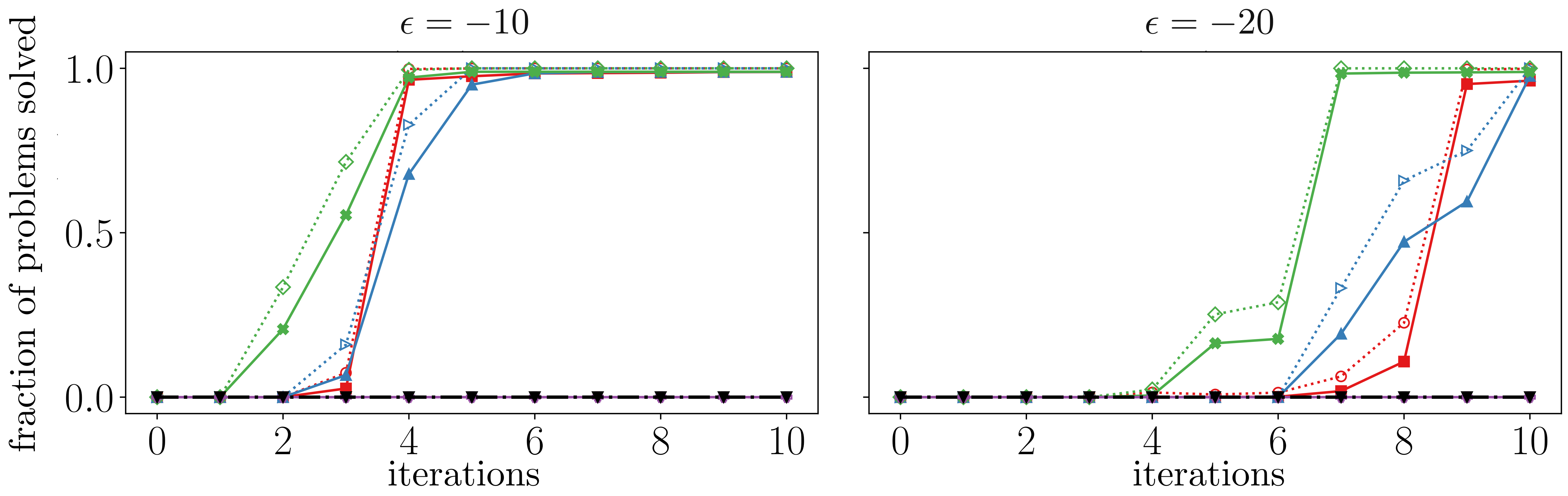
Success rates for learned optimizers in signal reconstruction

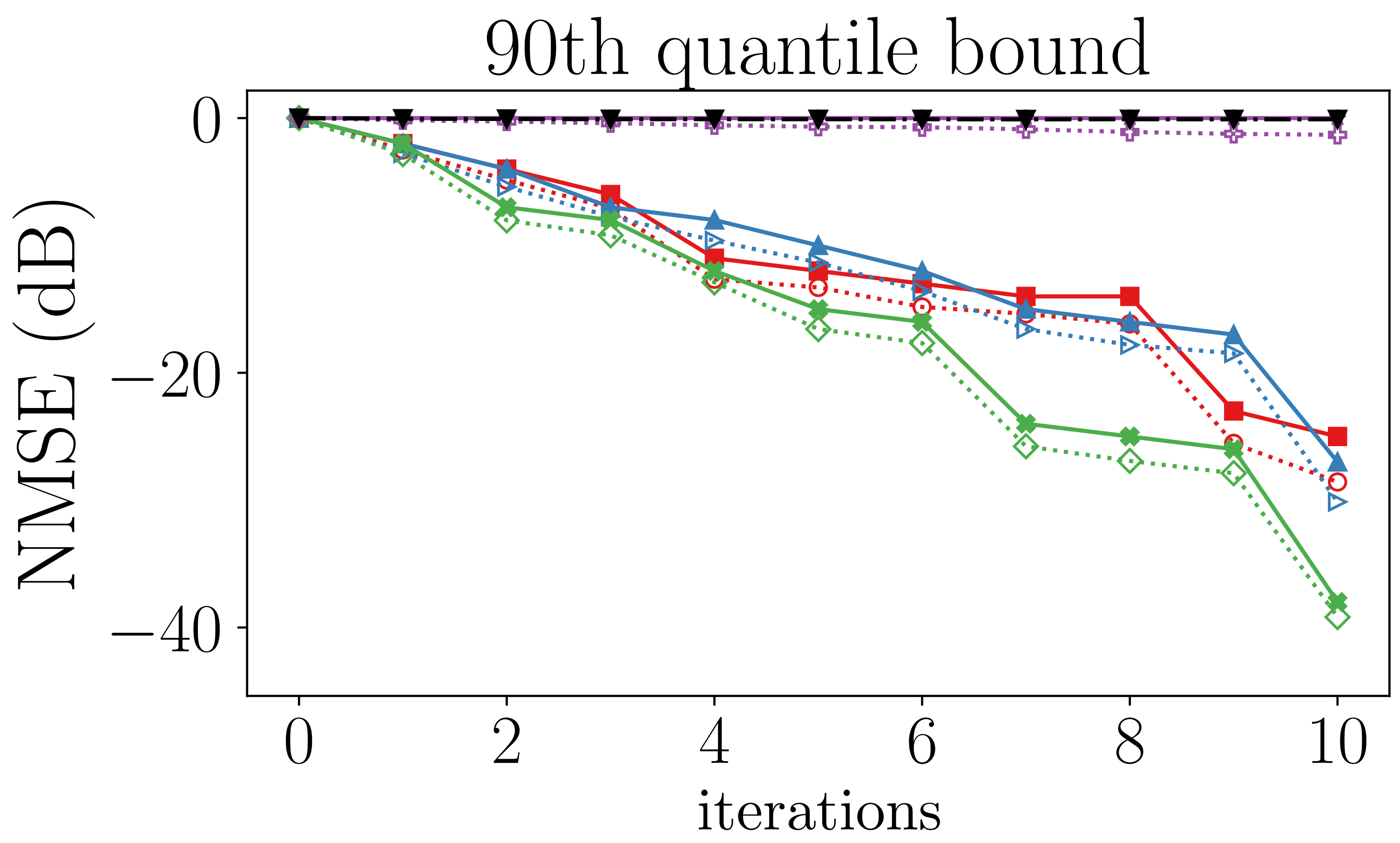
fraction of problems solved

$$1 - \underset{\theta\sim\Theta}{\mathbf{E}}\ \underset{x\sim P}{\mathbf{E}}\left(e_\theta(x)\right) \longleftarrow$$

error

$$e_\theta(x) = \mathbf{1}(\mathrm{NMSE}_{\mathrm{dB}}(z^K(x)) > \epsilon)$$





	Learned				Not learned
	LISTA	ALISTA	TiLISTA	GLISTA	ISTA
Sample bound	✚	▷	○	◇	▼
Our bound (high confidence, $1-\delta=0.999$)	★	▲	■	✖	

our bound are close to the empirical performance

learned optimizers provably perform well in just 10 iterations

Data-Driven Performance Guarantees for Classical and Learned Optimizers
R. Sambharya and B. Stellato
arxiv.org: 2404.13831 (2024)
github.com/stellatogrp/data_driven_optimizer_guarantees

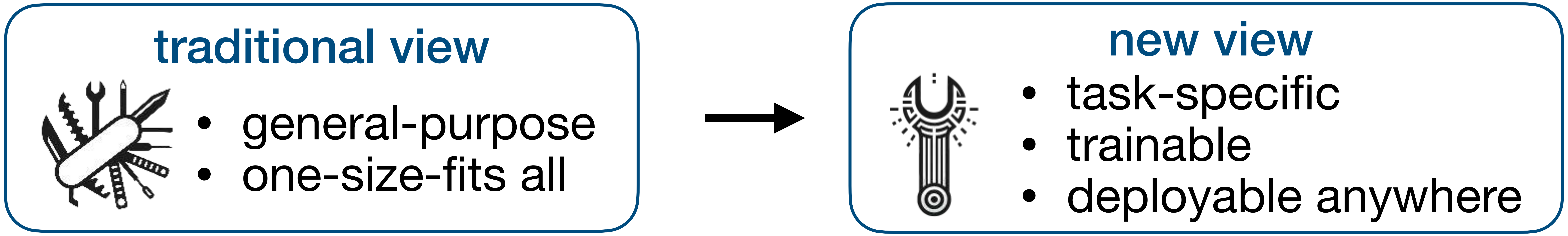
Conclusions

Algorithm Design and Verification for Parametric Convex Optimization

1. parametric structure matters

2. data can help us
   - design optimization algorithms ⚙
   - verify their performance ✓

3. we should rethink optimization algorithms



traditional view

- general-purpose
- one-size-fits all

→

new view

- task-specific
- trainable
- deployable anywhere

Backup

PAC-Bayes generalization guarantees for learned warm starts

β-contractive case $\|Tx - Ty\|_2 \leq \beta \|x - y\|_2 \quad \forall x, y$

$\beta \in (0,1)$

Theorem: for any $\gamma > 0$ with probability at least $1 - \delta$

$$\underbrace{\mathop{\mathbf{E}}_{x \sim \mathcal{X}} \ell_\theta^k(x)}_{\text{risk}} \leq \underbrace{\frac{1}{N} \sum_{i=1}^{N} \ell_\theta^k(x_i)}_{\text{empirical risk}} + \underbrace{2\beta^k \gamma + \mathcal{O}\left(\frac{\beta^k}{\gamma} (2D + 1) \sqrt{\frac{c_2(\theta) + \log(\frac{LN}{\delta})}{N}} \right)}_{\text{penalty term}}$$

bound on $\|z^\star(x)\|_2$

As the number of iterations $k \rightarrow \infty$ the penalty term goes to zero

The contractive factor β directly affects the penalty term

We combine operator theory with PAC-Bayes theory to get the bound

Computing the KL Inverse with Convex Optimization

KL divergence between Bernoulli distributions

$$\mathrm{kl}(q \,||\, p) := \mathrm{KL}(\mathrm{Bernoulli}(q), \mathrm{Bernoulli}(p))$$

Many PAC-Bayes-type bounds bound the risk implicitly

$$\mathrm{kl}(q \,||\, p) \leq c$$

empirical risk risk regularizer

Inverting the KL divergence

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