On PI controllers for updating Lagrange multipliers in constrained optimization

Jose Gallego-Posada

March 2024

gallego-posada.github.io

Today's agenda

- Constrained optimization
- Dynamics of gradient descent-ascent
- The νPI controller
- Applications of νPI in constrained optimization

"If I had been rich, I probably would not have devoted myself to mathematics."

Collaborators

Motahareh Sohrabi*

Juan Ramirez*

Helen Zhang

Simon Lacoste-Julien

Mila

Université de Montréal

CIFAR

IVADO

Constrained optimization

$$\underset{x}{\text{minimize}}\ f(\boldsymbol{x})$$

subject to $\boldsymbol{g}(\boldsymbol{x}) \leq \boldsymbol{0}_m$ and $\boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{0}_n$

Feasible set

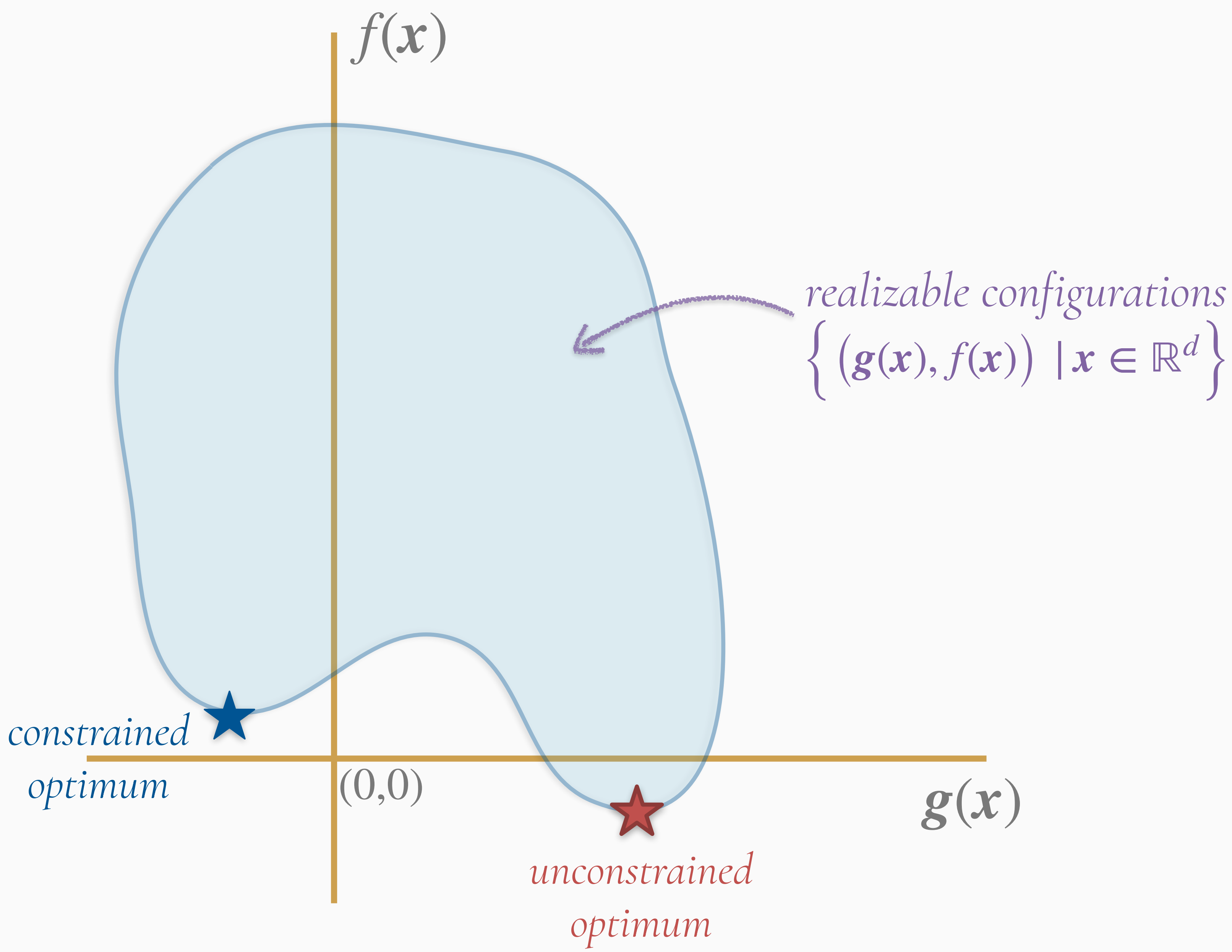
$$\mathscr{X} = \{\boldsymbol{x} \in \mathbb{R}^d \,|\, \boldsymbol{g}(\boldsymbol{x}) \leq \boldsymbol{0} \text{ and } \boldsymbol{h}(\boldsymbol{x}) = \boldsymbol{0}\}$$

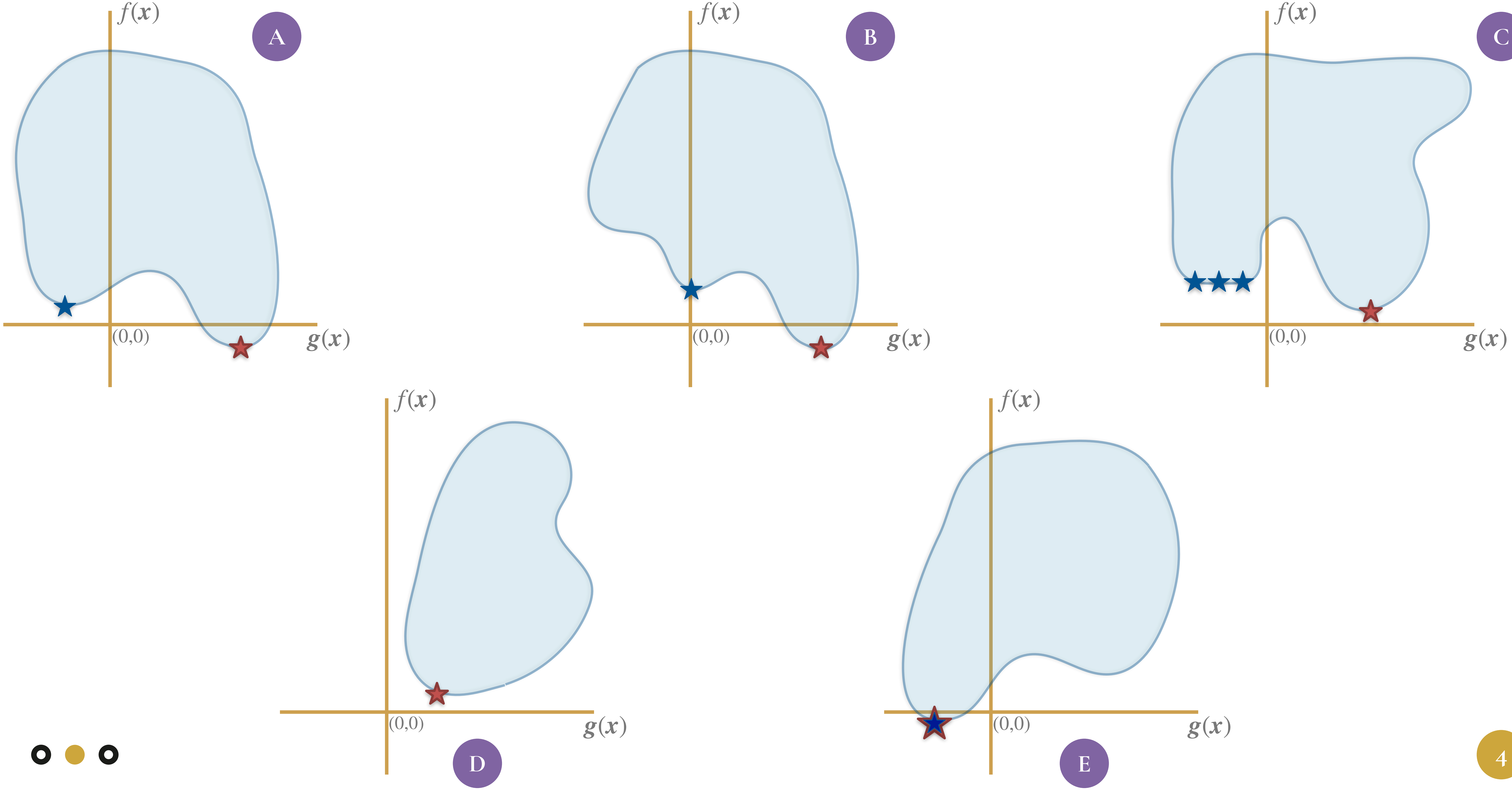
Optimality condition (necessary)

If $\boldsymbol{x}^*$ is a local minimum of f over $\mathscr{X}$, then

$$\nabla f(\boldsymbol{x}^*)^\top z \geq 0 \ \ \forall z \in \mathscr{F}(\boldsymbol{x}^*)$$

feasible directions at $\boldsymbol{x}^*$




f(x)
A
(0,0)
g(x)
f(x)
B
(0,0)
g(x)
f(x)
C
(0,0)
g(x)
f(x)
(0,0)
g(x)
D
f(x)
(0,0)
g(x)
E

Lagrangian problem

$$\min_{\boldsymbol{x}} f(\boldsymbol{x})$$
$$\text{subject to } \boldsymbol{g}(\boldsymbol{x}) \leq \mathbf{0}_m \text{ and } \boldsymbol{h}(\boldsymbol{x}) = \mathbf{0}_n$$

$\Leftrightarrow$

Lagrangian

$$\min_{\boldsymbol{x}} \max_{\boldsymbol{\lambda} \geq \mathbf{0}, \boldsymbol{\mu}} \mathfrak{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) \triangleq f(\boldsymbol{x}) + \boldsymbol{\lambda}^\top \boldsymbol{g}(\boldsymbol{x}) + \boldsymbol{\mu}^\top \boldsymbol{h}(\boldsymbol{x})$$

"Lagrange multipliers" or "dual variables"

Role of the multipliers (cf. Karush-Kuhn-Tucker necessary conditions)

$$\nabla f(\boldsymbol{x}^*) + \sum_{i=1}^{m} \lambda_i^* \nabla \boldsymbol{g}_i(\boldsymbol{x}^*) + \sum_{i=1}^{n} \boldsymbol{\mu}_i^* \nabla \boldsymbol{h}_i(\boldsymbol{x}^*) = \mathbf{0}$$

Algorithmic approach

Saddle points of the Lagrangian correspond to constrained optima. Find them!

Gradient Descent-Ascent (GDA)

Lagrangian $\min_{\boldsymbol{x}} \max_{\boldsymbol{\lambda} \geq \boldsymbol{0}, \boldsymbol{\mu}} \mathfrak{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) \triangleq f(\boldsymbol{x}) + \boldsymbol{\lambda}^\top \boldsymbol{g}(\boldsymbol{x}) + \boldsymbol{\mu}^\top \boldsymbol{h}(\boldsymbol{x})$

Algorithm

Initialize $\boldsymbol{x}_0$, $\boldsymbol{\lambda}_0 = \boldsymbol{0}$ and $\boldsymbol{\mu}_0 = \boldsymbol{0}$

Repeat

$$\boldsymbol{\mu}_{k+1} \leftarrow \boldsymbol{\mu}_k + \alpha_d \nabla_{\boldsymbol{\mu}} \mathfrak{L}(\boldsymbol{x}_k, \boldsymbol{\mu}_k, \boldsymbol{\mu}_k) = \boldsymbol{\mu}_k + \alpha \boldsymbol{h}(\boldsymbol{x}_k)$$

$$\boldsymbol{\lambda}_{k+1} \leftarrow \left[\boldsymbol{\lambda}_k + \alpha_d \nabla_{\boldsymbol{\lambda}} \mathfrak{L}(\boldsymbol{x}_k, \boldsymbol{\lambda}_k, \boldsymbol{\mu}_k)\right]^+ = \left[\boldsymbol{\lambda}_k + \alpha \boldsymbol{g}(\boldsymbol{x}_k)\right]^+$$

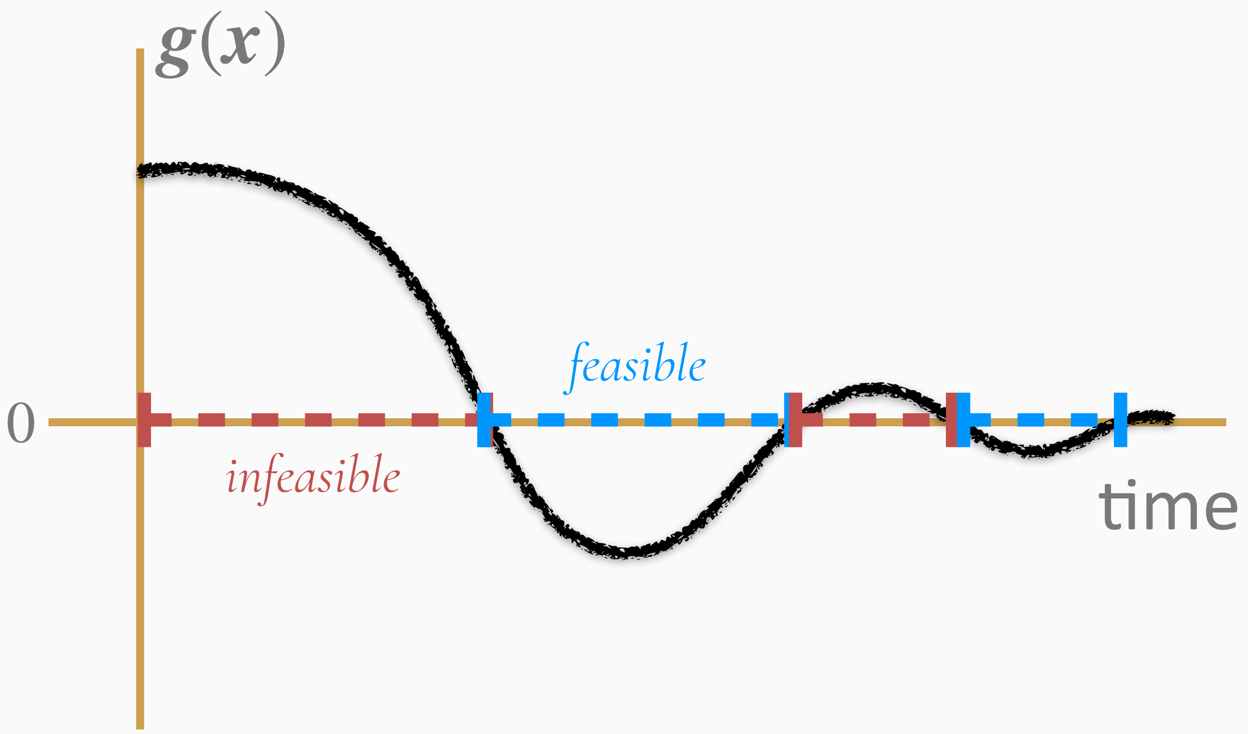
projected gradient ascent maintains non-negativity of inequality multipliers

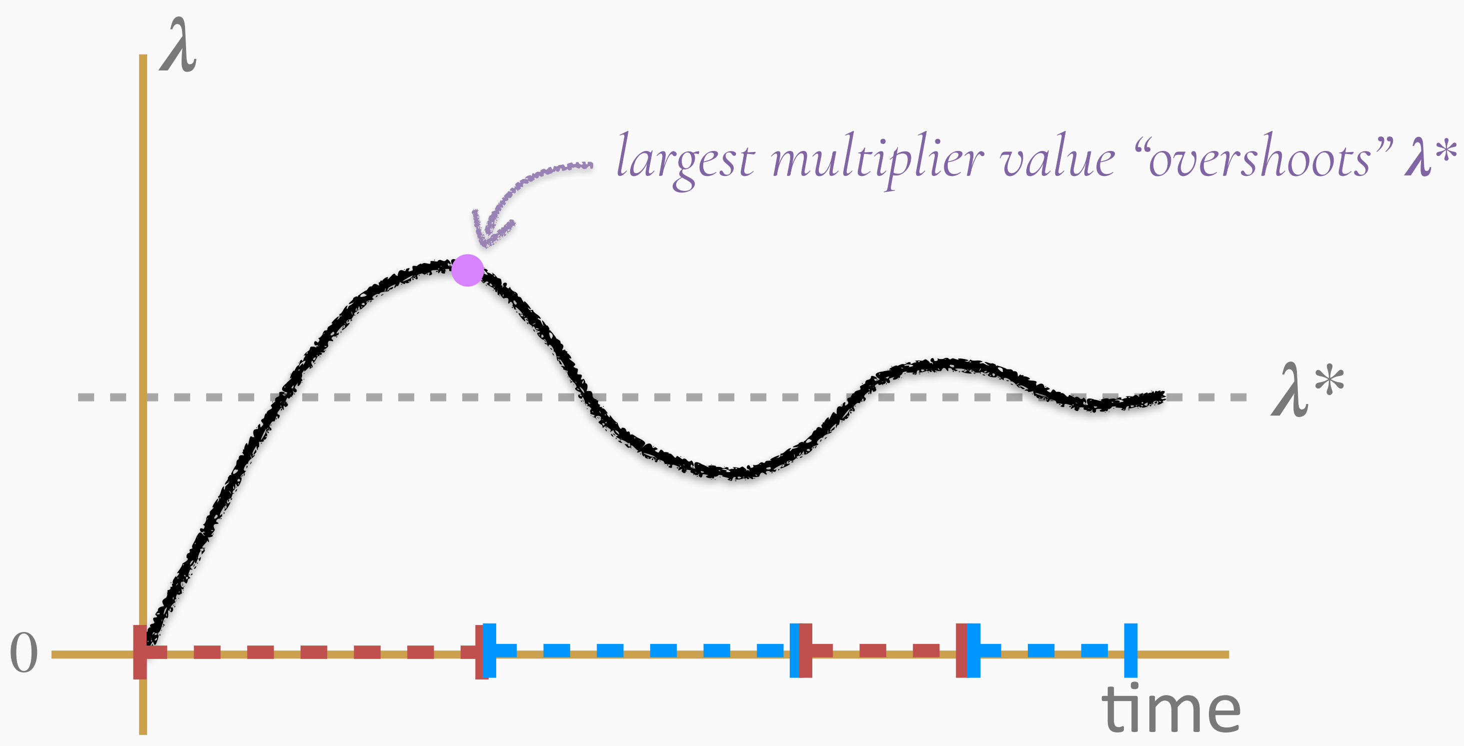
$$\boldsymbol{x}_{k+1} \leftarrow \boldsymbol{x}_k - \alpha_p \nabla_{\boldsymbol{x}} \mathfrak{L}(\boldsymbol{x}_k, \boldsymbol{\lambda}_k, \boldsymbol{\mu}_k)$$

If convergence check satisfied; **stop**

Dynamics of GDA

$$\lambda_{k+1} = \left[\lambda_k + \alpha_d \nabla_\lambda \mathfrak{L}(x_k, \lambda_k, \mu_k)\right]^+ = \left[\lambda_k + \alpha g(x_k)\right]^+$$





The multiplier accumulates/*integrates* the sequence of observed constraint violations

What we are looking for

Shortcomings of GDA

- GDA may result in overshoot and oscillations (Gidel et al. 2019; Stooke at al. 2020)
- Especially problematic in safety-related applications

Goal and scope

- **Reliable and robust** approach for solving Lagrangian optimization problems
- That **does not modify** training “recipe” for primal variables

Achieving this goal enables wider adoption of Lagrangian optimization in deep learning!

Gidel, G., Askari, R., Pezeshki, M., LePriol, R., Huang, G., Lacoste-Julien, S., and Mitliagkas, I. *Negative Momentum for Improved Game Dynamics.* In AISTATS, 2019.
Stooke, A., Achiam, J., and Abbeel, P. *Responsive Safety in Reinforcement Learning by PID Lagrangian Methods.* In ICML, 2020.

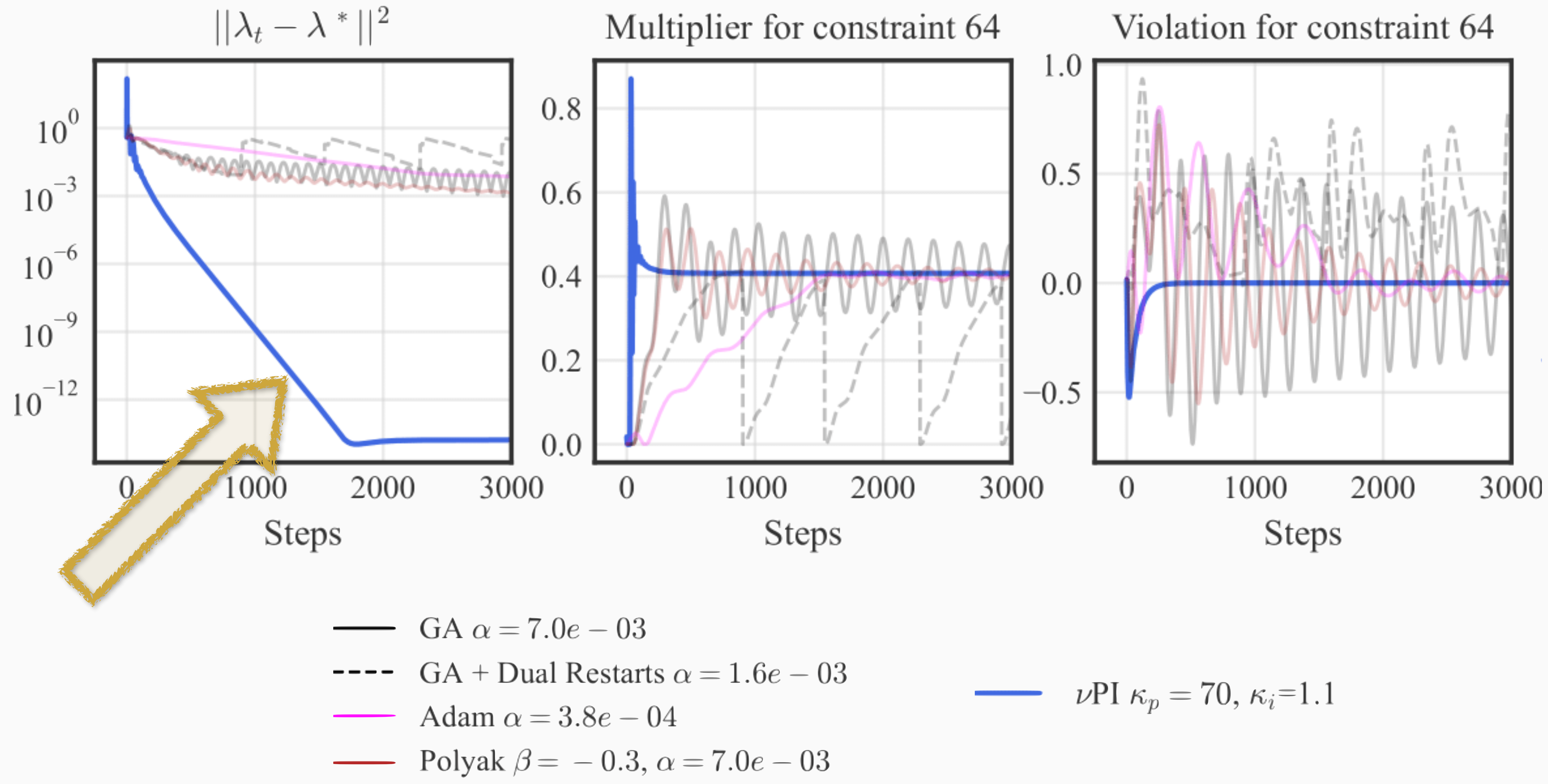
TLDR of our paper

- Stooke et al. (2020) propose **updating the Lagrange multipliers based on PID control,** improving stability on RL tasks with safety constraints.
- We provide an **optimization-oriented analysis of νPI**, our proposed PI controller
  - νPI yields stable dynamics and allows for monotonic control on the degree of overshoot
  - Conceptual insights explaining *why* using νPI helps
  - Experimental evidence in SVMs and sparsity-constrained ResNets
- We prove that **νPI generalizes standard optimization techniques** (including momentum)
  - We provide insights as to why momentum methods may aggravate the issues of GDA

Stooke, A., Achiam, J., and Abbeel, P. *Responsive Safety in Reinforcement Learning by PID Lagrangian Methods*. In ICML, 2020.

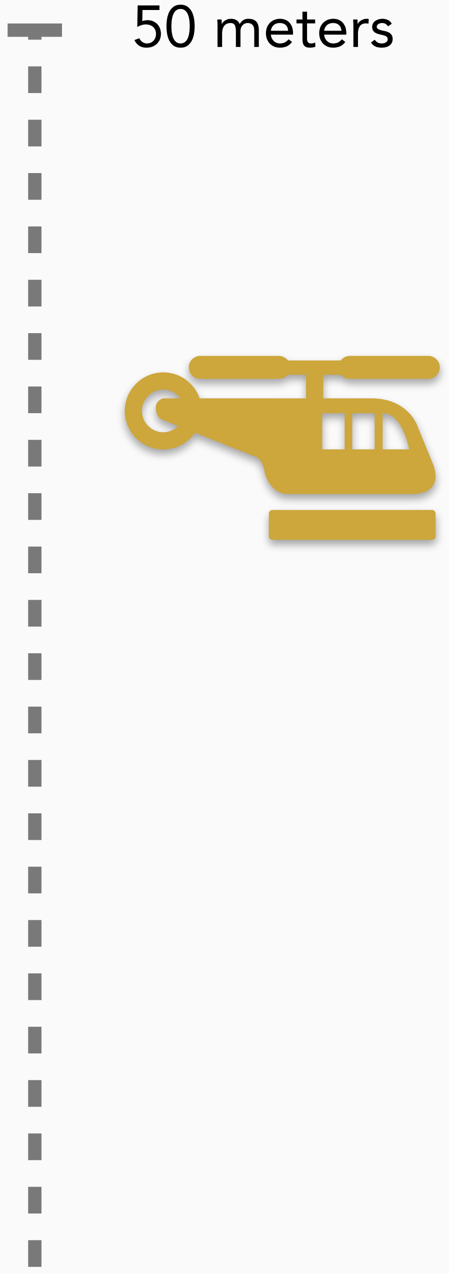
Overview of results

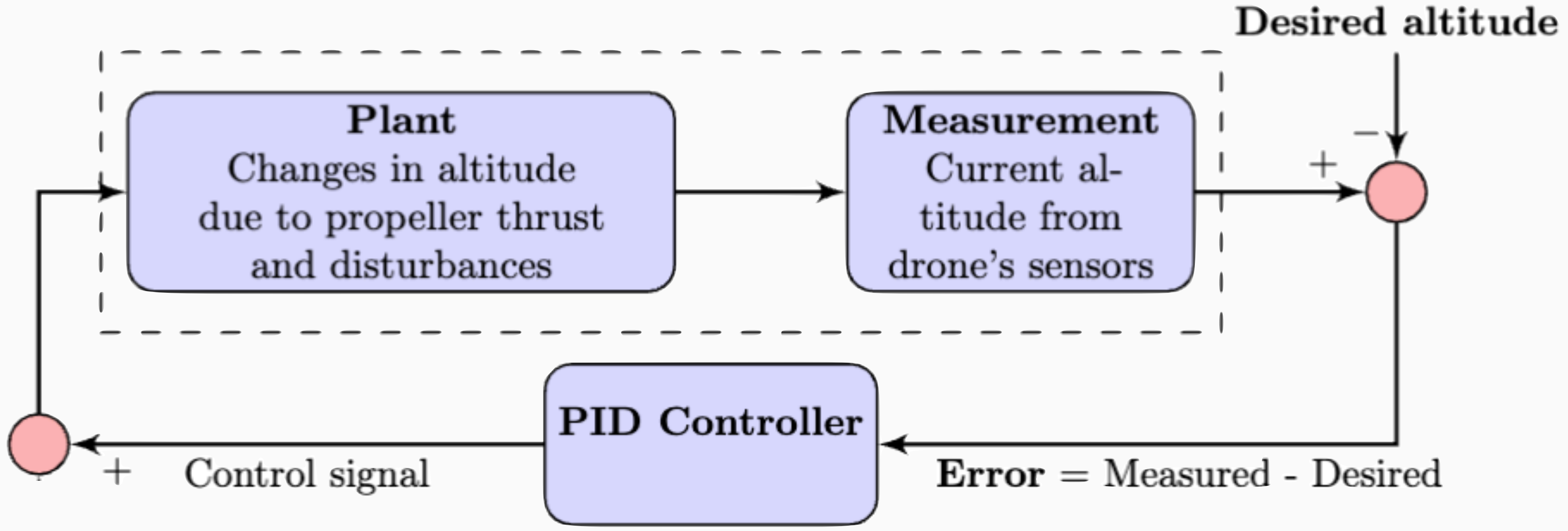
Of all attempted optimizers*, **only νPI converged successfully to the true solution!**



*Showing best hyperparameters for each optimizer after grid-search aiming to minimize the distance to λ^* after 5.000 iterations

PID control in one slide





Continuous-time (Analog)

$$u_t = \kappa_p e_t + \kappa_i \int_0^t e_\tau \, d\tau + \kappa_d \frac{de_t}{dt}$$

Discrete-time (Digital)

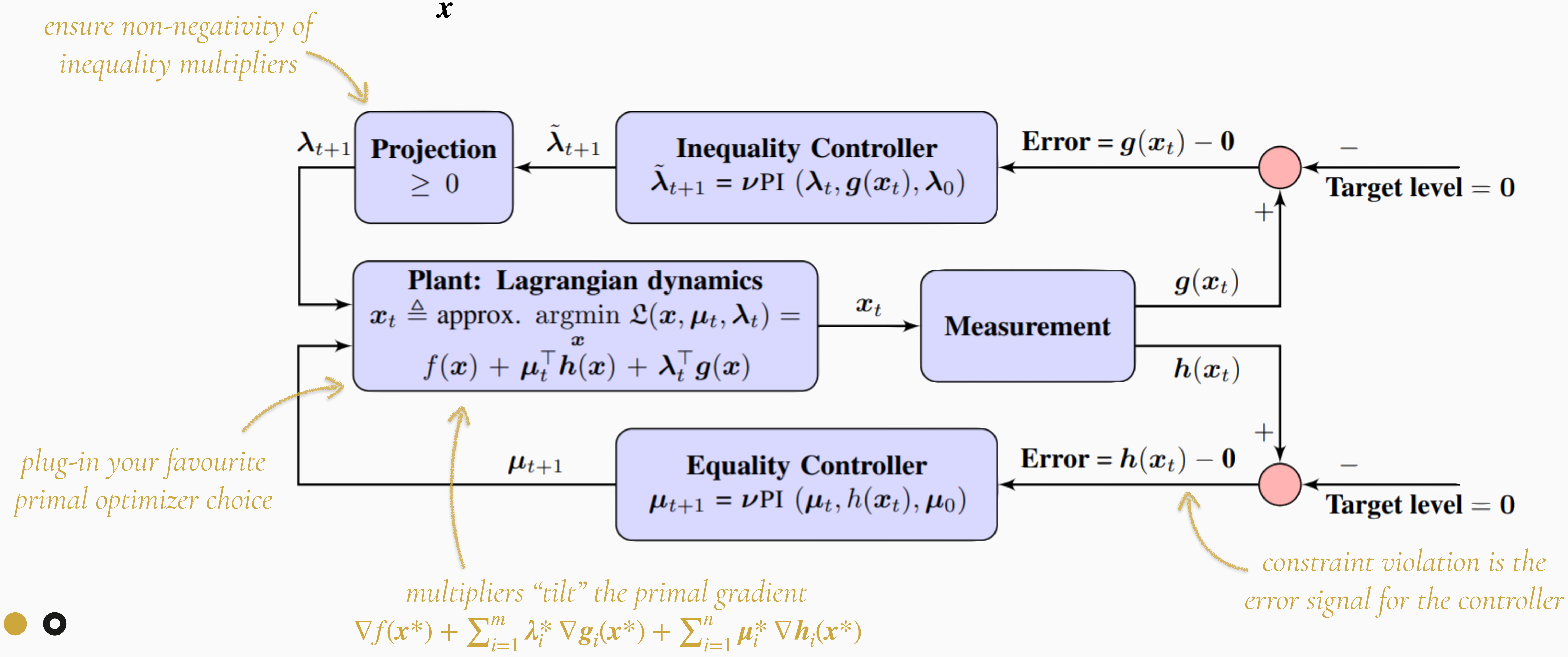
$$u_t = \kappa_p e_t + \kappa_i \sum_{\tau=0}^{t} e_\tau + \kappa_d \left(e_t - e_{t-1} \right)$$

The *Understanding PID Control* playlist by Matlab on YouTube provide an excellent, much more detailed introduction to PID control.

A control theory view of constrained optimization

Dynamical system's view of CO

$$\min_{x} f(x) \quad \text{subject to } g(x) \leq 0 \text{ and } h(x) = 0$$



νPI control for constrained optimization

Algorithm: νPI update on parameter $\boldsymbol{\theta}$

Args: EMA coefficient ν, proportional (κ_p) and integral (κ_i) gains; initial conditions $\boldsymbol{\xi}_0$ and $\boldsymbol{\theta}_0$

1. Measure the current system error $\boldsymbol{e}_t$
2. $\boldsymbol{\xi}_t \leftarrow \nu \boldsymbol{\xi}_{t-1} + (1-\nu)\boldsymbol{e}_t$ (for $t \geq 1$)
3. $\boldsymbol{\theta}_{t+1} \leftarrow \boldsymbol{\theta}_0 + \kappa_p \boldsymbol{\xi}_t + \kappa_i \sum_{\tau=0}^{t} \boldsymbol{e}_\tau$

Recursively, $\boldsymbol{\theta}_1 \leftarrow \boldsymbol{\theta}_0 + \kappa_p \boldsymbol{\xi}_0 + \kappa_i \boldsymbol{e}_0$

$$\boldsymbol{\theta}_{t+1} \leftarrow \boldsymbol{\theta}_t + \kappa_i \boldsymbol{e}_t + \kappa_p \left(\boldsymbol{\xi}_t - \boldsymbol{\xi}_{t-1}\right)$$

General case

like ∇-ascent

*new term looks at **change** in constraint satisfaction!*

$$\boldsymbol{\theta}_{t+1} \leftarrow \boldsymbol{\theta}_t + \kappa_i \boldsymbol{e}_t + \kappa_p \left(\boldsymbol{e}_t - \boldsymbol{e}_{t-1}\right)$$

Case $\nu = 0$

Two low-hanging fruits

$$\boldsymbol{\theta}_{t+1} \leftarrow \boldsymbol{\theta}_t + \kappa_i \boldsymbol{e}_t + \kappa_p \left(\boldsymbol{\xi}_t - \boldsymbol{\xi}_{t-1}\right) = \boldsymbol{\theta}_t + \kappa_i \boldsymbol{e}_t + \kappa_p (1-\nu)\left(\boldsymbol{e}_t - \boldsymbol{\xi}_{t-1}\right)$$

Suppose that the error signal is the negative gradient of a loss function f: $\boldsymbol{e}_t = -\nabla_{\boldsymbol{\theta}} f$

Gradient descent: $\kappa_p = 0$

$$\boldsymbol{\theta}_{t+1} \leftarrow \boldsymbol{\theta}_0 + \kappa_i \boldsymbol{e}_t$$

Optimistic gradient method (Popov, 1980): $\kappa_p = \kappa_i; \nu = 0$

$$\boldsymbol{\theta}_{t+1} \leftarrow \boldsymbol{\theta}_t + \kappa_i \left[\boldsymbol{e}_t + \left(\boldsymbol{e}_t - \boldsymbol{e}_{t-1}\right)\right]$$

Popov, L. D. *A modification of the Arrow-Hurwicz method for search of saddle points.* Mathematical notes of the Academy of Sciences of the USSR, 1980.

The updates of νPI

The entries of $\boldsymbol{\theta}$ can be updated in parallel,
but evolve collectively!

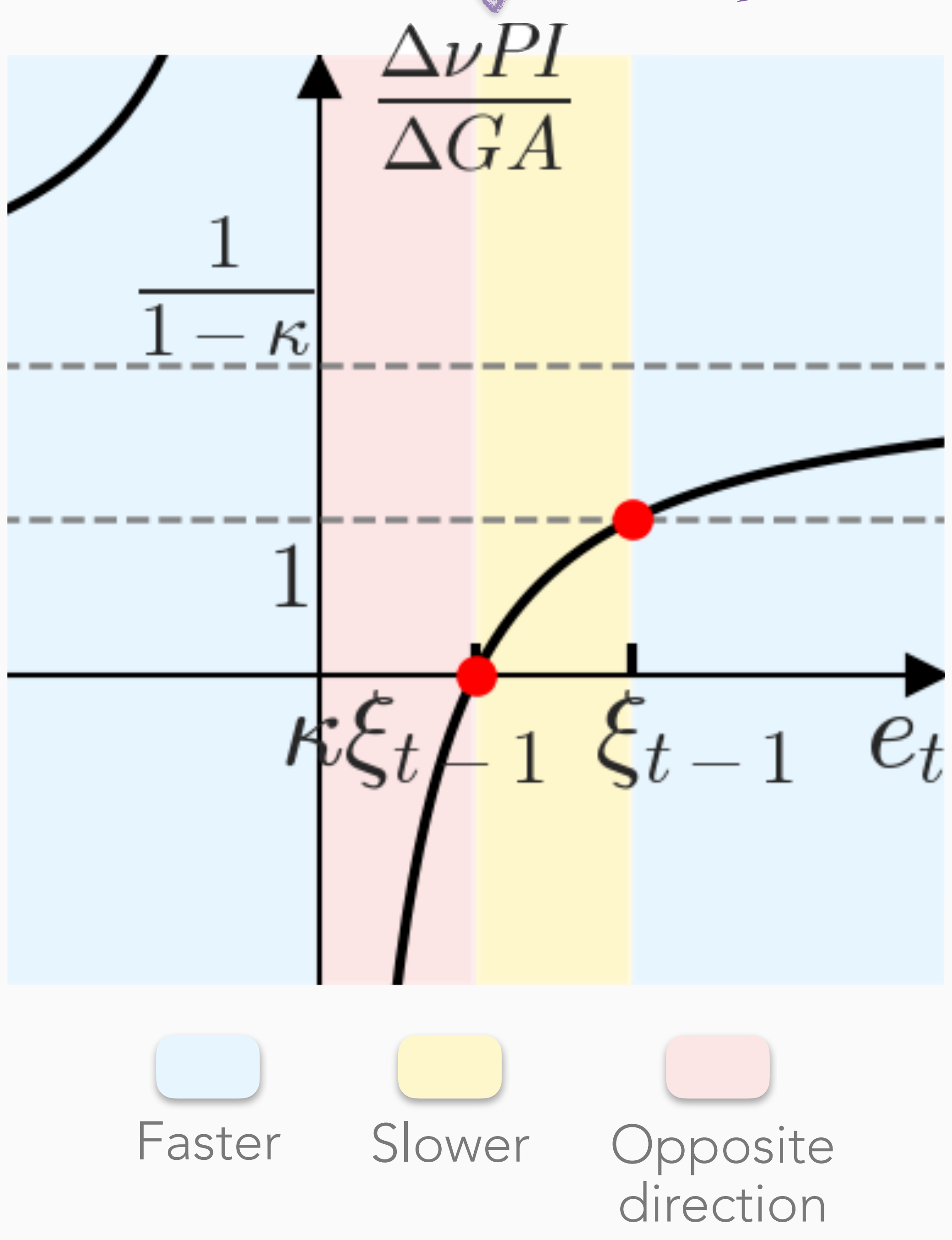
$$\theta_{t+1}^{\nu\mathrm{PI}} \leftarrow \theta_t + \kappa_i e_t + \kappa_p (1-\nu)\left(e_t - \xi_{t-1}\right)$$

$$\theta_{t+1}^{\mathrm{GA}} \leftarrow \theta_t + \kappa_i e_t$$

$$\frac{\Delta\nu\mathrm{PI}}{\Delta\mathrm{GA}} = \frac{\theta_{t+1}^{\nu\mathrm{PI}} - \theta_t}{\theta_{t+1}^{\mathrm{GA}} - \theta_t} = \frac{1}{1-\kappa}\left[1 - \frac{\kappa\xi_{t-1}}{e_t}\right]$$

constant that depends on κ_i and κ_p

"how large is the νPI update compared to GA?"



νPI generalizes momentum methods

Theorem 1

Polyak $\gamma = 0$; Nesterov $\gamma = 1$

Under the same initialization $\boldsymbol{\theta}_0$, UnifiedMomentum(α, $\beta \neq 1$, γ) is a special case of the νPI algorithm with the hyperparameter choices:

$$\nu \leftarrow \beta \qquad \boldsymbol{\xi}_0 \leftarrow (1-\beta)\boldsymbol{e}_0$$

$$\kappa_i \leftarrow \frac{\alpha}{1-\beta} \qquad \kappa_p \leftarrow -\frac{\alpha\beta}{(1-\beta)^2}[1-\gamma(1-\beta)]$$

Shen, L., Chen, C., Zou, F., Jie, Z., Sun, J., and Liu, W. A. *Unified Analysis of AdaGrad with Weighted Aggregation and Momentum Acceleration.* In IEEE TNNLS, 2018.

νPI generalizes momentum methods

Algorithm	$\boldsymbol{\xi}_0$	κ_p	κ_i	ν
UNIFIEDMOMENTUM(α, β, γ)	$(1-\beta)\boldsymbol{e}_0$	$-\frac{\alpha\beta}{(1-\beta)^2}[1-\gamma(1-\beta)]$	$\frac{\alpha}{1-\beta}$	β
POLYAK(α, β)	$(1-\beta)\boldsymbol{e}_0$	$-\frac{\alpha\beta}{(1-\beta)^2}$	$\frac{\alpha}{1-\beta}$	β
NESTEROV(α, β)	$(1-\beta)\boldsymbol{e}_0$	$-\frac{\alpha\beta^2}{(1-\beta)^2}$	$\frac{\alpha}{1-\beta}$	β
PI	$\boldsymbol{e}_0$	κ_p	κ_i	0
OPTIMISTICGRADIENTASCENT(α)	$\boldsymbol{e}_0$	α	α	0
$\boldsymbol{\nu}$PI $(\kappa_i, \kappa_p, \nu)$ in practice	$\mathbf{0}$	κ_i	κ_p	ν
GRADIENTASCENT(α)	–	0	α	0

νPI generalizes momentum methods

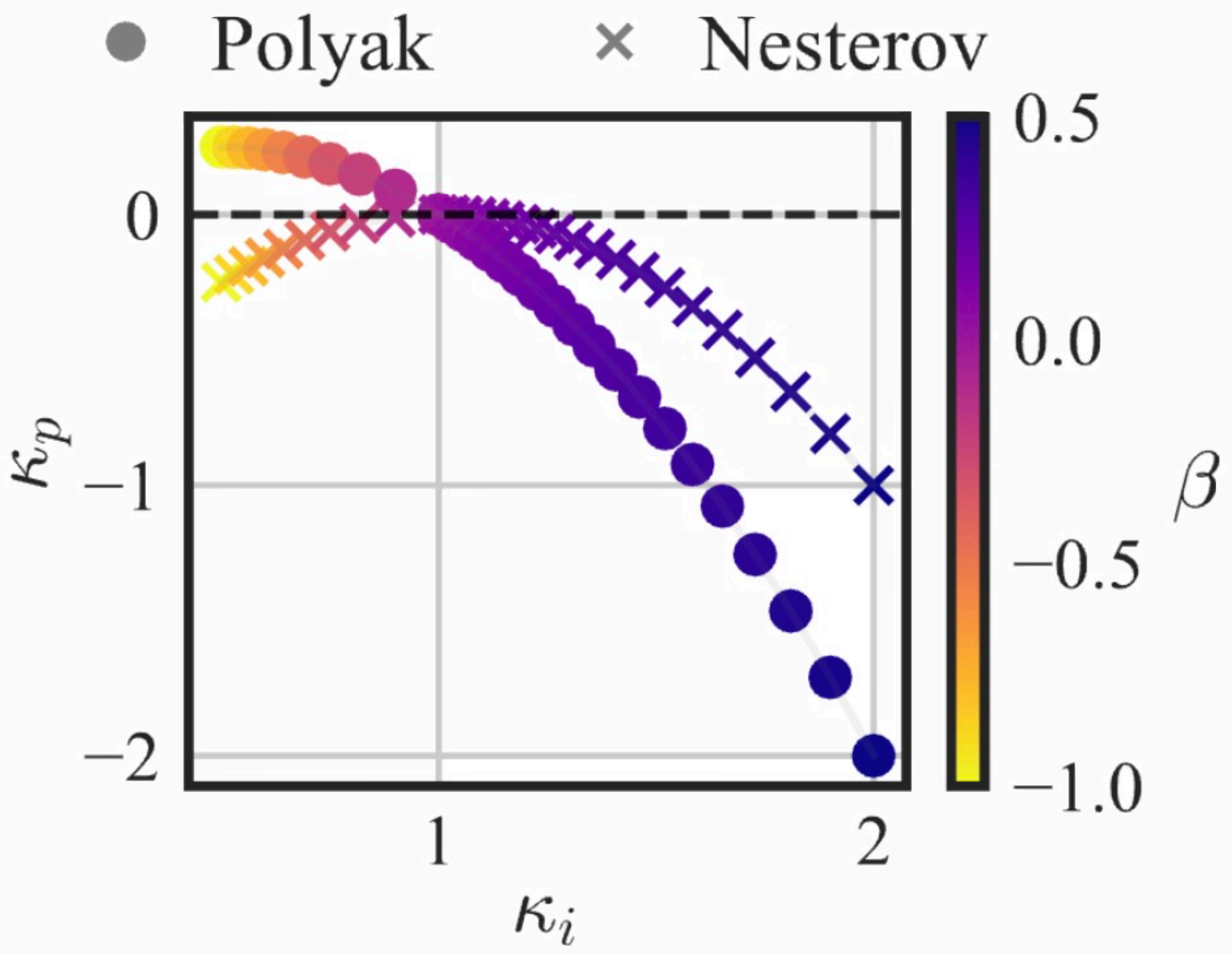
$$\kappa_i \leftarrow \frac{\alpha}{1-\beta}$$

Note the sign of the κ_p coefficient for Polyak and Nesterov:

$$\kappa_p^{\text{Polyak}} \leftarrow -\frac{\alpha\beta}{(1-\beta)^2}$$

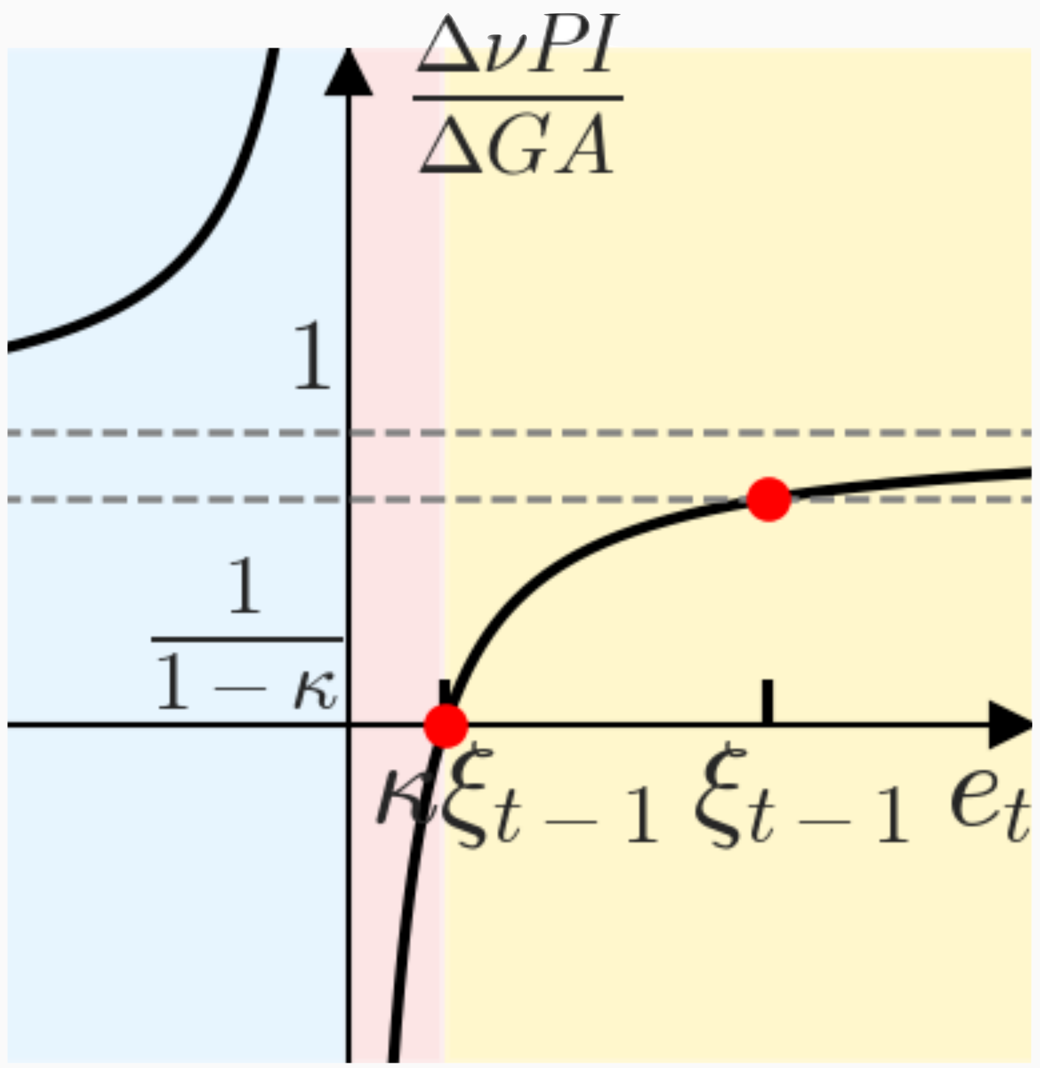
$$\kappa_p^{\text{Nesterov}} \leftarrow -\frac{\alpha\beta^2}{(1-\beta)^2} \leq 0$$

non-positive for both positive and negative momentum

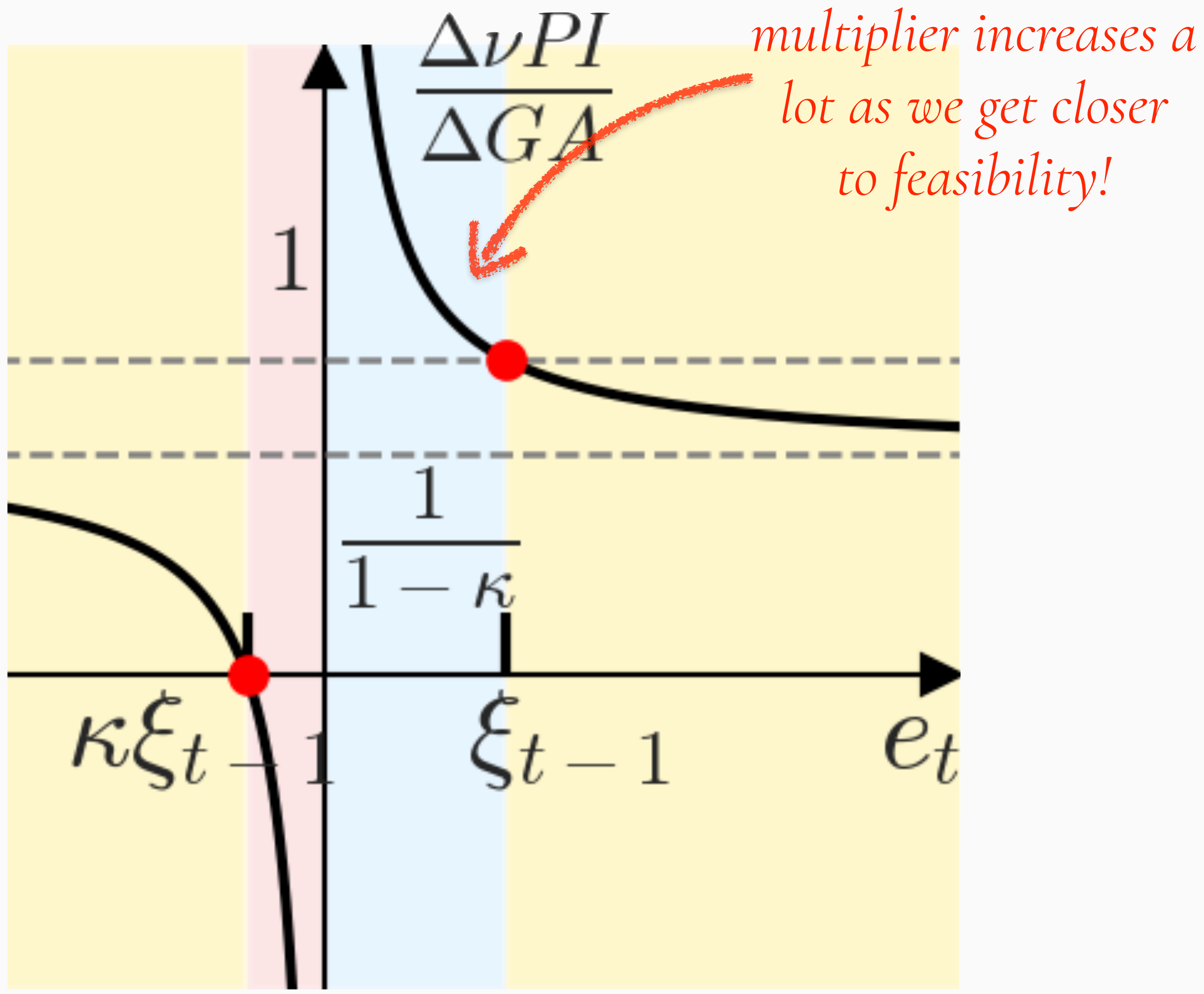


Spanned κ_i and κ_p coefficients for fixed α and changing β

The (undesirable?) effect of momentum



Polyak $\nu = -0.3$



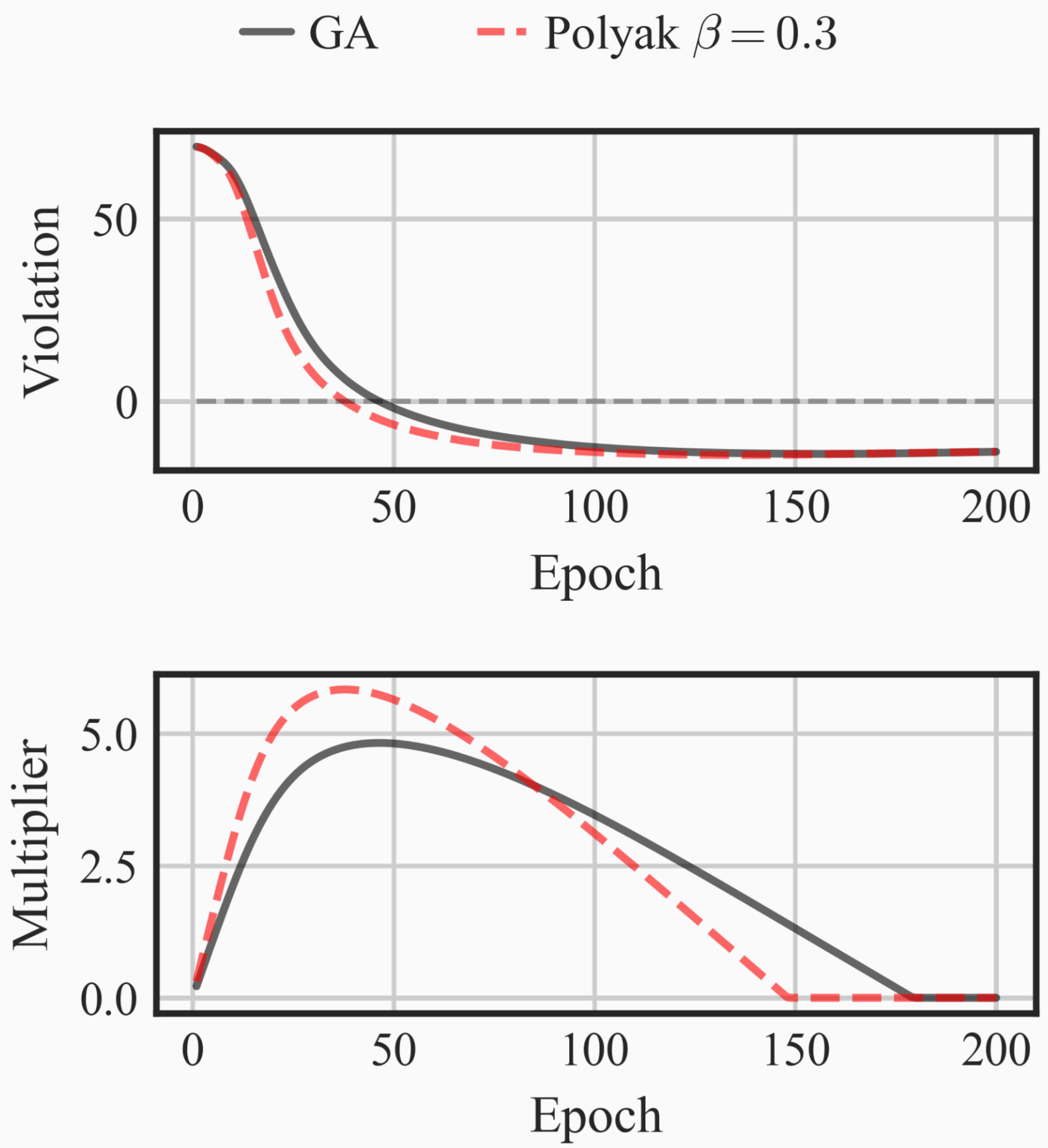
Polyak $\nu = +0.3$

Faster Slower Opposite direction

νPI in context

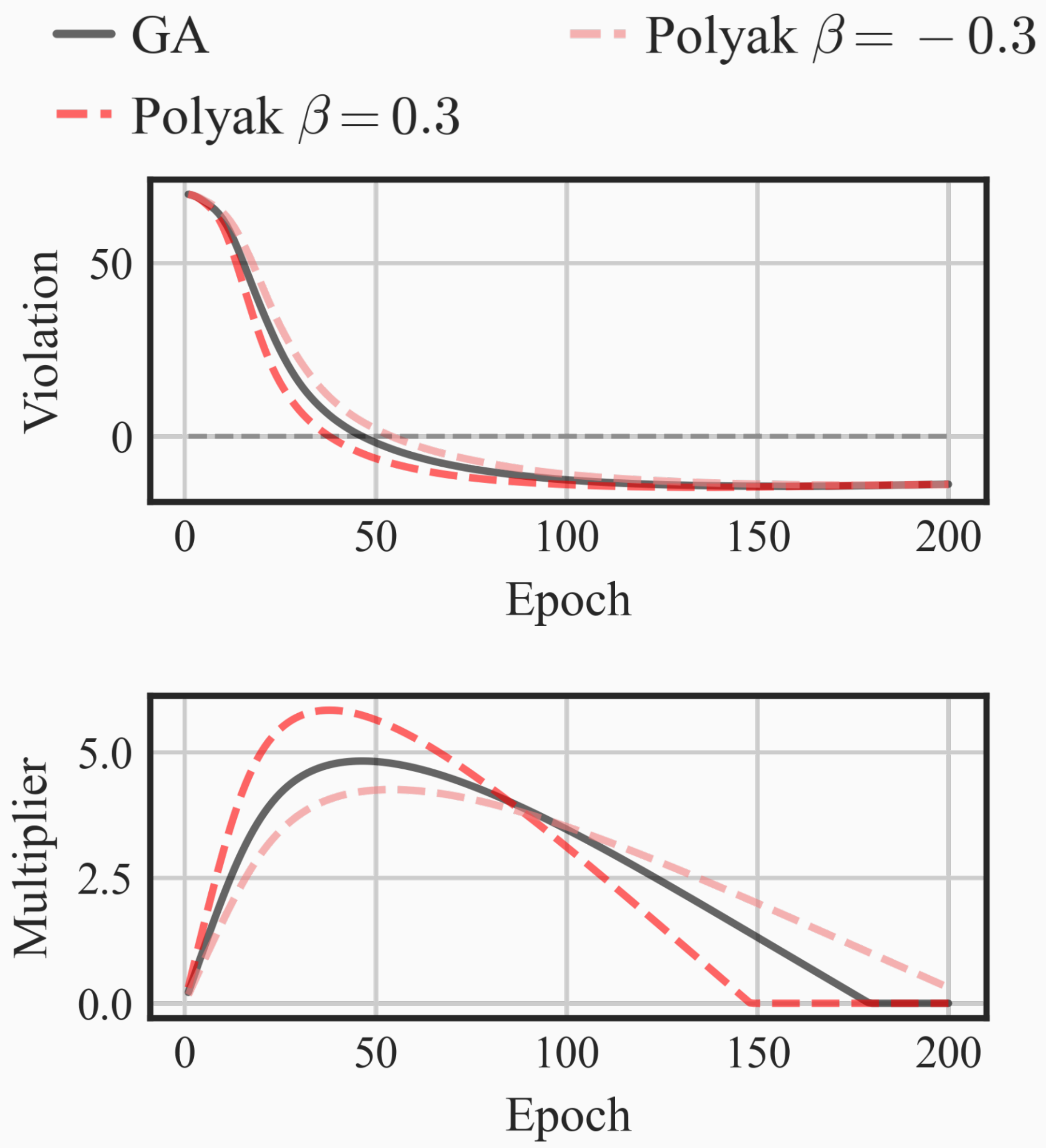
Positive momentum

- Is a special case of νPI
- Induces a negative κ_p
- Has been shown to be counterproductive for (bi-linear) games
- **Makes overshoot problem worse**



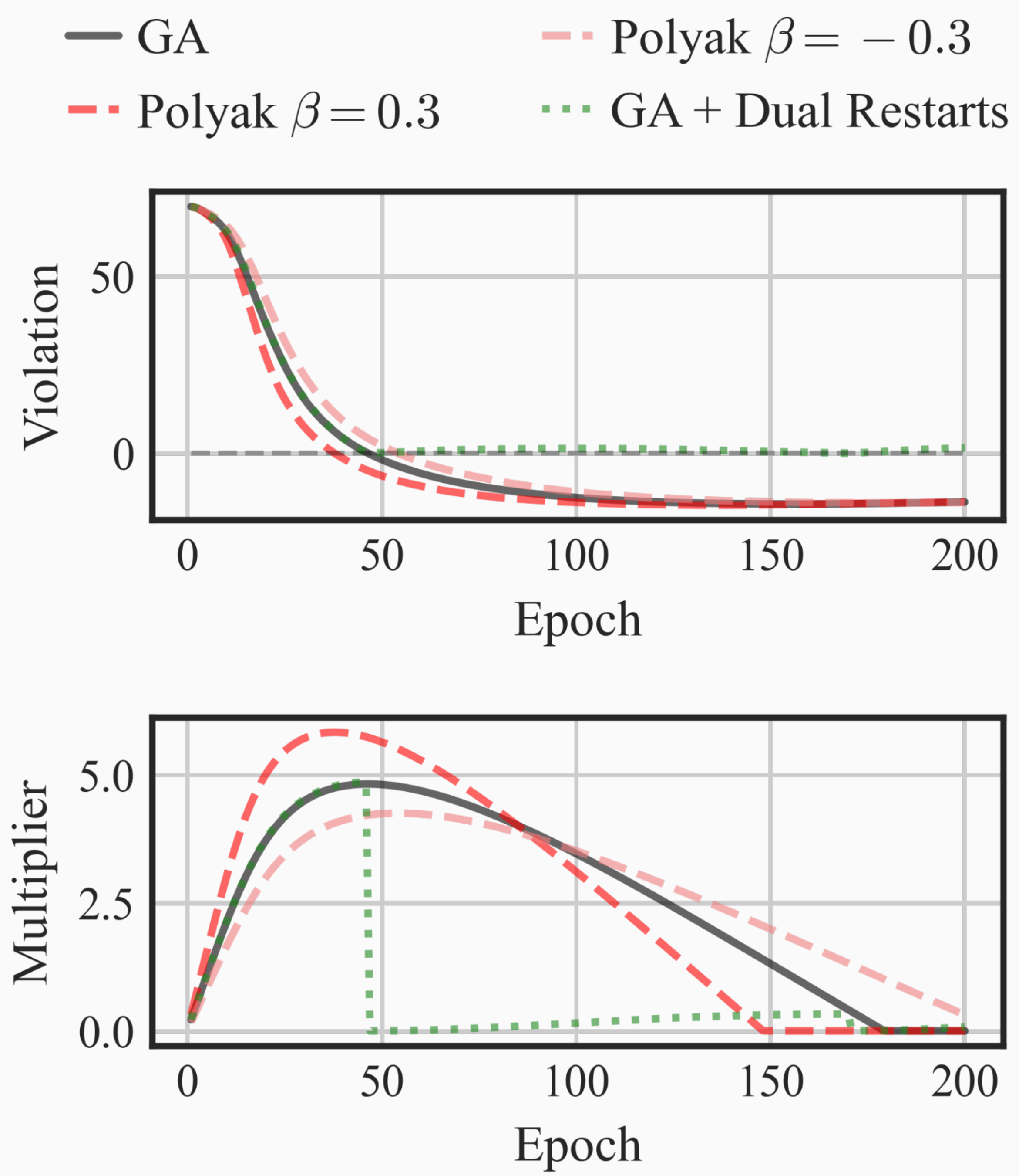
Negative momentum

- Is a special case of νPI
- Induces a positive κ_p
- Suboptimal for strongly convex games (Zhang et al., 2021)
- Alleviates multiplier overshooting, but not "over-enforcement" of the constraint



Dual restarts

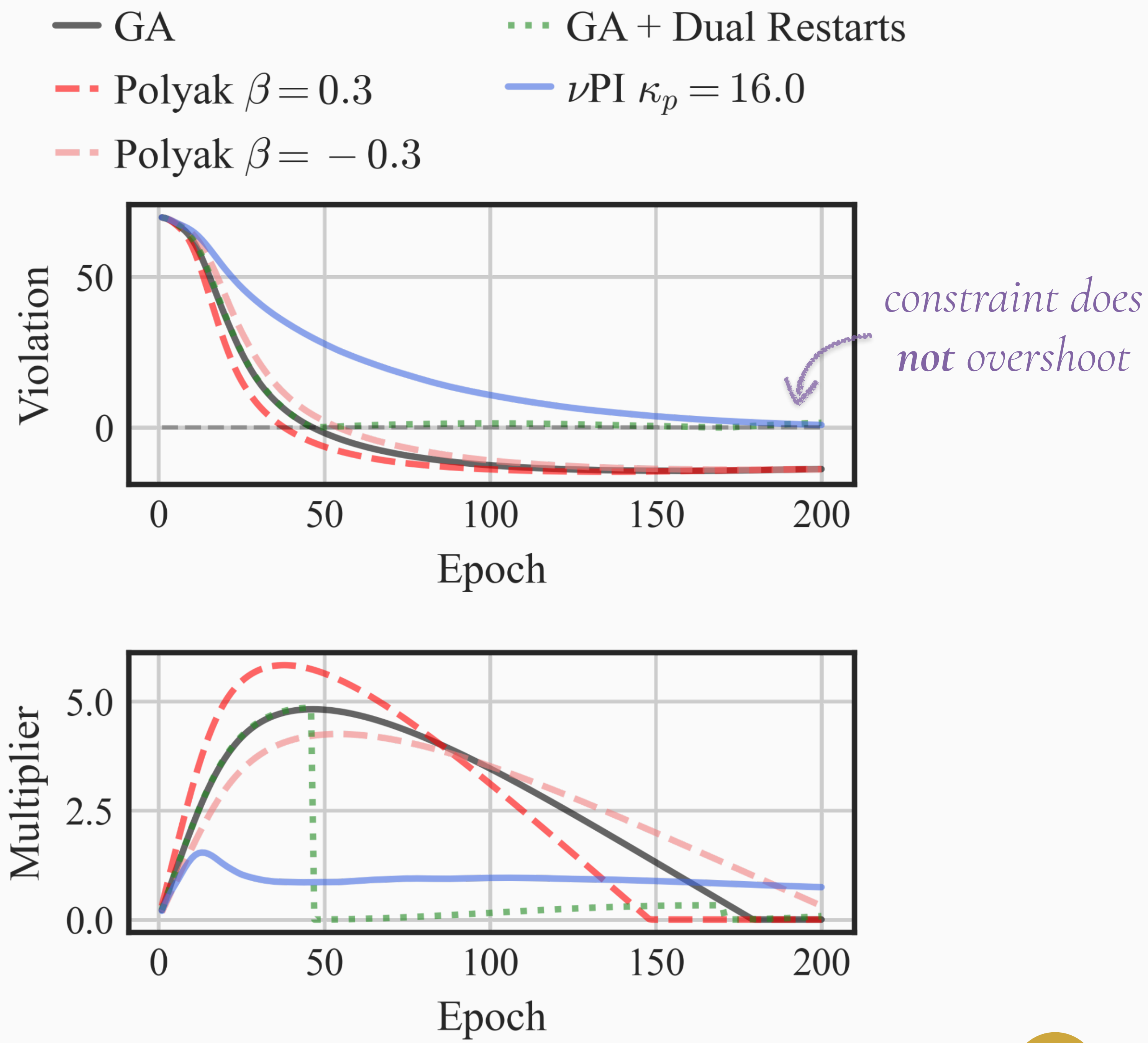
- Once a constraint is strictly satisfied, set its corresponding multiplier to zero (Gallego-Posada et al., 2022)
- Only applicable to (strictly feasible) inequality constraints.
- Relies on exact assessment of constraint satisfaction
  - Stochasticity; numerical precision; "temporary satisfaction"



Gallego-Posada, J., Ramirez, J., Erraqabi, A., Bengio, Y., Lacoste-Julien, S. *Controlled Sparsity via Constrained Optimization or: How I Learned to Stop Tuning Penalties and Love Constraints* . In NeurIPS, 2022.

νPI controller

- Natural generalization of the optimistic gradient method, which is (near) optimal for games (Mokhtari et al., 2020)
- Monotonic effect of κ_p on the degree of overshoot
- One fewer degree of freedom than full PID



Mokhtari, A., Ozdaglar, A. E., and Pattathil, S. *Convergence Rate of O(1/k) for Optimistic Gradient and Extragradient Methods in Smooth Convex-Concave Saddle Point Problems.* SIAM Journal on Optimization, 2020.

Experiments

Hard-margin SVMs

$$\min_{w} \frac{1}{2}||w||^2 \text{ subject to } y_i\left(w^\top x_i + b\right) \geq 1 \text{ for } i = 1,\ldots,N$$

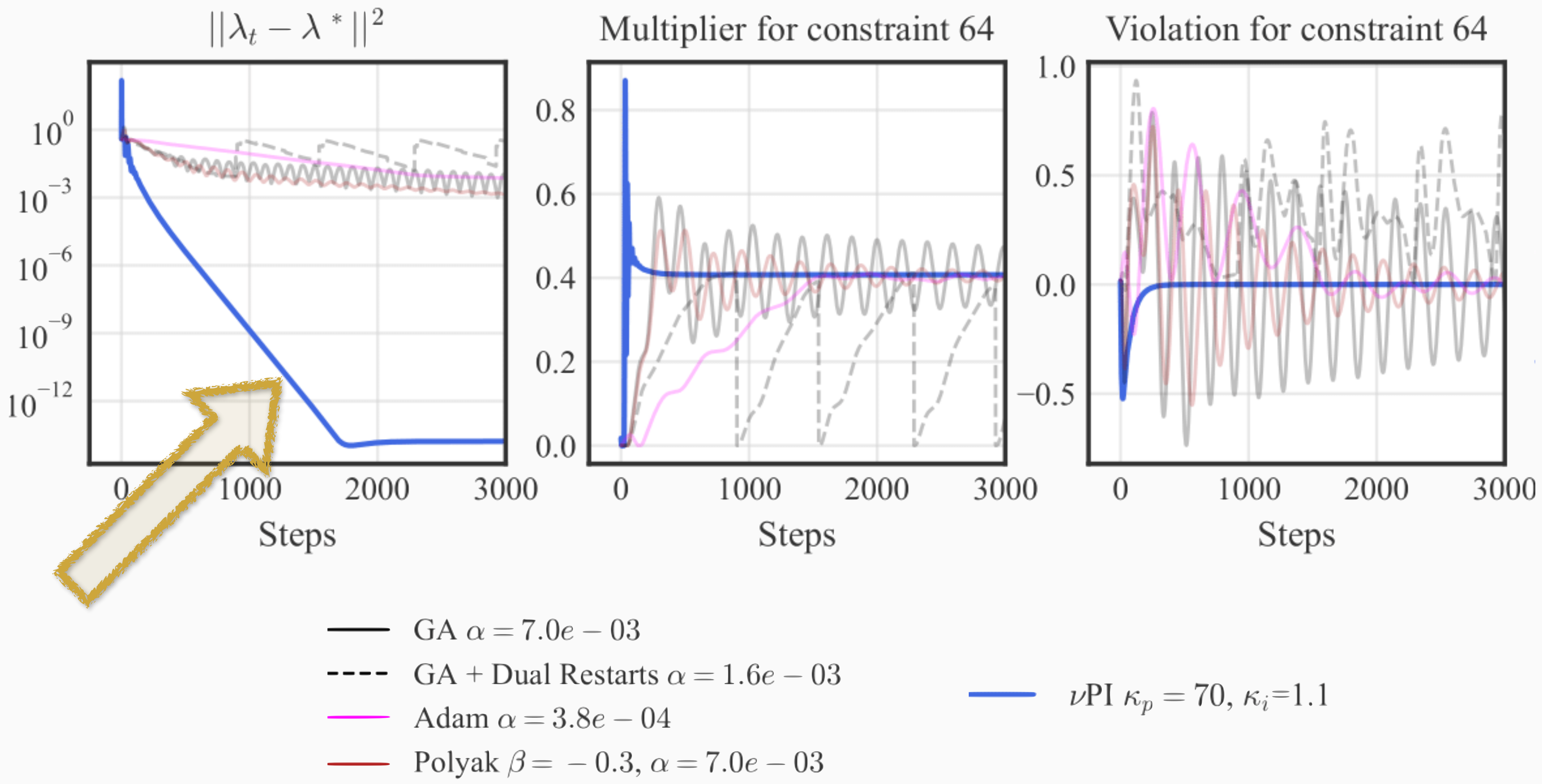
Motivation

- Simple, well behaved convex problem with unique* KKT tuple
- Specialized solvers exist for QCQPs, we use this task for illustration
- Cheap experiment allows fine grid-search to test influence of hyperparameters of different algorithms

Experimental setup

- Linearly-separable subset of the Iris dataset
- 70 training samples ⇒ 70 inequality constraints

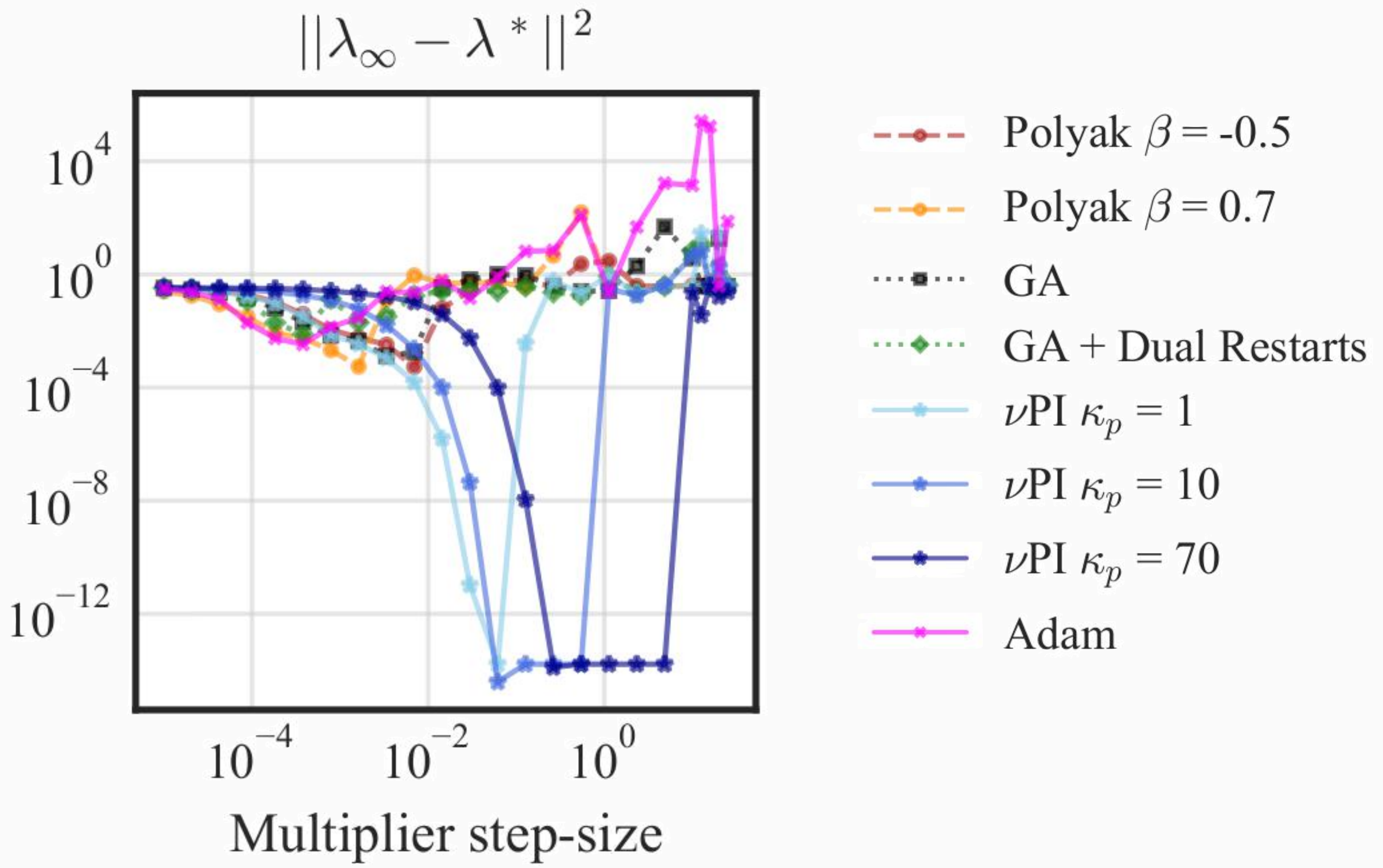
Of all attempted optimizers*, **only νPI converged successfully to the true solution!**

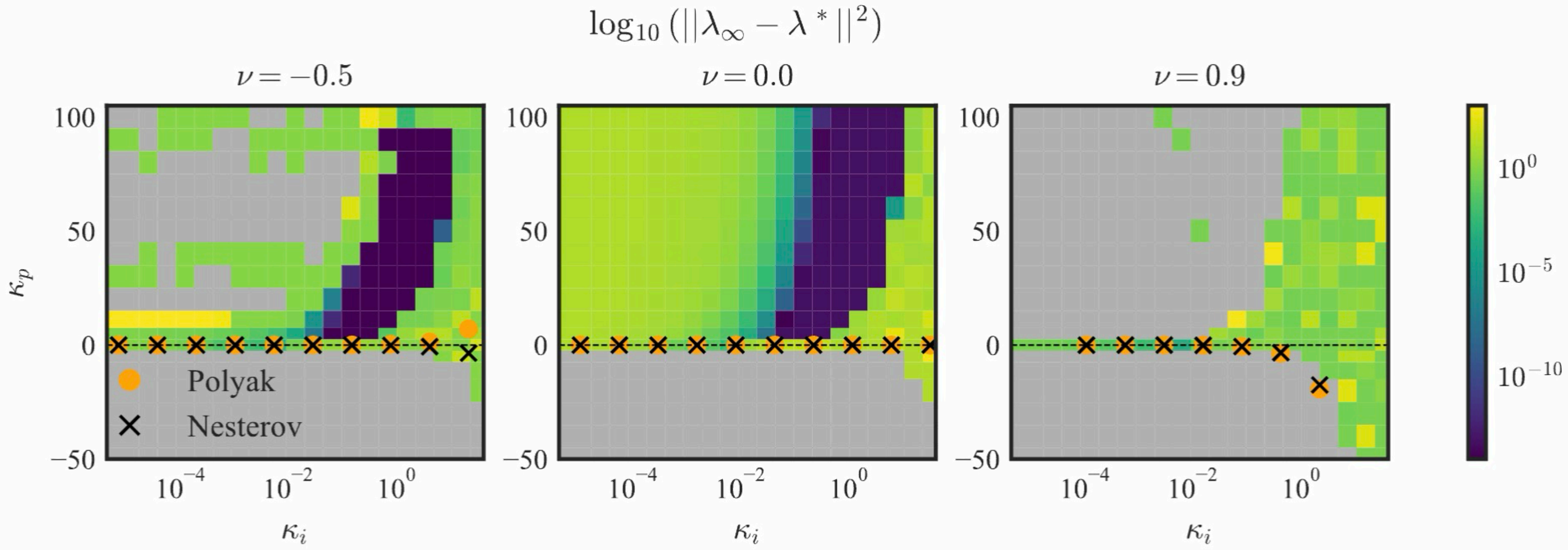


*Showing best hyperparameters for each optimizer after grid-search aiming to minimize the distance to λ^* after 5.000 iterations

Robustness

Higher values of κ_p allow for choosing **larger values of κ_i** (multiplier step-size) and **over a wider range**, while still achieving convergence.





νPI **provides additional flexibility** compared to Polyak and Nesterov which is **crucial for achieving convergence** in this task.

Training sparse ResNets

$$\min_{x,\phi} \mathbb{E}_{z|\phi}\left[L(x \odot z \,|\, \mathscr{D})\right] \text{ subject to} \frac{\mathbb{E}_{z|\phi}\left[||z||_0\right]}{\#(x)} \leq \epsilon$$

Motivation

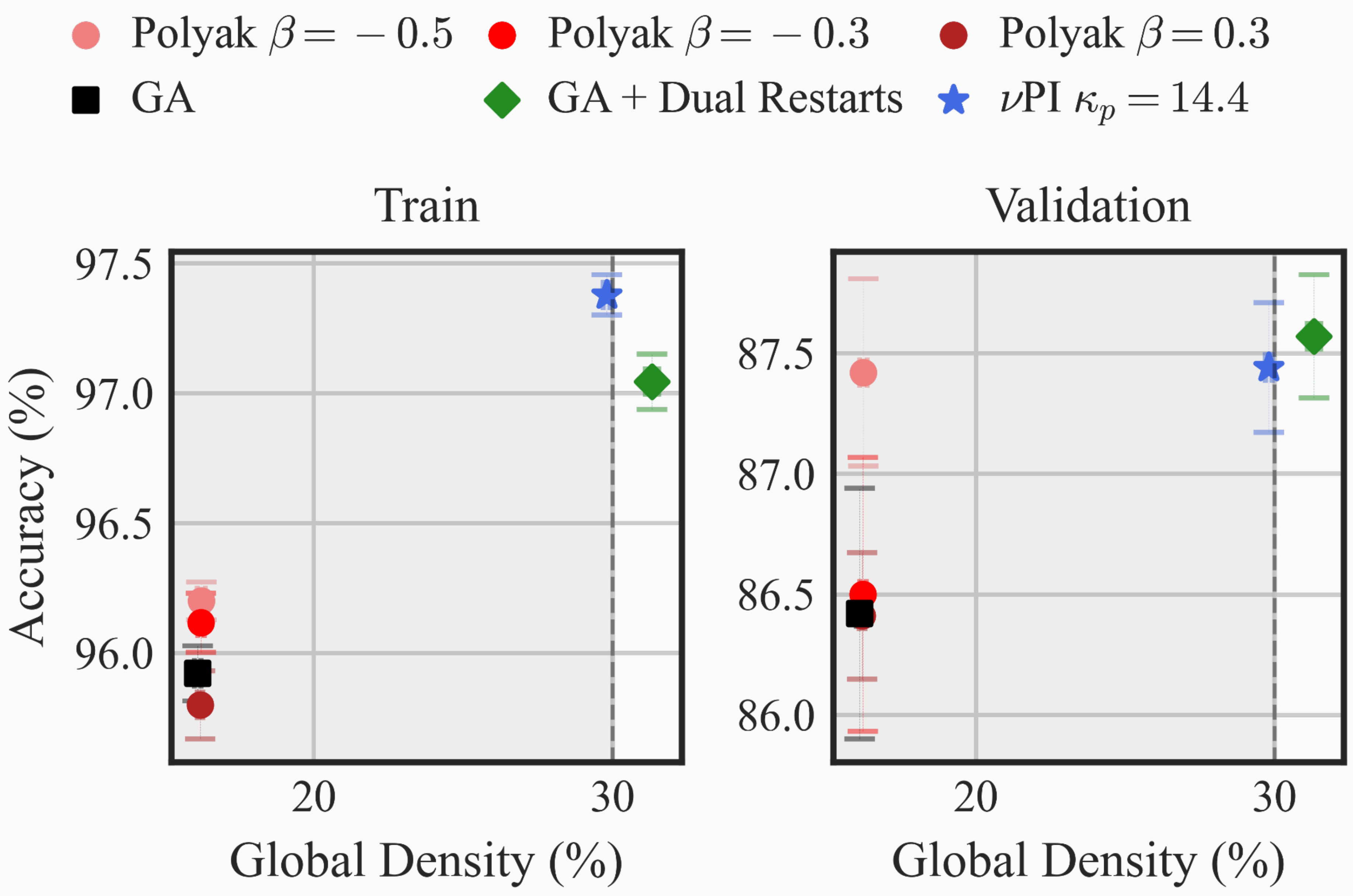
- More realistic deep application with non-convex constraints
- In our prior work we document the issue of overshoot and propose "dual restarts" heuristic

Experimental setup

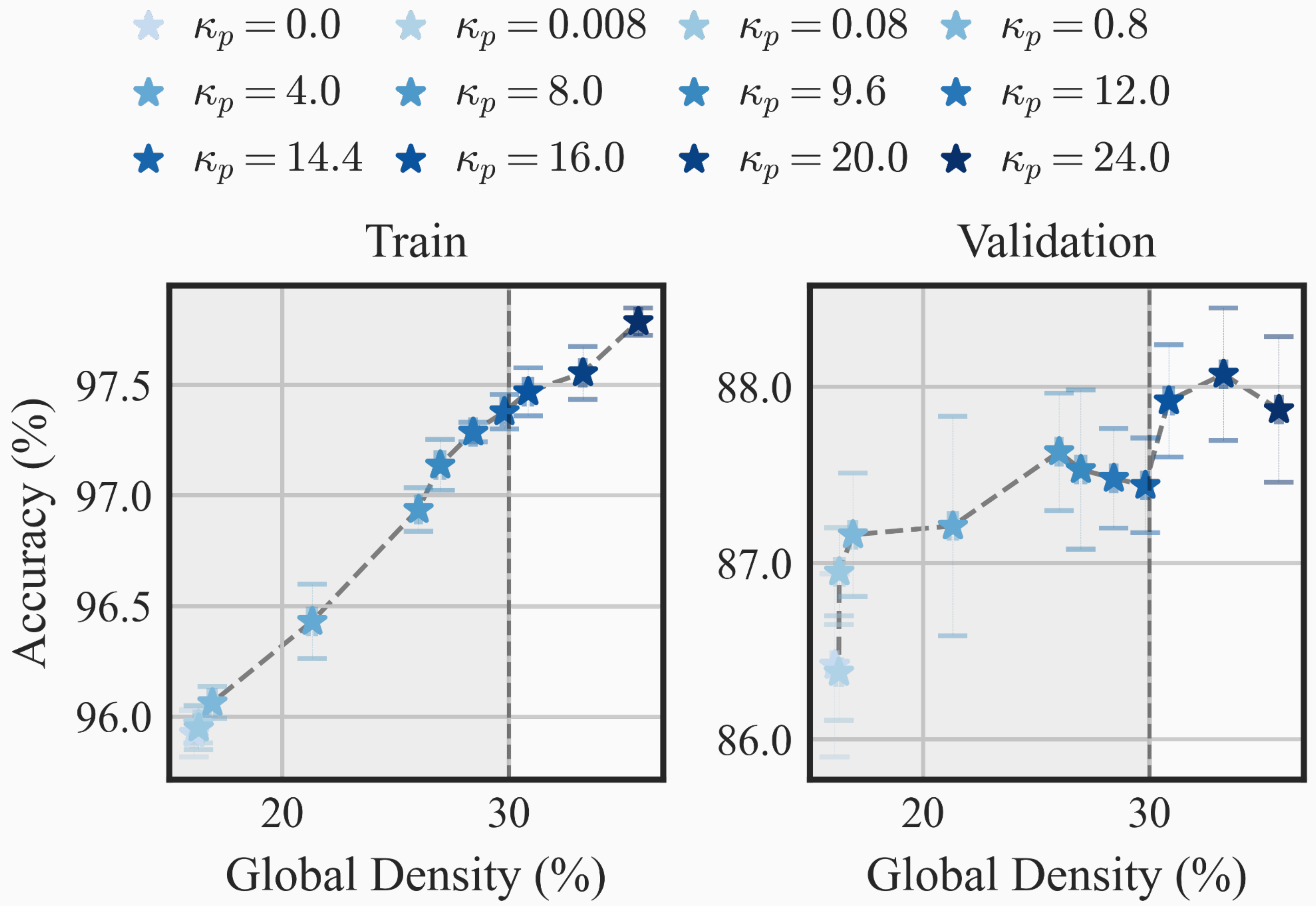
- Training a ResNet-18 model on CIFAR10
- Structured sparsity with layer-wise or model-wise constraints

Addressing constraint overshooting

νPI achieves high accuracy and tightly respects the constraints, without overshooting



Monotonicity on κ_p



Cooper

a library for Lagrangian-based constrained optimization in PyTorch