Interactively Learning Preference Constraints in Linear Bandits

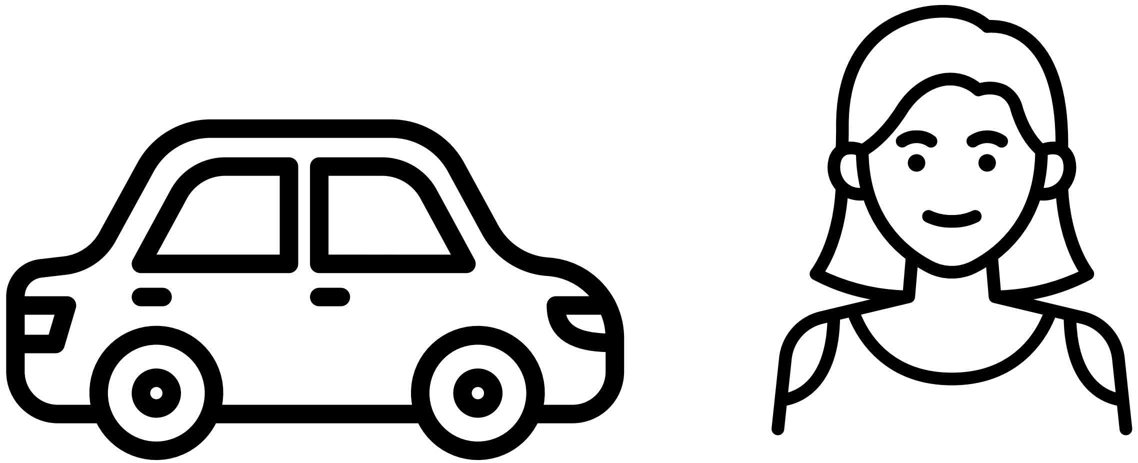
International Conference on Machine Learning (ICML), 2022

David Lindner[1], Sebastian Tschiatschek[2], Katja Hofmann[3], Andreas Krause[1]
[1]ETH Zurich; [2]University of Vienna; [3]Microsoft Research Cambridge

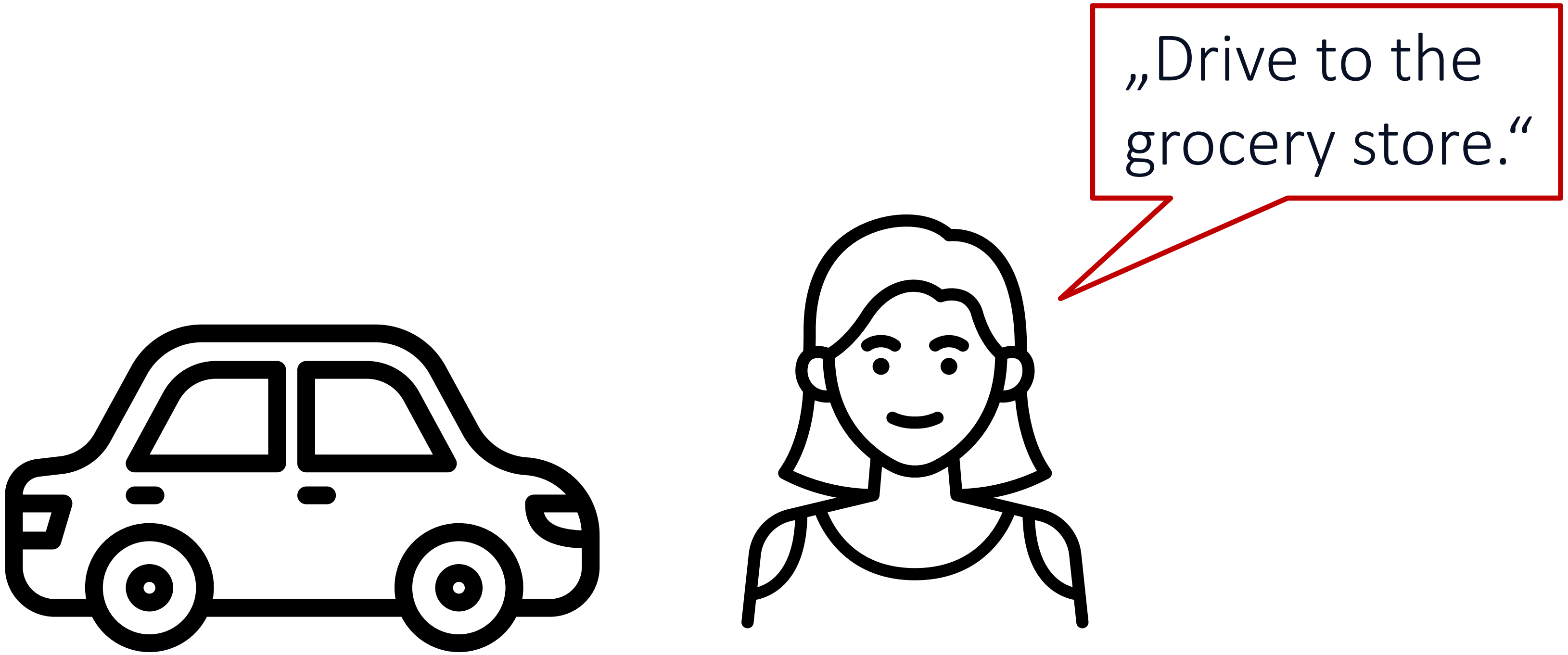
ETH zürich   Learning & Adaptive Systems   universität wien   Microsoft

Human preferences often naturally decompose into rewards and constraints

Human preferences often naturally decompose into rewards and constraints



Human preferences often naturally decompose into rewards and constraints

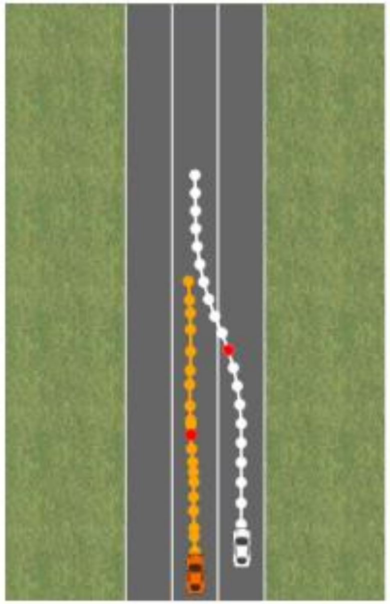


Human preferences often naturally decompose into rewards and constraints

Constraint can be more transferable and robust than rewards

Constraint can be more transferable and robust than rewards

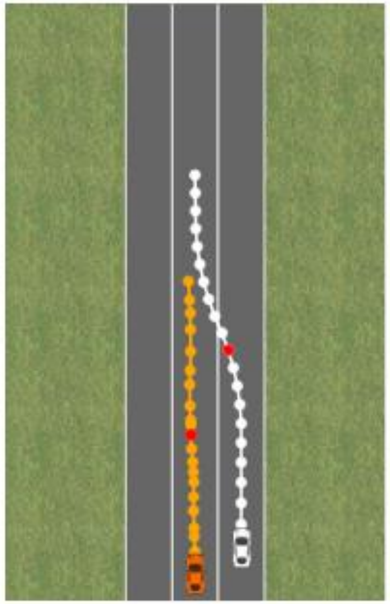
Encode task as reward + penalty



Drive at target velocity

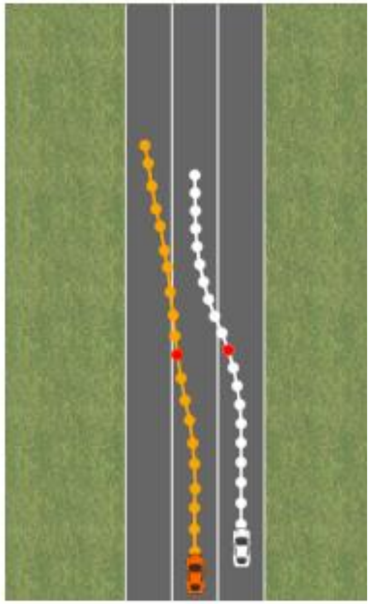
Constraint can be more transferable and robust than rewards

Encode task as reward + penalty



Drive at target velocity

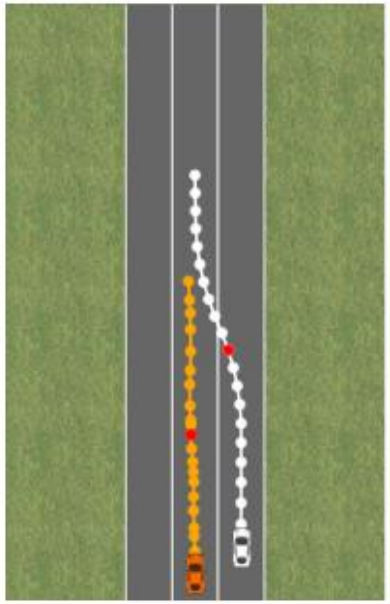
Encode task as reward + constraint



Drive at target velocity

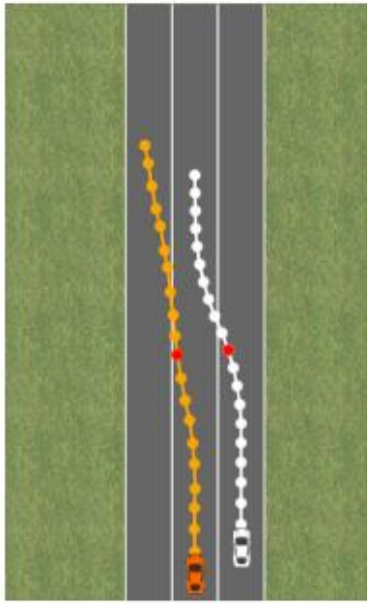
Constraint can be more transferable and robust than rewards

Encode task as reward + penalty



Drive at target velocity

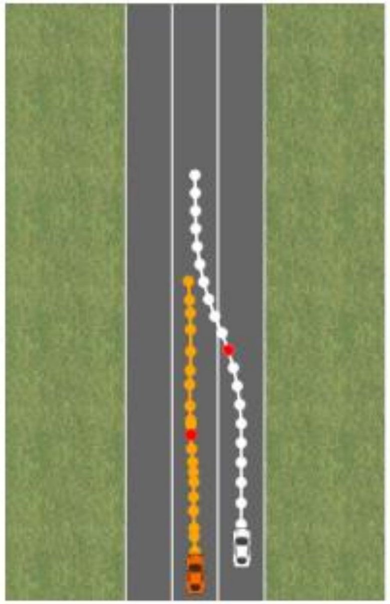
Encode task as reward + constraint

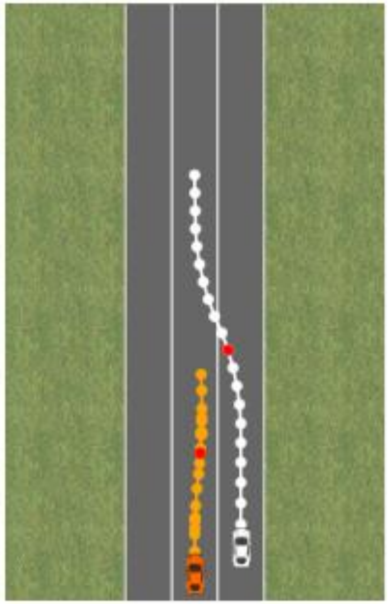


Drive at target velocity

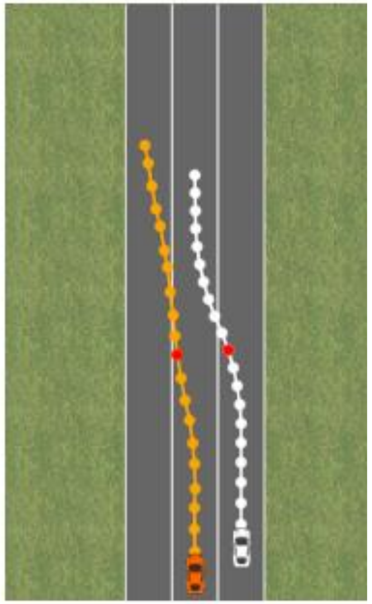
Constraint can be more transferable and robust than rewards

Encode task as reward + penalty


Drive at target velocity

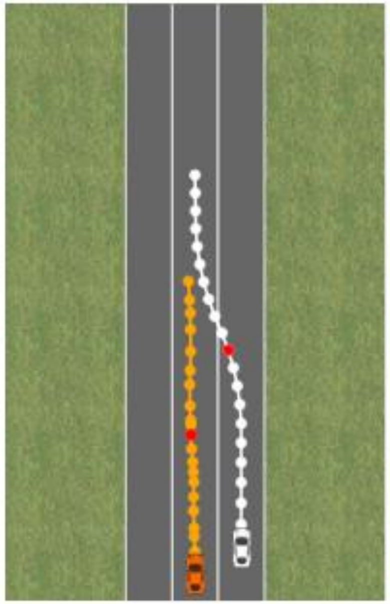

Pull over

Encode task as reward + constraint

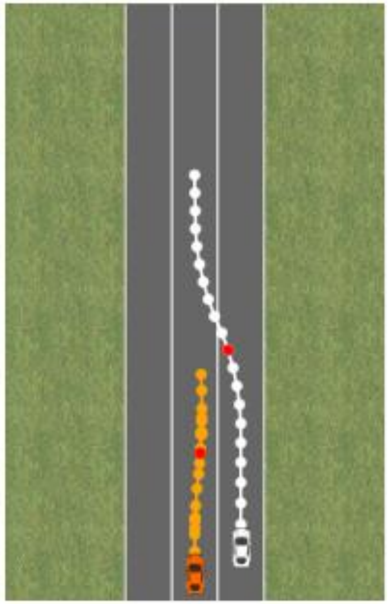

Drive at target velocity

Constraint can be more transferable and robust than rewards

Encode task as reward + penalty

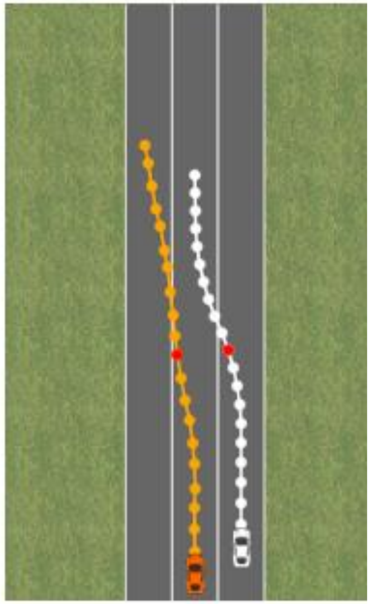


Drive at target velocity



Pull over

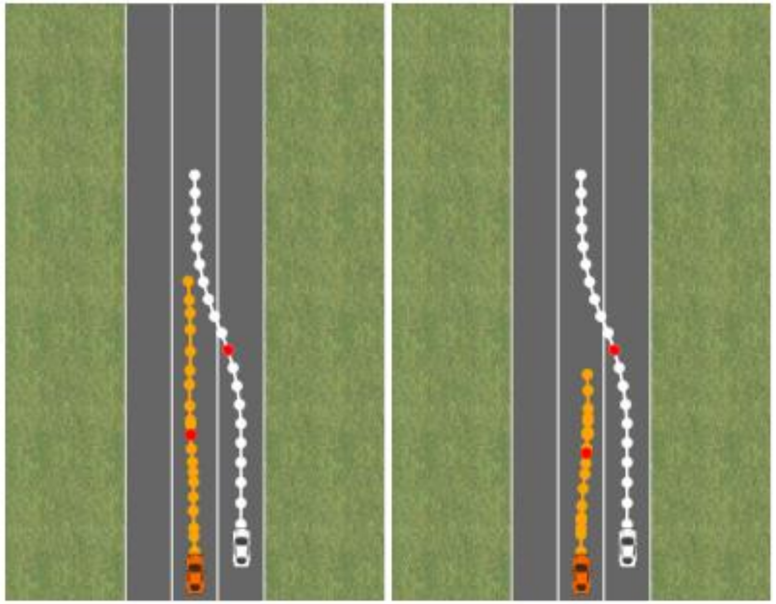
Encode task as reward + constraint



Drive at target velocity

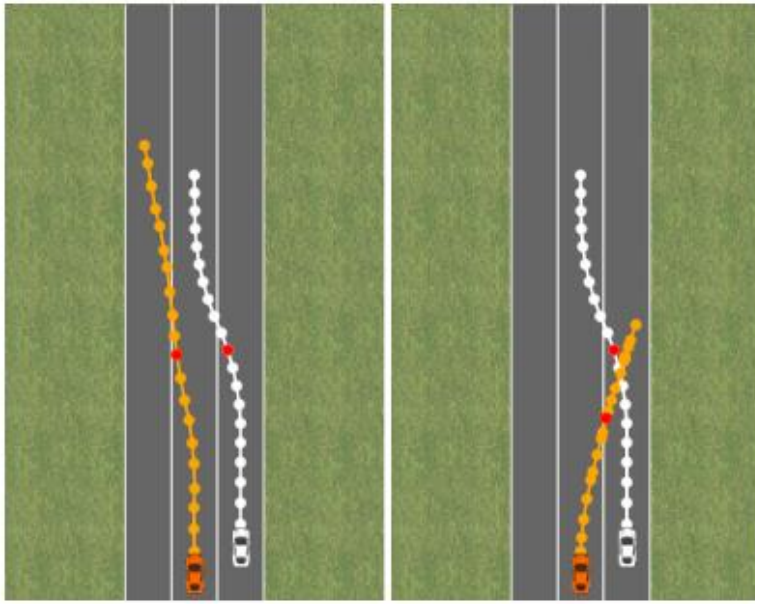
Constraint can be more transferable and robust than rewards

Encode task as reward + penalty



Drive at target velocity   Pull over

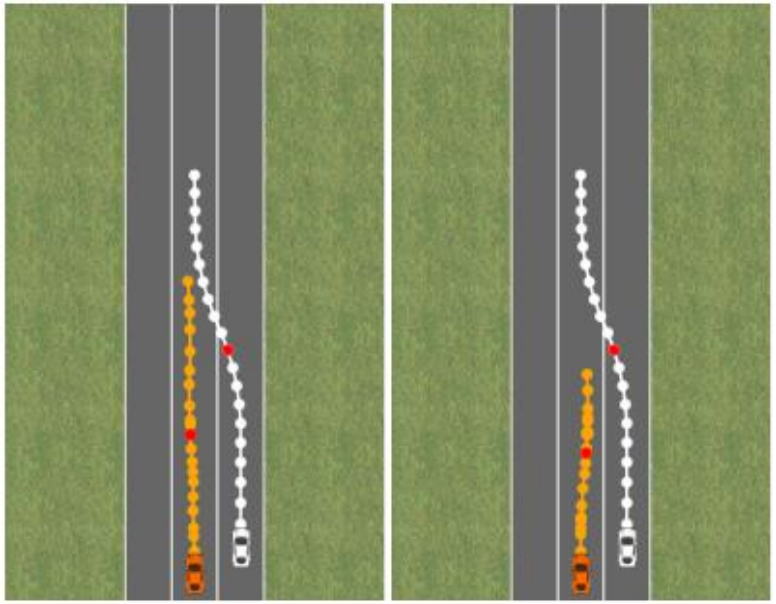
Encode task as reward + constraint



Drive at target velocity   Pull over

Constraint can be more transferable and robust than rewards

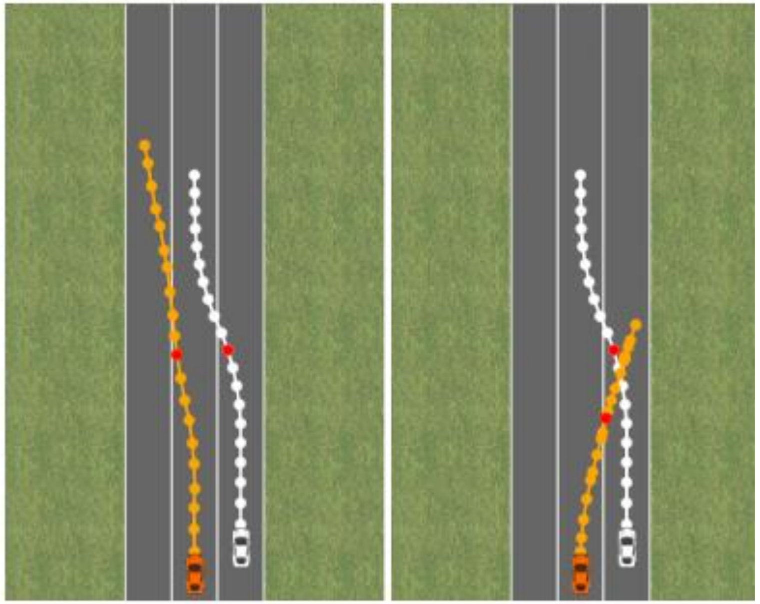
Encode task as reward + penalty



Drive at target velocity

Pull over

Encode task as reward + constraint

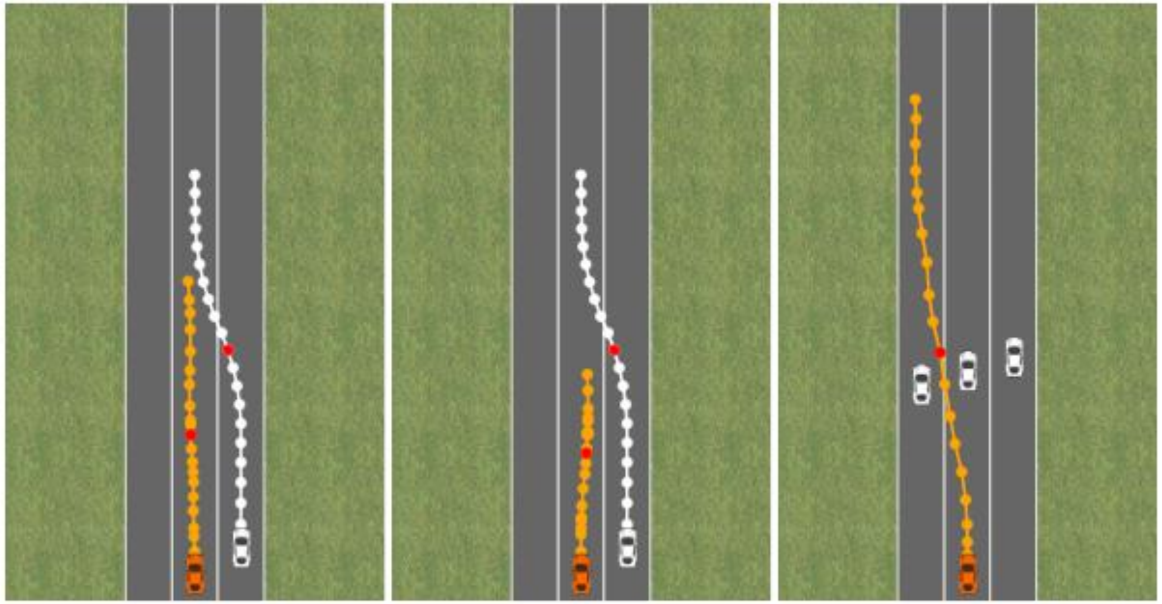


Drive at target velocity

Pull over

Constraint can be more transferable and robust than rewards

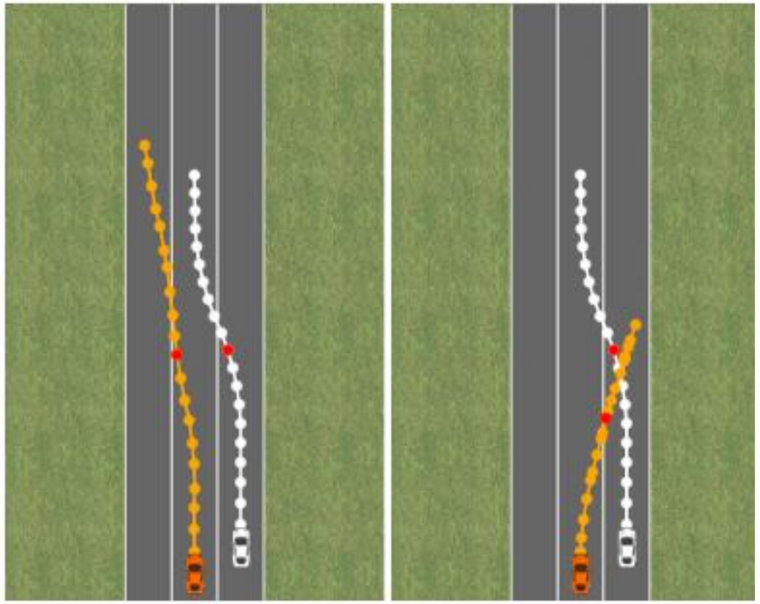
Encode task as reward + penalty



Drive at target velocity | Pull over | Blocked street

Encode task as reward + constraint

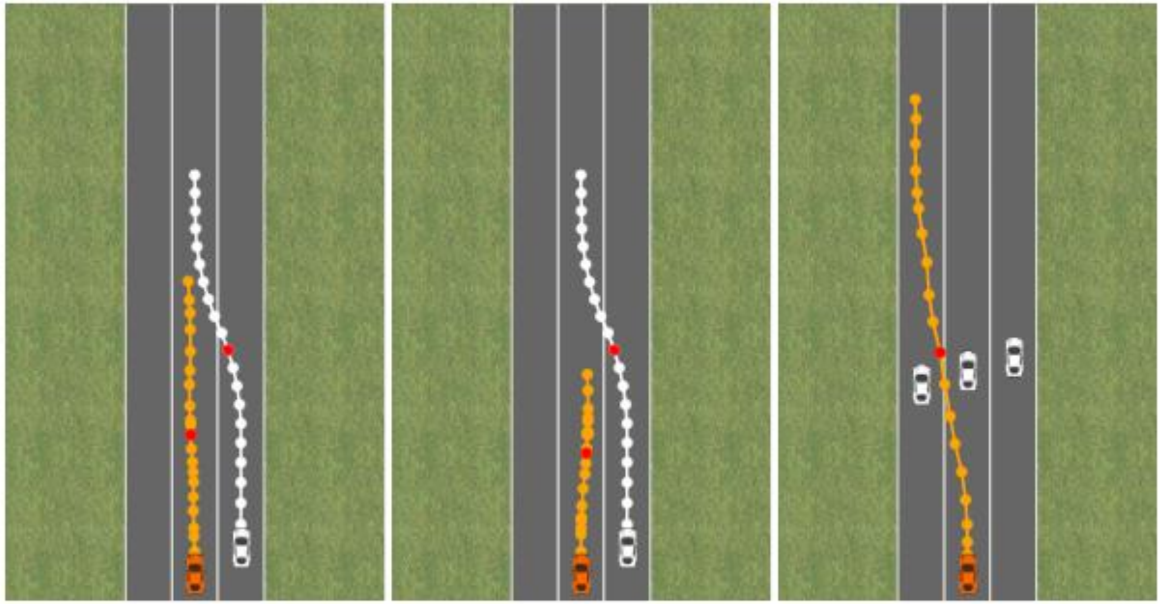


Drive at target velocity | Pull over

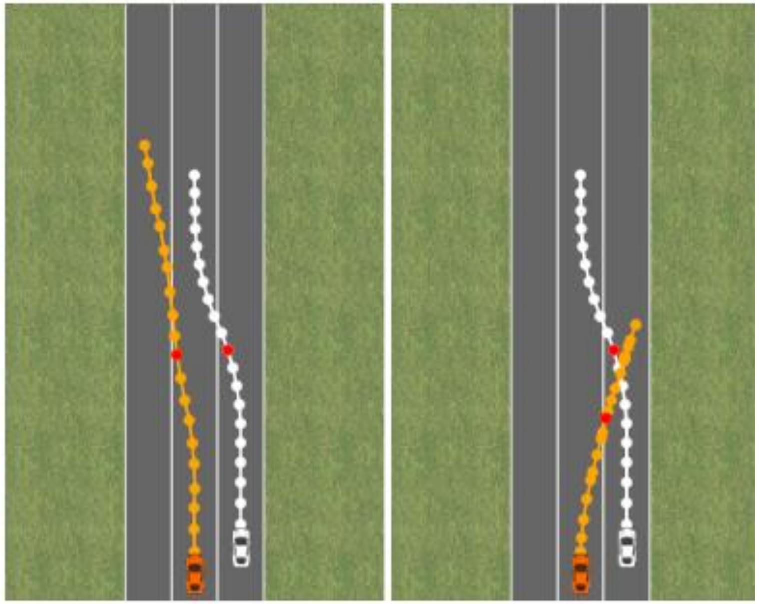
Constraint can be more transferable and robust than rewards

Encode task as reward + penalty



Drive at target velocity | Pull over | Blocked street

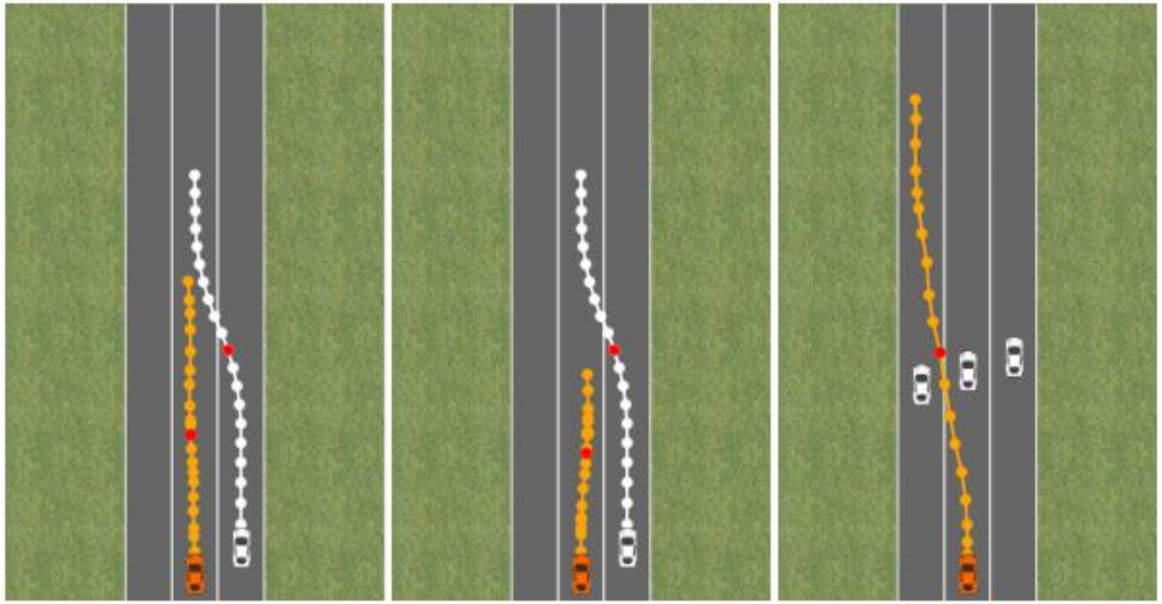
Encode task as reward + constraint



Drive at target velocity | Pull over

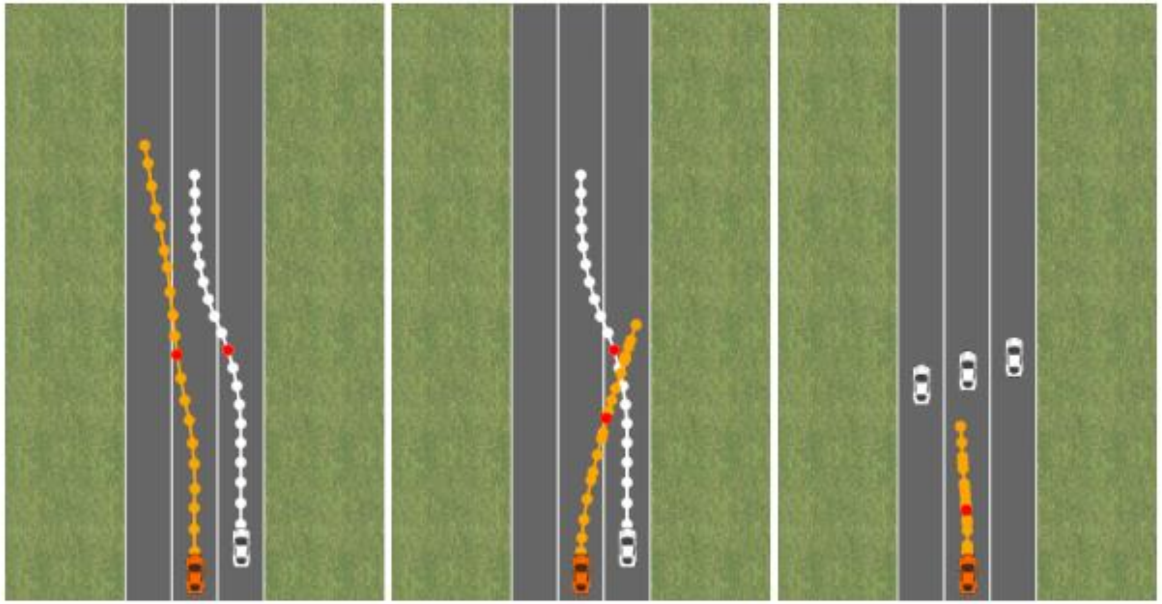
Constraint can be more transferable and robust than rewards

Encode task as reward + penalty



Drive at target velocity | Pull over | Blocked street

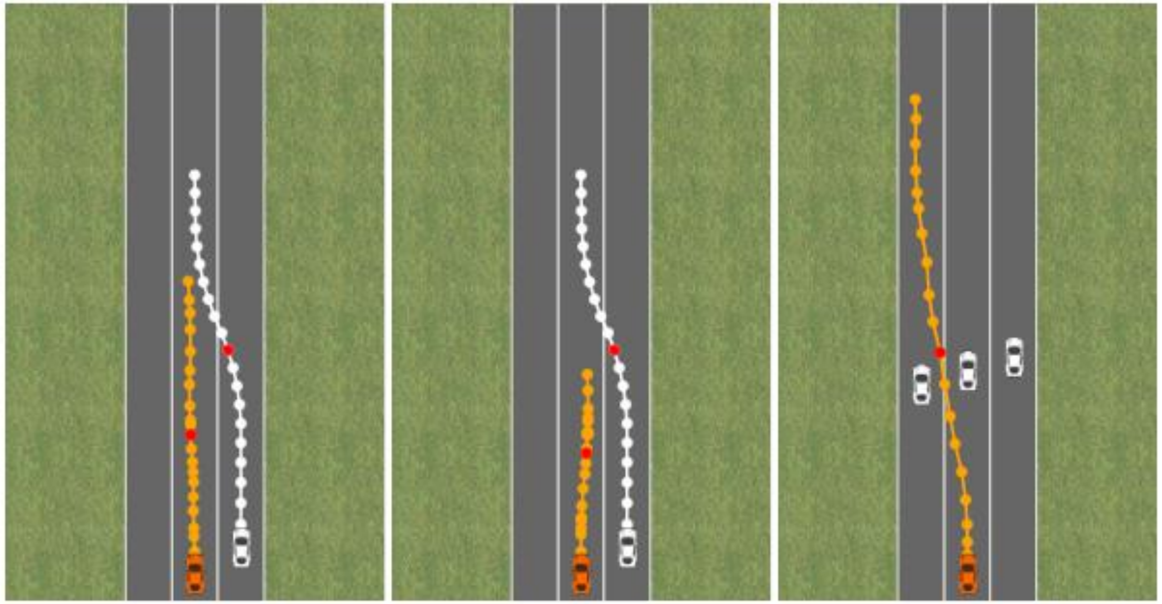
Encode task as reward + constraint



Drive at target velocity | Pull over | Blocked street

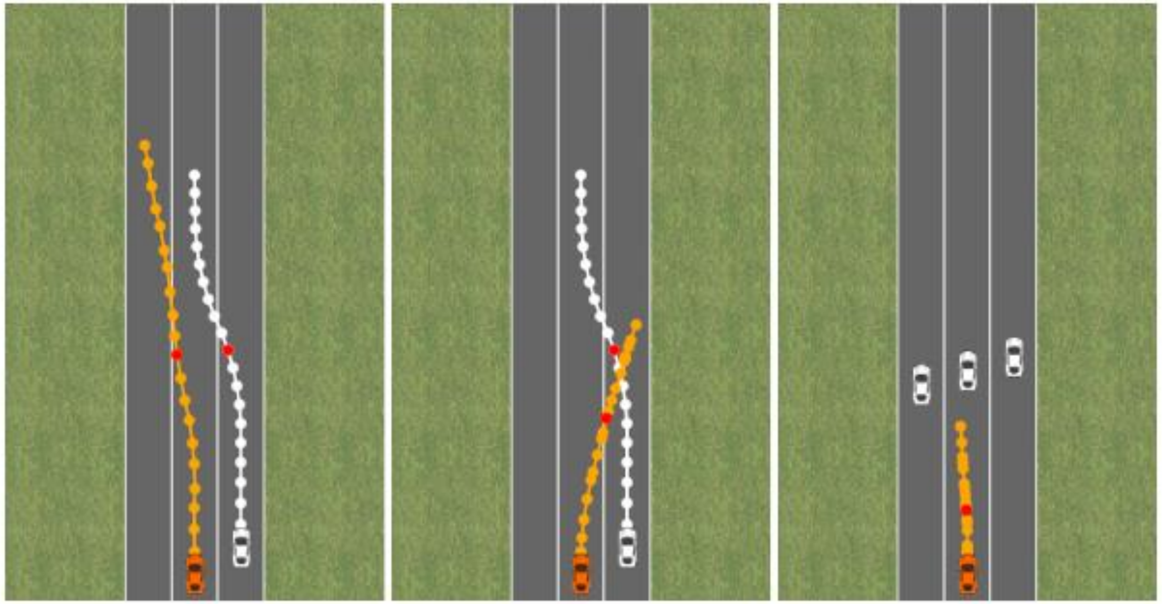
Constraint can be more transferable and robust than rewards

Encode task as reward + penalty



Drive at target velocity | Pull over | Blocked street

Encode task as reward + constraint

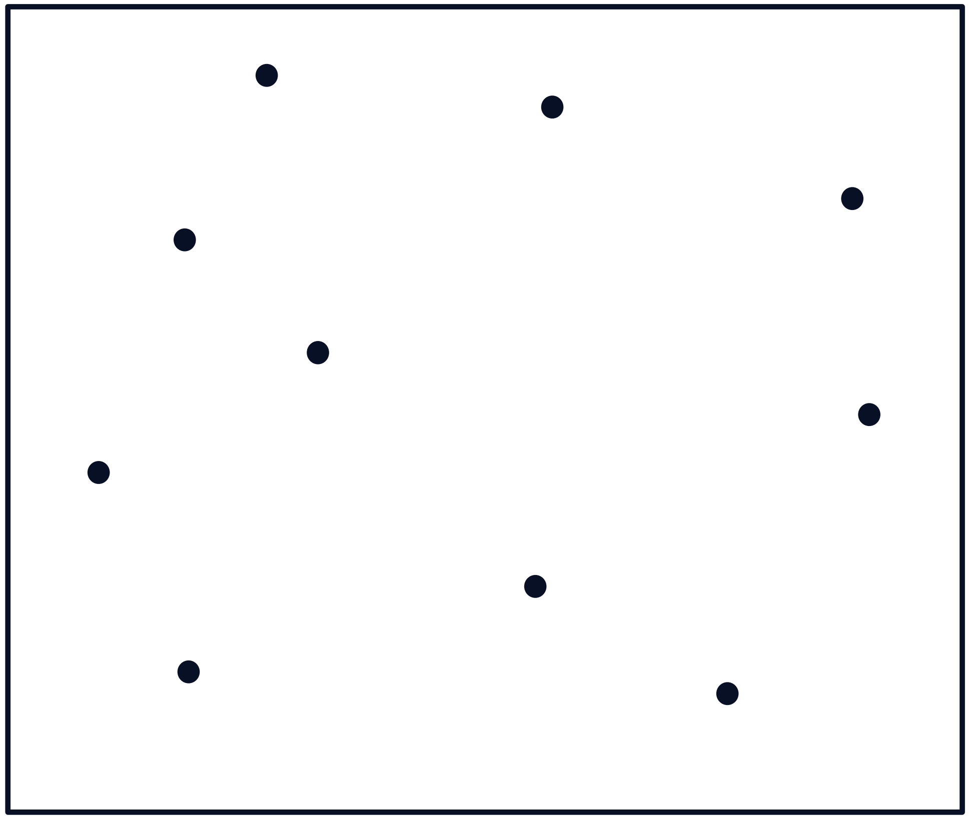


Drive at target velocity | Pull over | Blocked street

Constrained linear best-arm identification (CBAI)

Constrained linear best-arm identification (CBAI)

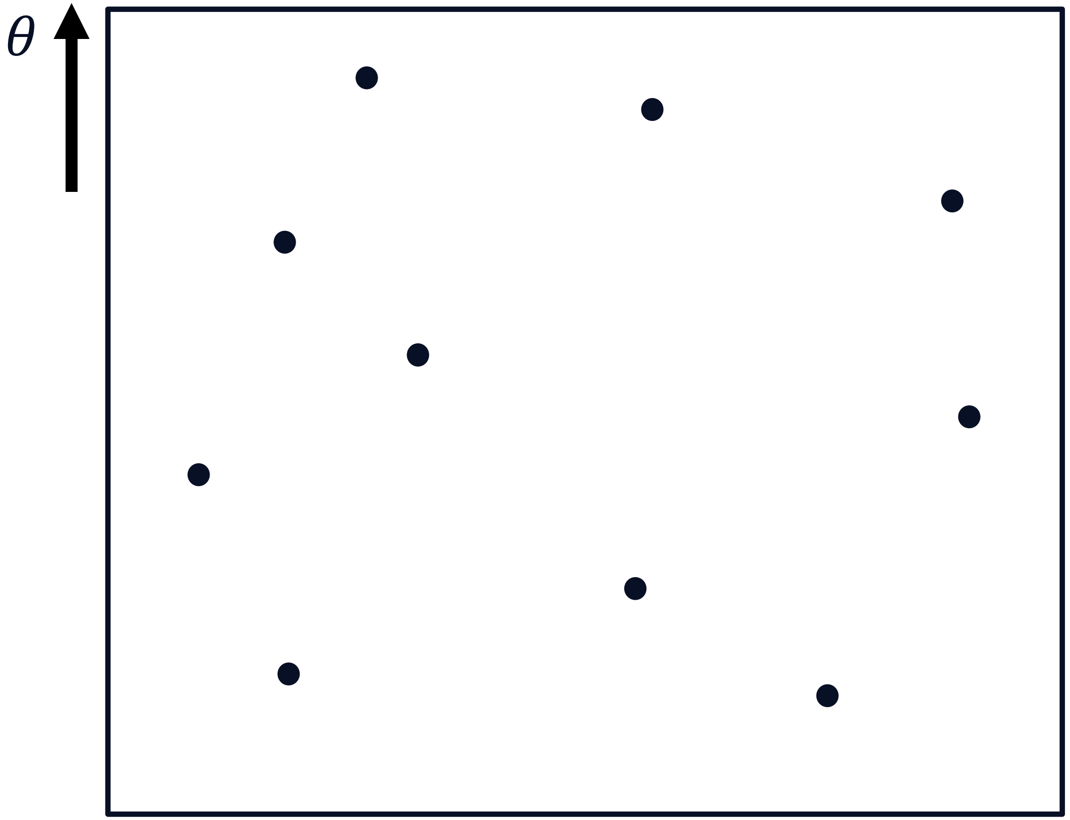
Known set of arms $\mathcal{X} \subset \mathbb{R}^d$



Constrained linear best-arm identification (CBAI)

Known set of arms $\mathcal{X} \subset \mathbb{R}^d$

Known reward parameter $\theta \in \mathbb{R}^d$

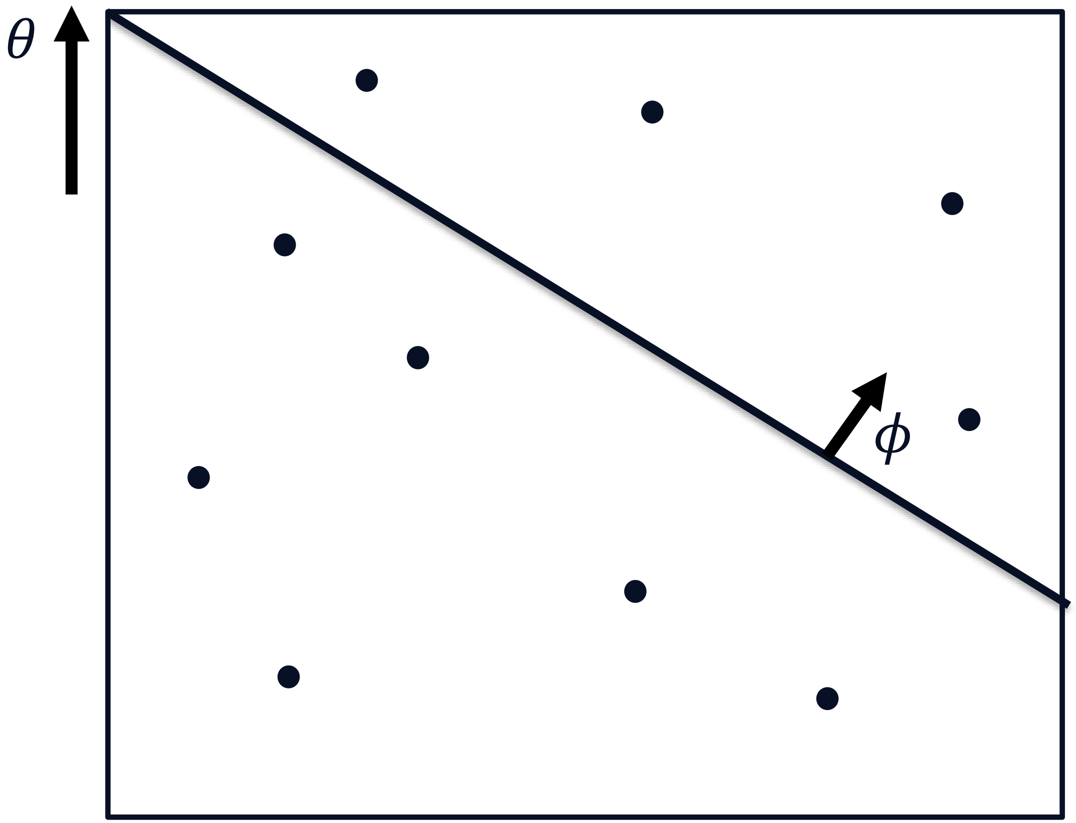


Constrained linear best-arm identification (CBAI)

Known set of arms $\mathcal{X} \subset \mathbb{R}^d$

Known reward parameter $\theta \in \mathbb{R}^d$

Unknown constraint parameter $\phi \in \mathbb{R}^d$

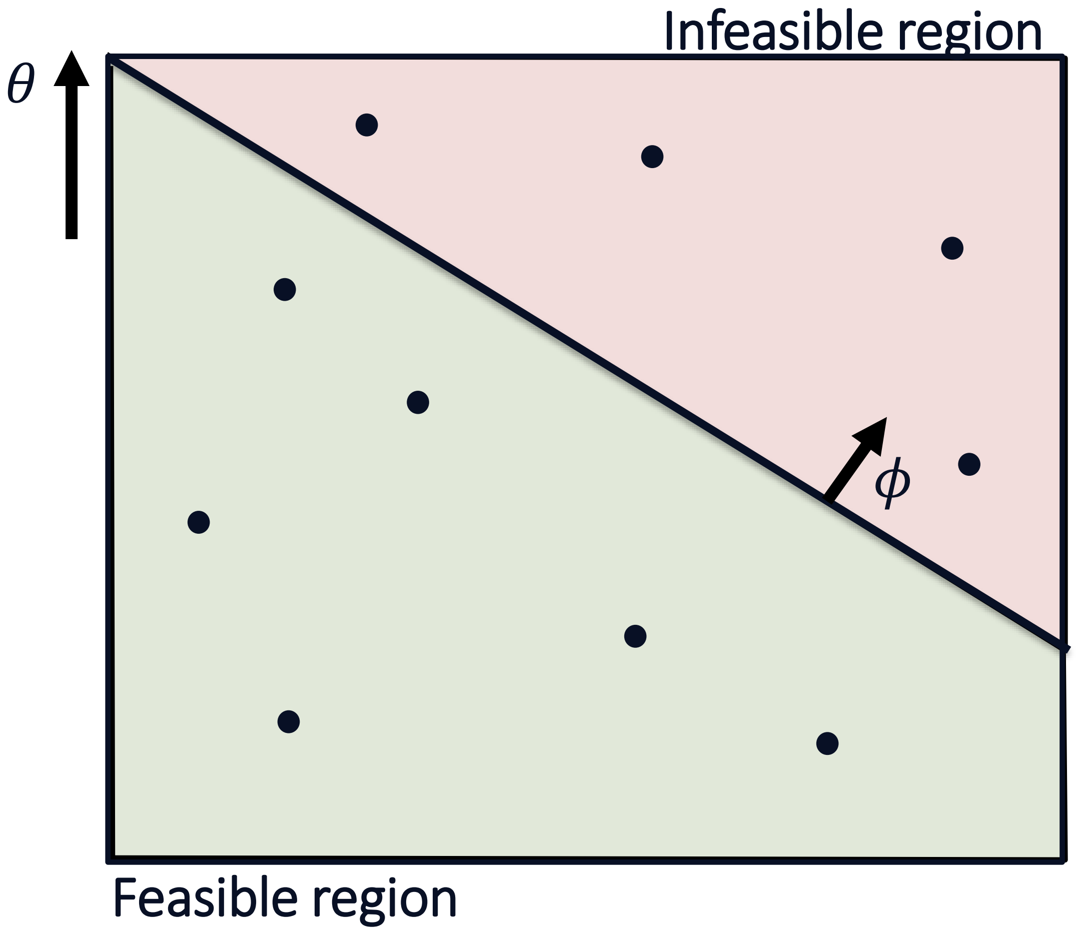


Constrained linear best-arm identification (CBAI)

Known set of arms $\mathcal{X} \subset \mathbb{R}^d$

Known reward parameter $\theta \in \mathbb{R}^d$

Unknown constraint parameter $\phi \in \mathbb{R}^d$



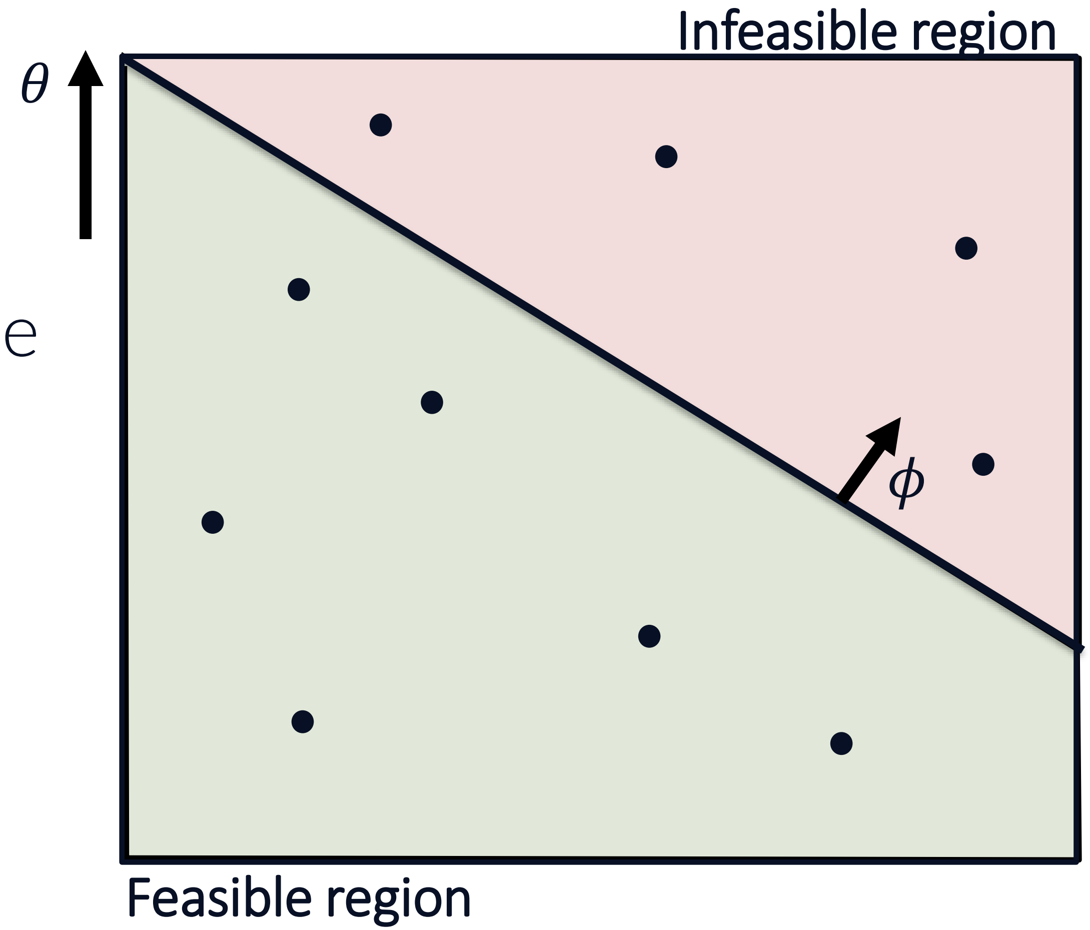
Constrained linear best-arm identification (CBAI)

Known set of arms $\mathcal{X} \subset \mathbb{R}^d$

Known reward parameter $\theta \in \mathbb{R}^d$

Unknown constraint parameter $\phi \in \mathbb{R}^d$

In each iteration, choose $x \in \mathcal{X}$ & observe $\phi^T x$ plus noise



Constrained linear best-arm identification (CBAI)

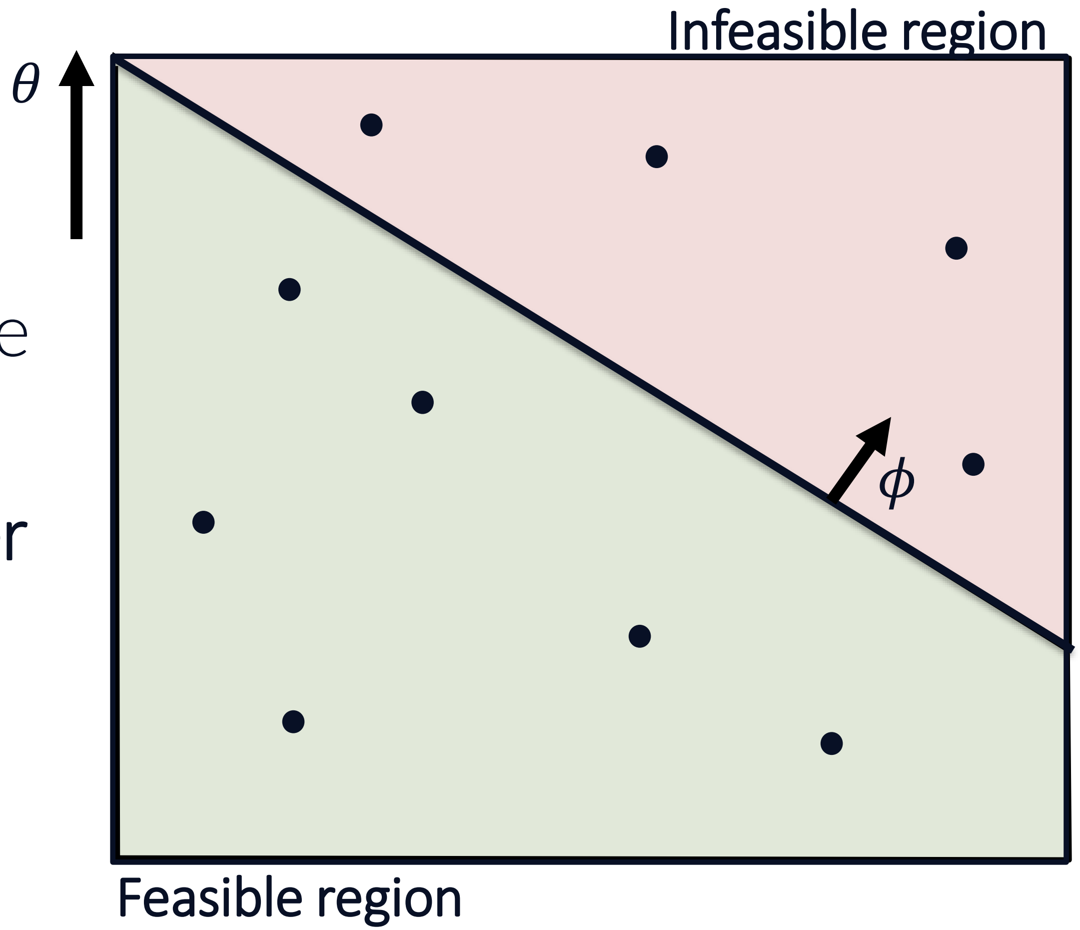
Known set of arms $\mathcal{X} \subset \mathbb{R}^d$

Known reward parameter $\theta \in \mathbb{R}^d$

Unknown constraint parameter $\phi \in \mathbb{R}^d$

In each iteration, choose $x \in \mathcal{X}$ & observe $\phi^T x$ plus noise

→ Need to estimate constraint parameter



Constrained linear best-arm identification (CBAI)

Known set of arms $\mathcal{X} \subset \mathbb{R}^d$

Known reward parameter $\theta \in \mathbb{R}^d$

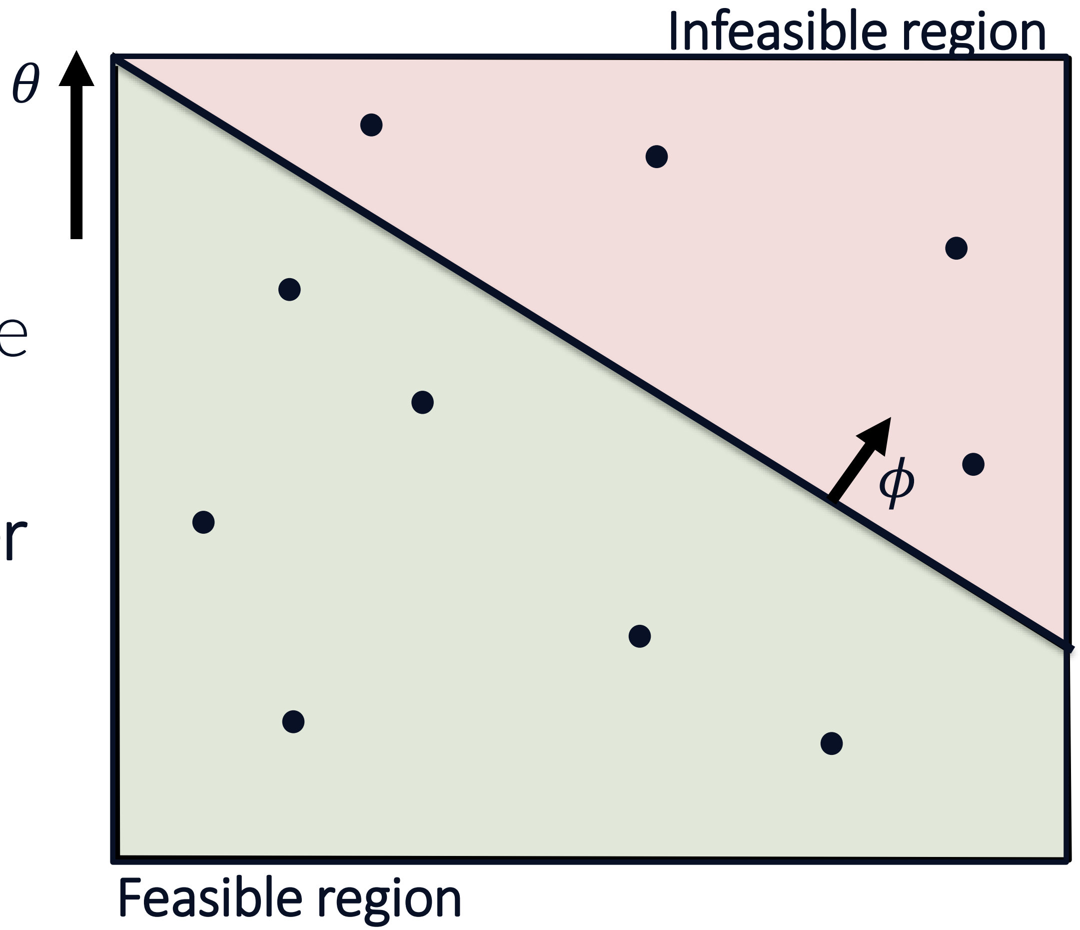
Unknown constraint parameter $\phi \in \mathbb{R}^d$

In each iteration, choose $x \in \mathcal{X}$ & observe $\phi^T x$ plus noise

→ Need to estimate constraint parameter

Goal: Find $x^* \in \underset{x \in \mathcal{X}, \phi^T x \leq 0}{\operatorname{argmax}} \; \theta^T x$

in as few iterations as possible



Constrained linear best-arm identification (CBAI)

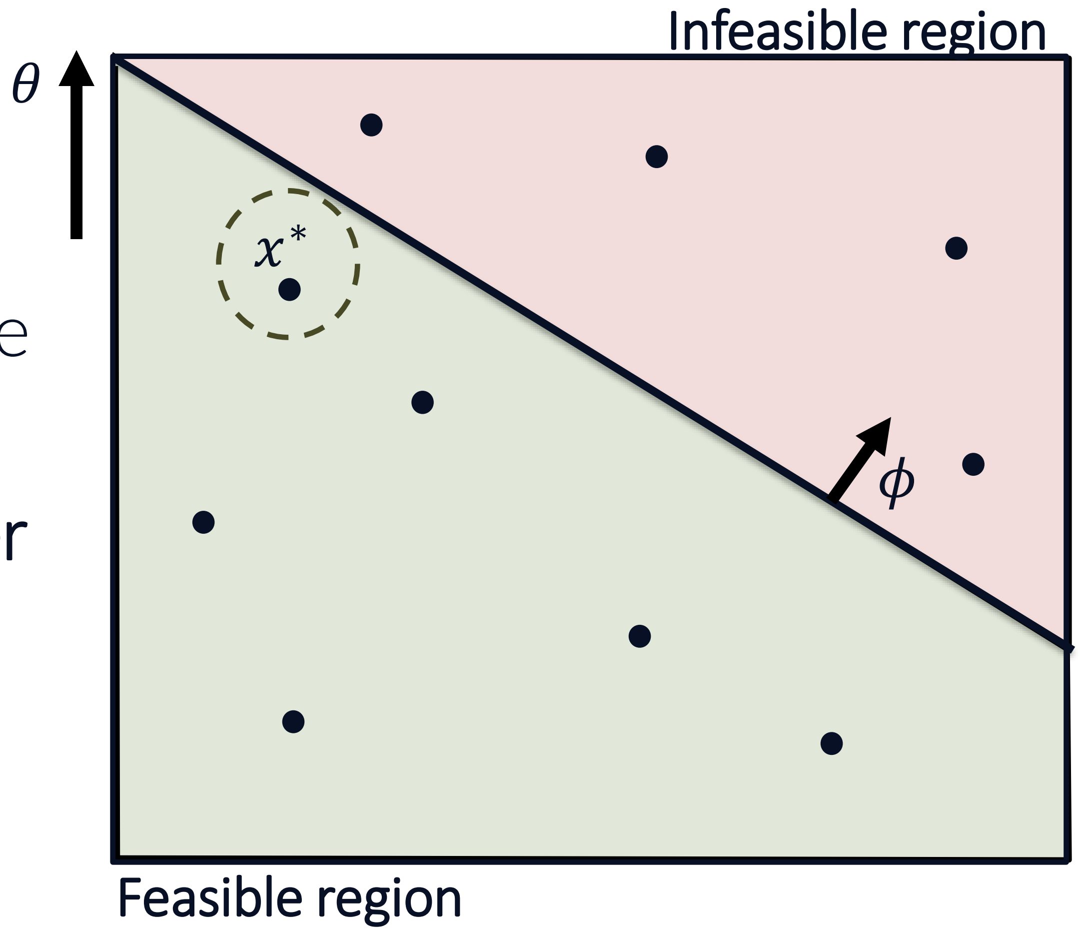
Known set of arms $\mathcal{X} \subset \mathbb{R}^d$

Known reward parameter $\theta \in \mathbb{R}^d$

Unknown constraint parameter $\phi \in \mathbb{R}^d$

In each iteration, choose $x \in \mathcal{X}$ & observe $\phi^T x$ plus noise

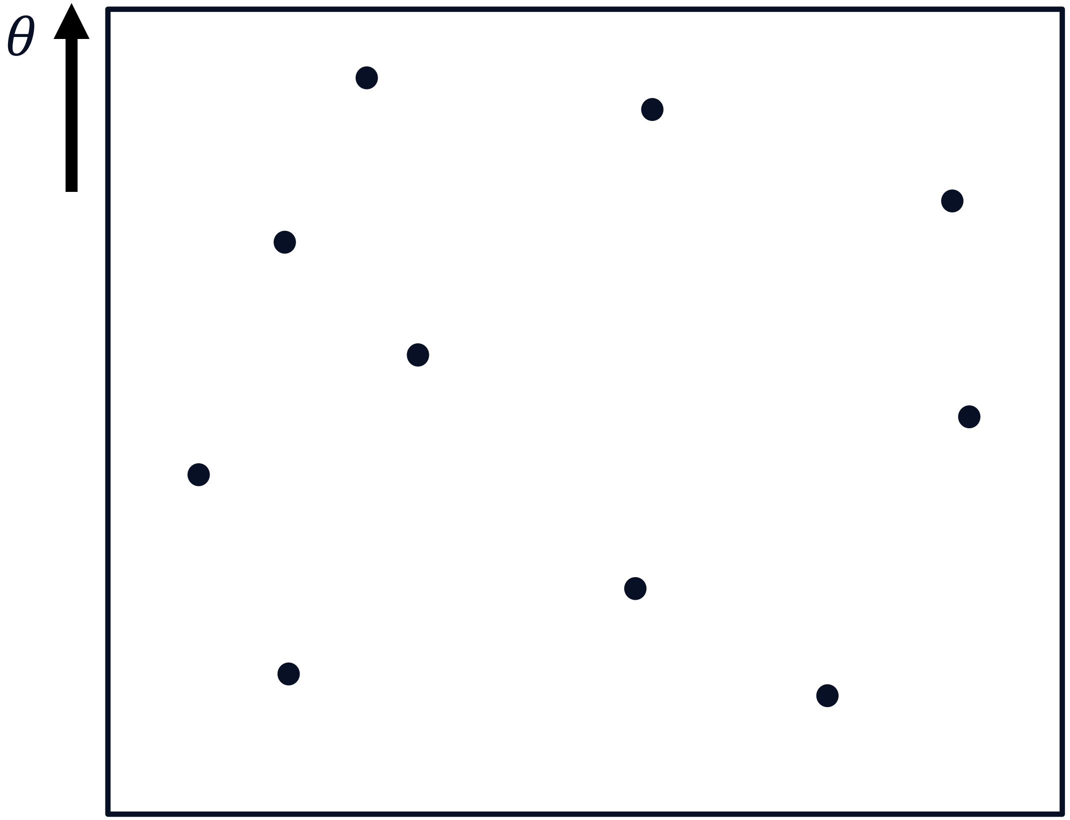
→ Need to estimate constraint parameter



Goal: Find $x^* \in \underset{x \in \mathcal{X}, \phi^T x \leq 0}{\operatorname{argmax}} \ \theta^T x$

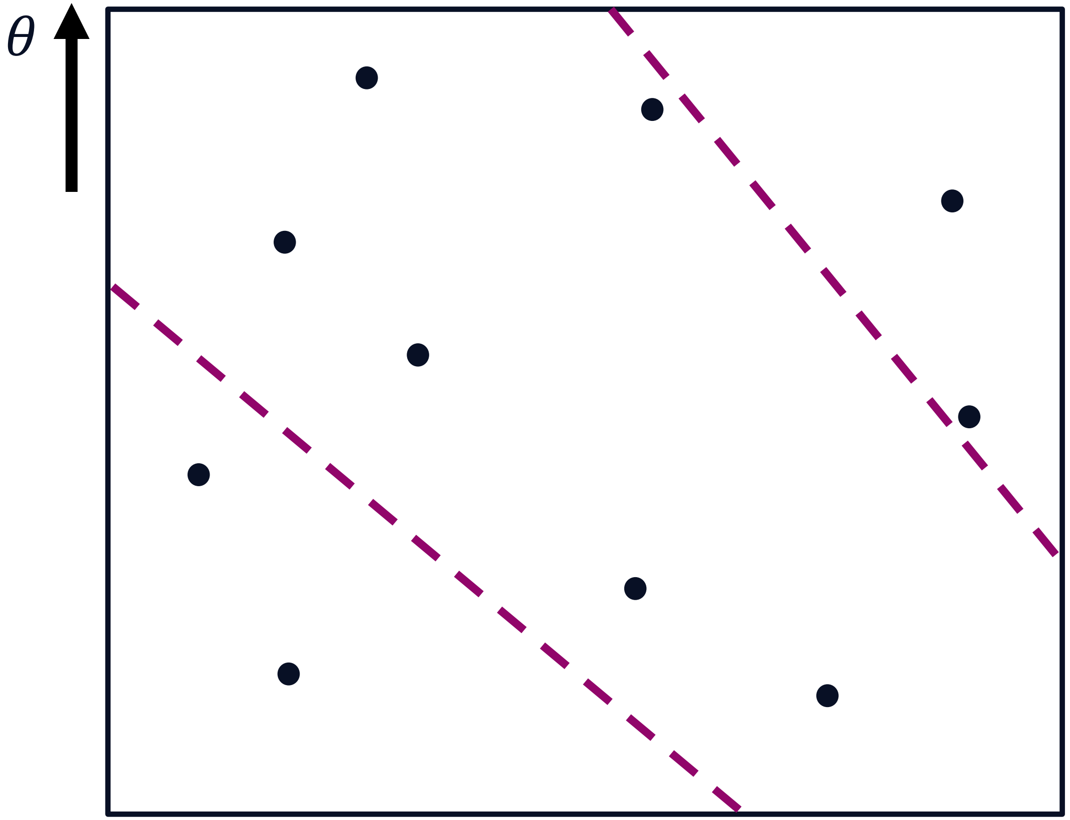
in as few iterations as possible

Adaptive constraint learning (ACOL)



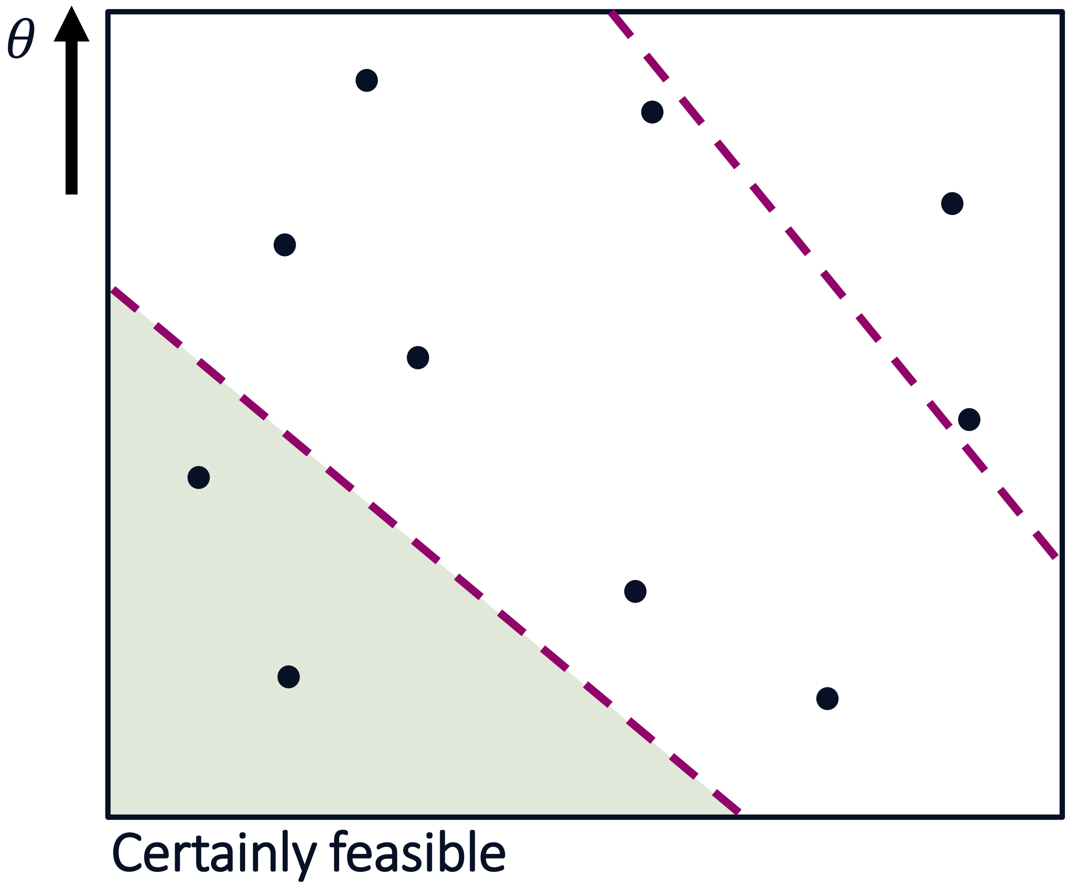
Adaptive constraint learning (ACOL)

1. Estimate confidence regions around the estimated constraint



Adaptive constraint learning (ACOL)

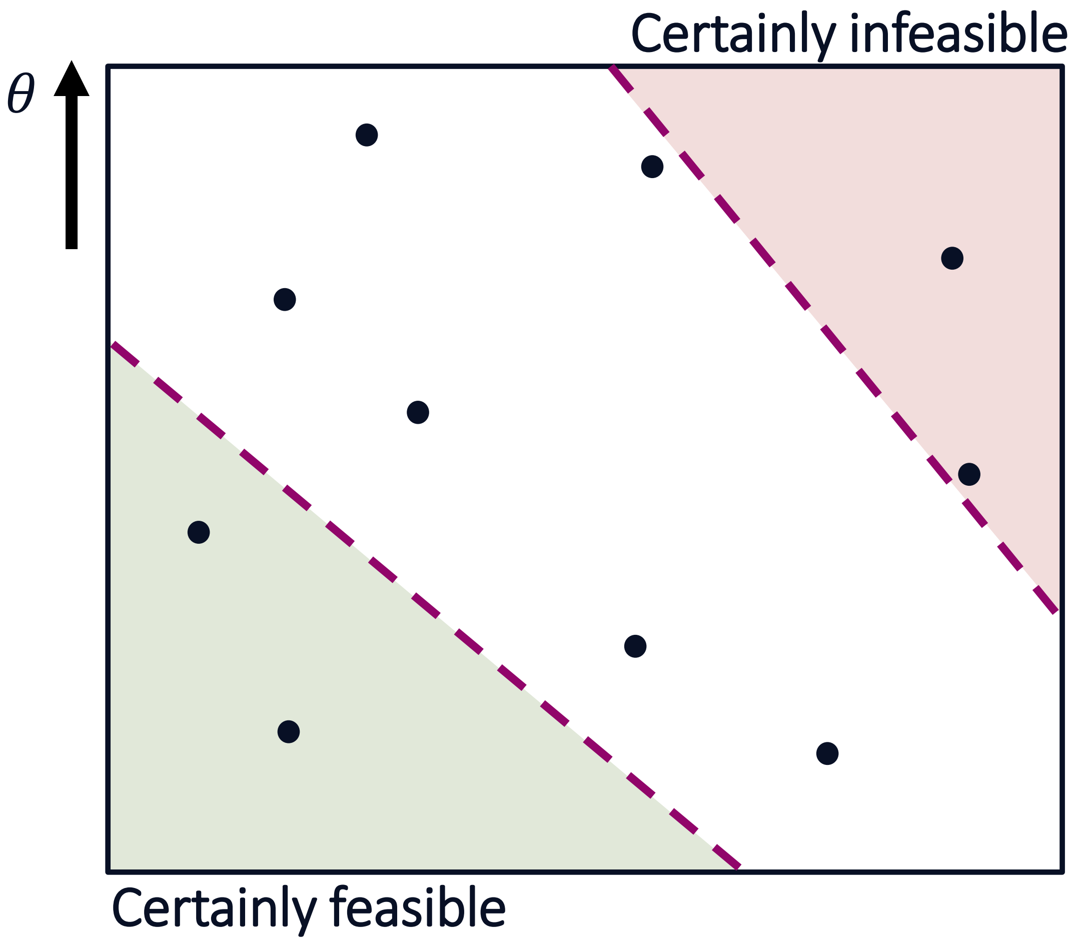
1. Estimate confidence regions around the estimated constraint
2. Determine certainly feasible arms



„Certainly“ ≙ with high probability

Adaptive constraint learning (ACOL)

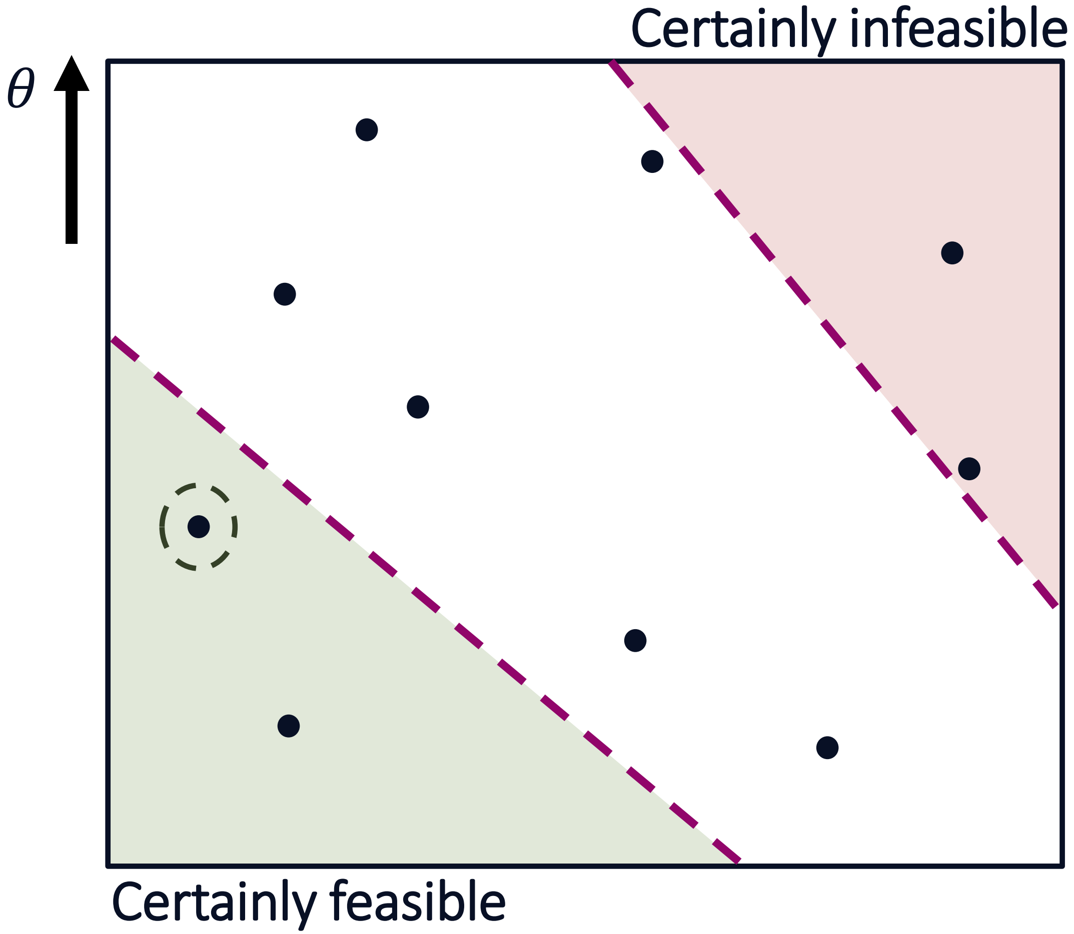
1. Estimate confidence regions around the estimated constraint
2. Determine certainly feasible arms
3. Determine certainly infeasible arms



„Certainly“ ≙ with high probability

Adaptive constraint learning (ACOL)

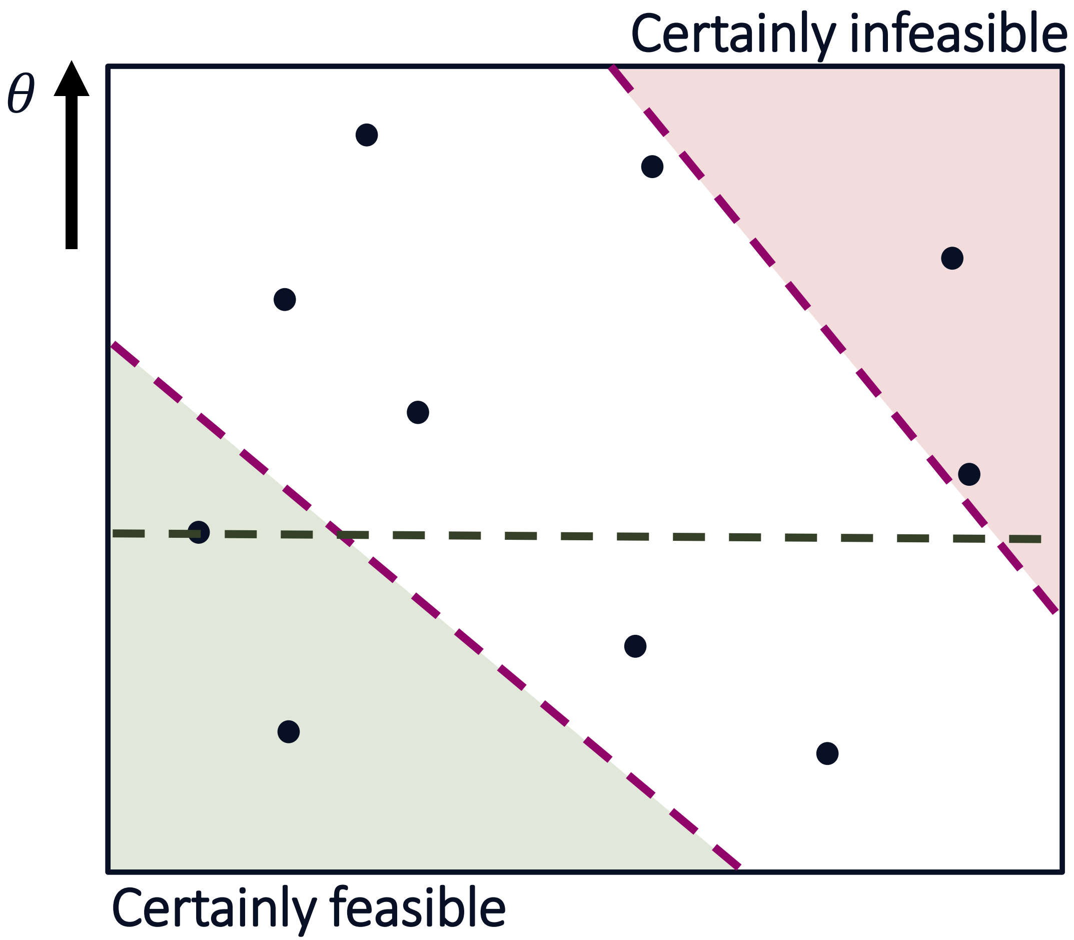
1. Estimate confidence regions around the estimated constraint
2. Determine certainly feasible arms
3. Determine certainly infeasible arms



„Certainly“ ≙ with high probability

Adaptive constraint learning (ACOL)

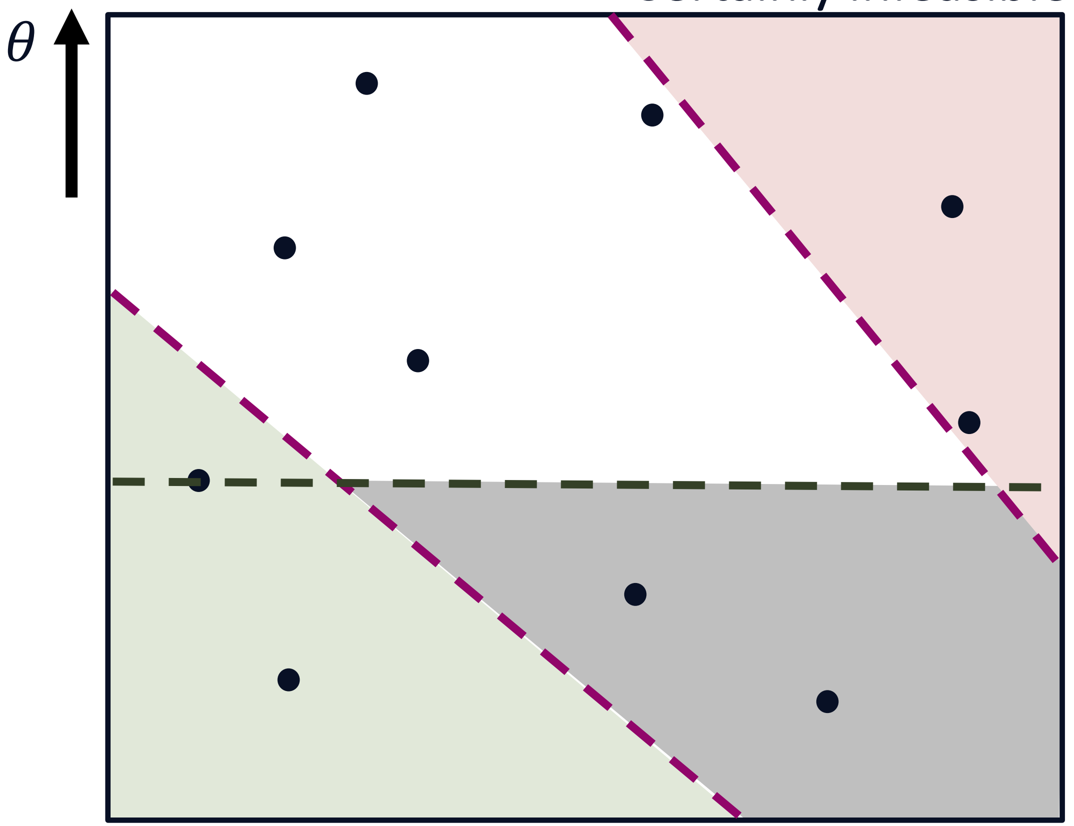
1. Estimate confidence regions around the estimated constraint
2. Determine certainly feasible arms
3. Determine certainly infeasible arms



„Certainly" ≙ with high probability

Adaptive constraint learning (ACOL)

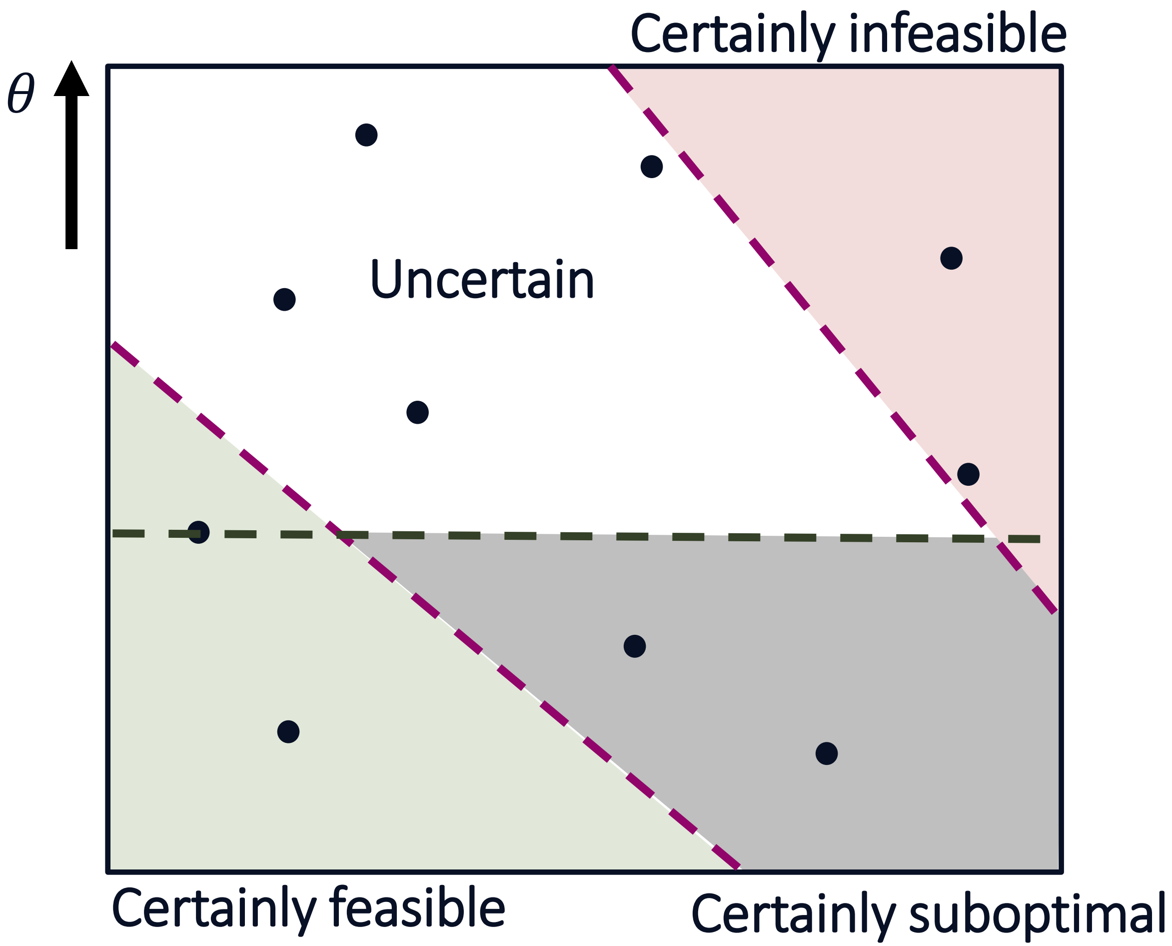
1. Estimate confidence regions around the estimated constraint
2. Determine certainly feasible arms
3. Determine certainly infeasible arms
4. Determine certainly suboptimal arms



„Certainly“ ≙ with high probability

Adaptive constraint learning (ACOL)

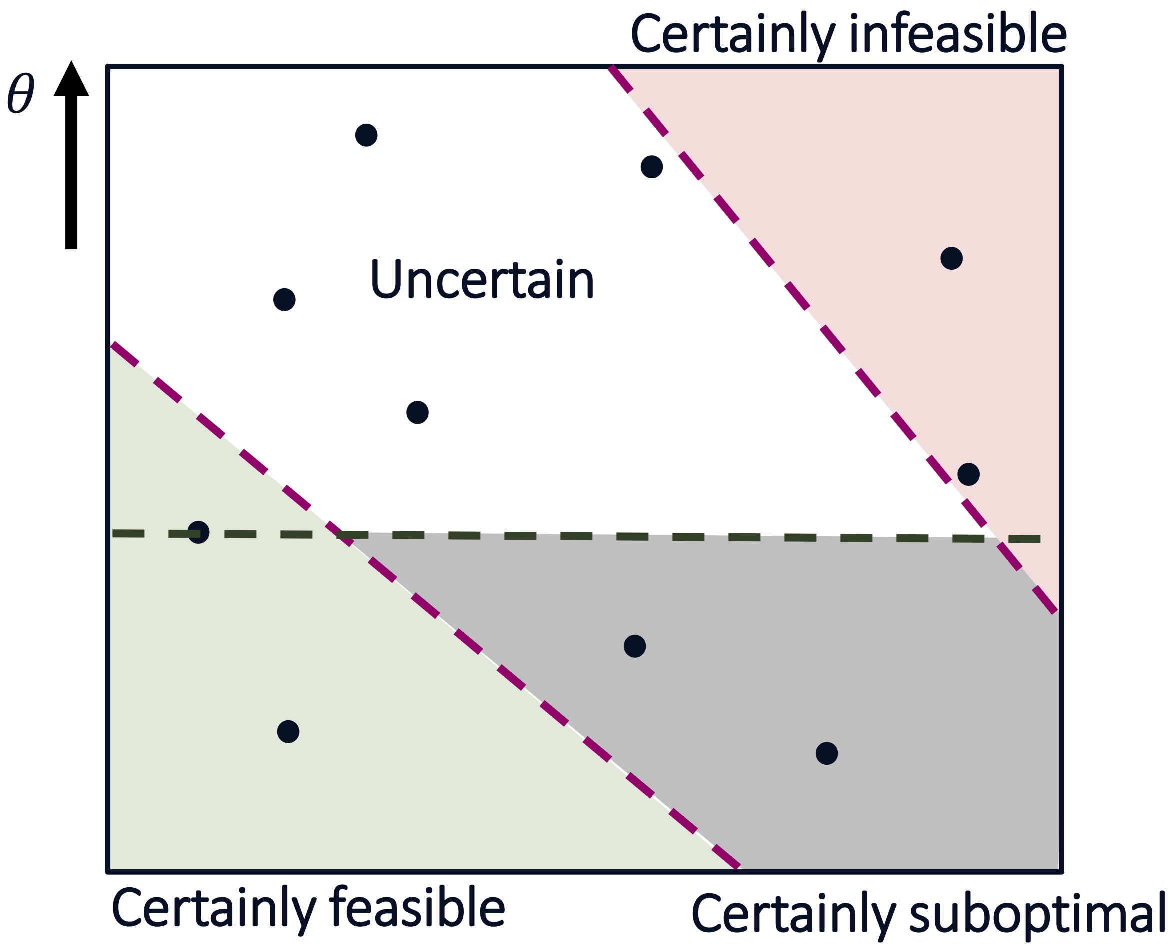
1. Estimate confidence regions around the estimated constraint
2. Determine certainly feasible arms
3. Determine certainly infeasible arms
4. Determine certainly suboptimal arms
5. Determine uncertain arms



„Certainly“ ≙ with high probability

Adaptive constraint learning (ACOL)

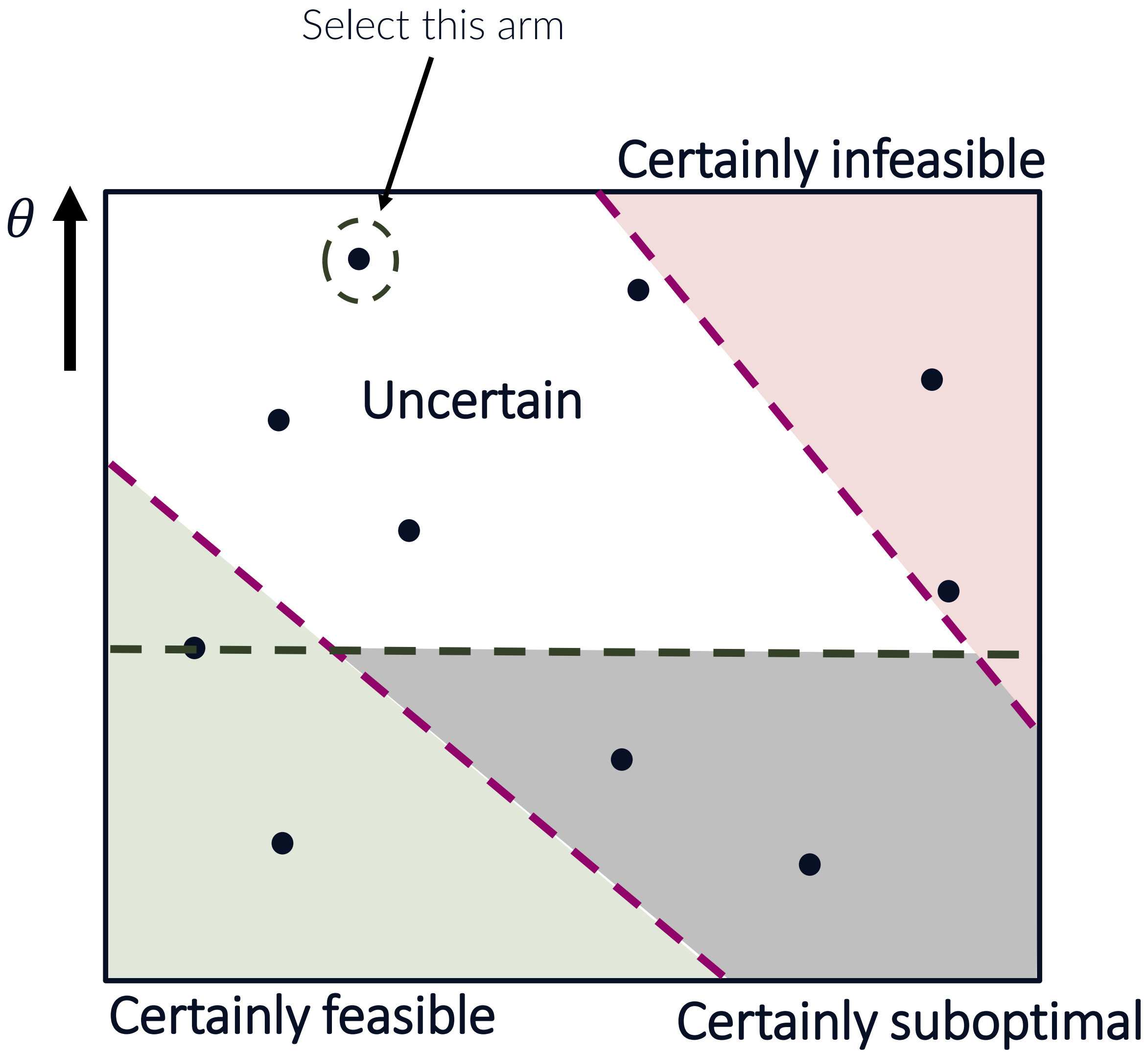
1. Estimate confidence regions around the estimated constraint
2. Determine certainly feasible arms
3. Determine certainly infeasible arms
4. Determine certainly suboptimal arms
5. Determine uncertain arms
6. Select arms to reduce uncertainty about uncertain arms



„Certainly“ ≙ with high probability

Adaptive constraint learning (ACOL)

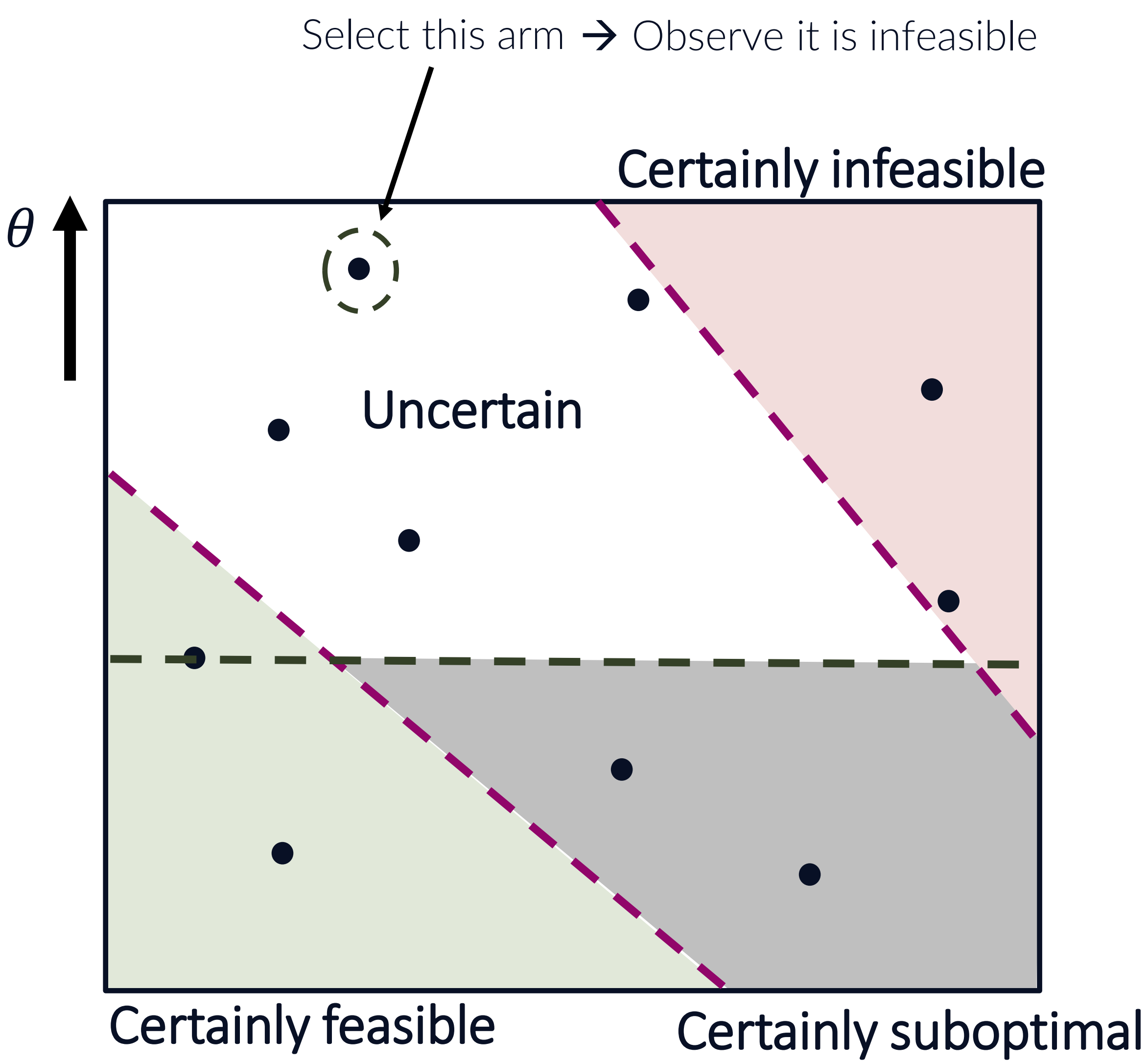
1. Estimate confidence regions around the estimated constraint
2. Determine certainly feasible arms
3. Determine certainly infeasible arms
4. Determine certainly suboptimal arms
5. Determine uncertain arms
6. Select arms to reduce uncertainty about uncertain arms



„Certainly" ≙ with high probability

Adaptive constraint learning (ACOL)

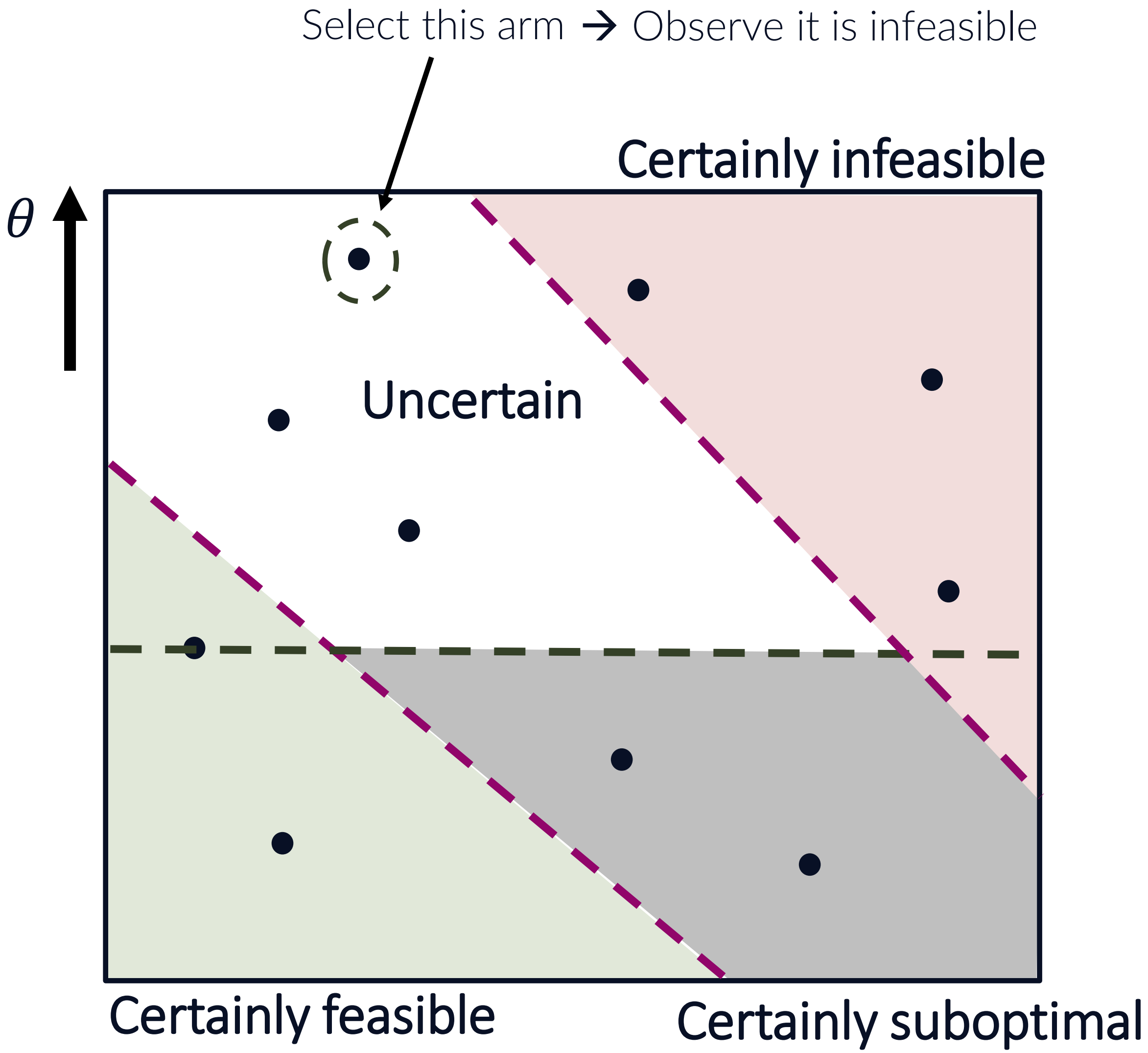
1. Estimate confidence regions around the estimated constraint
2. Determine certainly feasible arms
3. Determine certainly infeasible arms
4. Determine certainly suboptimal arms
5. Determine uncertain arms
6. Select arms to reduce uncertainty about uncertain arms



„Certainly" ≙ with high probability

Adaptive constraint learning (ACOL)

1. Estimate confidence regions around the estimated constraint
2. Determine certainly feasible arms
3. Determine certainly infeasible arms
4. Determine certainly suboptimal arms
5. Determine uncertain arms
6. Select arms to reduce uncertainty about uncertain arms



„Certainly“ ≙ with high probability

We provide a tight sample complexity guarantee for ACOL

CBAI lower bound (informal)

The number of samples necessary to solve a CBAI problem is lower bounded by

$$\min_{\lambda} \max_{x \in \mathcal{X}_{\theta}^{\geq}(x^*)} \frac{|x|_{A_\lambda^{-1}}^2}{(\phi^T x)^2}$$

We provide a tight sample complexity guarantee for ACOL

CBAI lower bound (informal)

The number of samples necessary to solve a CBAI problem is lower bounded by

$$\min_{\lambda} \max_{x \in \mathcal{X}_\theta^{\geq}(x^*)} \frac{|x|^2_{A_\lambda^{-1}}}{(\phi^T x)^2}$$

ACOL sample complexity (informal)

ACOL returns the correct optimal arm with a number of samples upper bounded by

We provide a tight sample complexity guarantee for ACOL

CBAI lower bound (informal)

The number of samples necessary to solve a CBAI problem is lower bounded by

$$\min_{\lambda} \max_{x \in \mathcal{X}_{\theta}^{\gtrless}(x^*)} \frac{|x|^2_{A_\lambda^{-1}}}{(\phi^T x)^2}$$

ACOL sample complexity (informal)

ACOL returns the correct optimal arm with a number of samples upper bounded by

$$\min_{\lambda} \max_{x \in \mathcal{X}} \frac{|x|^2_{A_\lambda^{-1}}}{(\phi^T x)^2}$$

We provide a tight sample complexity guarantee for ACOL

CBAI lower bound (informal)

The number of samples necessary to solve a CBAI problem is lower bounded by

$$\min_{\lambda} \max_{x \in \mathcal{X}_{\theta}^{\gtrless}(x^*)} \frac{|x|_{A_\lambda^{-1}}^2}{(\phi^T x)^2}$$

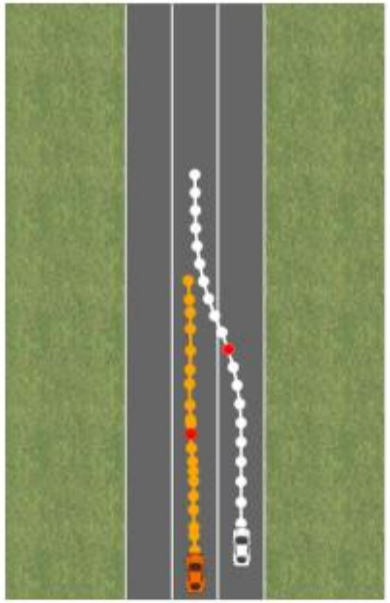
ACOL sample complexity (informal)

ACOL returns the correct optimal arm with a number of samples upper bounded by

$$\min_{\lambda} \max_{x \in \mathcal{X}} \frac{|x|_{A_\lambda^{-1}}^2}{(\phi^T x)^2}$$

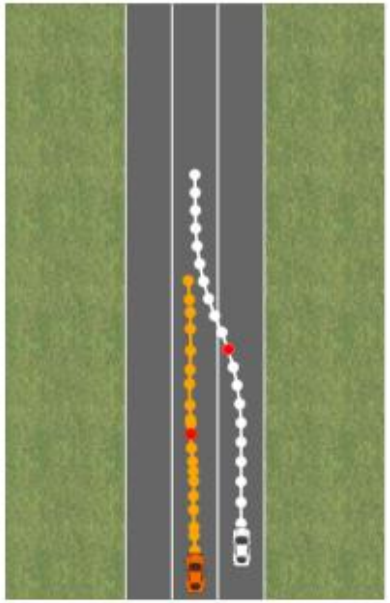
Adaptive constraint learning works well in simulations

Adaptive constraint learning works well in simulations



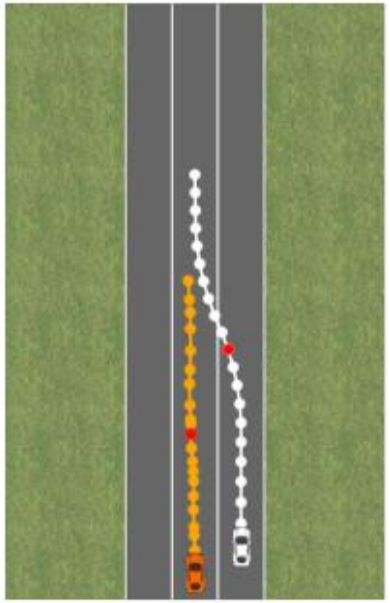
- Driving simulation with known reward but unknown constraints

Adaptive constraint learning works well in simulations

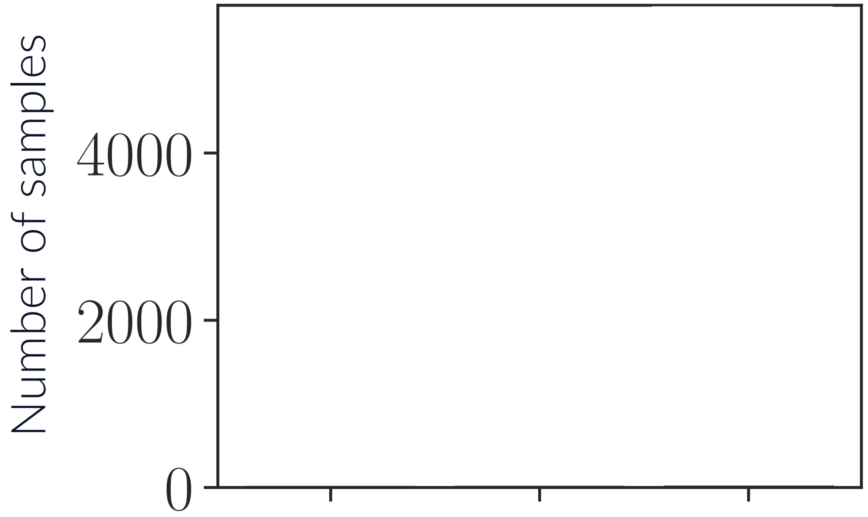


- Driving simulation with known reward but unknown constraints
- We obtain binary observations whether a trajectory is feasible

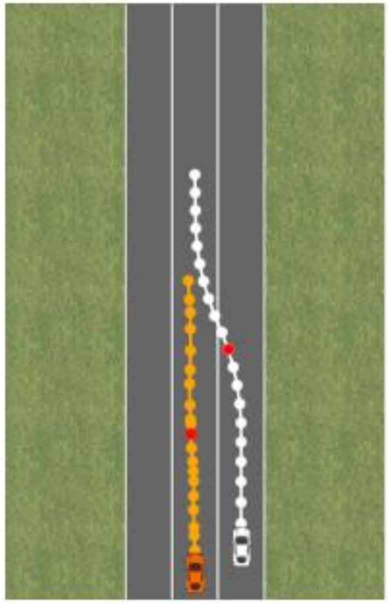
Adaptive constraint learning works well in simulations



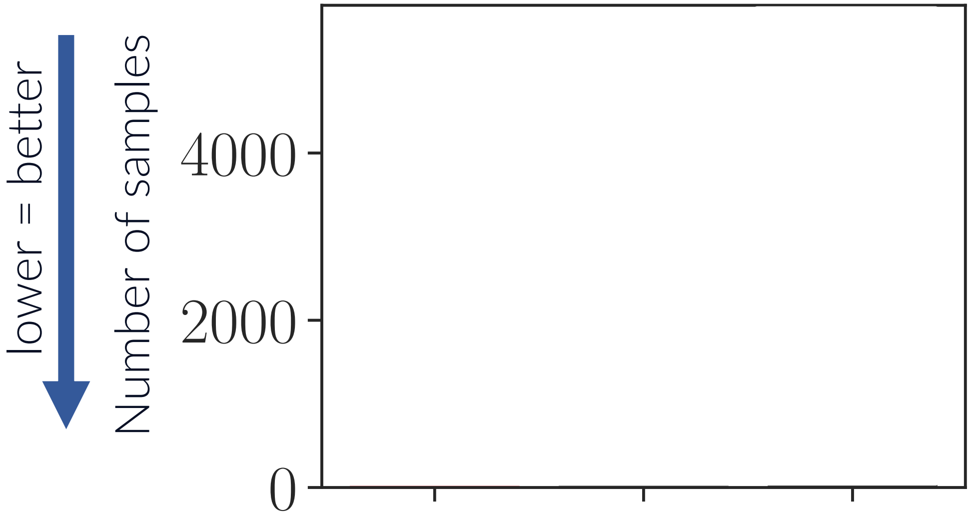
- Driving simulation with known reward but unknown constraints
- We obtain binary observations whether a trajectory is feasible
- How many samples until we can identify the best controller?



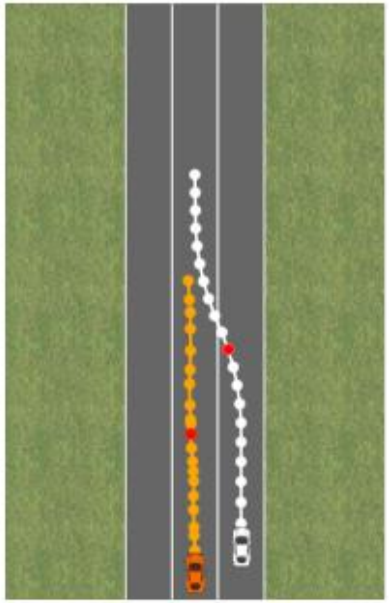
Adaptive constraint learning works well in simulations



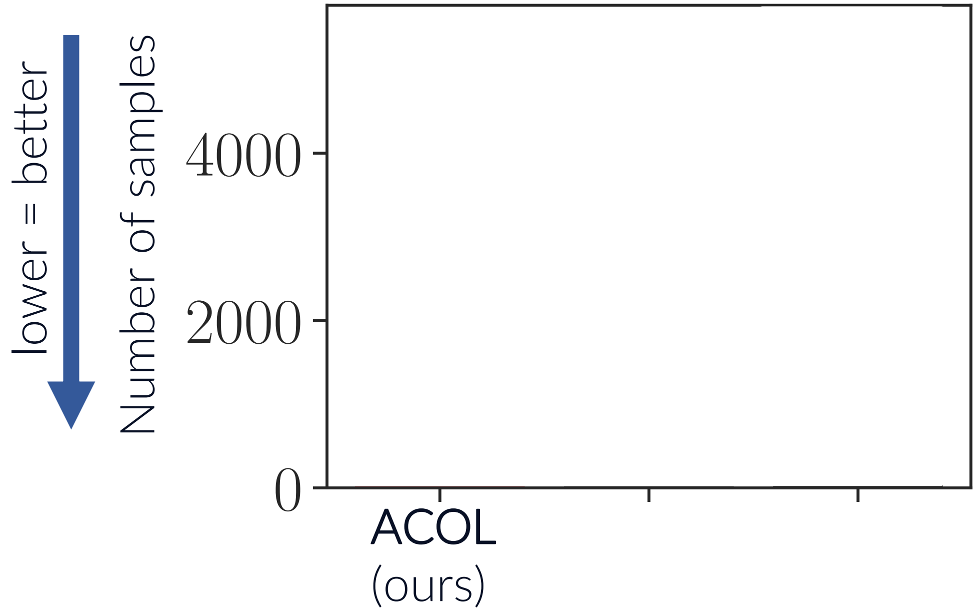
- Driving simulation with known reward but unknown constraints
- We obtain binary observations whether a trajectory is feasible
- How many samples until we can identify the best controller?



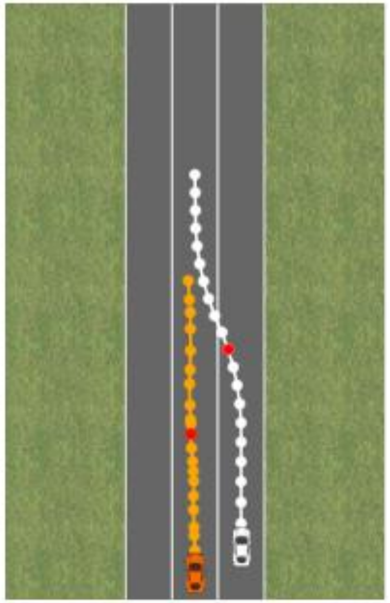
Adaptive constraint learning works well in simulations



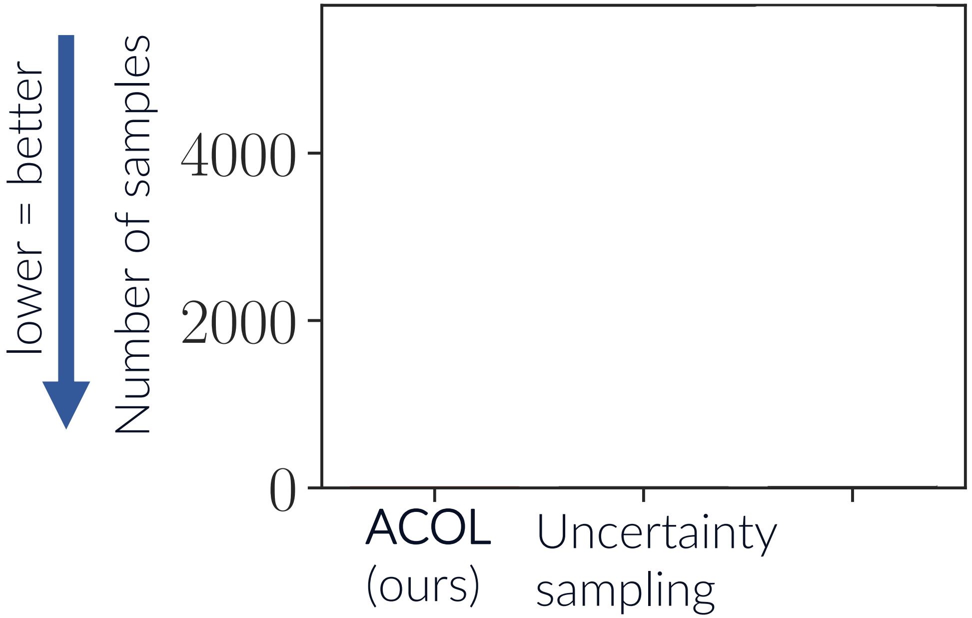
- Driving simulation with known reward but unknown constraints
- We obtain binary observations whether a trajectory is feasible
- How many samples until we can identify the best controller?



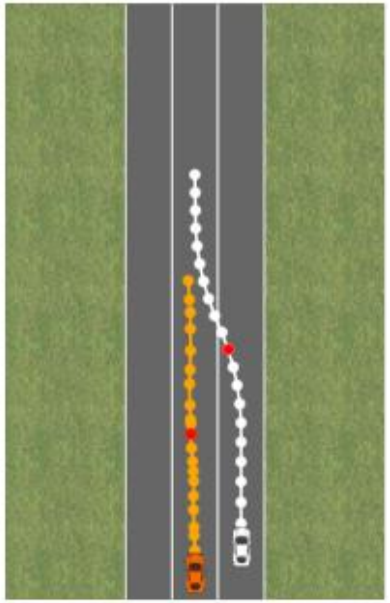
Adaptive constraint learning works well in simulations



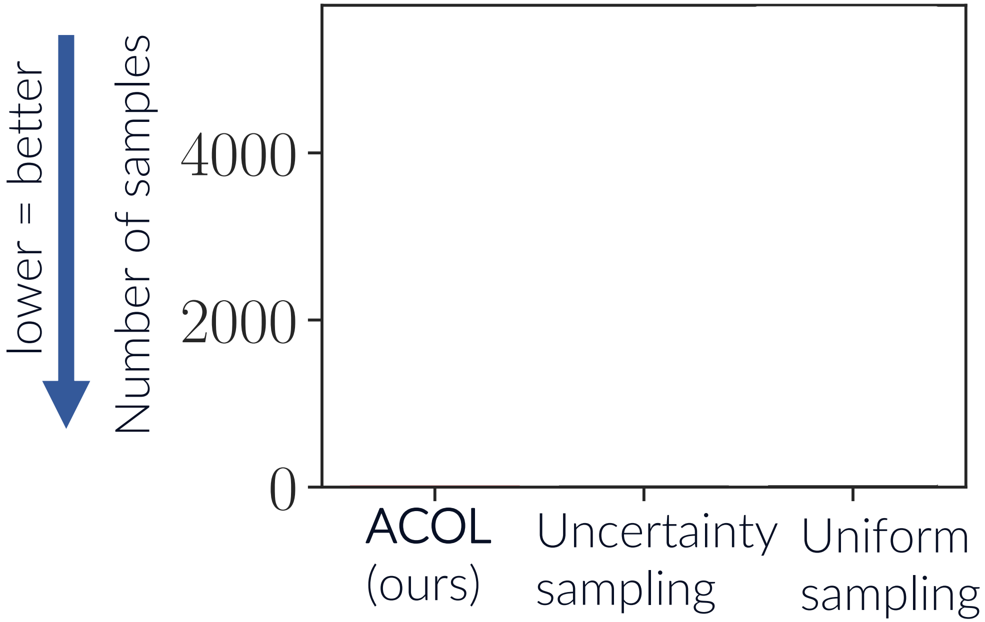
- Driving simulation with known reward but unknown constraints
- We obtain binary observations whether a trajectory is feasible
- How many samples until we can identify the best controller?



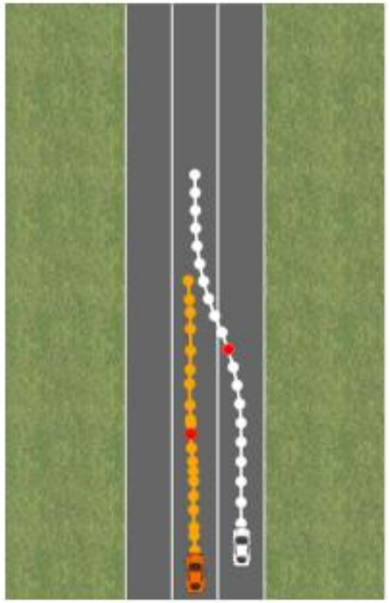
Adaptive constraint learning works well in simulations



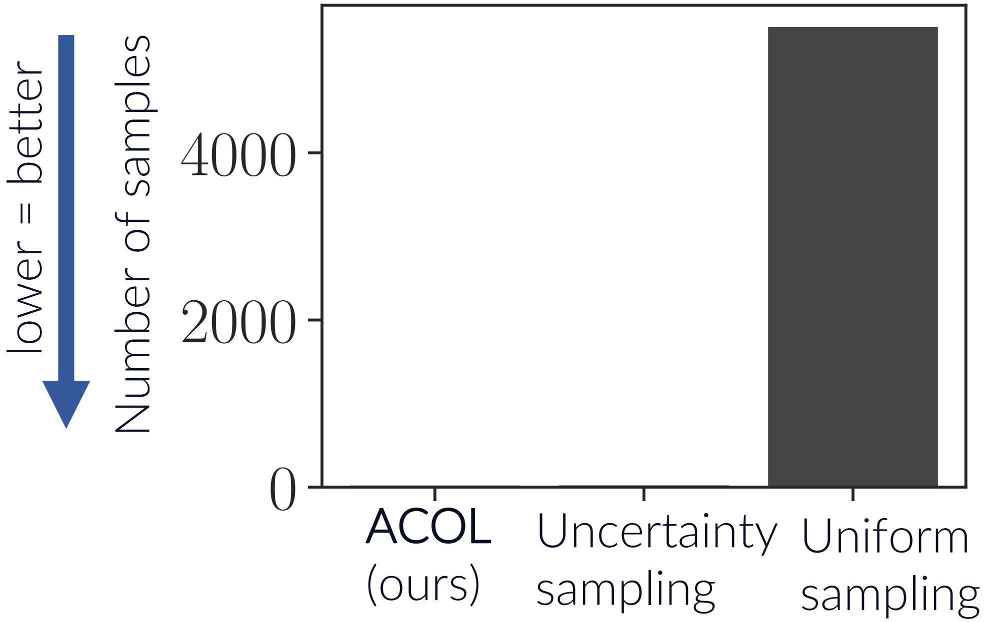
- Driving simulation with known reward but unknown constraints
- We obtain binary observations whether a trajectory is feasible
- How many samples until we can identify the best controller?



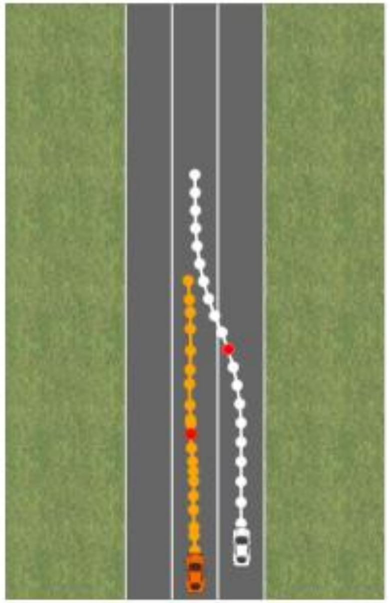
Adaptive constraint learning works well in simulations



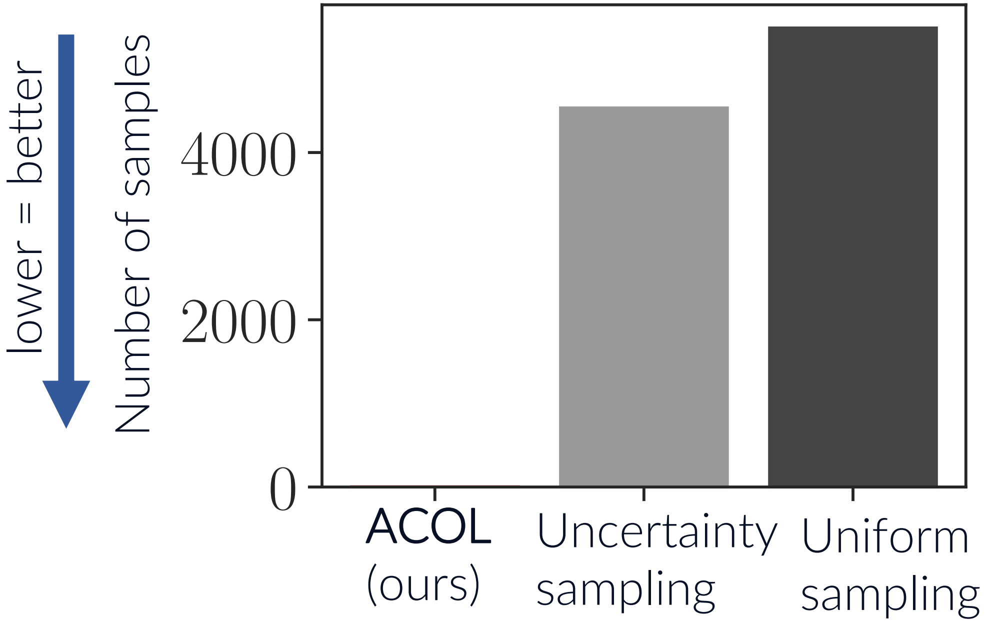
- Driving simulation with known reward but unknown constraints
- We obtain binary observations whether a trajectory is feasible
- How many samples until we can identify the best controller?



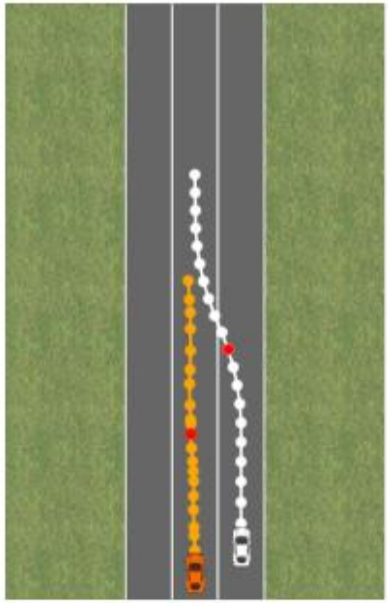
Adaptive constraint learning works well in simulations



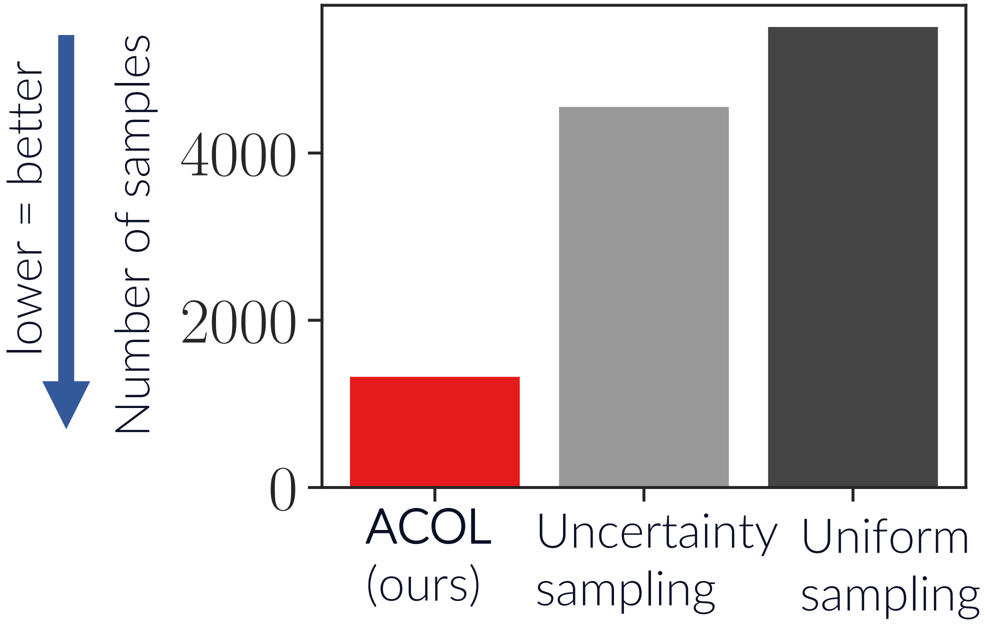
- Driving simulation with known reward but unknown constraints
- We obtain binary observations whether a trajectory is feasible
- How many samples until we can identify the best controller?



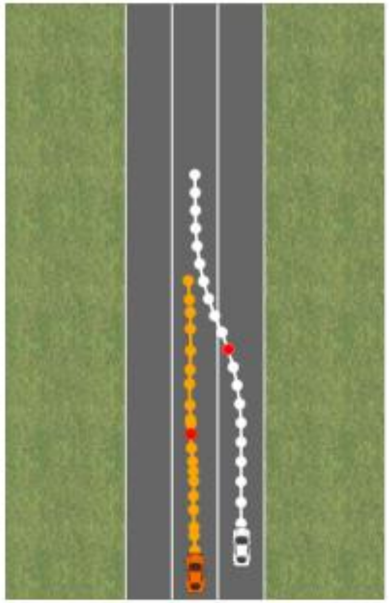
Adaptive constraint learning works well in simulations



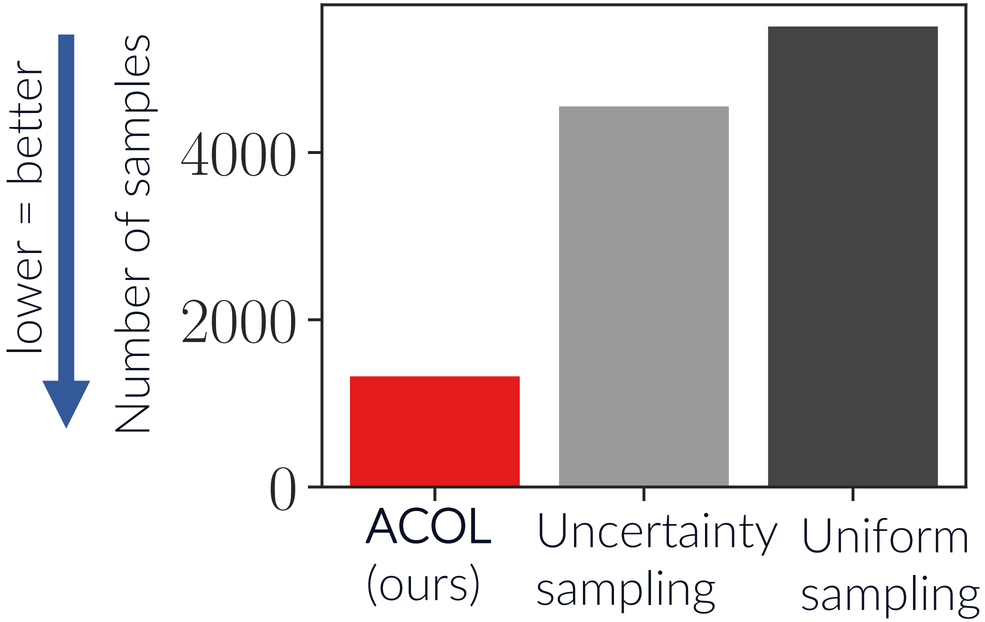
- Driving simulation with known reward but unknown constraints
- We obtain binary observations whether a trajectory is feasible
- How many samples until we can identify the best controller?



Adaptive constraint learning works well in simulations

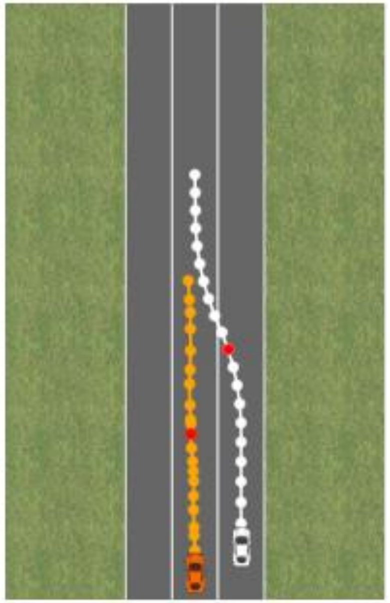


- Driving simulation with known reward but unknown constraints
- We obtain binary observations whether a trajectory is feasible
- How many samples until we can identify the best controller?

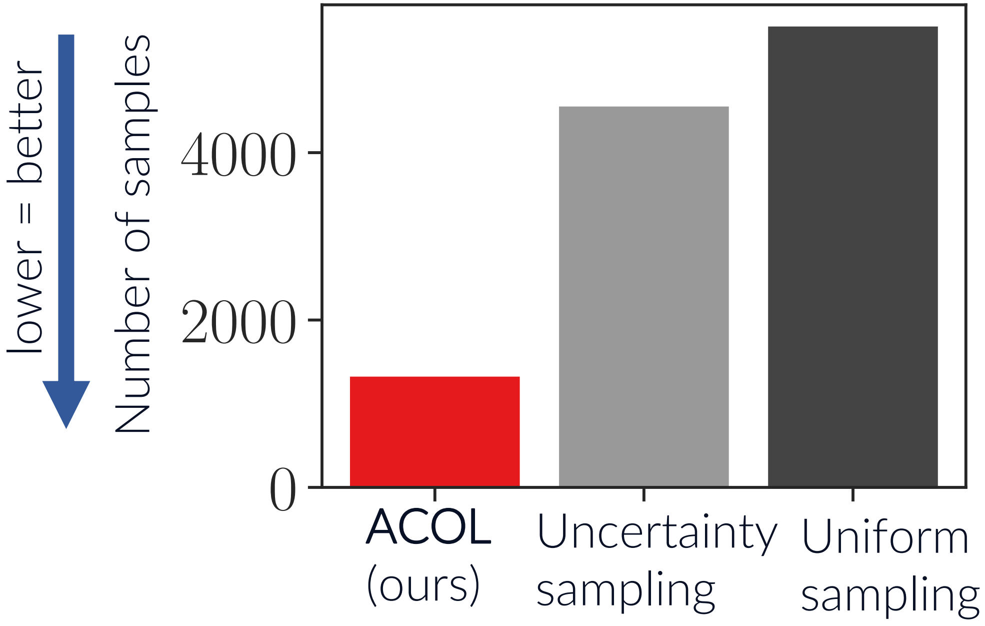


- Constraints are a natural and robust way to represent human preferences

Adaptive constraint learning works well in simulations

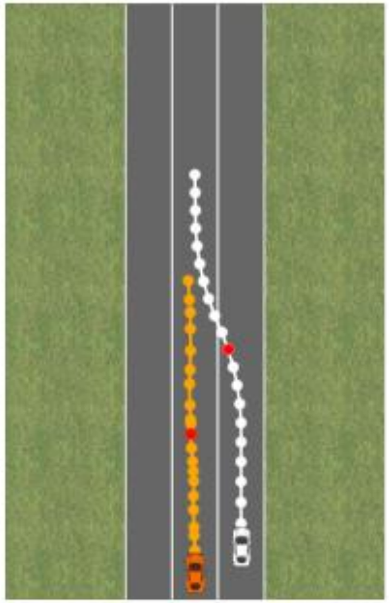


- Driving simulation with known reward but unknown constraints
- We obtain binary observations whether a trajectory is feasible
- How many samples until we can identify the best controller?

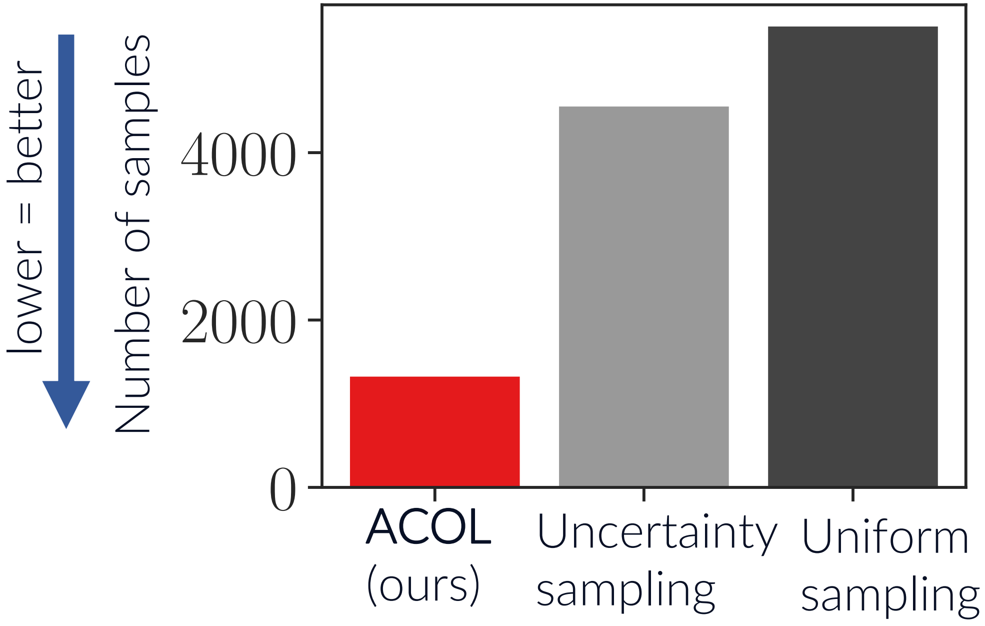


- Constraints are a natural and robust way to represent human preferences
- Adaptive Constraint Learning can efficiently learn constraints in linear bandits

Adaptive constraint learning works well in simulations



- Driving simulation with known reward but unknown constraints
- We obtain binary observations whether a trajectory is feasible
- How many samples until we can identify the best controller?



- Constraints are a natural and robust way to represent human preferences
- Adaptive Constraint Learning can efficiently learn constraints in linear bandits

https://arxiv.org/abs/2206.05255