GROUPE RENAULT

Reinforcement Learning for Safe Decision-Making in Autonomous Driving

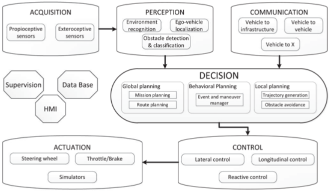
Edouard Leurent[1,2,3],
Odalric-Ambrym Maillard[1],
Denis Efimov[2]

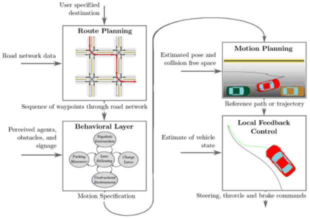
[1]Inria SequeL,
[2]Inria Valse,
[3]Renault Group

01

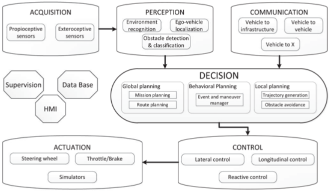
Motivation and Scope

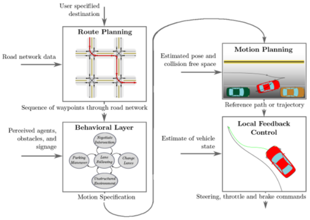
Classic Autonomous Driving Pipeline





Classic Autonomous Driving Pipeline

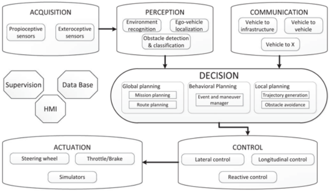


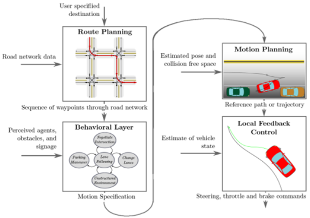


(Bold?) Claim

If we remove the humans on the road, the problem becomes easy.

Classic Autonomous Driving Pipeline





(Bold?) Claim

If we remove the humans on the road, the problem becomes easy.

- ✓ Even with obstacles, partial observability, disturbances, etc.
- ✓ The problems of Route Planning, Motion Planning, Local Feedback Control are basically solved.

✗ We focus instead on the (arguably) harder challenge:

Behavioural Planning

What we have

- In practice, often a hand-crafted rule-based system (FSM).
- Won't scale to complex scenes

What we want

- Handle human agents with uncertain behaviours
- Handle the interactions between agents

↳ We turn to learning-based approaches

Markov Decision Processes

1. Observe state $s \in S$;

Markov Decision Processes

1. Observe state $s \in S$;
2. Pick action $a \in A$ according to our policy $\pi(a|s)$;

Markov Decision Processes

1. Observe state $s \in S$;
2. Pick action $a \in A$ according to our policy $\pi(a|s)$;
3. Transition to a next state $s' \sim \mathbb{P}\left(s'|s, a\right)$;

Markov Decision Processes

1. Observe state $s \in S$;
2. Pick action $a \in A$ according to our policy $\pi(a|s)$;
3. Transition to a next state $s' \sim \mathbb{P}\left(s'|s, a\right)$;
4. Receive a reward r.

Markov Decision Processes

1. Observe state $s \in S$;
2. Pick action $a \in A$ according to our policy $\pi(a|s)$;
3. Transition to a next state $s' \sim \mathbb{P}(s'|s, a)$;
4. Receive a reward r.

Objective: maximise $V = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$

Markov Decision Processes

1. Observe state $s \in S$;
2. Pick action $a \in A$ according to our policy $\pi(a|s)$;
3. Transition to a next state $s' \sim \mathbb{P}\left(s'|s, a\right)$;
4. Receive a reward r.

Objective: maximise $V = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r_t\right]$

- **States**: Ground truth for vehicles, roads, signals, etc.
  ↳ Continuous
- **Actions**: Semantic decisions: change lane, yield, pass, etc.
  ↳ Discrete

Model-free

1. Directly optimise $\pi(a|s)$ through policy evaluation and policy improvement

Model-free

1. Directly optimise $\pi(a|s)$ through policy evaluation and policy improvement

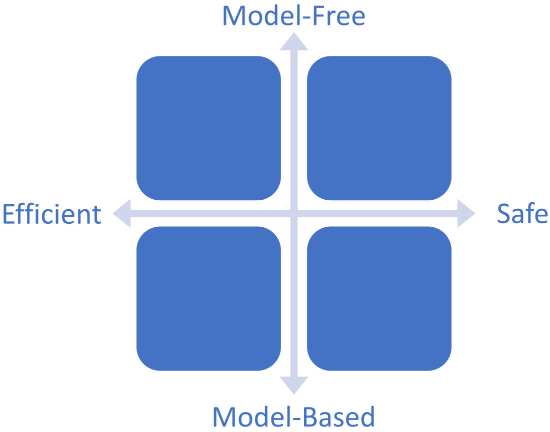
Model-based

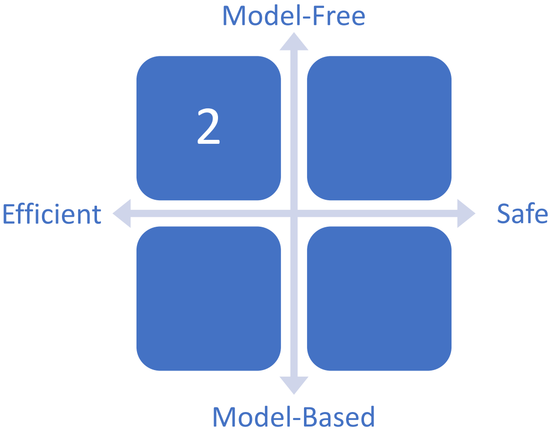
1. Learn a model for the dynamics $\hat{T}(s_{t+1}|s_t, a_t)$,
2. (*Planning*) Leverage it to compute

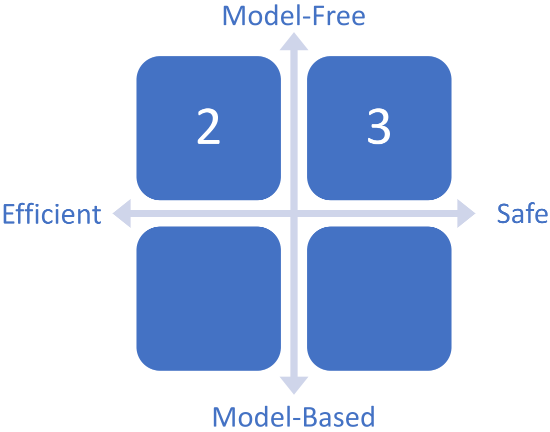
$$\max_{\pi} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \,\middle|\, a_t \sim \pi(s_t), s_{t+1} \sim \hat{T}(s_t, a_t)\right]$$

\+ Better sample efficiency, interpretability, priors.

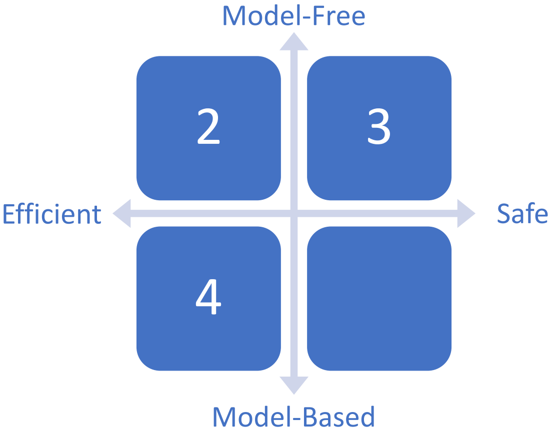
Outline

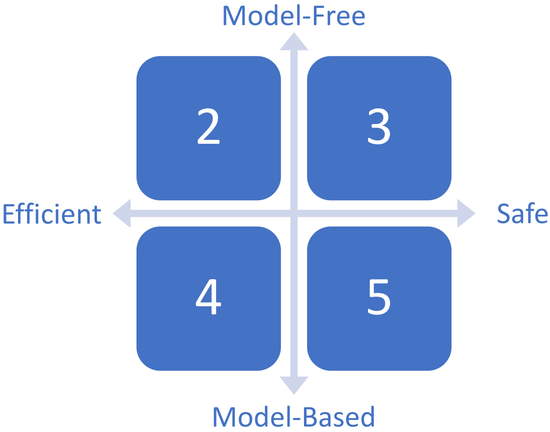



Model-Free
2
Efficient
Safe
Model-Based


Model-Free
2
3
Efficient
Safe
Model-Based

Outline




Model-Free
2
3
Efficient
Safe
4
5
Model-Based

02

Efficient Model-Free

Definition (Optimal State-action Value Function Q^*)

$$Q^*(s,a) = \max_\pi \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \,\middle|\, s_0 = s, a_0 = a \right]$$

How to learn Q^* ?

Proposition (Bellman Optimality Equation)

$$Q^*(s,a) = R(s,a) + \gamma \mathop{\mathbb{E}}_{s'} \max_{a'} Q^*(s', a')$$

↳ Represent Q^* with function approximation (e.g. a neural network in DQN)

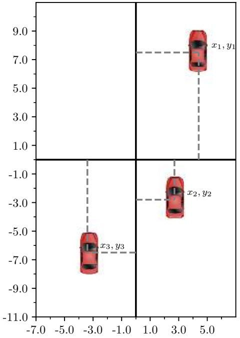
↳ Apply fixed-point iteration over samples (s, a, s') until convergence

The list of features representation

A joint state s of $N+1$ observed vehicles

$$s = (s_i)_{i\in[0,N]}$$

$$s_i = \begin{bmatrix} x_i & y_i & v_i^x & v_i^y & \cos\psi_i & \sin\psi_i \end{bmatrix}^T$$

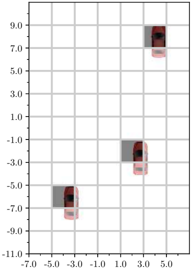


Issues related to function approximation

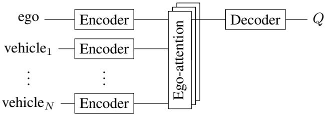
1. Variable size
   ↳ usual models accept fixed-size inputs
2. Sensitivity to the ordering
   ↳ we want the policy to be permutation-invariant:

$$\forall \tau \in \mathfrak{S}_N, \quad \pi(\cdot|(s_0, s_1, \dots, s_N)) = \pi(\cdot|(s_0, s_{\tau(1)}, \dots, s_{\tau(N)}))$$

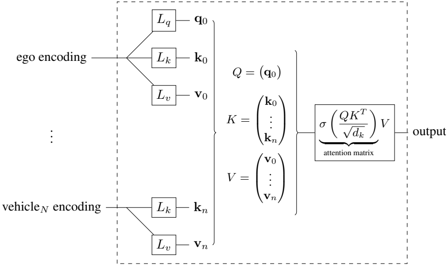
Occupancy grid representation



- ✓ Fixed-size
- ✓ Does not depend on an ordering
- ✗ Suffers from an accuracy / size tradeoff



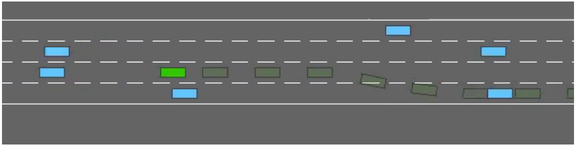
Model architecture



Ego-attention block

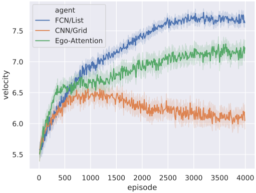
- ✓ Inputs can have a variable size
- ✓ Based on a dot product
  - ↳ permutation-invariant
- ✓ Compact size with no accuracy loss

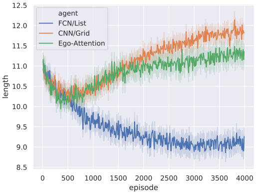
The highway-env environment



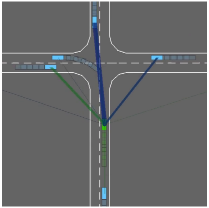
Agent	FCN/List	CNN/Grid	Ego-Attention
Input sizes	[15, 7]	[32, 32, 7]	[· , 7]
Layers sizes	[128, 128]	Convolutional layers: 3 Kernel Size: 2 Stride: 2 Head: [20]	Encoder: [64, 64] Attention: 2 heads $d_k = 32$ Decoder: [64, 64]
Number of parameters	3.0e4	3.2e4	3.4e4
Variable input size	No	No	Yes
Permutation invariant	No	Yes	Yes

Performances

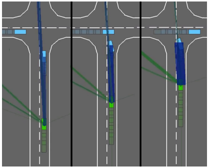




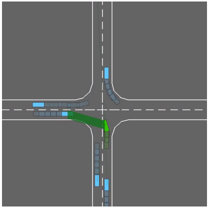
Head specialisation



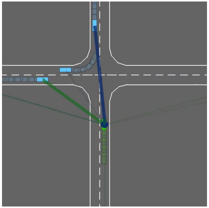
Distance



Sensitivity to uncertainty



A full episode



03

Safe Model-Free

Reinforcement learning relies on a single reward function R

Reinforcement learning relies on a single reward function R

✓ A convenient formulation, but;

Reinforcement learning relies on a single reward function R

- ✓ A convenient formulation, but;
- ✗ R is not always easy to design.

Reinforcement learning relies on a single reward function R

- ✓ A convenient formulation, but;
- ✗ R is not always easy to design.

Conflicting Objectives

Complex tasks require multiple contradictory aspects. Typically:

Task completion vs Safety

Reinforcement learning relies on a single reward function R

- ✓ A convenient formulation, but;
- ✗ R is not always easy to design.

Conflicting Objectives

Complex tasks require multiple contradictory aspects. Typically:

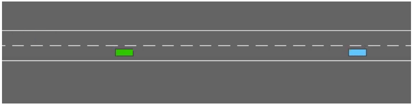
Task completion vs Safety

For example...

Two-Way Road

The agent is driving on a two-way road with a car in front of it,

- it can stay behind (safe/slow);
- it can overtake (unsafe/fast).



Reinforcement learning relies on a single reward function R

- ✓ A convenient formulation, but;
- ✗ R is not always easy to design.

Conflicting Objectives

Complex tasks require multiple contradictory aspects. Typically:

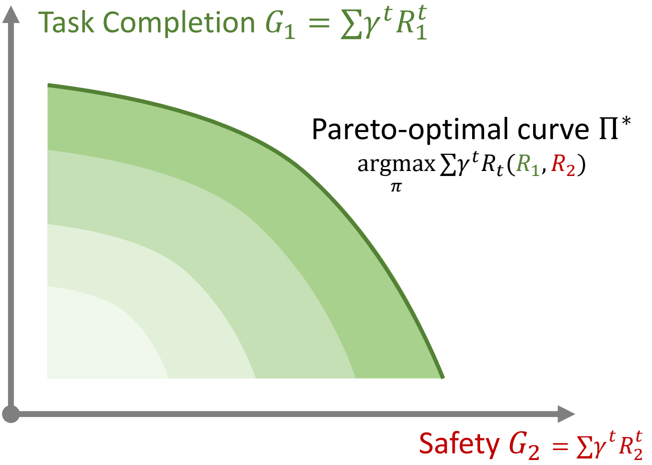
Task completion vs Safety

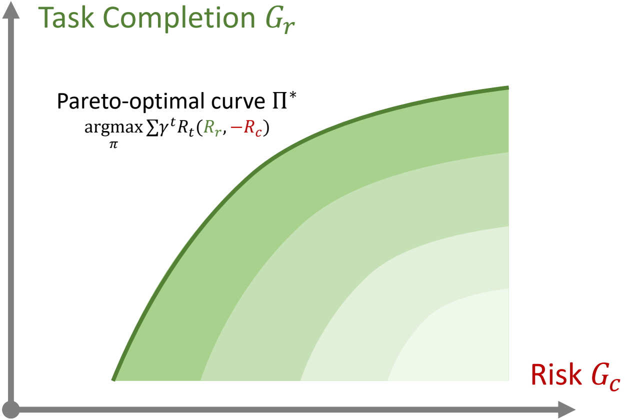
For example...

For a fixed reward function R,

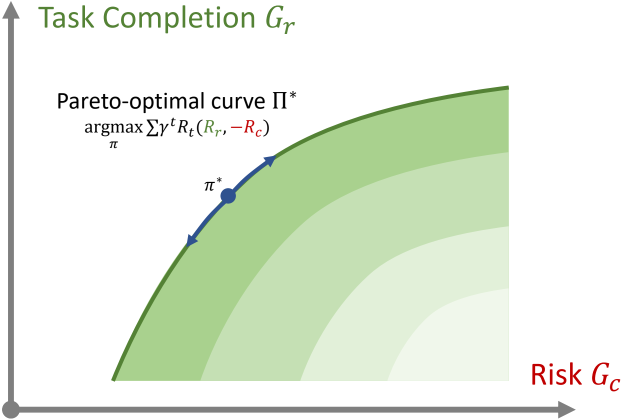
↳ π^* is only guaranteed to lie on a Pareto front Π^*

↳ no control over the $\frac{\text{Task Completion}}{\text{Safety}}$ trade-off

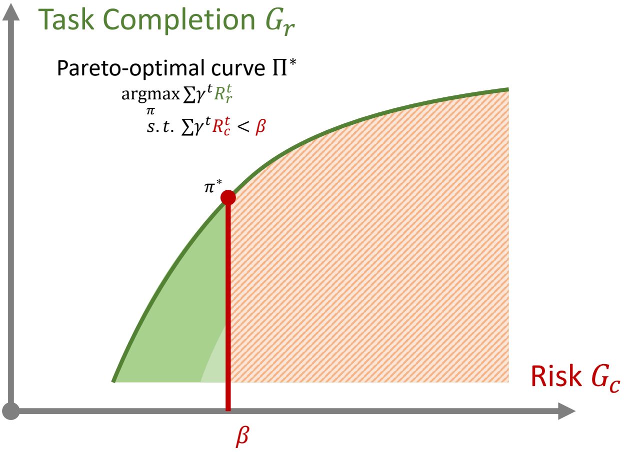

Task Completion $G_1 = \sum \gamma^t R_1^t$
Pareto-optimal curve Π^*
$\underset{\pi}{\operatorname{argmax}} \sum \gamma^t R_t(R_1, R_2)$
Safety $G_2 = \sum \gamma^t R_2^t$


Task Completion G_r
Pareto-optimal curve Π^*
$\underset{\pi}{\text{argmax}} \sum \gamma^t R_t(R_r, -R_c)$
Risk G_c

The optimal policy can move freely along Π^*



How to choose a desired trade-off



Markov Decision Process

An MDP is a tuple $(\mathcal{S}, \mathcal{A}, P, R_r, \gamma)$ with:

- Rewards $R_r \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$

Objective

Maximise rewards

$$\max_{\pi\in\mathcal{M}(\mathcal{A})^\mathcal{S}} \quad \mathbb{E}\left[\sum_{t=0}^\infty \gamma^t R_r(s_t, a_t) \mid s_0 = s\right]$$

Constrained Markov Decision Process

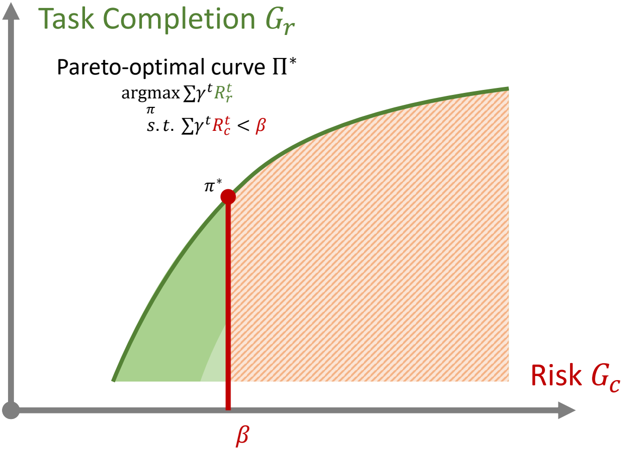
A CMDP is a tuple $(\mathcal{S}, \mathcal{A}, P, R_r, R_c, \gamma, \beta)$ with:

- Rewards $R_r \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$
- Costs $R_c \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$
- Budget β

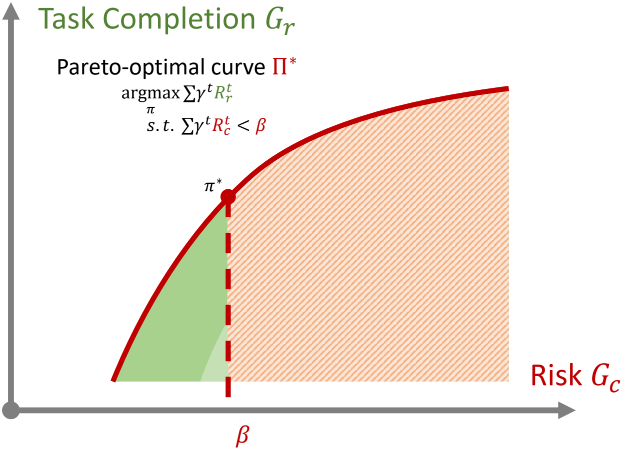
Objective

Maximise rewards while keeping costs under a fixed budget

$$\begin{aligned} \max_{\pi\in\mathcal{M}(\mathcal{A})^\mathcal{S}} \quad & \mathbb{E}\left[\sum_{t=0}^\infty \gamma^t R_r(s_t, a_t) \mid s_0 = s\right] \\ \text{s.t.} \quad & \mathbb{E}\left[\sum_{t=0}^\infty \gamma^t R_c(s_t, a_t) \mid s_0 = s\right] \leq \beta \end{aligned}$$


Task Completion G_r
Pareto-optimal curve Π^*
$\operatorname*{argmax}_\pi \sum \gamma^t R_r^t$
$s.t. \sum \gamma^t R_c^t < \beta$
π^*
Risk G_c
β

We want to learn Π^* rather than π^*_β



Budgeted Markov Decision Process

A BMDP is a tuple $(\mathcal{S}, \mathcal{A}, P, R_r, R_c, \gamma, \mathcal{B})$ with:

- Rewards $R_r \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$
- Costs $R_c \in \mathbb{R}^{\mathcal{S}\times\mathcal{A}}$
- Budget space $\mathcal{B}$

Objective

Maximise rewards while keeping costs under an adjustable budget.
$\forall \beta \in \mathcal{B}$,

$$\begin{aligned} \max_{\pi\in\mathcal{M}(\mathcal{A}\times\mathcal{B})^{\mathcal{S}\times\mathcal{B}}} \quad & \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R_r(s_t, a_t) \mid s_0 = s, \beta_0 = \beta\right] \\ \text{s.t.} \quad & \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R_c(s_t, a_t) \mid s_0 = s, \beta_0 = \beta\right] \leq \beta \end{aligned}$$

Budgeted policies π

- Take a budget β as an additional input
- Output a next budget β'
- $\pi : \underbrace{(s, \beta)}_{\overline{s}} \to \underbrace{(a, \beta')}_{\overline{a}}$

↳ Augment the spaces with the budget β

Definition (Augmented spaces)

- States $\overline{\mathcal{S}} = \mathcal{S} \times \mathcal{B}$.
- Actions $\overline{\mathcal{A}} = \mathcal{A} \times \mathcal{B}$.
- Dynamics $\overline{P}$

$$\text{state } (s, \beta),\ \text{ action } (a, \beta_a) \rightarrow \text{ next state } \begin{cases} s' \sim P(s'|s, a) \\ \beta' = \beta_a \end{cases}$$

Definition (Augmented signals)

1. Rewards $R = (R_r, R_c)$
2. Returns $G^\pi = (G_r^\pi, G_c^\pi) \stackrel{\text{def}}{=} \sum_{t=0}^\infty \gamma^t R(\overline{s}_t, \overline{a}_t)$
3. Value $V^\pi(\overline{s}) = (V_r^\pi, V_c^\pi) \stackrel{\text{def}}{=} \mathbb{E}\left[G^\pi \mid \overline{s_0} = \overline{s}\right]$
4. Q-Value $Q^\pi(\overline{s}, \overline{a}) = (Q_r^\pi, Q_c^\pi) \stackrel{\text{def}}{=} \mathbb{E}\left[G^\pi \mid \overline{s_0} = \overline{s}, \overline{a_0} = \overline{a}\right]$

Definition (Budgeted Optimality)

In that order, we want to:

(i) Respect the budget β:

$$\Pi_a(\overline{s}) \stackrel{\text{def}}{=} \{\pi \in \Pi : V_c^\pi(s, \beta) \leq \beta\}$$

(ii) Maximise the rewards:

$$V_r^*(\overline{s}) \stackrel{\text{def}}{=} \max_{\pi \in \Pi_a(\overline{s})} V_r^\pi(\overline{s}) \qquad \Pi_r(\overline{s}) \stackrel{\text{def}}{=} \arg\max_{\pi \in \Pi_a(\overline{s})} V_r^\pi(\overline{s})$$

(iii) Minimise the costs:

$$V_c^*(\overline{s}) \stackrel{\text{def}}{=} \min_{\pi \in \Pi_r(\overline{s})} V_c^\pi(\overline{s}), \qquad \Pi^*(\overline{s}) \stackrel{\text{def}}{=} \arg\min_{\pi \in \Pi_r(\overline{s})} V_c^\pi(\overline{s})$$

We define the budgeted action-value function Q^* similarly

Theorem (Budgeted Bellman Optimality Equation)

Q^* verifies the following equation:

$$
\begin{aligned}
Q^*(\overline{s},\overline{a}) &= \mathcal{T}Q^*(\overline{s},\overline{a}) \\
&\stackrel{def}{=} R(\overline{s},\overline{a}) + \gamma \sum_{\overline{s}'\in\overline{\mathcal{S}}} \overline{P}(\overline{s'}|\overline{s},\overline{a}) \sum_{\overline{a'}\in\overline{\mathcal{A}}} \pi_{greedy}(\overline{a'}|\overline{s'};Q^*)Q^*(\overline{s'},\overline{a'})
\end{aligned}
$$

where the greedy policy π_{greedy} is defined by:

$$\pi_{\text{greedy}}(\overline{a}|\overline{s};Q) \in \operatorname{arg\,min}_{\rho\in\Pi_r^Q} \mathop{\mathbb{E}}_{\overline{a}\sim\rho} Q_c(\overline{s},\overline{a}),$$

$$\text{where} \quad \Pi_r^Q \stackrel{\text{def}}{=} \operatorname{arg\,max}_{\rho\in\mathcal{M}(\overline{\mathcal{A}})} \mathop{\mathbb{E}}_{\overline{a}\sim\rho} Q_r(\overline{s},\overline{a})$$

$$\text{s.t.} \quad \mathop{\mathbb{E}}_{\overline{a}\sim\rho} Q_c(\overline{s},\overline{a}) \leq \beta$$

The optimal policy

Proposition (Optimality of the policy)

$\pi_{greedy}(\cdot\, ; Q^*)$ *is simultaneously optimal in all states* $\overline{s} \in \overline{\mathcal{S}}$*:*

$$\pi_{greedy}(\cdot\, ; Q^*) \in \Pi^*(\overline{s})$$

In particular, $V^{\pi_{greedy}(\cdot;Q^*)} = V^*$ *and* $Q^{\pi_{greedy}(\cdot;Q^*)} = Q^*$*.*

Proposition (Solving the non-linear program)

π_{greedy} *can be computed efficiently, as a mixture* π_{hull} *of two points that lie on the convex hull of* Q*.*

$$\pi_{greedy} = \pi_{hull}$$

Recall what we've shown so far:

$$\mathcal{T} \xrightarrow[fixed-point]{} Q^* \xrightarrow[tractable]{} \pi_{\text{hull}}(Q^*) \xrightarrow[equal]{} \pi_{\text{greedy}}(Q^*) \xrightarrow[optimal]{}$$

Recall what we've shown so far:

$$\mathcal{T} \xrightarrow[fixed-point]{} Q^* \xrightarrow[tractable]{} \pi_{\text{hull}}(Q^*) \xrightarrow[equal]{} \pi_{\text{greedy}}(Q^*) \xrightarrow[optimal]{}$$

We're almost there!

All that is left is to perform Fixed-Point Iteration to compute Q^*.

Recall what we've shown so far:

$$\mathcal{T} \xrightarrow[\text{fixed-point}]{} Q^* \xrightarrow[\text{tractable}]{} \pi_{\text{hull}}(Q^*) \xrightarrow[\text{equal}]{} \pi_{\text{greedy}}(Q^*) \xrightarrow[\text{optimal}]{}$$

We're almost there!
All that is left is to perform Fixed-Point Iteration to compute Q^*.

Theorem (Non-Contractivity)

For any BMDP $(\mathcal{S}, \mathcal{A}, P, R_r, R_c, \gamma)$ with $|\mathcal{A}| \geq 2$, $\mathcal{T}$ is ***not*** *a contraction.*

$$\forall \varepsilon > 0, \exists Q^1, Q^2 \in (\mathbb{R}^2)^{\overline{\mathcal{SA}}} : \|\mathcal{T}Q^1 - \mathcal{T}Q^2\|_\infty \geq \frac{1}{\varepsilon}\|Q^1 - Q^2\|_\infty$$

✗ We cannot guarantee the convergence of $\mathcal{T}^n(Q_0)$ to Q^*

Thankfully,

Theorem (Contractivity on smooth Q-functions)

$\mathcal{T}$ is a contraction when restricted to the subset $\mathcal{L}_\gamma$ of Q-functions such that "Q_r is L-Lipschitz with respect to Q_c", with $L < \frac{1}{\gamma} - 1$.

$$\mathcal{L}_\gamma = \left\{ \begin{array}{c} Q \in (\mathbb{R}^2)^{\overline{\mathcal{S}\mathcal{A}}} \text{ s.t. } \exists L < \frac{1}{\gamma} - 1 : \forall \overline{s} \in \overline{\mathcal{S}}, \overline{a}_1, \overline{a}_2 \in \overline{\mathcal{A}}, \\ |Q_r(\overline{s}, \overline{a}_1) - Q_r(\overline{s}, \overline{a}_2)| \leq L|Q_c(\overline{s}, \overline{a}_1) - Q_c(\overline{s}, \overline{a}_2)| \end{array} \right\}$$

- ✓ We guarantee convergence under some (strong) assumptions
- ✓ We observe empirical convergence

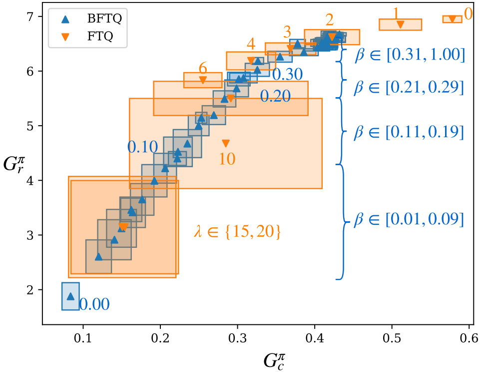
Lagrangian Relaxation Baseline

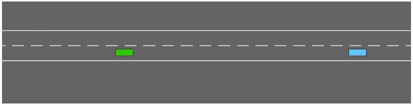
Consider the dual problem so as to replace the hard constraint by a soft constraint penalised by a Lagrangian multiplier λ:

$$\max_{\pi} \mathbb{E} \sum_{t} \gamma^t R_r(s, a) - \lambda \gamma^t R_c(s, a)$$

- Train many policies π_k with penalties λ_k and recover the cost budgets β_k
- Very data/memory-heavy

Experiments





04

Efficient Model-Based

Model estimation

Learn a model for the dynamics $\hat{T}(s_{t+1}|s_t, a_t)$. For instance:

1. Least-square estimate: $\min_{\hat{T}} \sum_t \|s_{t+1} - \hat{T}(s_t, a_t)\|_2^2$
2. Maximum Likelihood estimate: $\max_{\hat{T}} \sum_t \hat{T}(s_{t+1}|s_t, a_t)$

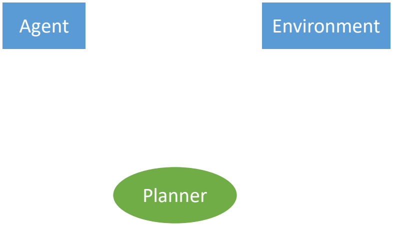
Planning

Leverage $\hat{T}$ to compute

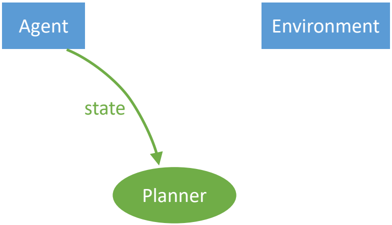
$$\max_\pi \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \,\middle|\, a_t \sim \pi(s_t), s_{t+1} \sim \hat{T}(s_t, a_t)\right]$$

How?

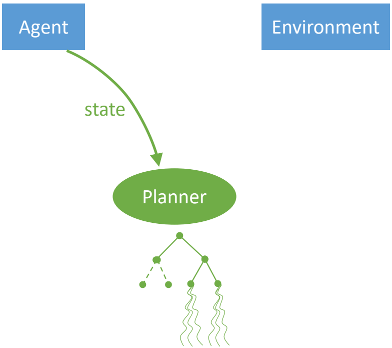
We can use $\hat{T}$ as a generative model:



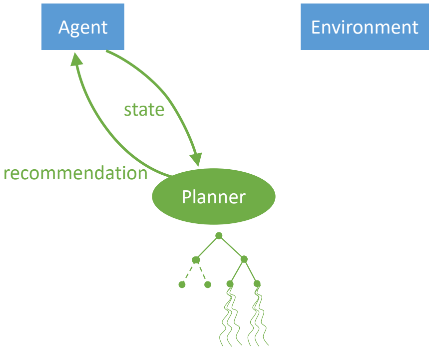
We can use $\hat{T}$ as a generative model:



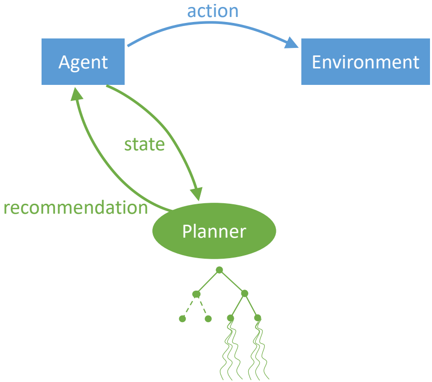
We can use $\hat{T}$ as a generative model:



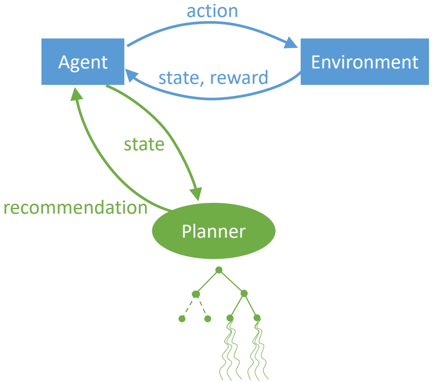
We can use $\hat{T}$ as a generative model:



We can use $\hat{T}$ as a generative model:



We can use $\hat{T}$ as a generative model:



Online *Planning*

- fixed budget: the model can only be queried n times

$$\text{Objective: minimize } \mathbb{E}\ \underbrace{V^* - V(n)}_{\text{Simple Regret } r_n}$$

An exploration-exploitation problem.

Optimism in the Face of Uncertainty

Given a set of options $a \in A$ with uncertain outcomes, try the one with the highest possible outcome.

Optimism in the Face of Uncertainty

Given a set of options $a \in A$ with uncertain outcomes, try the one with the highest possible outcome.

- Either you performed well;

Optimism in the Face of Uncertainty

Given a set of options $a \in A$ with uncertain outcomes, try the one with the highest possible outcome.

- Either you performed well;
- or you learned something.

Optimism in the Face of Uncertainty

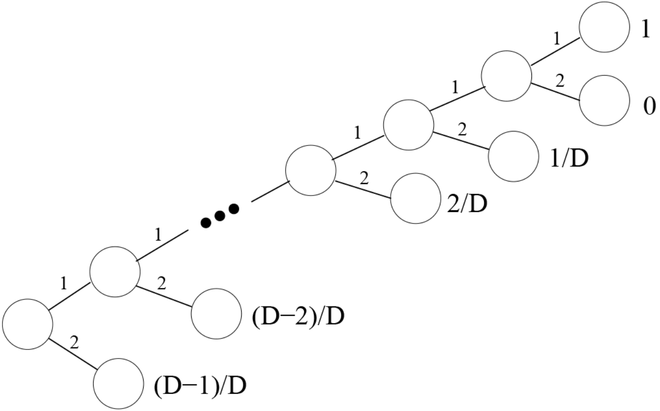
Given a set of options $a \in A$ with uncertain outcomes, try the one with the highest possible outcome.

- Either you performed well;
- or you learned something.

Instances

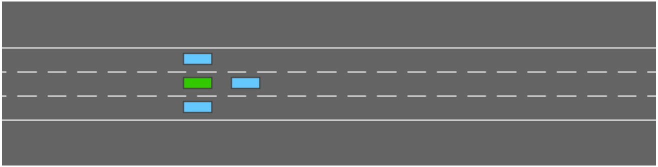
- Monte-carlo tree search (`MCTS`) (Coulom, 2006): `CrazyStone`
- Reframed in the bandit setting as `UCT` (Kocsis and Szepesvári, 2006), still very popular (e.g. `Alpha Go`).
- Proved asymptotic consistency, but no regret bound.

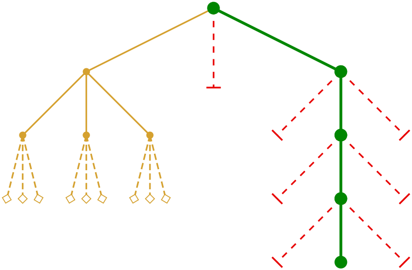
It was analysed in (Coquelin and Munos, 2007)]



The sample complexity of is lower-bounded by $O(\exp(\exp(D)))$.

Not just a theoretical counter-example.





OPD: Optimistic Planning for Deterministic systems

- Introduced by (Hren and Munos, 2008)
- Another optimistic algorithm
- Only for deterministic MDPs

Theorem (OPD sample complexity)

$$\mathbb{E}\, r_n = \mathcal{O}\left(n^{-\frac{\log 1/\gamma}{\log \kappa}}\right), \textit{if } \kappa > 1$$

OPD: Optimistic Planning for Deterministic systems

- Introduced by (Hren and Munos, 2008)
- Another optimistic algorithm
- Only for deterministic MDPs

Theorem (OPD sample complexity)

$$\mathbb{E}\, r_n = \mathcal{O}\left(n^{-\frac{\log 1/\gamma}{\log \kappa}}\right), \textit{if } \kappa > 1$$

OLOP: Open-Loop Optimistic Planning

- Introduced by (Bubeck and Munos, 2010)
- Extends OPD to the stochastic setting
- Only considers open-loop policies, i.e. sequences of actions

A direct application of Optimism in the Face of Uncertainty

1. We want

$$\max_a V(a)$$

A direct application of Optimism in the Face of Uncertainty

1. We want

$$\max_a V(a)$$

2. Form upper confidence-bounds of sequence values:

$$V(a) \leq U_a \quad \text{w.h.p}$$

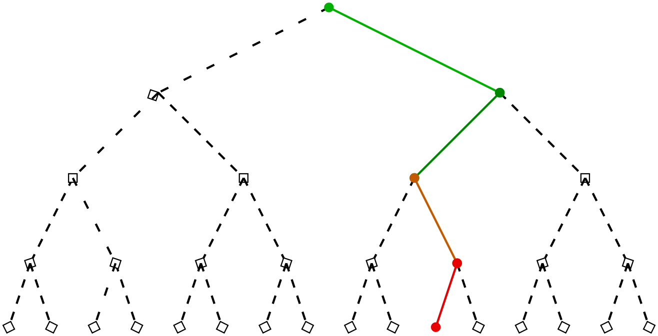
A direct application of Optimism in the Face of Uncertainty

1. We want
$$\max_a V(a)$$

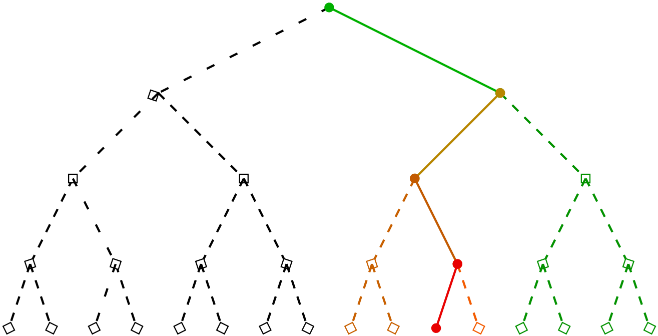
2. Form upper confidence-bounds of sequence values:
$$V(a) \leq U_a \quad \text{w.h.p}$$

3. Sample the sequence with highest UCB:
$$\arg\max_a U_a$$

The idea behind OLOP



The idea behind OLOP



Upper-bounding the value of sequences

$$V(a) = \overbrace{\sum_{t=1}^{h} \gamma^t \mu_{a_{1:t}}}^{\text{follow the sequence}} + \overbrace{\sum_{t\geq h+1} \gamma^t \mu_{a^*_{1:t}}}^{\text{act optimally}}$$

Upper-bounding the value of sequences

$$V(a) = \overbrace{\sum_{t=1}^{h} \gamma^t \underbrace{\mu_{a_{1:t}}}_{\leq U^\mu}}^{\text{follow the sequence}} + \overbrace{\sum_{t \geq h+1} \gamma^t \underbrace{\mu_{a^*_{1:t}}}_{\leq 1}}^{\text{act optimally}}$$

OLOP main tool: the Chernoff-Hoeffding deviation inequality

$$\underbrace{U_a^\mu(m)}_{\text{Upper bound}} \stackrel{\text{def}}{=} \underbrace{\hat{\mu}_a(m)}_{\text{Empirical mean}} + \underbrace{\sqrt{\frac{2\log M}{T_a(m)}}}_{\text{Confidence interval}}$$

OLOP main tool: the Chernoff-Hoeffding deviation inequality

$$\underbrace{U_a^{\mu}(m)}_{\text{Upper bound}} \stackrel{\text{def}}{=} \underbrace{\hat{\mu}_a(m)}_{\text{Empirical mean}} + \underbrace{\sqrt{\frac{2\log M}{T_a(m)}}}_{\text{Confidence interval}}$$

OPD: upper-bound all the future rewards by 1

$$U_a(m) \stackrel{\text{def}}{=} \sum_{t=1}^{h} \underbrace{\gamma^t U_{a_{1:t}}^{\mu}(m)}_{\text{Past rewards}} + \underbrace{\frac{\gamma^{h+1}}{1-\gamma}}_{\text{Future rewards}}$$

OLOP main tool: the Chernoff-Hoeffding deviation inequality

$$\underbrace{U_a^{\mu}(m)}_{\text{Upper bound}} \overset{\text{def}}{=} \underbrace{\hat{\mu}_a(m)}_{\text{Empirical mean}} + \underbrace{\sqrt{\frac{2\log M}{T_a(m)}}}_{\text{Confidence interval}}$$

OPD: upper-bound all the future rewards by 1

$$U_a(m) \overset{\text{def}}{=} \sum_{t=1}^{h} \underbrace{\gamma^t U^{\mu}_{a_{1:t}}(m)}_{\text{Past rewards}} + \underbrace{\frac{\gamma^{h+1}}{1-\gamma}}_{\text{Future rewards}}$$

Bounds sharpening

$$B_a(m) \overset{\text{def}}{=} \inf_{1\leq t\leq L} U_{a_{1:t}}(m)$$

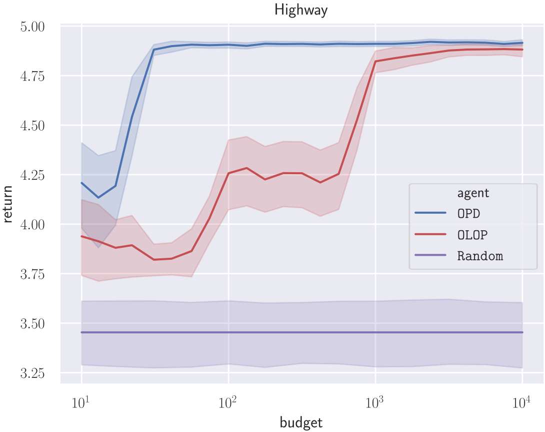
Theorem (OLOP Sample complexity)

OLOP satisfies:

$$\mathbb{E}\, r_n = \begin{cases} \widetilde{\mathcal{O}}\left(n^{-\frac{\log 1/\gamma}{\log \kappa'}}\right), & \textit{if } \gamma\sqrt{\kappa'} > 1 \\ \widetilde{\mathcal{O}}\left(n^{-\frac{1}{2}}\right), & \textit{if } \gamma\sqrt{\kappa'} \leq 1 \end{cases}$$

"Remarkably, in the case $\kappa\gamma^2 > 1$, we obtain the same rate for the simple regret as Hren and Munos (2008). Thus, in this case, we can say that planning in stochastic environments is not harder than planning in deterministic environments".

Does it work?



Our objective: understand and bridge this gap.

Make `OLOP` *practical*.

Explanation: inconsistency

- Unintended behaviour happens when $U_a^\mu(m) > 1$, $\forall a$.

$$U_a^\mu(m) = \underbrace{\hat{\mu}_a(m)}_{\in[0,1]} + \underbrace{\sqrt{\frac{2\log M}{T_a(m)}}}_{>0}$$

Explanation: inconsistency

- Unintended behaviour happens when $U_a^\mu(m) > 1, \forall a$.

$$U_a^\mu(m) = \underbrace{\hat{\mu}_a(m)}_{\in[0,1]} + \underbrace{\sqrt{\frac{2\log M}{T_a(m)}}}_{>0}$$

- Then the sequence $(U_{a_{1:t}}(m))_t$ is increasing

$$U_{a_{1:1}}(m) = \gamma U_{a_1}^\mu(m) + \gamma^2 1 \qquad + \gamma^3 1 + \dots$$

$$U_{a_{1:2}}(m) = \gamma U_{a_1}^\mu(m) + \gamma^2 \underbrace{U_{a_2}^\mu}_{>1} \qquad + \gamma^3 1 + \dots$$

Explanation: inconsistency

- Unintended behaviour happens when $U_a^\mu(m) > 1, \forall a$.

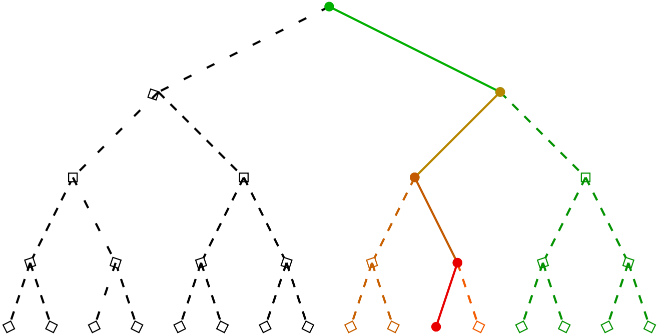
$$U_a^\mu(m) = \underbrace{\hat{\mu}_a(m)}_{\in[0,1]} + \underbrace{\sqrt{\frac{2\log M}{T_a(m)}}}_{>0}$$

- Then the sequence $(U_{a_{1:t}}(m))_t$ is increasing

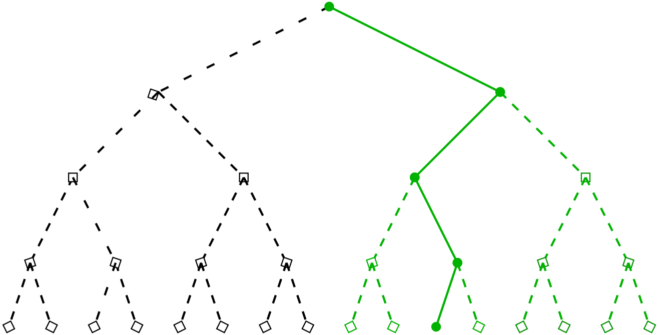
$$\begin{aligned}
U_{a_{1:1}}(m) &= \gamma U_{a_1}^\mu(m) + \gamma^2 1 && +\gamma^3 1 + \dots \\
U_{a_{1:2}}(m) &= \gamma U_{a_1}^\mu(m) + \gamma^2 \underbrace{U_{a_2}^\mu}_{>1} && +\gamma^3 1 + \dots
\end{aligned}$$

- Then $B_a(m) = U_{a_{1:1}}(m)$

What we were promised



What we actually get



OLOP behaves as uniform planning!

We summon the upper-confidence bound from kl-UCB (Cappé et al., 2013):

$$U_a^{\mu}(m) \stackrel{\text{def}}{=} \max\{q \in I : T_a(m) d(\hat{\mu}_a(m), q) \leq f(m)\}$$

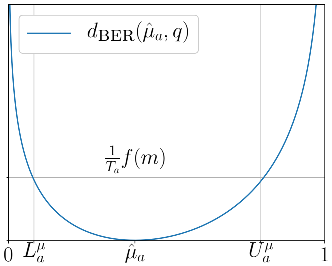
We summon the upper-confidence bound from kl-UCB (Cappé et al., 2013):

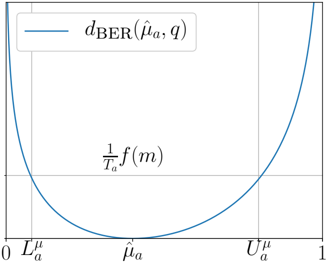
$$U_a^\mu(m) \stackrel{\text{def}}{=} \max\{q \in I : T_a(m) d(\hat{\mu}_a(m), q) \leq f(m)\}$$

Algorithm	OLOP	KL-OLOP
Interval I	$\mathbb{R}$	$[0, 1]$
Divergence d	$d_{\texttt{QUAD}}$	$d_{\texttt{BER}}$
$f(m)$	$4 \log M$	$2 \log M + 2 \log \log M$

$$d_{\texttt{QUAD}}(p, q) \stackrel{\text{def}}{=} 2(p-q)^2$$

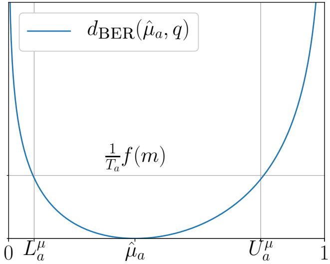
$$d_{\texttt{BER}}(p, q) \stackrel{\text{def}}{=} p \log \frac{p}{q} + (1-p) \log \frac{1-p}{1-q}$$


$d_{\mathrm{BER}}(\hat{\mu}_a, q)$
$\frac{1}{T_a} f(m)$
0
L_a^μ
$\hat{\mu}_a$
U_a^μ
1



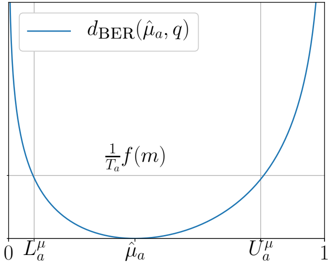
And now,

- $U_a^\mu(m) \in I = [0, 1], \forall a$.



And now,

- $U_a^\mu(m) \in I = [0, 1], \forall a$.
- The sequence $(U_{a_{1:t}}(m))_t$ is non-increasing



And now,

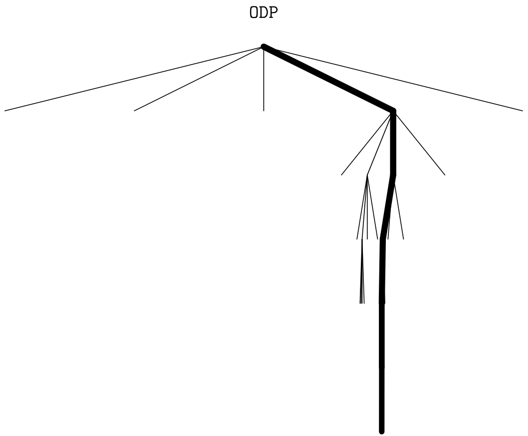
- $U_a^\mu(m) \in I = [0, 1], \forall a$.
- The sequence $(U_{a_{1:t}}(m))_t$ is non-increasing
- $B_a(m) = U_a(m)$, the *bound sharpening* step is superfluous.

Theorem (Sample complexity)

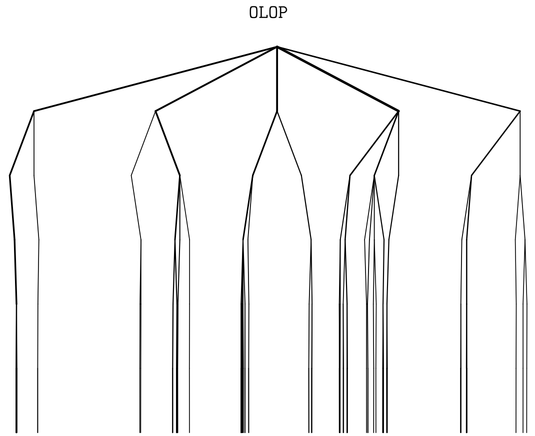
KL-OLOP enjoys the same regret bounds as OLOP. More precisely, KL-OLOP satisfies:

$$\mathbb{E}\, r_n = \begin{cases} \widetilde{\mathcal{O}}\left(n^{-\frac{\log 1/\gamma}{\log \kappa'}}\right), & \text{if } \gamma\sqrt{\kappa'} > 1 \\ \widetilde{\mathcal{O}}\left(n^{-\frac{1}{2}}\right), & \text{if } \gamma\sqrt{\kappa'} \leq 1 \end{cases}$$

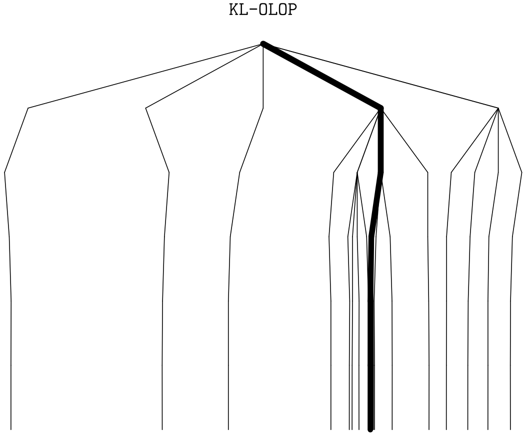
Experiments — Expanded Trees



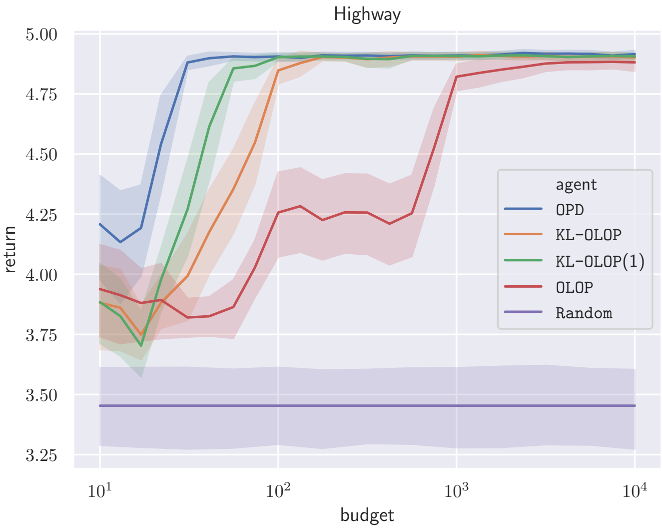
Experiments — Expanded Trees



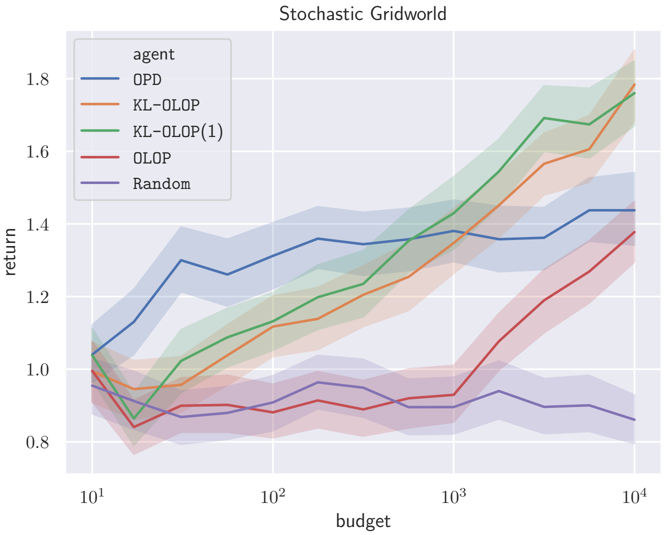
Experiments — Expanded Trees



Experiments — Performances



Experiments — Performances



05

Safe Model-Based

Model-based RL learns the dynamics $\hat{T}$ and optimizes

$$\max_{\pi} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \,\middle|\, a_t \sim \pi(s_t), s_{t+1} \sim \hat{T}(s_t, a_t)\right]$$

Definition (Model Bias)

$$T \neq \hat{T}$$

- Video example

1. Build a confidence region C_δ around the true dynamics T

$$\mathbb{P}\left(T \in C_\delta\right) > 1 - \delta$$

2. Plan robustly with respect to this ambiguity

$$\max_{\pi} \underbrace{\min_{T \in C_\delta} \sum_{t=0}^{\infty} \gamma^t r_t}_{v^r(\pi)}$$

In order to build C_δ, we rely on a structure assumption

Assumption (Structure)

$$\dot{x}(t) = A(\theta)x(t) + Bu(t) + d(t)$$

with

$$A(\theta) = \sum_{i=1}^{d} \theta_i \Phi_i$$

Having observed a history of $\dot{x}(t), x(t)$, we obtain a linear regression problem:

$$\min_\theta \|\dot{x}(t) - A(\theta)x(t) - Bu(t)\|_2^2$$

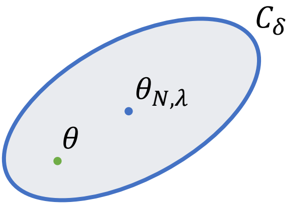
Proposition (Confidence ellipsoid (Abbasi-yadkori, Pál, and Szepesvári, 2011))

Under some assumptions on the disturbance $d(t)$, it holds with probability $1-\delta$ that:

$$\|\theta - \theta_{Np,\lambda}\|_{G_{Np,\lambda}} \leq \beta_t(\delta)$$

$$\text{where} \quad \theta_{Np,\lambda} = G_{Np,\lambda}^{-1}\Phi_{[Np]}^T Y_{[Np]};$$

$$G_{Np,\lambda} = \Phi_{[Np]}^T \Phi_{[Np]} + \lambda I_d.$$

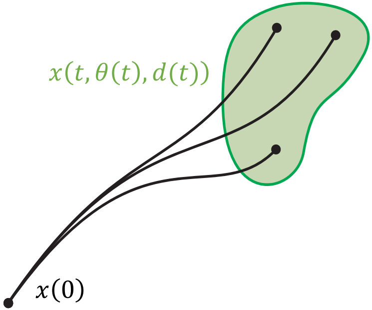


Possible trajectories

$$\dot{x}(t) = A(\theta)x(t) + Bu(t) + d(t)$$

There are two sources of uncertainty:

- Parametric uncertainty $A(\theta) \in C_\delta$
- External perturbations $d(t)$

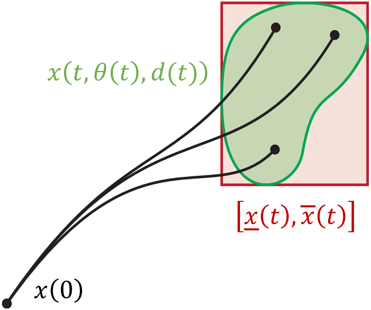


Interval Prediction

Can we design an interval predictor $[\underline{x}(t), \overline{x}(t)]$ that verifies:

- inclusion property: $\forall t, \underline{x}(t) \leq x(t) \leq \overline{x}(t)$;
- stable dynamics?

We want the predictor to be as tight as possible. How to proceed?



A first idea

Assume that $\underline{x}(t) \leq x(t) \leq \overline{x}(t)$, for some $t \geq 0$.

A first idea

Assume that $\underline{x}(t) \leq x(t) \leq \overline{x}(t)$, for some $t \geq 0$.

$\hookrightarrow$ To propagate the interval to $x(t+dt)$, we need to bound $A(\theta)x(t)$.

A first idea

Assume that $\underline{x}(t) \leq x(t) \leq \overline{x}(t)$, for some $t \geq 0$.

- $\hookrightarrow$ To propagate the interval to $x(t+dt)$, we need to bound $A(\theta)x(t)$.
- $\hookrightarrow$ Why not use interval arithmetics?

Assume that $\underline{x}(t) \leq x(t) \leq \overline{x}(t)$, for some $t \geq 0$.

↳ To propagate the interval to $x(t + dt)$, we need to bound $A(\theta)x(t)$.

↳ Why not use interval arithmetics?

Lemma (Image of an interval (Efimov et al., 2012))

If A a known matrix, then

$$A^+\underline{x} - A^-\overline{x} \leq Ax \leq A^+\overline{x} - A^-\underline{x}.$$

where $A^+ = \max(A, 0)$ *and* $A^- = A - A^+$.

Assume that $\underline{x}(t) \leq x(t) \leq \overline{x}(t)$, for some $t \geq 0$.

↳ To propagate the interval to $x(t + dt)$, we need to bound $A(\theta)x(t)$.

↳ Why not use interval arithmetics?

Lemma (Product of intervals (Efimov et al., 2012))

If A is unknown but bounded $\underline{A} \leq A \leq \overline{A}$,

$$\underline{A}^+\underline{x}^+ - \overline{A}^+\underline{x}^- - \underline{A}^-\overline{x}^+ + \overline{A}^-\overline{x}^- \leq Ax$$
$$\leq \overline{A}^+\overline{x}^+ - \underline{A}^+\overline{x}^- - \overline{A}^-\underline{x}^+ + \underline{A}^-\underline{x}^-.$$

Assume that $\underline{x}(t) \leq x(t) \leq \overline{x}(t)$, for some $t \geq 0$.

↳ To propagate the interval to $x(t+dt)$, we need to bound $A(\theta)x(t)$.

↳ Why not use interval arithmetics?

Lemma (Product of intervals (Efimov et al., 2012))

If A is unknown but bounded $\underline{A} \leq A \leq \overline{A}$,

$$\begin{aligned} \underline{A}^+\underline{x}^+ - \overline{A}^+\underline{x}^- - \underline{A}^-\overline{x}^+ + \overline{A}^-\overline{x}^- &\leq Ax \\ \leq \overline{A}^+\overline{x}^+ - \underline{A}^+\overline{x}^- - \overline{A}^-\underline{x}^+ + \underline{A}^-\underline{x}^-. \end{aligned}$$

✓ Since $A(\theta)$ belongs to a known C_δ,
we can easily compute such bounds $\underline{A} \leq A(\theta) \leq \overline{A}$

Following this result, define the predictor:

$$
\begin{aligned}
\dot{\underline{x}}(t) &= \underline{A}^+\underline{x}^+(t) - \overline{A}^+\underline{x}^-(t) - \underline{A}^-\overline{x}^+(t) \\
&\quad + \overline{A}^-\overline{x}^-(t) + B^+\underline{d}(t) - B^-\overline{d}(t), \\
\dot{\overline{x}}(t) &= \overline{A}^+\overline{x}^+(t) - \underline{A}^+\overline{x}^-(t) - \overline{A}^-\underline{x}^+(t) \\
&\quad + \underline{A}^-\underline{x}^-(t) + B^+\overline{d}(t) - B^-\underline{d}(t), \\
&\quad \underline{x}(0) = \underline{x}_0,\ \overline{x}(0) = \overline{x}_0,
\end{aligned}
\tag{1}
$$

Following this result, define the predictor:

$$\begin{aligned}
\dot{\underline{x}}(t) &= \underline{A}^{+}\underline{x}^{+}(t) - \overline{A}^{+}\underline{x}^{-}(t) - \underline{A}^{-}\overline{x}^{+}(t) \\
&\quad + \overline{A}^{-}\overline{x}^{-}(t) + B^{+}\underline{d}(t) - B^{-}\overline{d}(t), \\
\dot{\overline{x}}(t) &= \overline{A}^{+}\overline{x}^{+}(t) - \underline{A}^{+}\overline{x}^{-}(t) - \overline{A}^{-}\underline{x}^{+}(t) \\
&\quad + \underline{A}^{-}\underline{x}^{-}(t) + B^{+}\overline{d}(t) - B^{-}\underline{d}(t), \\
&\quad \underline{x}(0) = \underline{x}_0,\ \overline{x}(0) = \overline{x}_0,
\end{aligned} \tag{1}$$

Proposition (Inclusion property)

✓ *The predictor* (1) *satisfies* $\underline{x}(t) \leq x(t) \leq \overline{x}(t)(t)$

Following this result, define the predictor:

$$
\begin{aligned}
\dot{\underline{x}}(t) &= \underline{A}^{+}\underline{x}^{+}(t) - \overline{A}^{+}\underline{x}^{-}(t) - \underline{A}^{-}\overline{x}^{+}(t) \\
&\quad +\overline{A}^{-}\overline{x}^{-}(t) + B^{+}\underline{d}(t) - B^{-}\overline{d}(t), \\
\dot{\overline{x}}(t) &= \overline{A}^{+}\overline{x}^{+}(t) - \underline{A}^{+}\overline{x}^{-}(t) - \overline{A}^{-}\underline{x}^{+}(t) \\
&\quad +\underline{A}^{-}\underline{x}^{-}(t) + B^{+}\overline{d}(t) - B^{-}\underline{d}(t), \\
&\quad \underline{x}(0) = \underline{x}_0,\ \overline{x}(0) = \overline{x}_0,
\end{aligned}
\tag{1}
$$

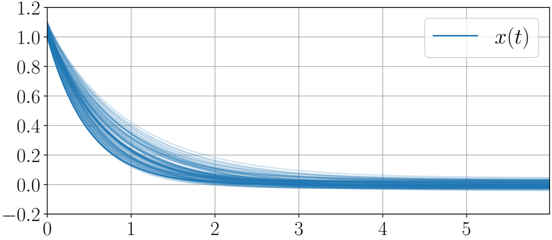
Proposition (Inclusion property)

✓ *The predictor* (1) *satisfies* $\underline{x}(t) \leq x(t) \leq \overline{x}(t)(t)$

? But is it stable?

Consider the scalar system, for all $t \geq 0$:

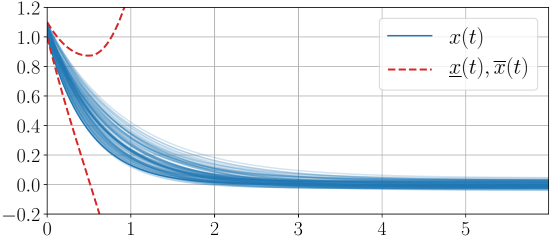
$$\dot{x}(t) = -\theta(t)x(t) + d(t), \text{ where } \begin{cases} x(0) \in [\underline{x}_0, \overline{x}_0] = [1.0, 1.1], \\ \theta(t) \in \Theta = [\underline{\theta}, \overline{\theta}] = [1, 2], \\ d(t) \in [\underline{d}, \overline{d}] = [-0.1, 0.1], \end{cases}$$



✓ The system is always stable

Consider the scalar system, for all $t \geq 0$:

$$\dot{x}(t) = -\theta(t)x(t) + d(t), \text{ where } \begin{cases} x(0) \in [\underline{x}_0, \overline{x}_0] = [1.0, 1.1], \\ \theta(t) \in \Theta = [\underline{\theta}, \overline{\theta}] = [1, 2], \\ d(t) \in [\underline{d}, \overline{d}] = [-0.1, 0.1], \end{cases}$$

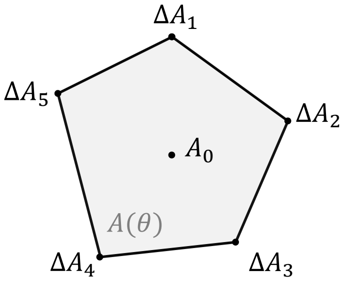


✓ The system is always stable ✗ The predictor (1) is unstable

Assumption (Polytopic Structure)

There exist A_0 Metzler and $\Delta A_0, \cdots, \Delta A_N$ such that:

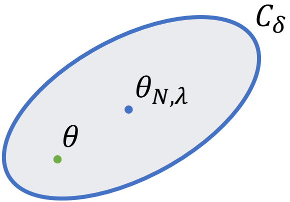
$$A(\theta) = \underbrace{A_0}_{\substack{\text{Nominal}\\ \text{dynamics}}} + \sum_{i=1}^{N} \lambda_i(\theta)\Delta A_i, \quad \sum_{i=1}^{N} \underbrace{\lambda_i(\theta)}_{\geq 0} = 1; \quad \forall \theta \in \Theta$$



Assumption (Polytopic Structure)

There exist A_0 *Metzler and* $\Delta A_0, \cdots, \Delta A_N$ *such that:*

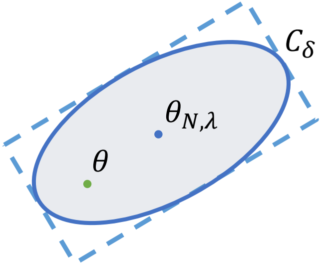
$$A(\theta) = \underbrace{A_0}_{\substack{\text{Nominal}\\ \text{dynamics}}} + \sum_{i=1}^{N} \lambda_i(\theta)\Delta A_i, \quad \sum_{i=1}^{N} \underbrace{\lambda_i(\theta)}_{\geq 0} = 1; \quad \forall \theta \in \Theta$$



Assumption (Polytopic Structure)

There exist A_0 Metzler and $\Delta A_0, \cdots, \Delta A_N$ such that:

$$A(\theta) = \underbrace{A_0}_{\substack{\text{Nominal}\\\text{dynamics}}} + \sum_{i=1}^{N} \lambda_i(\theta)\Delta A_i, \quad \sum_{i=1}^{N} \underbrace{\lambda_i(\theta)}_{\geq 0} = 1; \quad \forall\theta \in \Theta$$



Denote

$$\Delta A_+ = \sum_{i=1}^{N} \Delta A_i^+, \ \Delta A_- = \sum_{i=1}^{N} \Delta A_i^-,$$

We define the predictor

$$\begin{aligned} \dot{\underline{x}}(t) &= A_0\underline{x}(t) - \Delta A_+\underline{x}^-(t) - \Delta A_-\overline{x}^+(t) \\ &\quad +B^+\underline{d}(t) - B^-\overline{d}(t), \\ \dot{\overline{x}}(t) &= A_0\overline{x}(t) + \Delta A_+\overline{x}^+(t) + \Delta A_-\underline{x}^-(t) \\ &\quad +B^+\overline{d}(t) - B^-\underline{d}(t), \\ &\quad \underline{x}(0) = \underline{x}_0, \ \overline{x}(0) = \overline{x}_0 \end{aligned} \tag{2}$$

Theorem (Inclusion property)

The predictor (2) *ensures* $\underline{x}(t) \leq x(t) \leq \overline{x}(t)$.

Theorem (Stability)

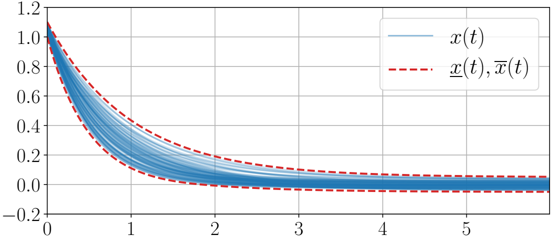
If there exist diagonal matrices P, Q, Q_+, Q_-, Z_+, Z_-, Ψ_+, Ψ_-, Ψ, $\Gamma \in \mathbb{R}^{2n\times 2n}$ *such that the following LMIs are satisfied:*

$$P + \min\{Z_+, Z_-\} > 0, \;\; \Upsilon \preceq 0, \;\; \Gamma > 0,$$
$$Q + \min\{Q_+, Q_-\} + 2\min\{\Psi_+, \Psi_-\} > 0,$$

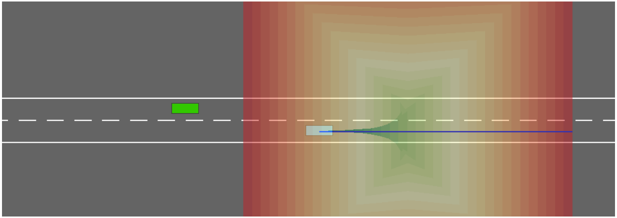
where $\Upsilon = \Upsilon(A_0, \Delta A_-, \Delta A_+, \Psi_-, \Psi_+, \Psi)$,
then the predictor (2) *is input-to-state stable with respect to the inputs* $\underline{d}$, $\overline{d}$.

Recall the scalar system:

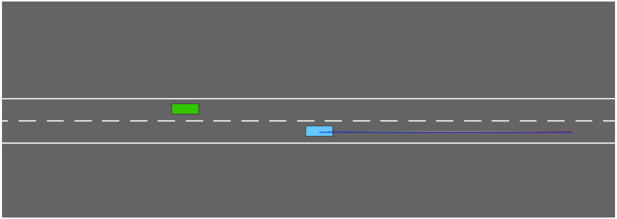
$$\dot{x}(t) = -\theta(t)x(t) + d(t), \text{ where } \begin{cases} x(0) \in [\underline{x}_0, \overline{x}_0] = [1.0, 1.1], \\ \theta(t) \in \Theta = [\underline{\theta}, \overline{\theta}] = [1, 2], \\ d(t) \in [\underline{d}, \overline{d}] = [-0.1, 0.1], \end{cases}$$



✓ The system is always stable ✓ The predictor (2) is stable

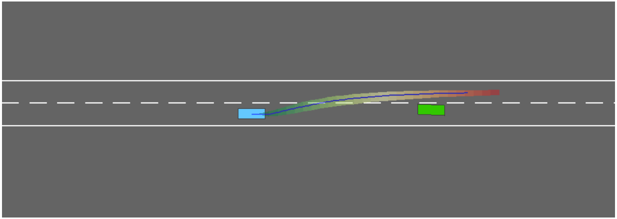


The naive predictor (1) quickly diverges

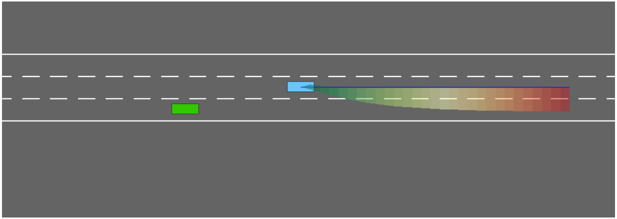


The proposed predictor (2) remains stable

Prediction Results



Prediction during a lane change maneuver



Prediction with uncertainty in the followed lane L_i

Approximate the robust objective by a tractable surrogate.

Definition (Robust objective v^r)

$$v^r(\pi) \stackrel{\text{def}}{=} \min_{A(\theta)\in C_\delta} \sum_{t=0}^{H} \gamma^t R(x_t, \pi(x_t)) \tag{3}$$

Definition (Surrogate objective $\hat{v}^r$)

$$\hat{v}^r(\pi) \stackrel{\text{def}}{=} \sum_{t=0}^{H} \gamma^t \min_{[x\in \underline{x}(t),\overline{x}(t)]} R(x, \pi(x)) \tag{4}$$

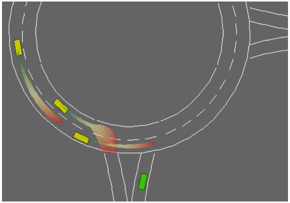
The approximate performance of a policy is guaranteed on the true environment.

Proposition (Lower bound)

The surrogate objective $\hat{v}^r$ is a lower bound of the true objective v^r:

$$\forall \pi, \;\; \hat{v}^r(\pi) \leq v^r(\pi) \tag{5}$$

Ambiguity	Agent	Worst-case	Mean $\pm$ std
None	Oracle	9.83	10.84 ± 0.16
Continuous	Nominal	1.99	9.95 ± 2.38
	Robust	7.88	10.73 ± 0.61



But what if...

Our linear structure assumption is wrong?

Model Adequacy: you can detect it with statistical tests

Our linear structure assumption is wrong?

Model Adequacy: you can detect it with statistical tests

Solution: Multi-Model Prediction

Use many linear models with different features. For instance:

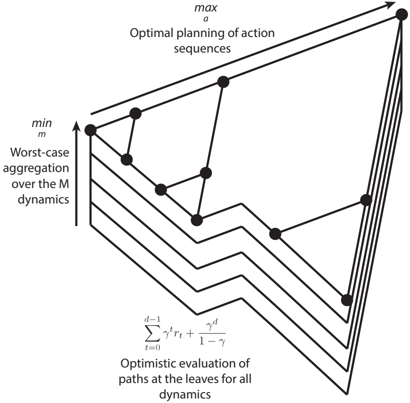
- Lane-dependent features
- Neural network features
- Random features

↳ Maintain a set of admissible experts

↳ Perform robust aggregation

Assumption (Discrete Ambiguity Set)

$$T \in \{T_1, \cdots, T_m\}$$



Definition (Robust sequence value upper-bound)

Given node $i \in \mathcal{T}$, define the robust B-value:

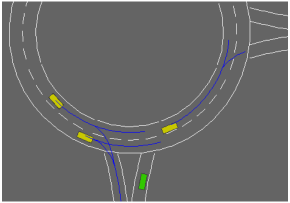
$$B_i^r(n) \stackrel{\text{def}}{=} \begin{cases} \min\limits_{m\in[1,M]} \sum_{t=0}^{d-1} \gamma^t r_t + \frac{\gamma^d}{1-\gamma} & \text{if } i \in \mathcal{L}_n \ ; \\ \max\limits_{a\in\mathcal{A}} b_{ia}^r(n) & \text{if } i \in \mathcal{T}_n \setminus \mathcal{L}_n \end{cases}$$

Theorem (Regret bound)

The corresponding planning algorithm enjoys a simple regret of:

$$\text{If } \kappa > 1, \qquad \mathcal{R}_n = O\left(n^{-\frac{\log 1/\gamma}{\log \kappa}}\right) \tag{6}$$

Ambiguity	Agent	Worst-case	Mean $\pm$ std
None	Oracle	9.83	10.84 ± 0.16
Continuous	Nominal	1.99	9.95 ± 2.38
	Robust	7.88	10.73 ± 0.61
Discrete	Nominal	2.09	8.85 ± 3.53
	Robust	8.99	10.78 ± 0.34



Decision-making among interacting drivers with behavioural uncertainty

Model-free

1. Self-attention model for permutation invariance and variable size
2. Budgeted reinforcement learning to constrain the expected risk

Model-based

3. Efficient tree-based planning with tight statistical bounds
4. Tackle the issue of model bias
   - ↳ Build a confidence region around the true model
   - ↳ Design a stable interval predictor
   - ↳ Perform robust control with respect to this uncertainty

Thank You!