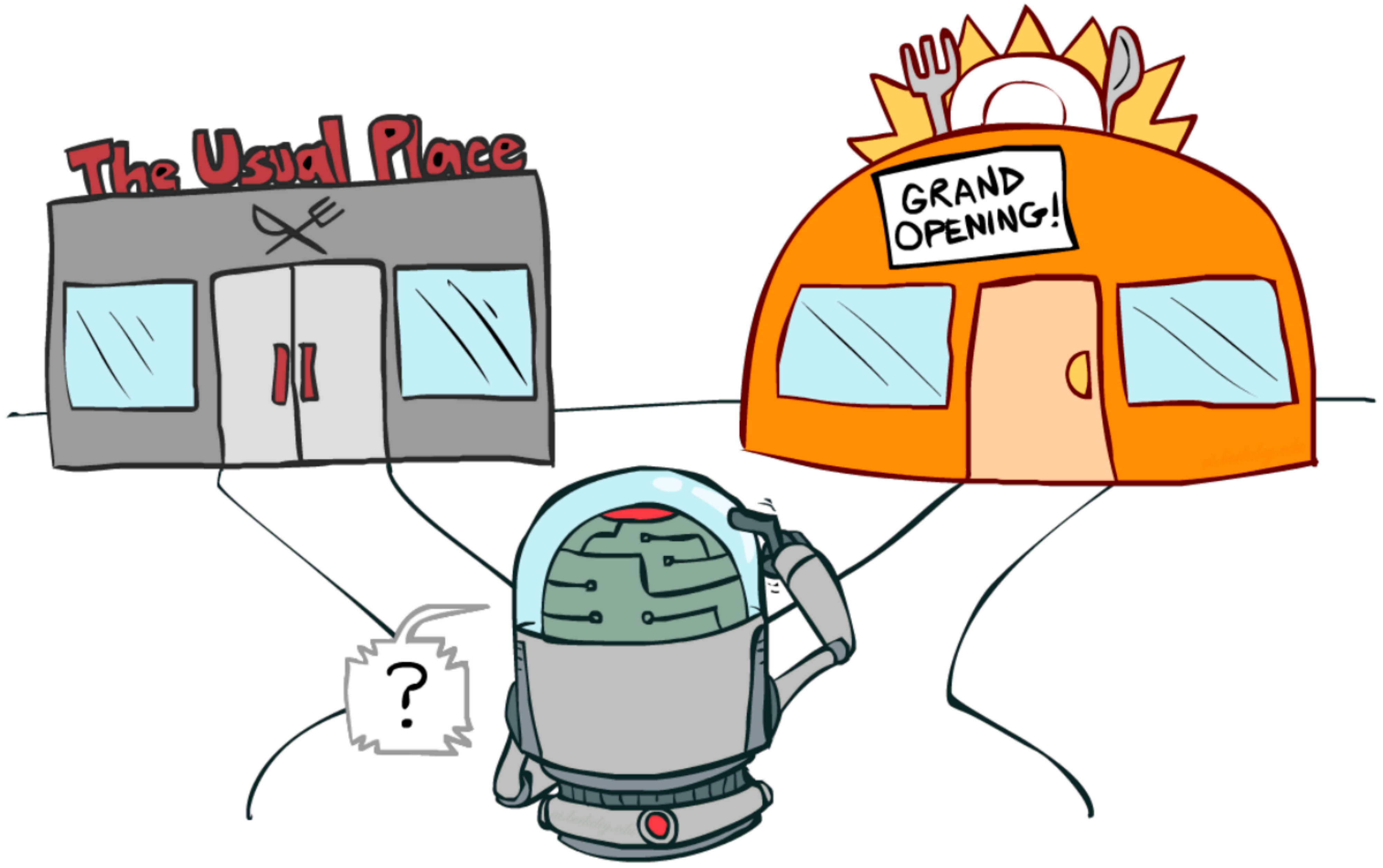
Black-box vs White-box Policy Optimization

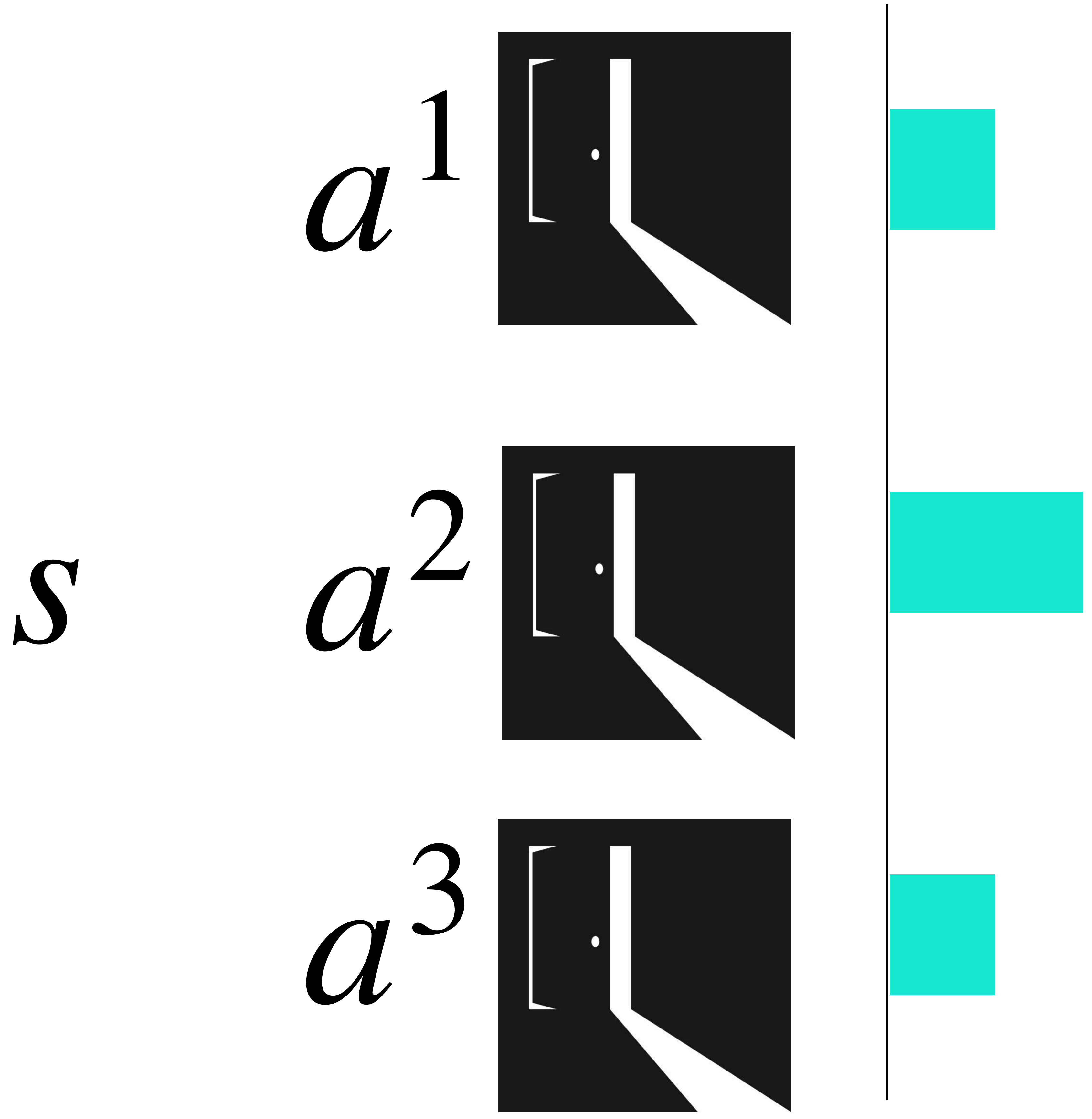
Sanjiban Choudhury

Cornell Bowers CIS
Computer Science

Recap: Two Ingredients of RL

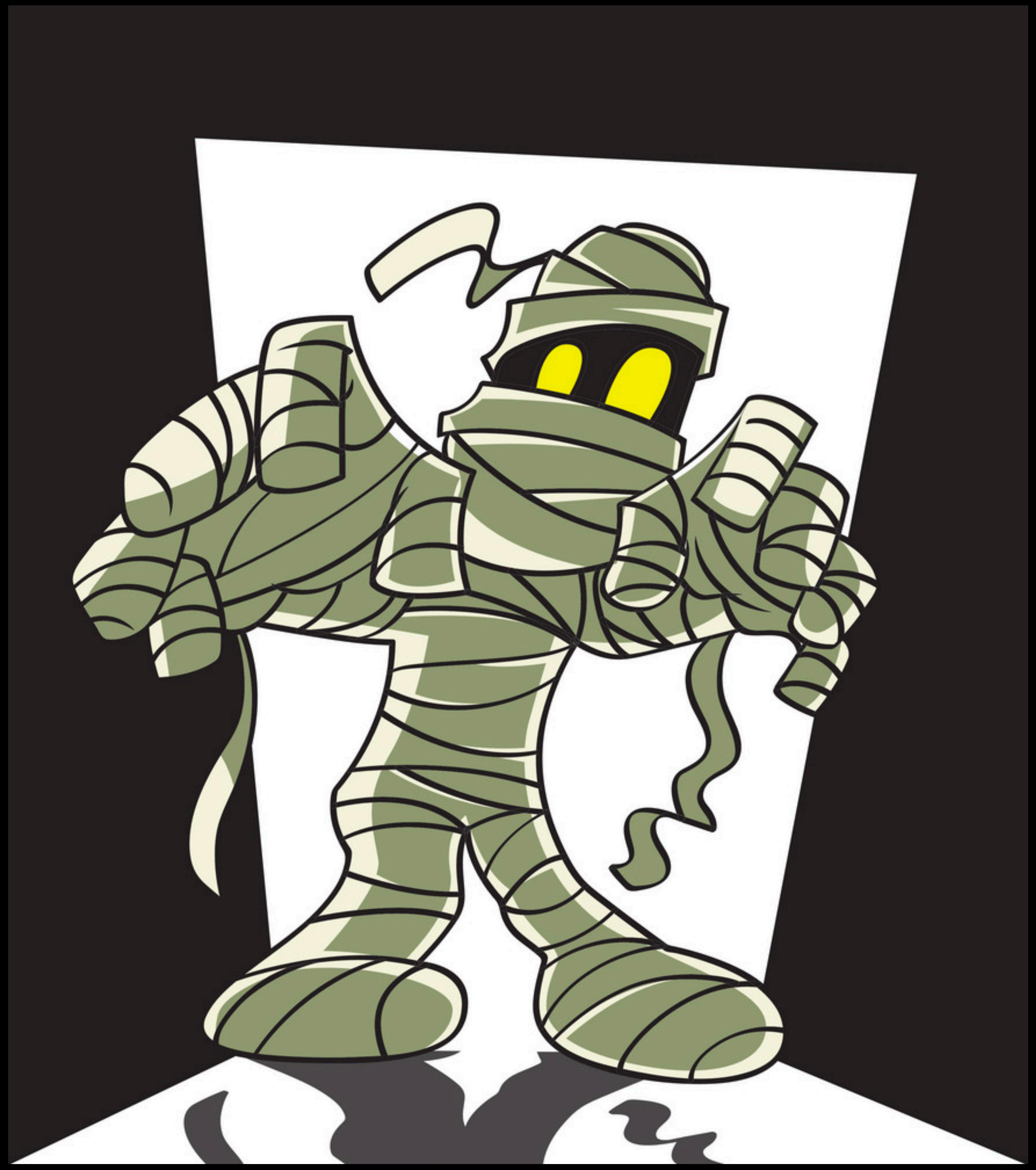


Exploration Exploitation



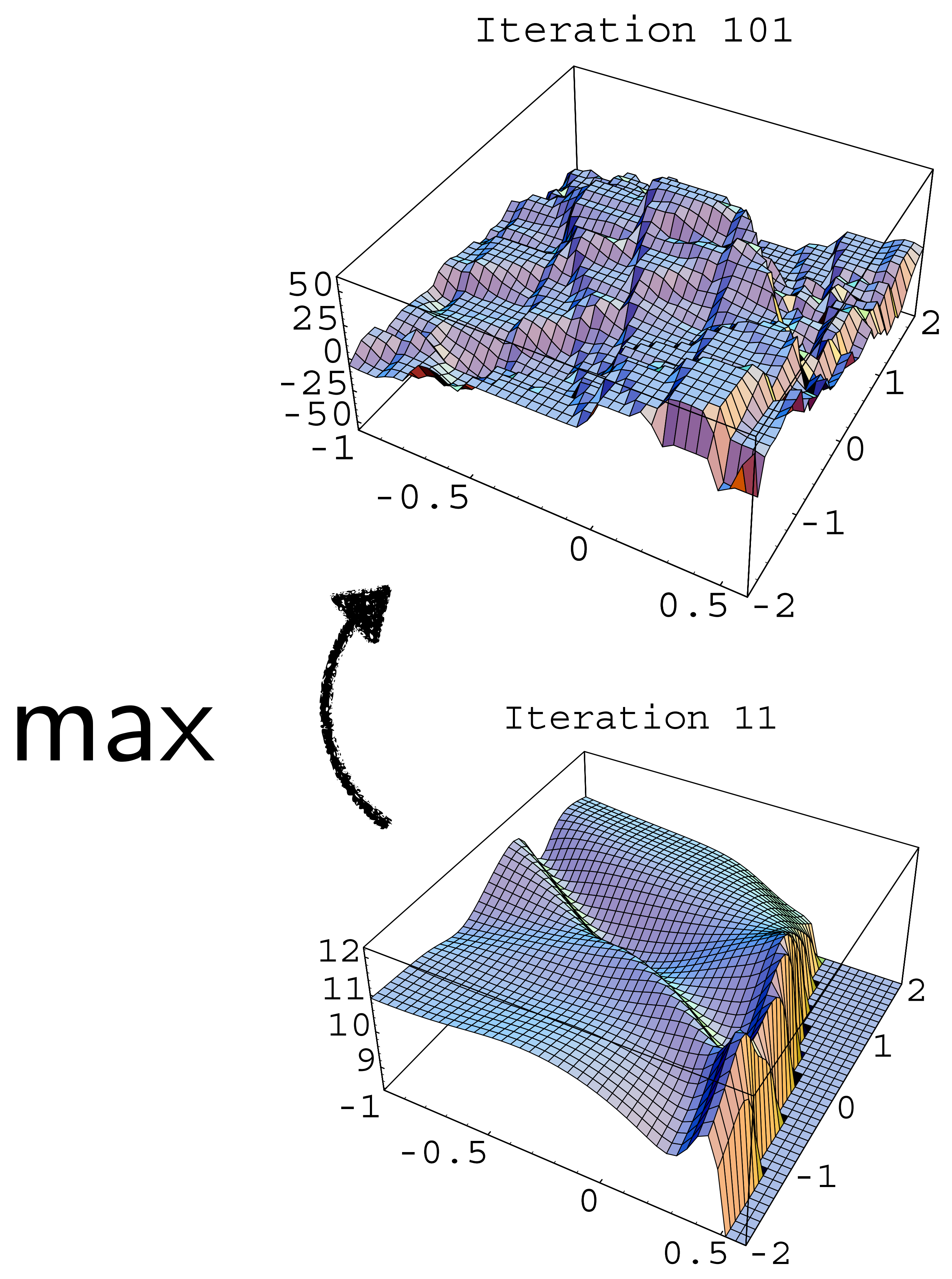
Estimate Values $Q(s, a)$

Curse of Value Approximation!

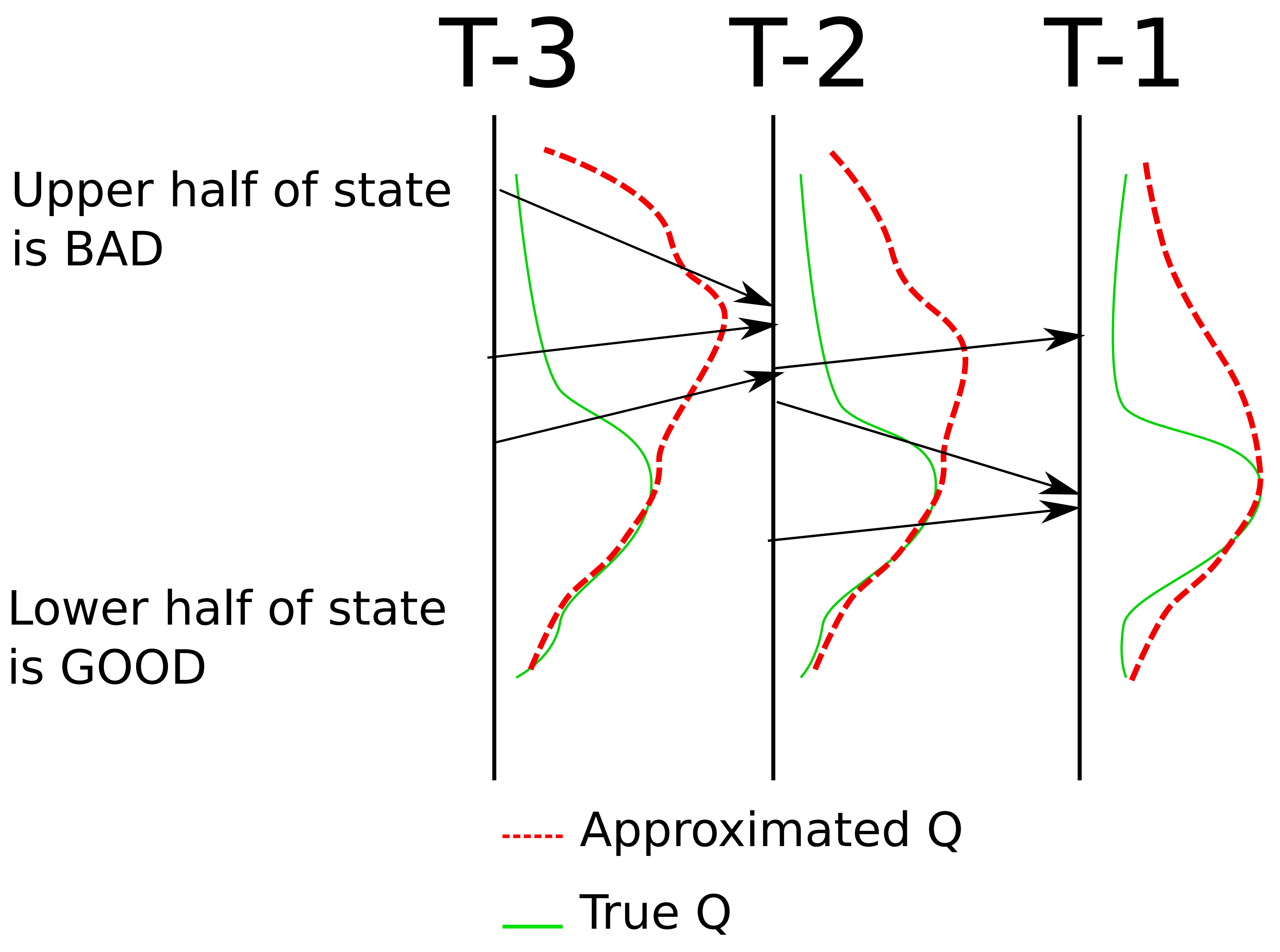


Two sides of the same coin

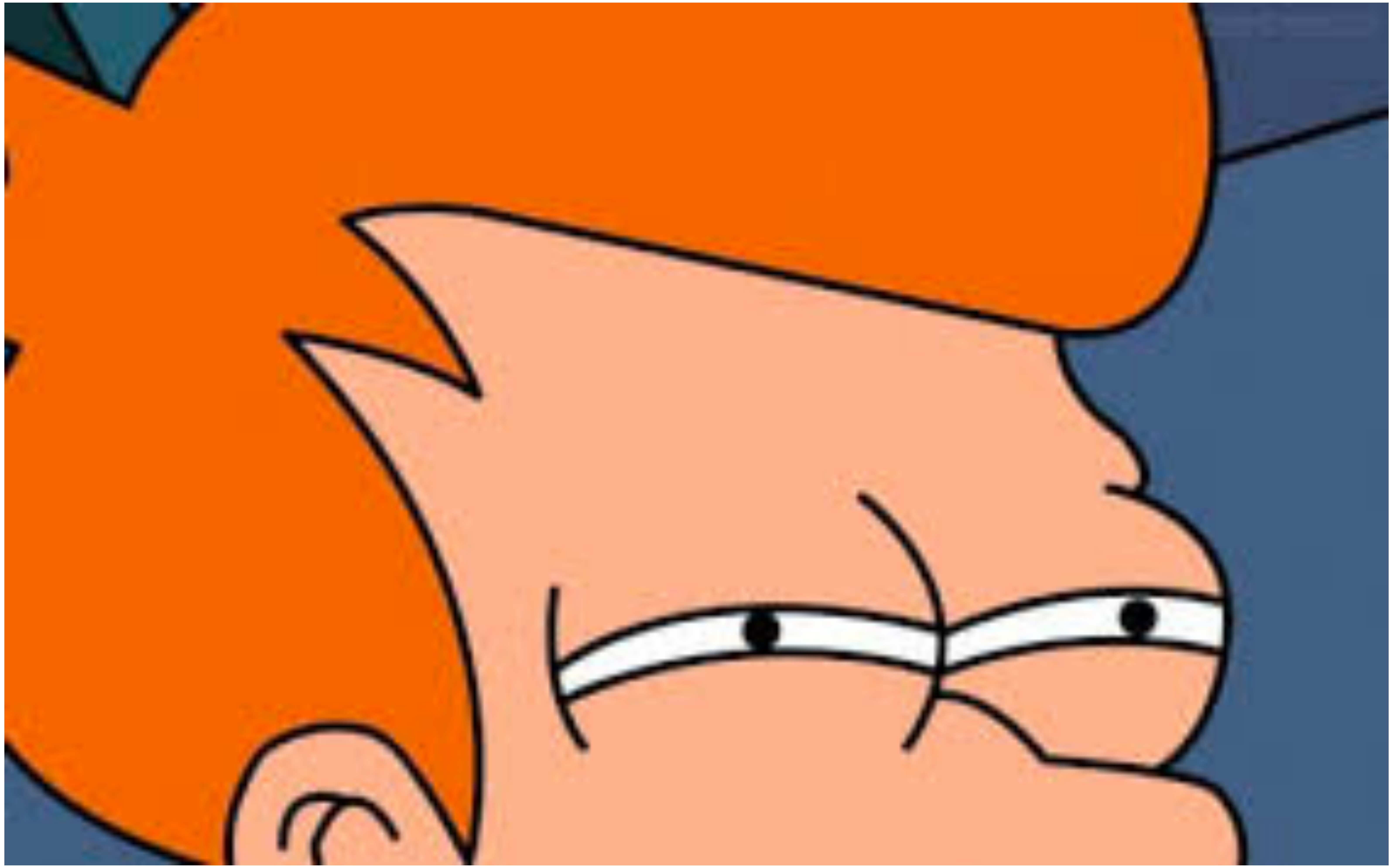
Bootstrapping



Distribution shift



To hell with Value Estimates!



Trust ONLY actual Returns

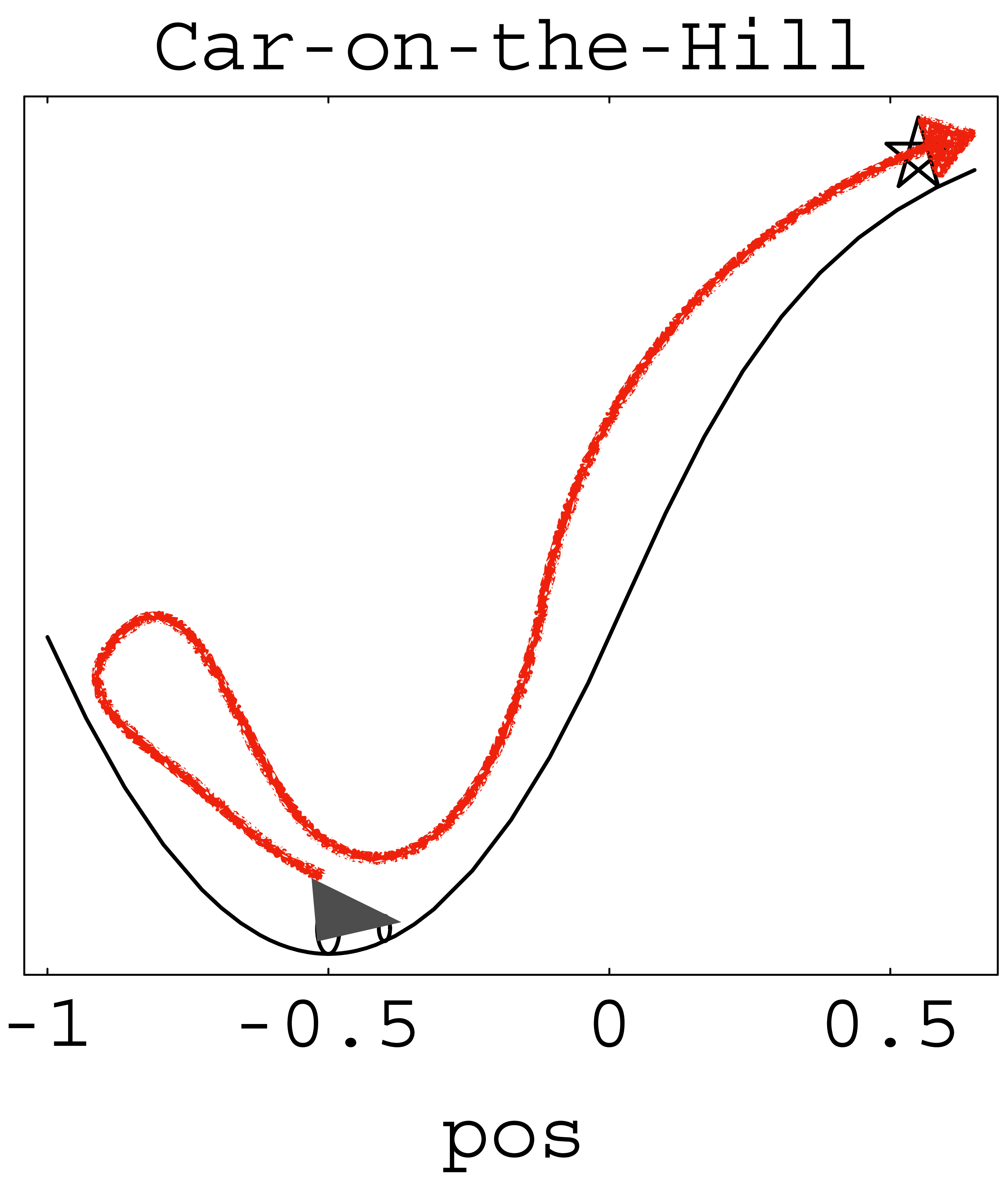
Bye Bye Bellman ...

"not to be blinded by the beauty of the Bellman equation"

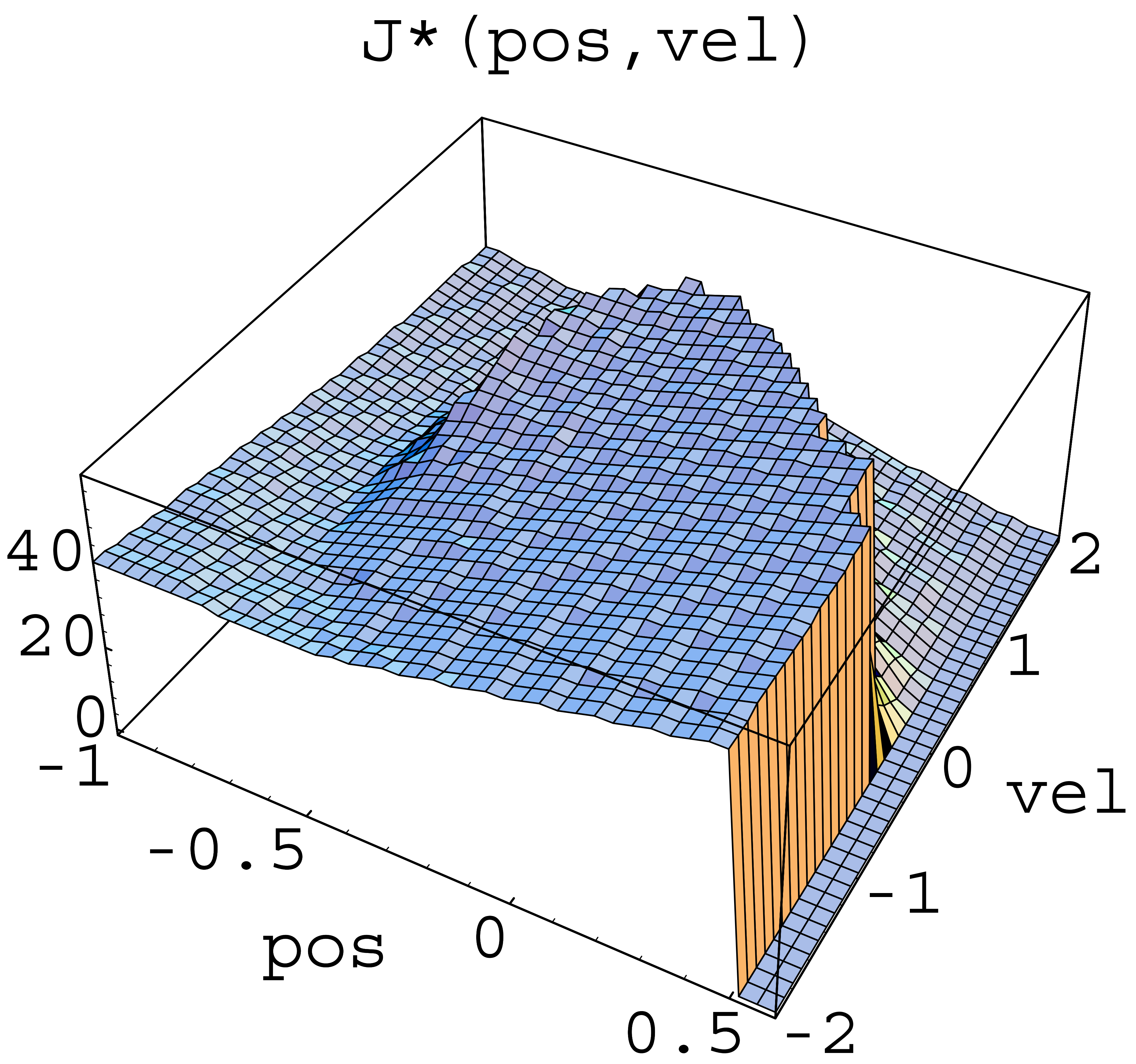
- Andrew Moore

What if we focused on finding good policies ... ?

Sometimes a policy is waaaaay simpler than the value



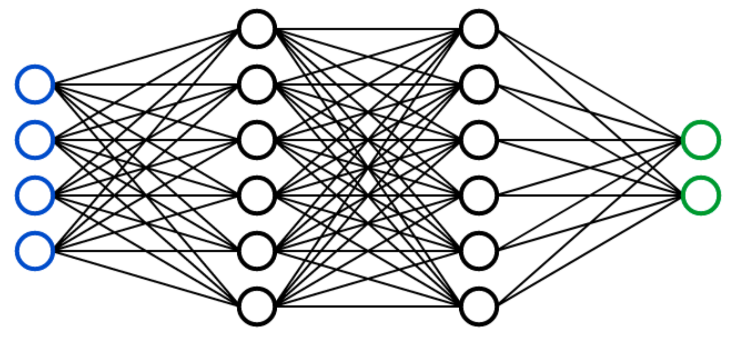
The Policy!



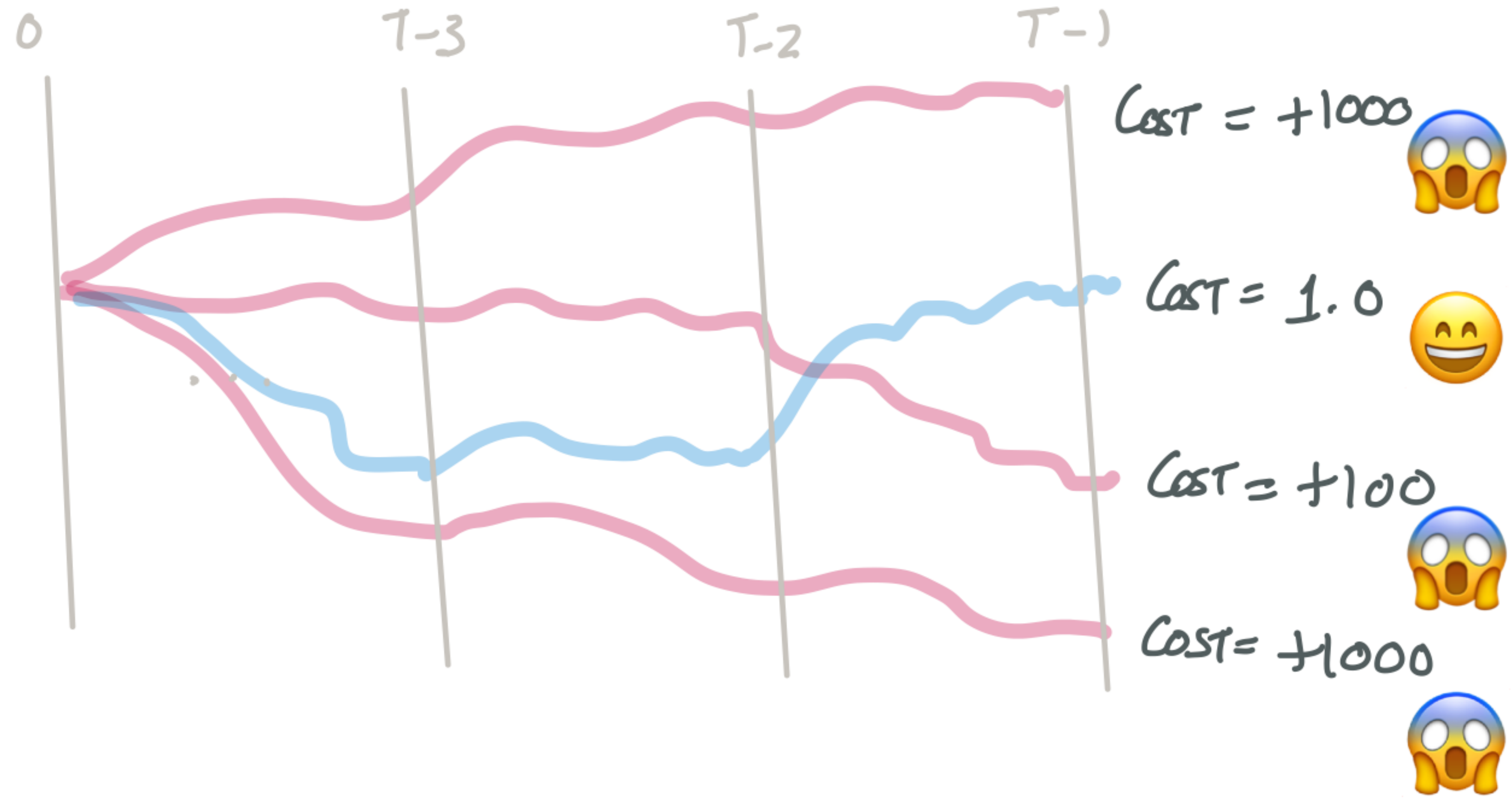
The Value!

Can we just focus on finding a good policy?

$$\pi_\theta : s_t \to a_t$$

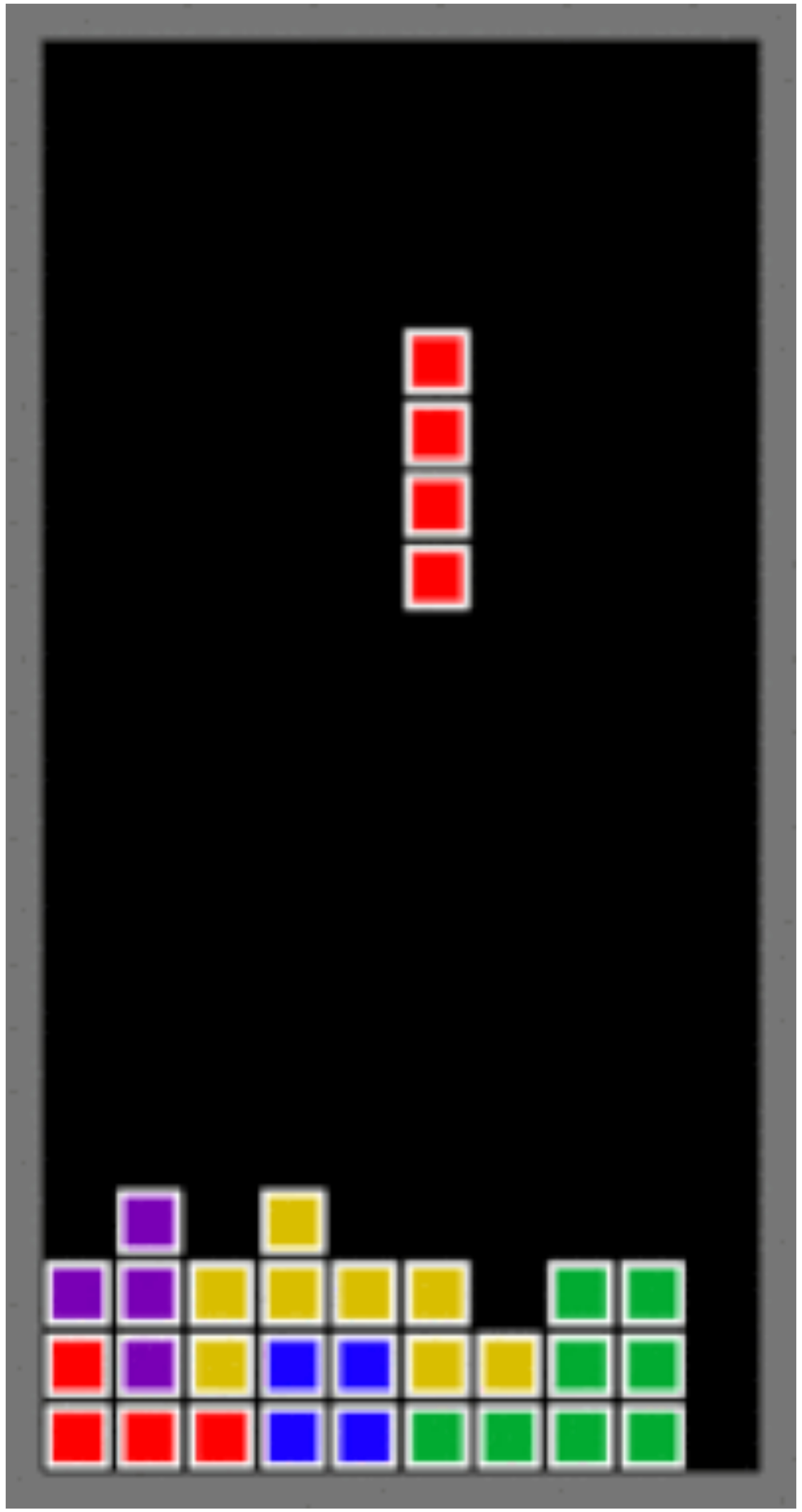


Learn a mapping from states to actions



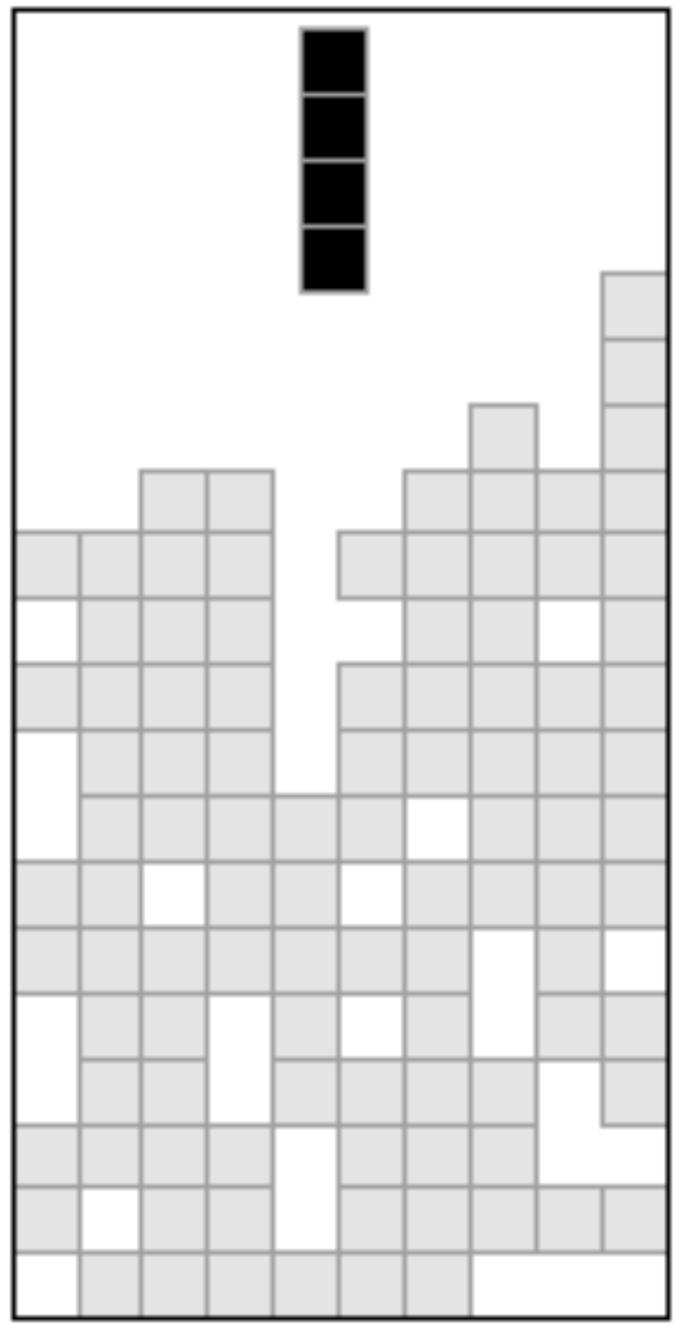
Roll-out policies in the real-world to estimate value

The Game of Tetris

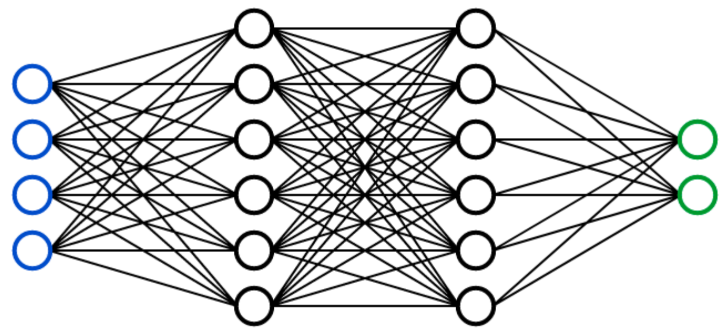


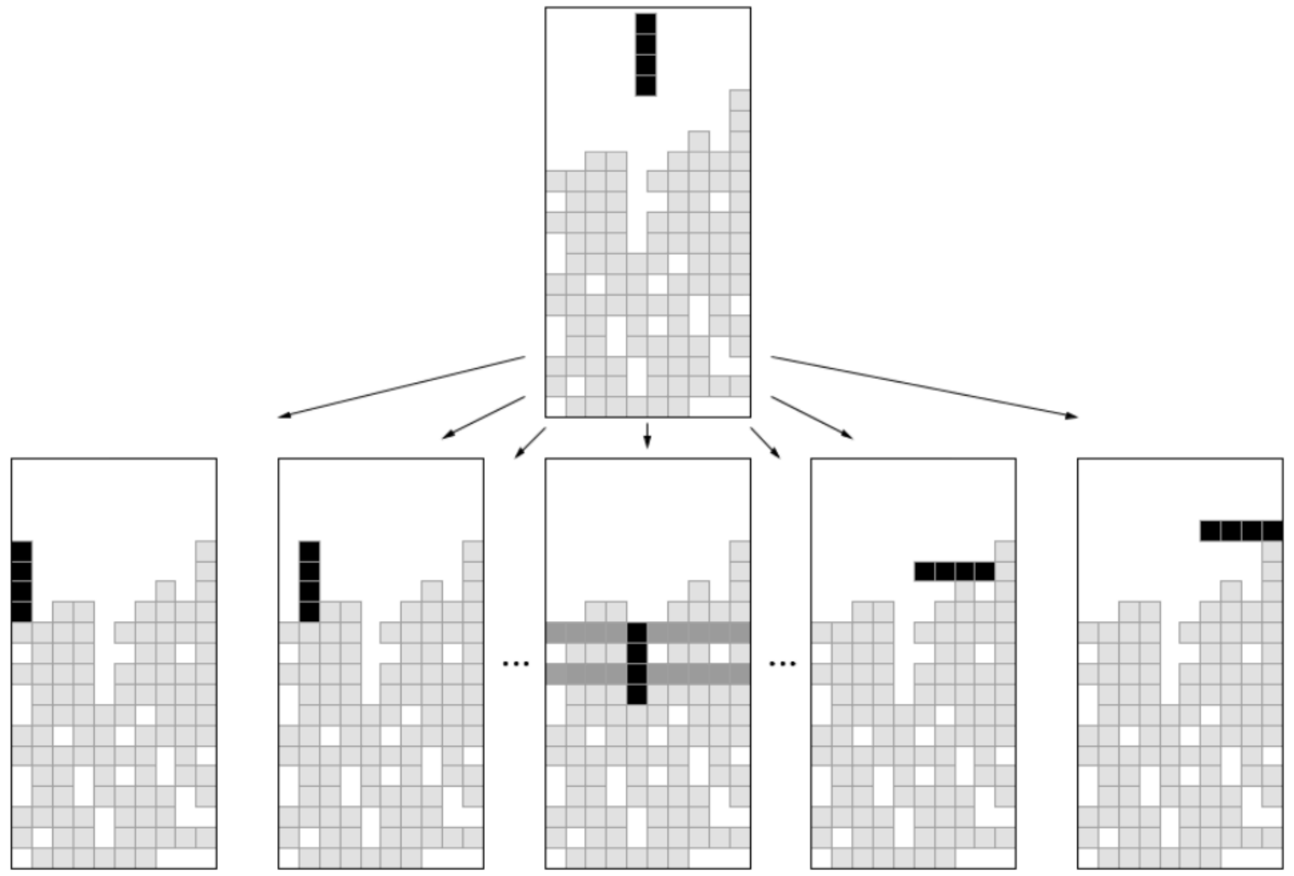
What's a good policy representation for Tetris?

(4 rotations)*(10 slots)
 - (6 impossible poses) = 34



$$\pi_\theta : s_t \rightarrow a_t$$





State (s_t)

Action (a_t)

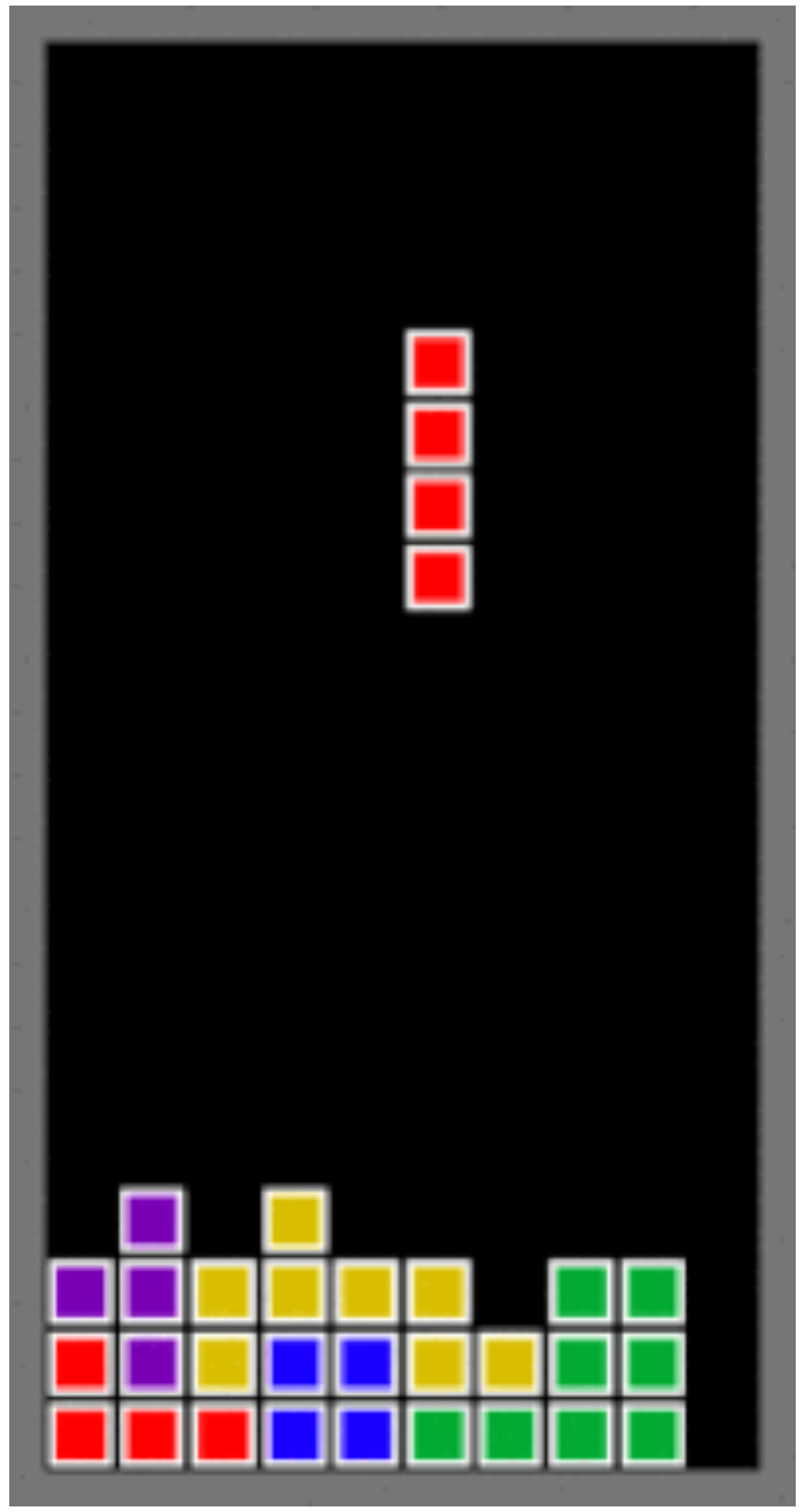
Activity!

Think-Pair-Share

Think (30 sec): Ideas for how to represent policy for tetris?

Pair: Find a partner

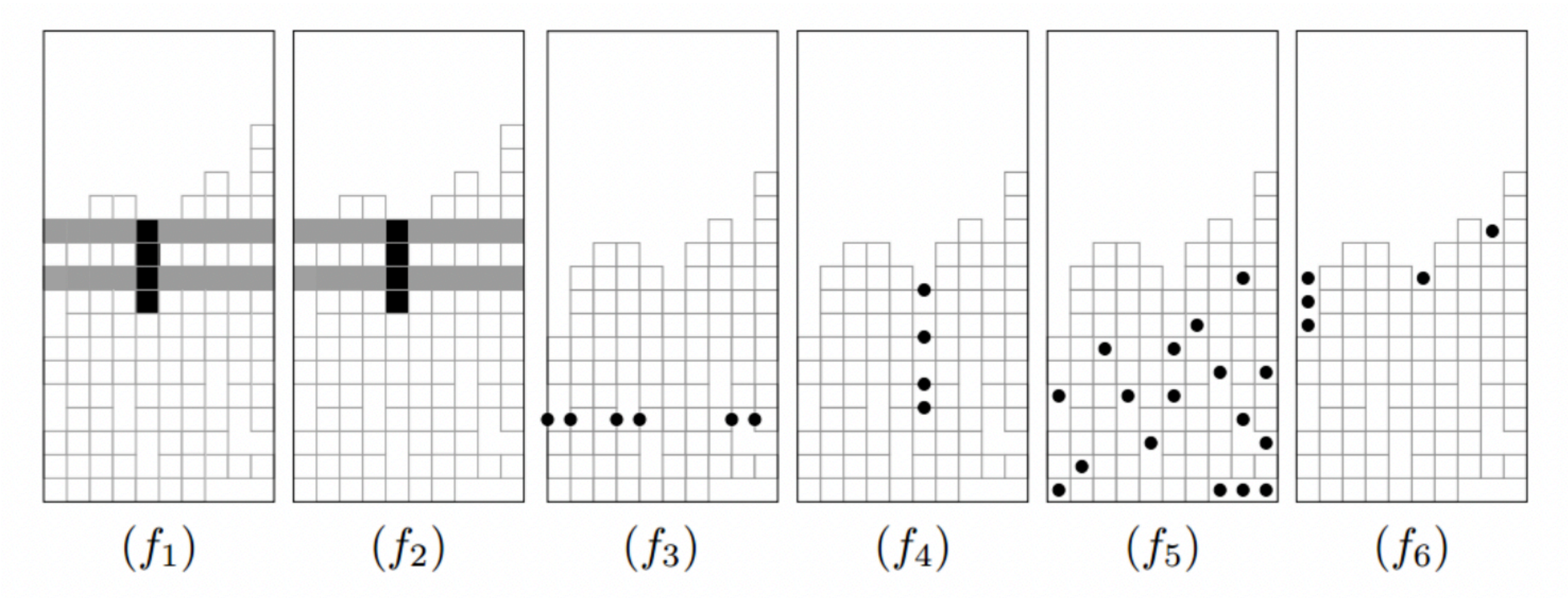
Share (45 sec): Partners exchange ideas



Some inspiration for Tetris policy

Until 2008, the best artificial Tetris player was handcrafted, as reported by Fahey (2003). Pierre Dellacherie, a self declared average Tetris player, identified six simple features and tuned the weights by trial and error.

Dellacherie Features



Landing Heights

Eroded Cells

Row Transitions

Column Transitions

Holes

Cumulative Wells

The contribution of the last piece to the cleared lines time the number of cleared lines.

The number of filled cells adjacent to the empty cells summed over all rows

A well is a succession of empty cells and the cells to the left and right are occupied

A *magic* formula ?!?

$$
\begin{aligned}
&-4 \times holes - cumulative\, wells \\
&- row\, transitions - column\, transitions \\
&- landing\, height + eroded\, cells
\end{aligned}
$$

A *magic* formula ?!?

$$-4 \times holes - cumulative\ wells - row\ transitions - column\ transitions - landing\ height + eroded\ cells$$

This linear evaluation function cleared an average of 660,000 lines on the full grid ...
... In the simplified implementation used by the approaches discussed earlier, the games would have continued further, until every placement would overflow the grid. Therefore, this report underrates this simple linear rule compared to other algorithms.

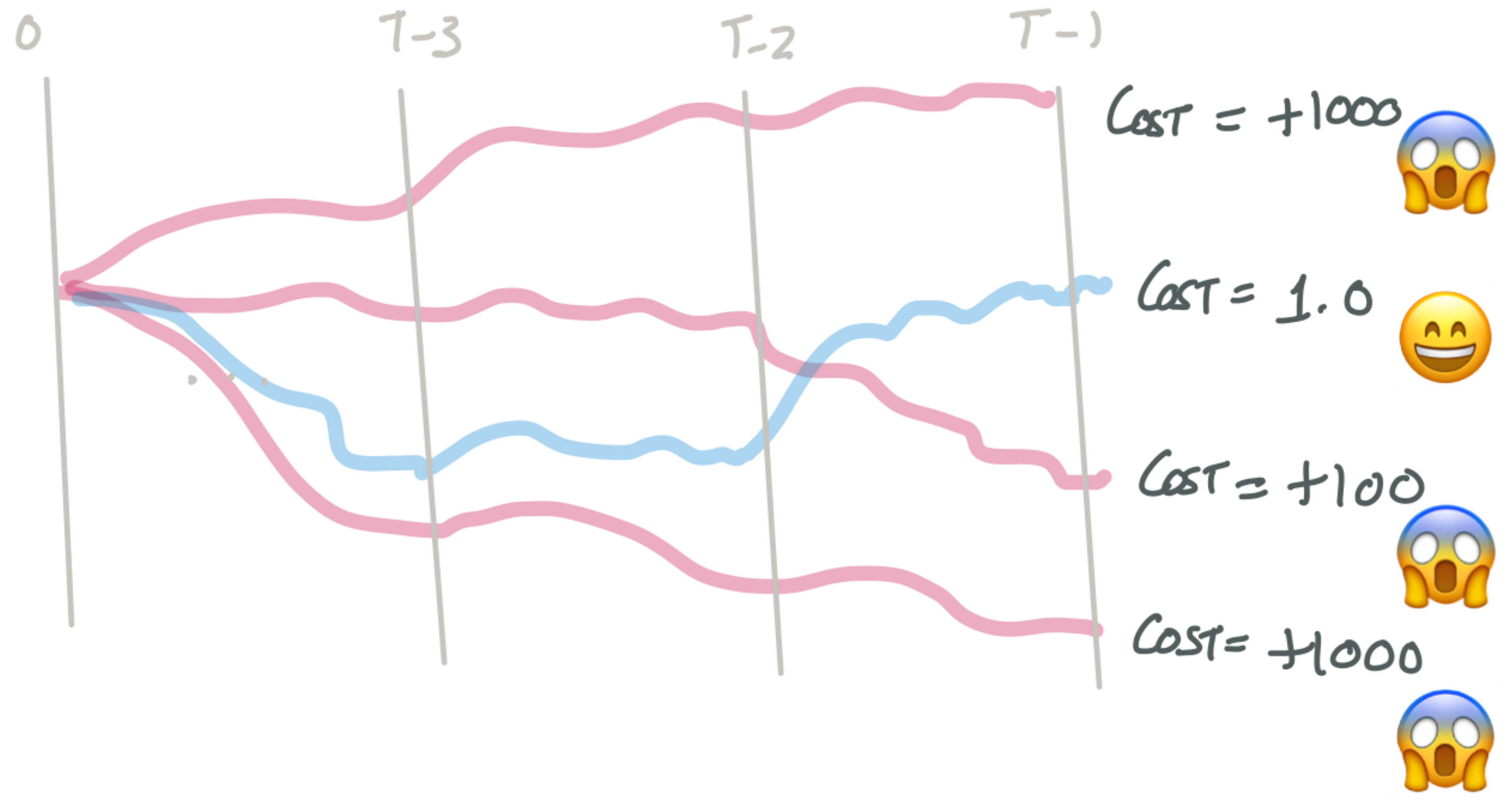
Can YOU do better than Dellacherie?

The Goal of Policy Optimization

$$\pi_\theta(s) = \arg\min_a \theta^T f(s, a)$$

#Think of f(s,a) being dellacherie features

$$\min_\theta J(\theta) = \sum_{t=0}^{T-1} \mathbb{E}_{\pi_\theta} c(s_t, a_t)$$



#Think of c(s,a) as
-num_rows_cleared

Is Policy Optimization a good idea?

😄 **Pros**

A policy is simpler than value function

Easy to bake in engineering knowledge

Easy to code up!

Cons 😱

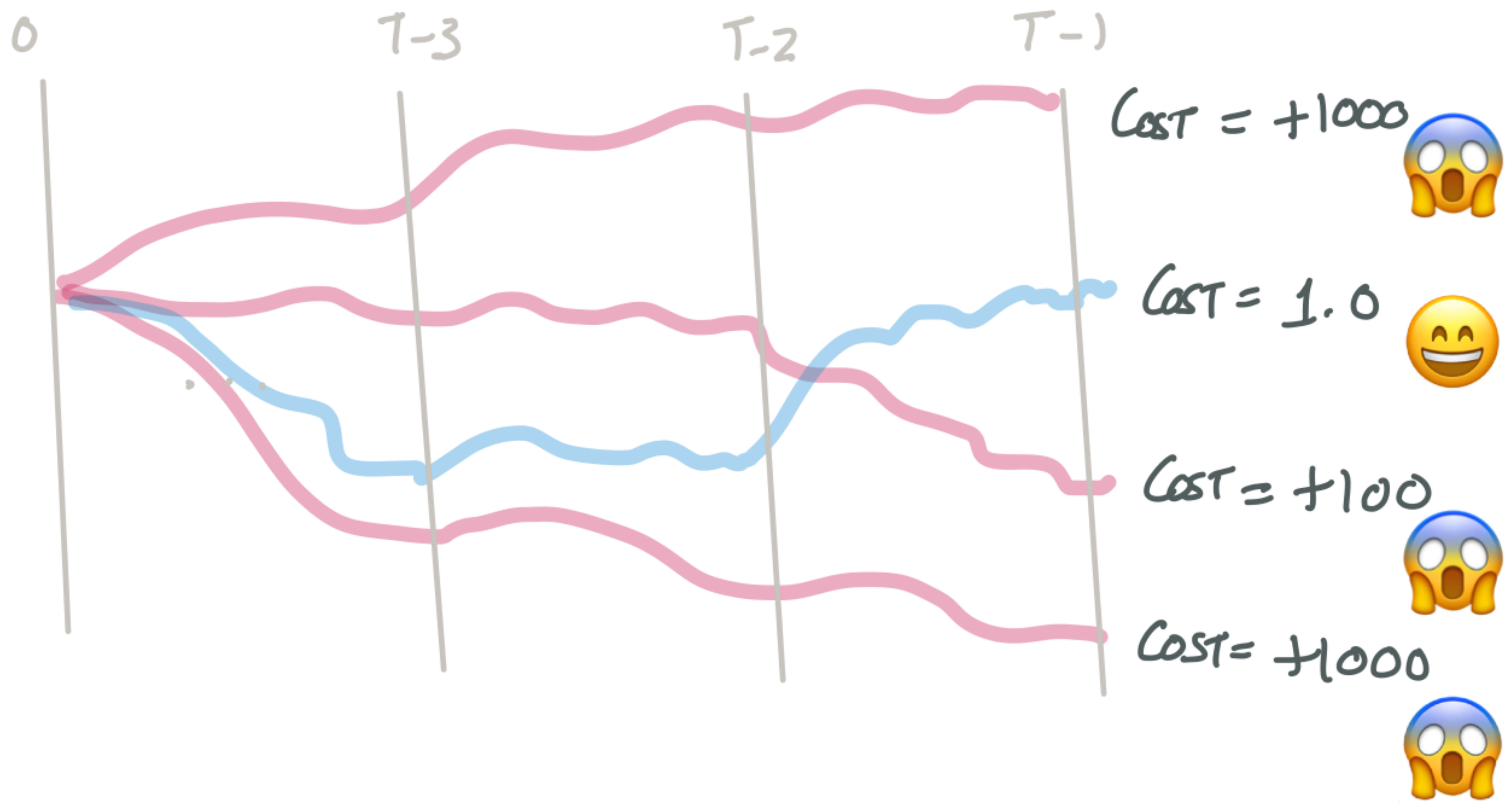
Careful feature engineering

Exploration is difficult in this setting

Ignoring Markov structure of states and costs

What are some ways to solve this optimization?

$$\min_{\theta} J(\theta) = \sum_{t=0}^{T-1} \mathbb{E}_{\pi_\theta} c(s_t, a_t)$$



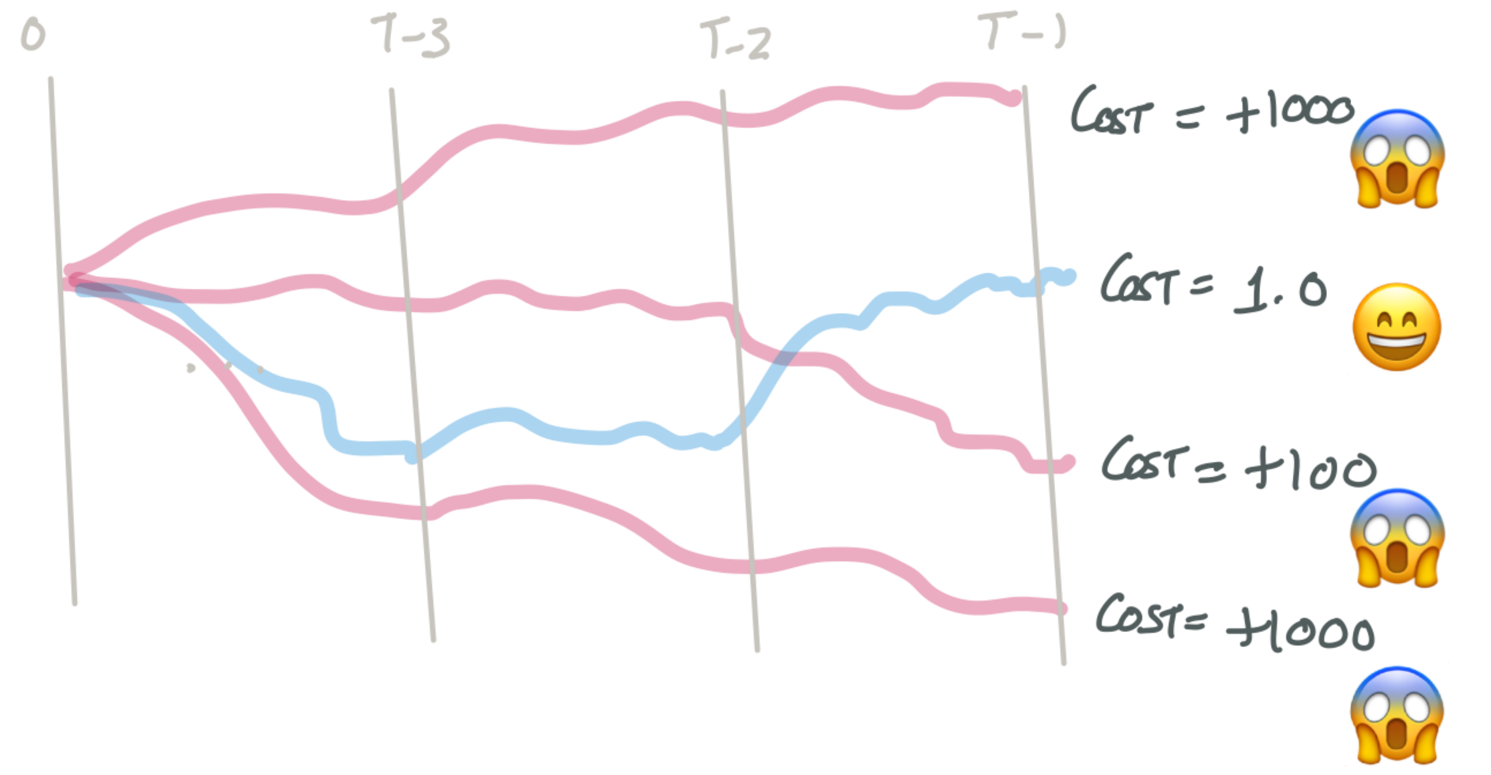
Activity!

Think-Pair-Share

Think (30 sec): What are some of the techniques we can use to solve the policy optimization?

$$\min_{\theta} J(\theta) = \sum_{t=0}^{T-1} \mathbb{E}_{\pi_\theta} c(s_t, a_t)$$

Pair: Find a partner



Share (45 sec): Partners exchange ideas

Option 1: Gradient Descent

$$\theta^+ = \theta - \eta \nabla_\theta J(\theta)$$

What could go wrong?

1. Is $J(\theta)$ differentiable?

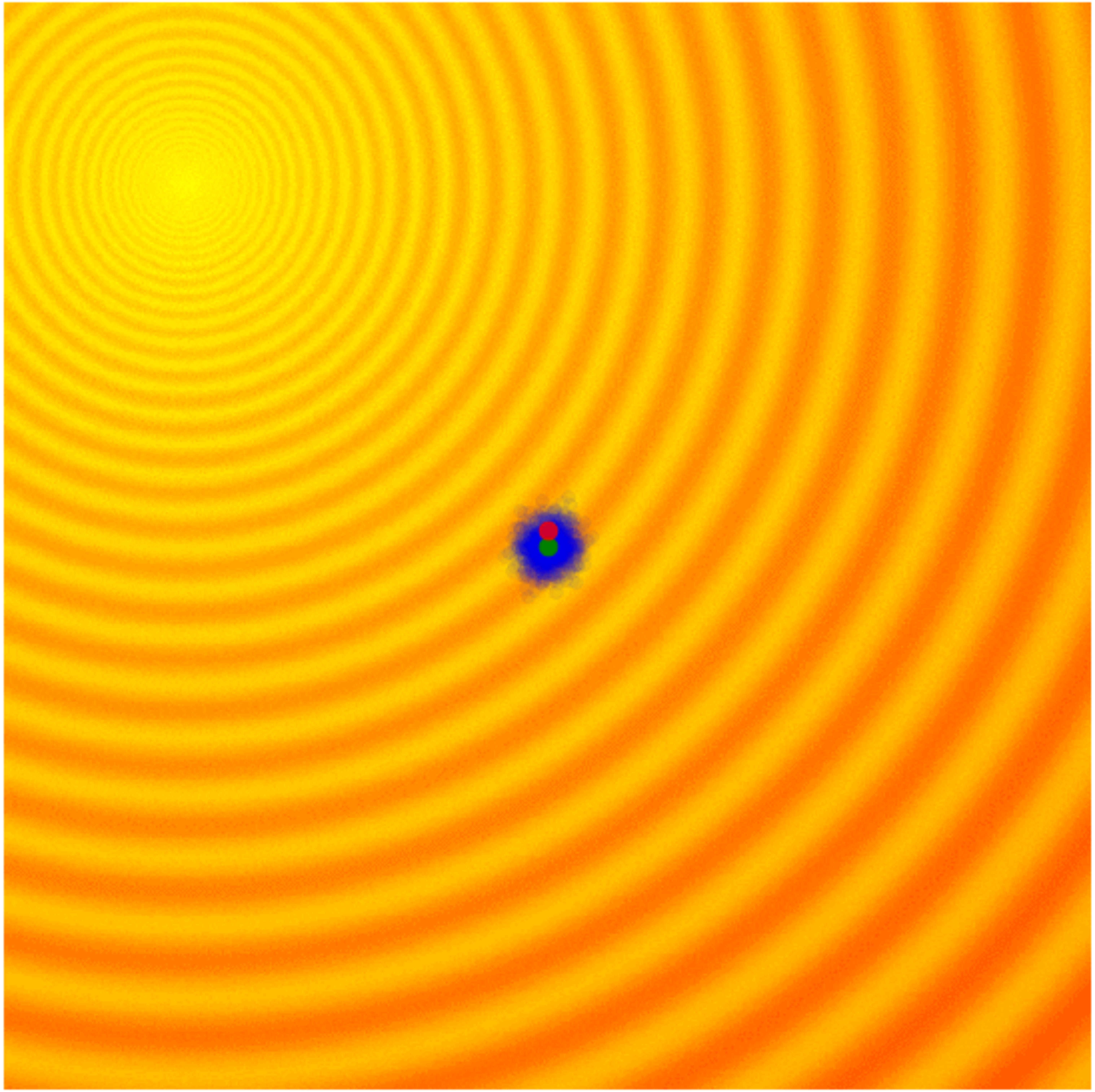
2. Is $J(\theta)$ noisy?

Option 2: Zeroth Order Methods

- Nelder Mead
- Cross Entropy
- Simulated Annealing
- Genetic Algorithm
- Response Surface Methods
- Coordinate Descent


Cross
Entropy
If you were ever stranded on an island ...

Let’s build some intuition!



Credit: https://blog.otoro.net/2017/10/29/visual-evolution-strategies/

The Cross Entropy Algorithm

Init D_θ

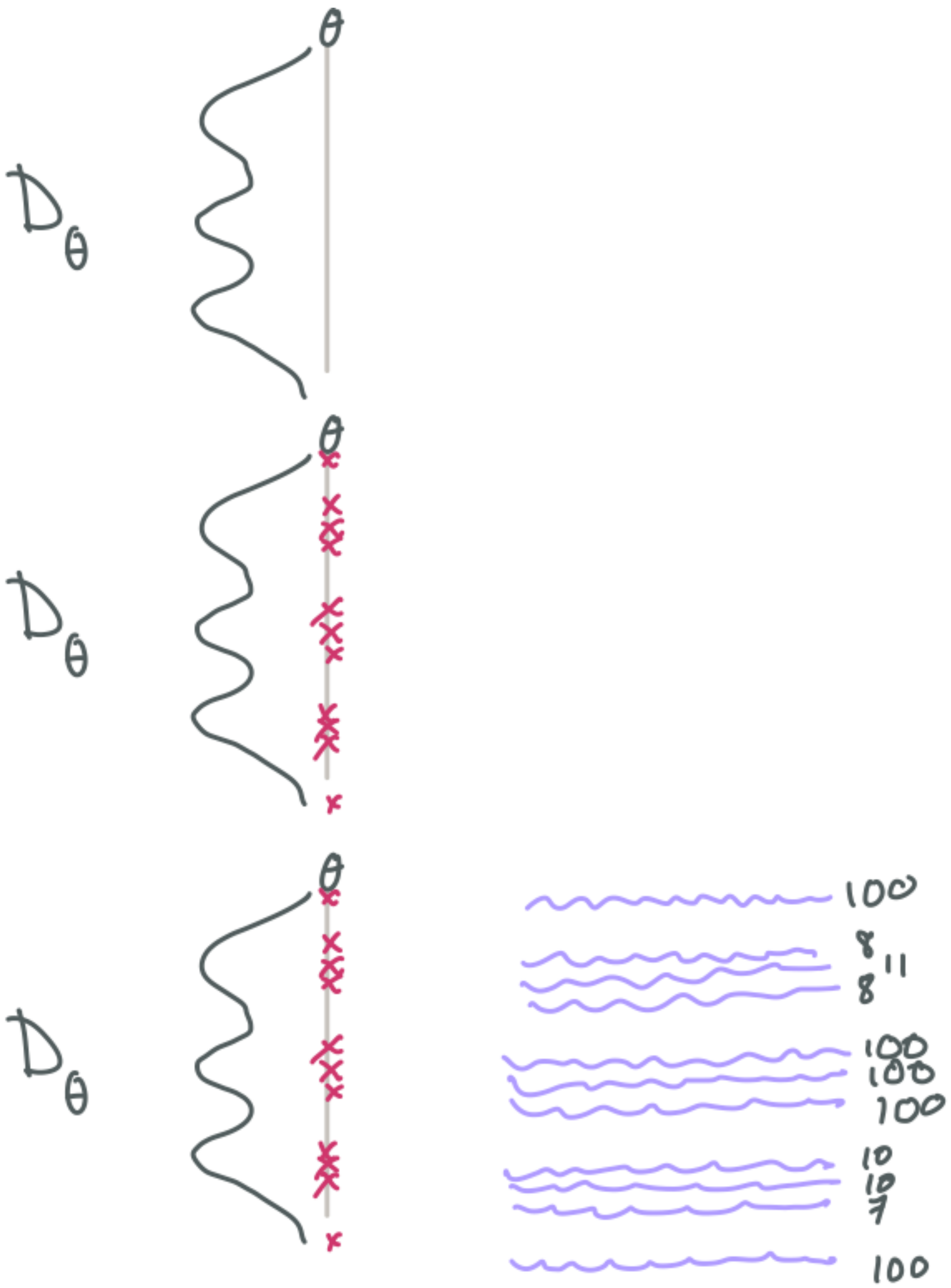
The Cross Entropy Algorithm

INIT D_θ

SAMPLE k TIMES

to get $\{\theta_i\}_{i=1}^{k}$

The Cross Entropy Algorithm



INIT D_θ

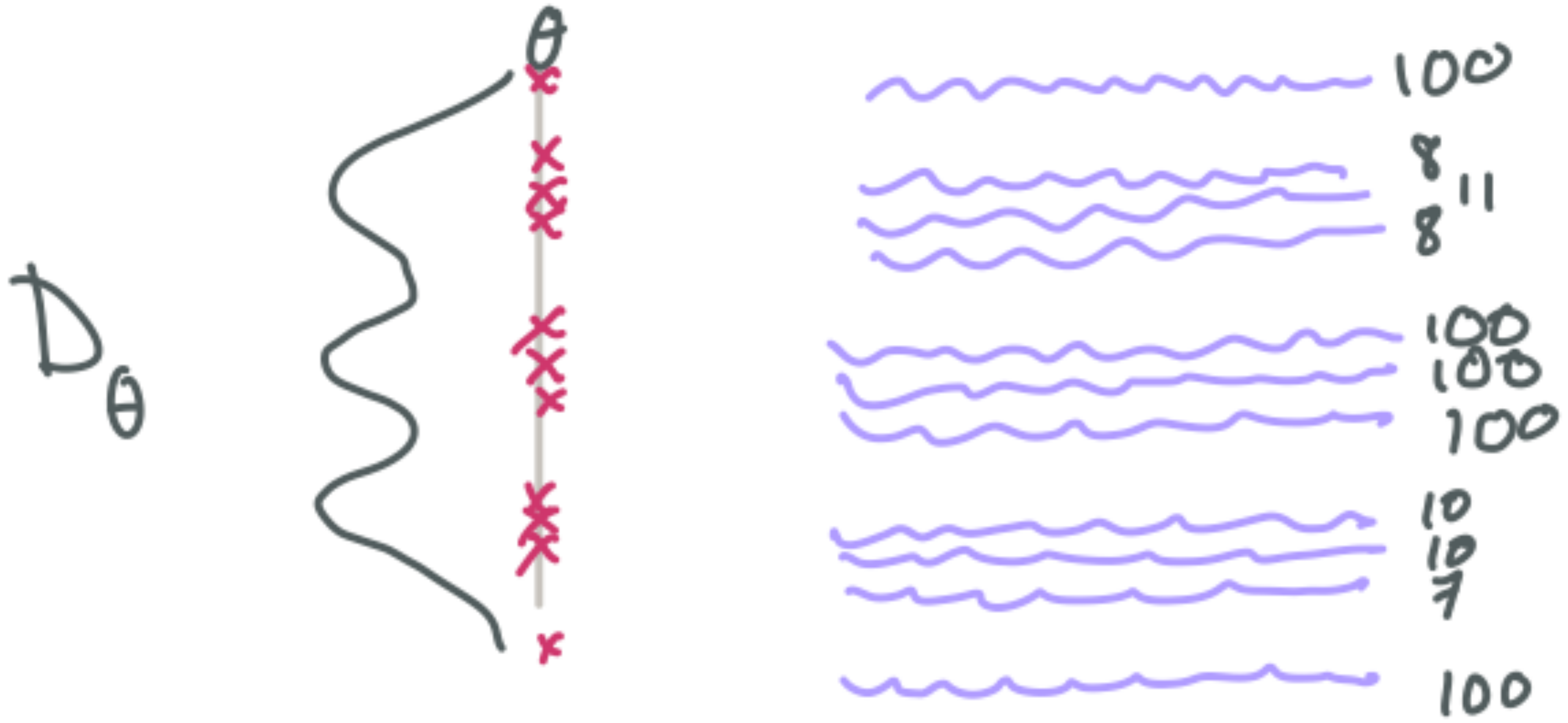
SAMPLE k TIMES

to get $\{\theta_i\}_{i=1}^{k}$

EVALUATE EACH θ_i:

- EXECUTE POLICY MULTIPLE TIMES

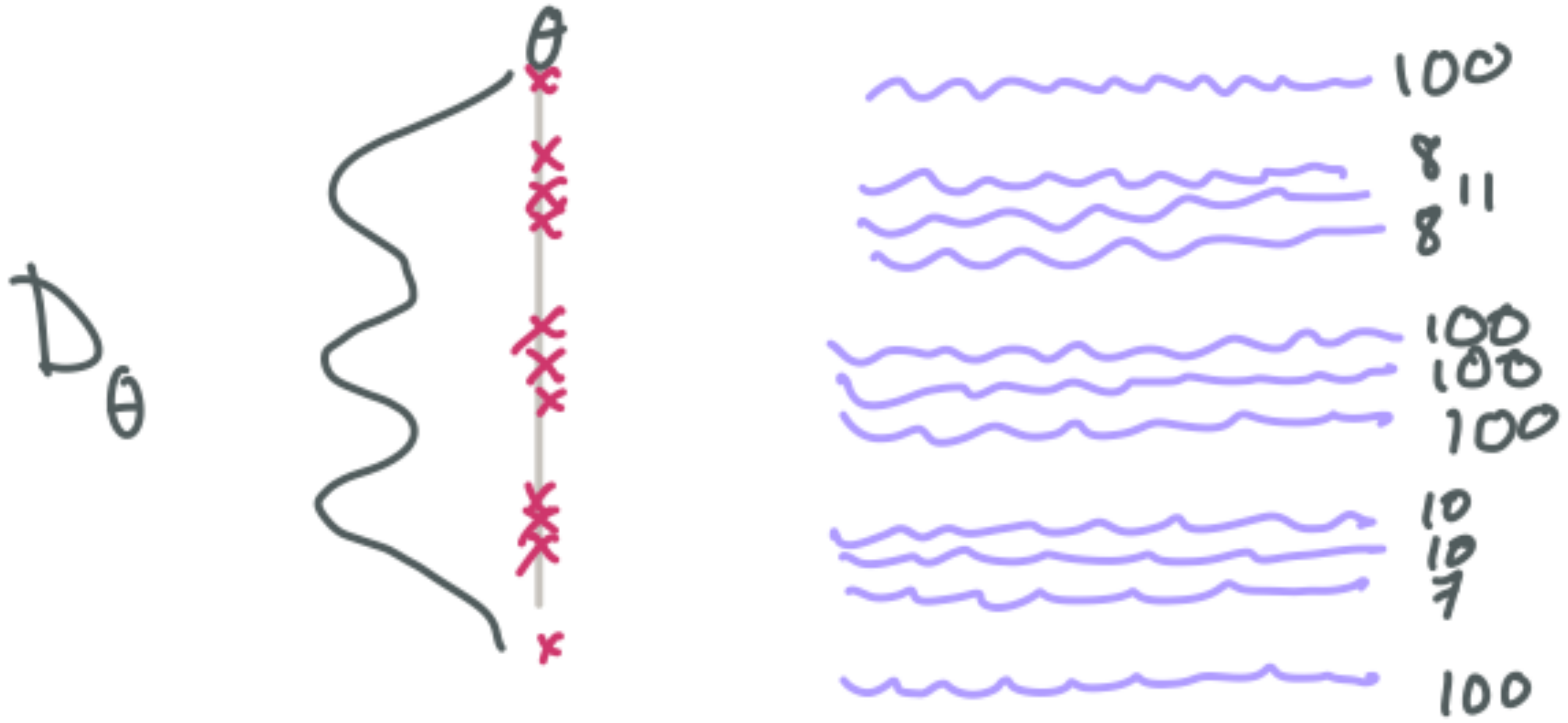
The Cross Entropy Algorithm



EVALUATE EACH θ_i:

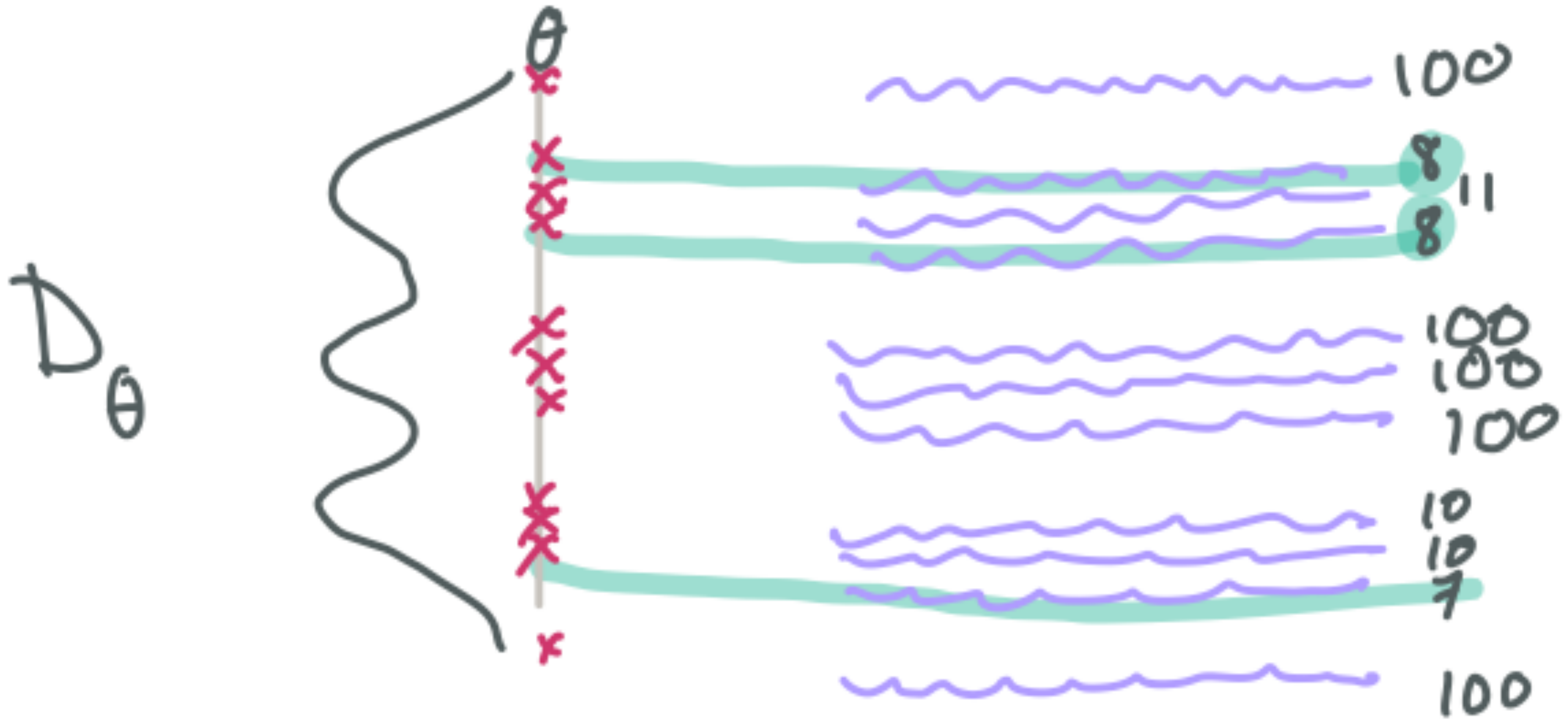
- EXECUTE POLICY MULTIPLE TIMES

The Cross Entropy Algorithm



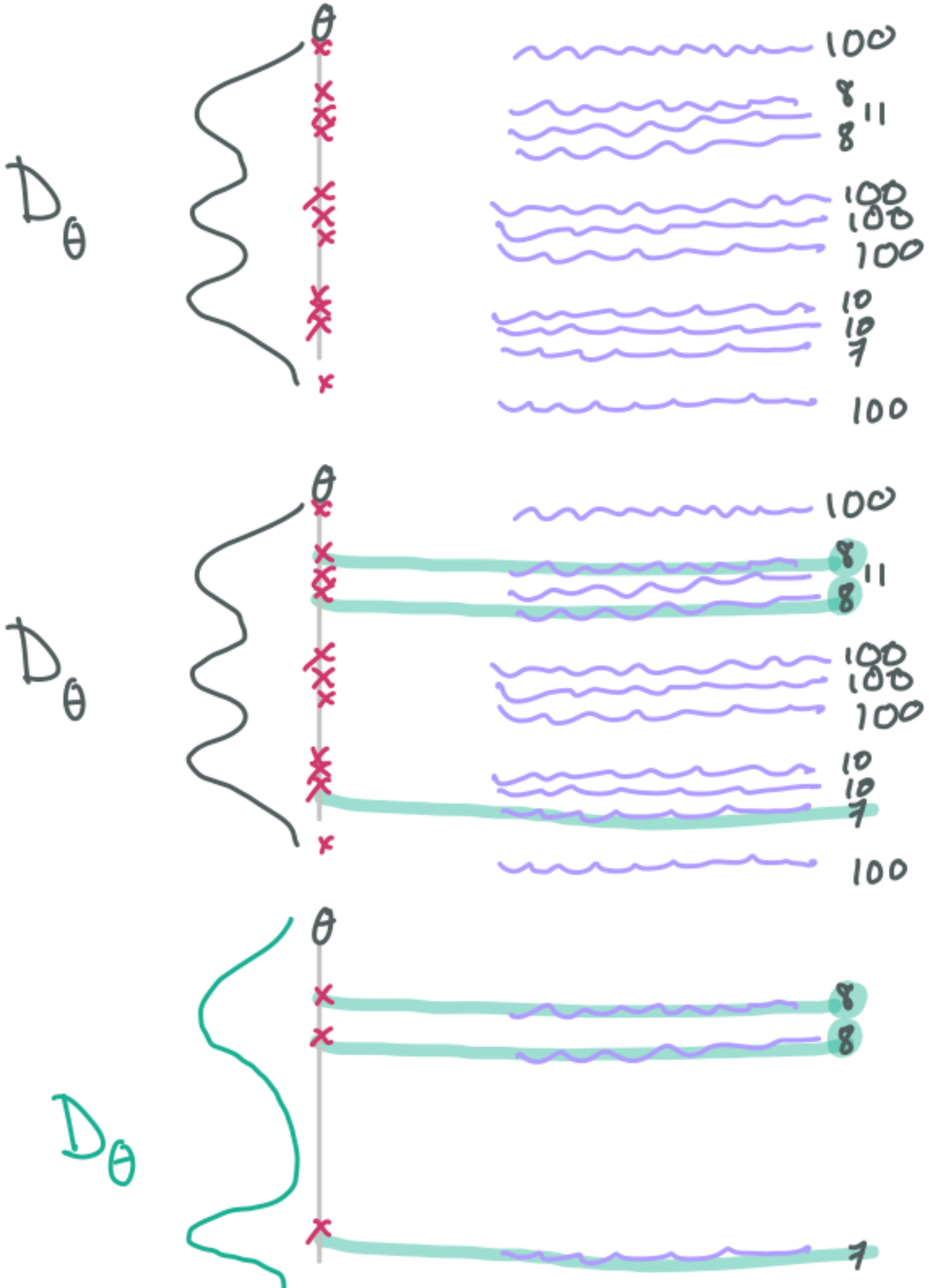
EVALUATE EACH θ_i:

- EXECUTE POLICY MULTIPLE TIMES



FIND TOP 'E' ELITES (e.g. 25%)

The Cross Entropy Algorithm



EVALUATE EACH θ_i:

- EXECUTE POLICY MULTIPLE TIMES

FIND TOP 'E' ELITES (e.g. 25%)

FIT A NEW DISTRIBUTION

D_θ

Cross Entropy for Gaussian

Gaussian Distribution $D_\theta := \mathcal{N}(\mu, \Sigma)$

Mean $$\mu^t = \frac{1}{e}\sum_{i=1}^{e}\theta_i$$

Variance $$\Sigma^t = \frac{1}{e}\sum_{i=1}^{e}(\theta_i - \mu^t)^2$$

Does it work?

Learning Tetris Using the Noisy Cross-Entropy Method

István Szita
szityu@eotvos.elte.hu
András Lőrincz
andras.lorincz@elte.hu
Department of Information Systems, Eötvös Loránd University, Budapest, Hungary H-1117

The cross-entropy method is an efficient and general optimization algorithm. However, its applicability in reinforcement learning (RL) seems to be limited because it often converges to suboptimal policies. We apply noise for preventing early convergence of the cross-entropy method, using Tetris, a computer game, for demonstration. The resulting policy outperforms previous RL algorithms by almost two orders of magnitude.

Does it work?

	Algorithm	Grid Size	Lines Cleared	Feature Set Used
Tsitsiklis & Van Roy (1996)	Approximate value iteration	16×10	30	Holes and pile height
Bertsekas & Tsitsiklis (1996)	λ - PI	19×10	2,800	Bertsekas
Lagoudakis et al. (2002)	Least-squares PI	20×10	$\approx$ 2,000	Lagoudakis
Kakade (2002)	Natural policy gradient	20×10	$\approx$ 5,000	Bertsekas
Dellacherie [Reported by Fahey (2003)]	Hand tuned	20×10	660,000	Dellacherie
Ramon & Driessens (2004)	Relational RL	20×10	$\approx$ 50	
Böhm et al. (2005)	Genetic algorithm	20×10	480,000,000 (Two Piece)	Böhm
Farias & Van Roy (2006)	Linear programming	20×10	4,274	Bertsekas
Szita & Lörincz (2006)	Cross entropy	20×10	348,895	Dellacherie
Romdhane & Lamontagne (2008)	Case-based reasoning and RL	20×10	$\approx$ 50	
Boumaza (2009)	CMA-ES	20×10	35,000,000	BCTS
Thiery & Scherrer (2009a;b)	Cross entropy	20×10	35,000,000	DT
Gabillon et al. (2013)	Classification-based policy iteration	20×10	51,000,000	DT for policy DT + RBF for value

Practical Issues and Fixes

Problem 1: What happens to the variance?

$$\Sigma^t = \frac{1}{e} \sum_{i=1}^{e} (\theta_i - \mu^t)^2$$

Collapses too quickly!

Simple fix: Add a bit of noise to the variance

$$\Sigma^t = \frac{1}{e} \sum_{i=1}^{e} (\theta_i - \mu^t)^2 + \Sigma_{noise}$$

Problem 2: What if we have a bad batch of samples?

$$\mu^t = \frac{1}{e}\sum_{i=1}^{e}\theta_i$$

The elites can be bad, and the mean can slingshot into a bad value

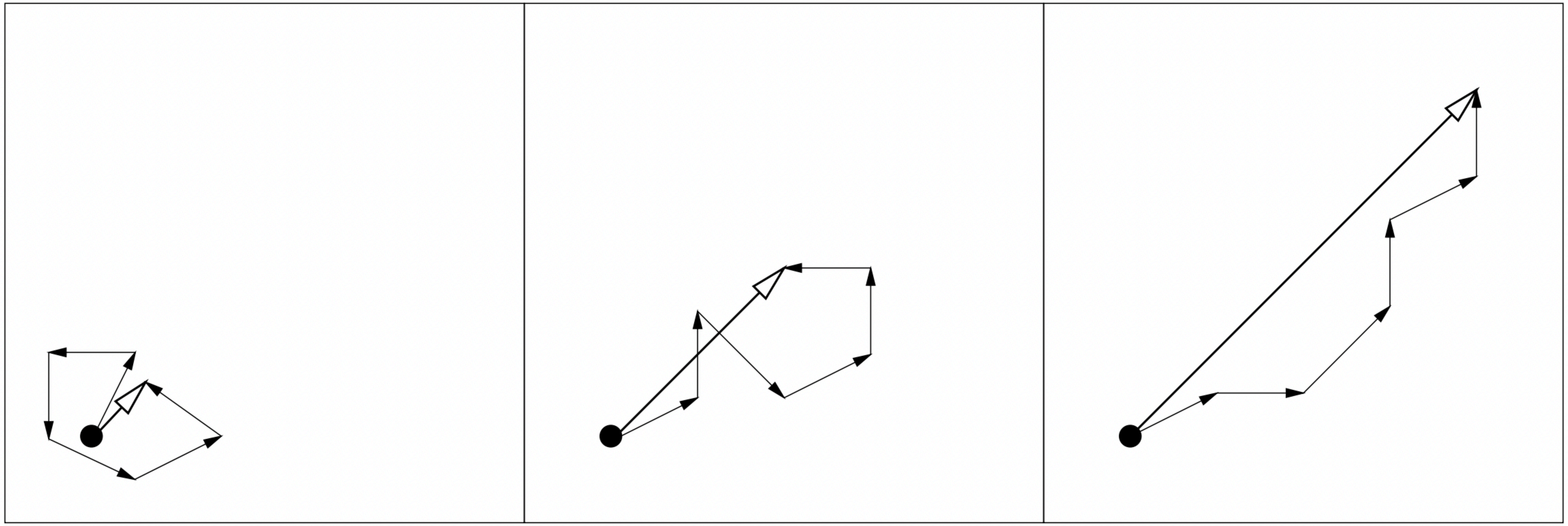
Simple fix: Slowly update mean

$$\mu^t = \mu^{t-1} + \eta\frac{1}{e}\sum_{i=1}^{e}\theta_i$$

Problem 3: What if we never converge and do random walks?

Single-steps cancel out
Use small Σ

Progress correlated
Use large Σ



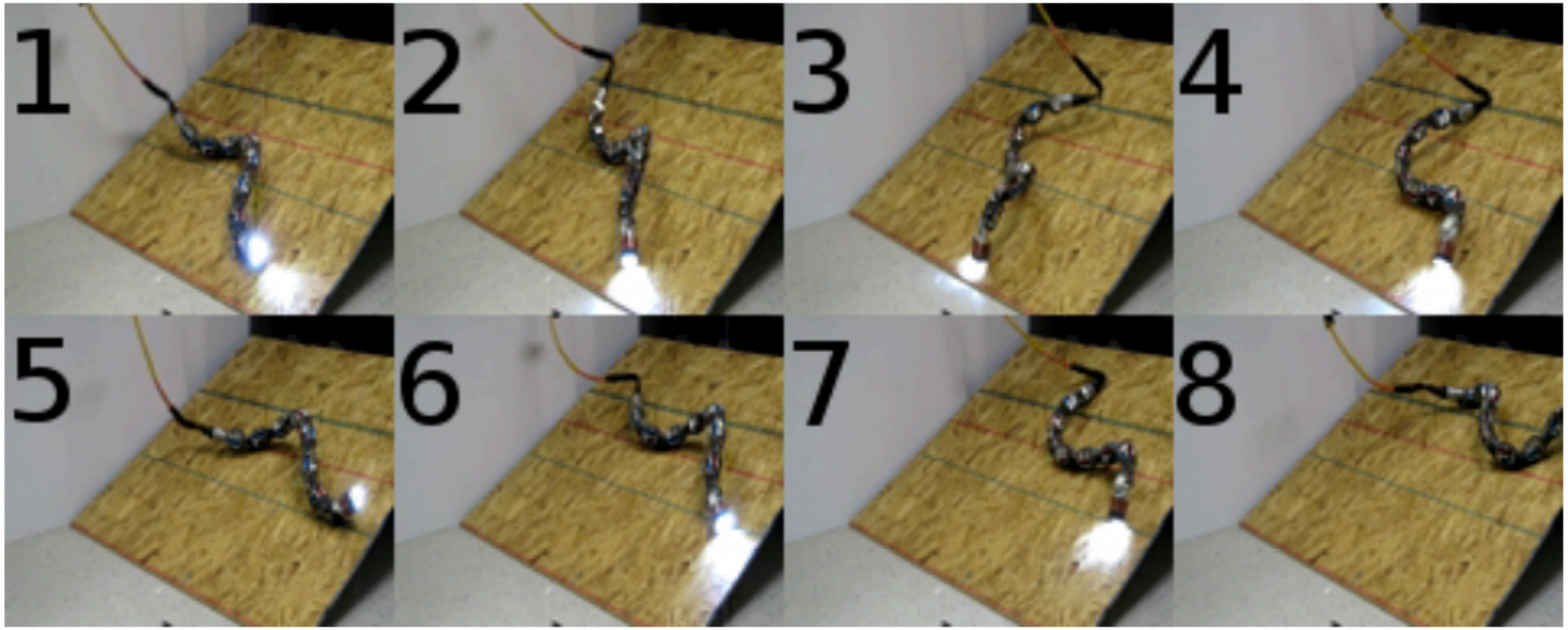
A very fancy version of Cross Entropy: CMA-ES

Tetris is cute...
But what about *real* robots?

Cross Entropy for Snake Robot Gaits

Using Response Surfaces and Expected Improvement to Optimize Snake Robot Gait Parameters

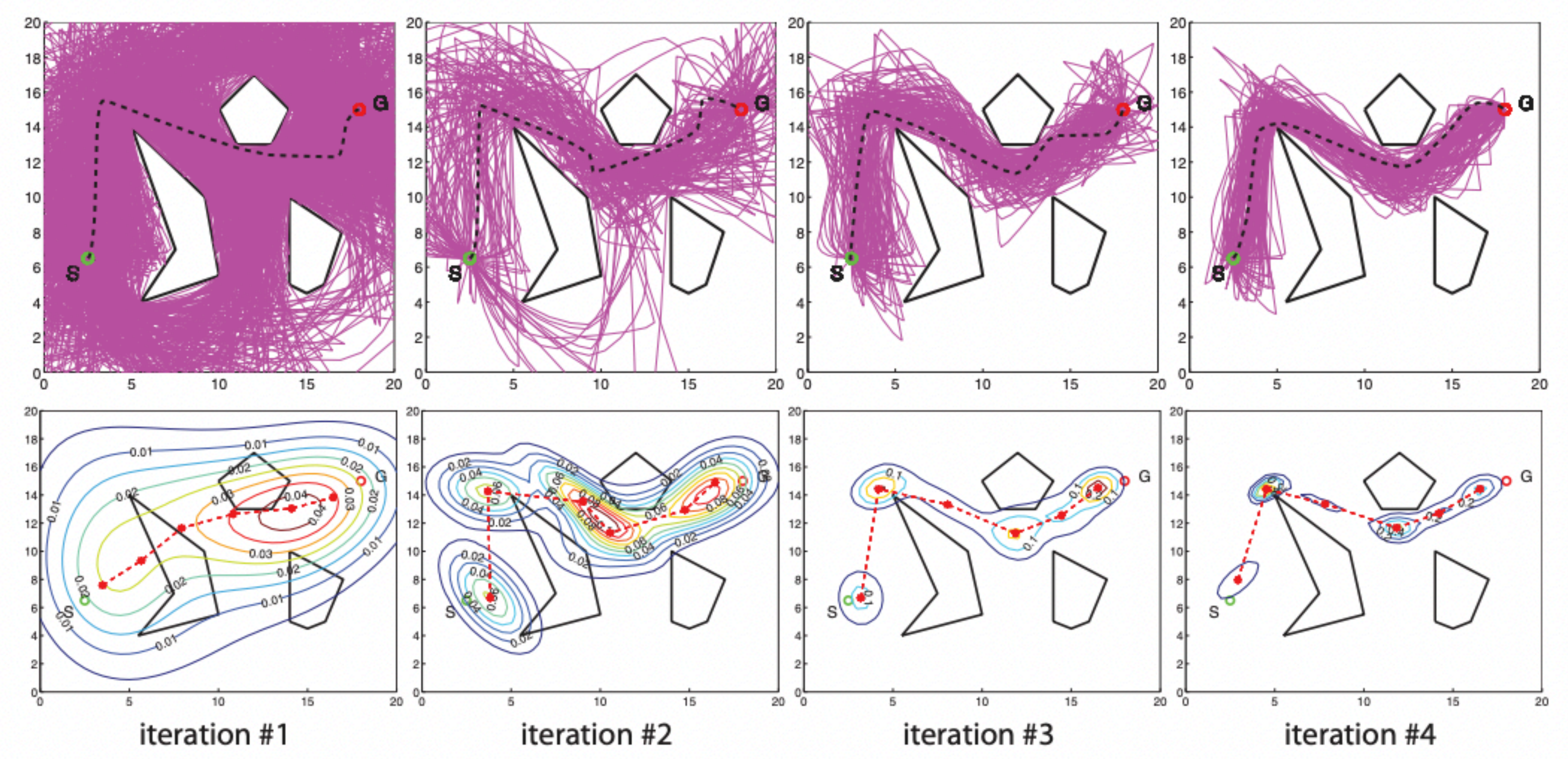
Matthew Tesch, Jeff Schneider, and Howie Choset

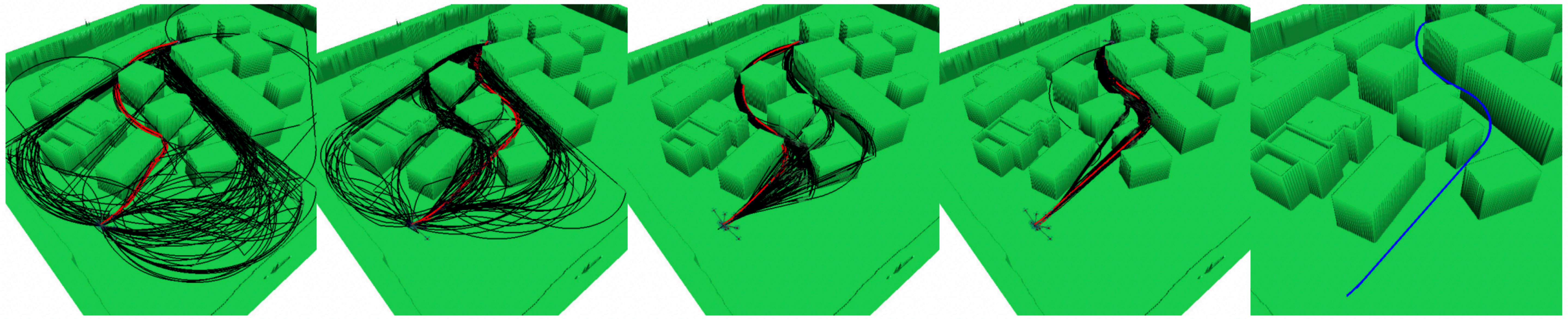


Uses a Gaussian Process to fit a distribution

Prove it can find the optimal gait with *minimal samples*

Cross Entropy Search for Motion Planning





iteration #1 iteration #2 iteration #5 iteration #8 optimal path

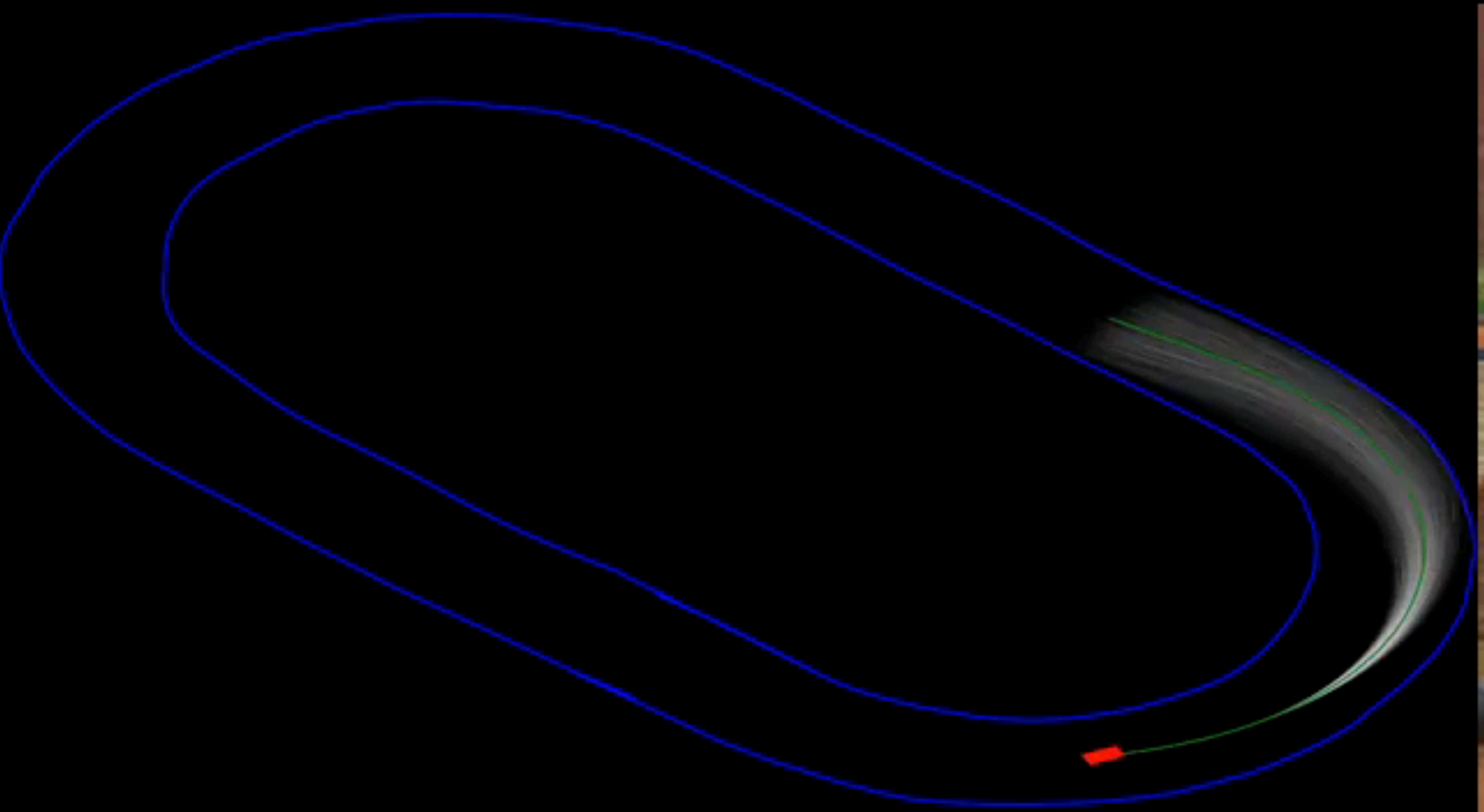
Cross-Entropy Randomized Motion Planning

Marin Kobilarov

Distribution over control trajectories

Cross Entropy for Control

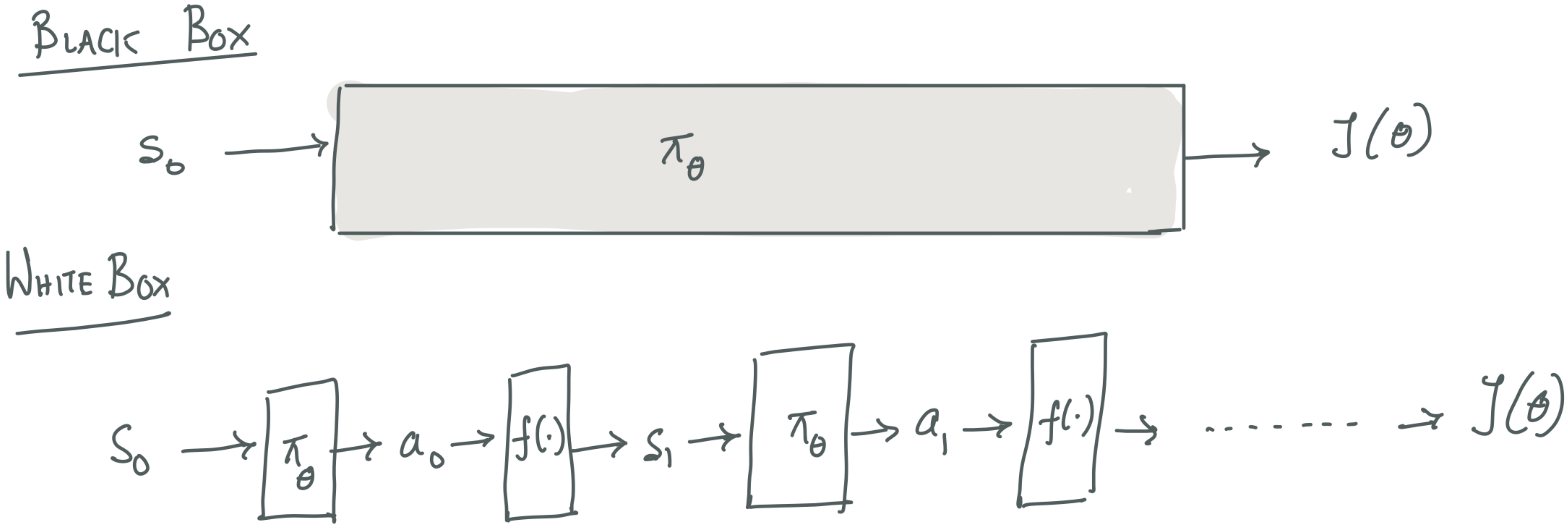
2560, 2.5 second trajectories sampled
with cost-weighted average @ 60 Hz



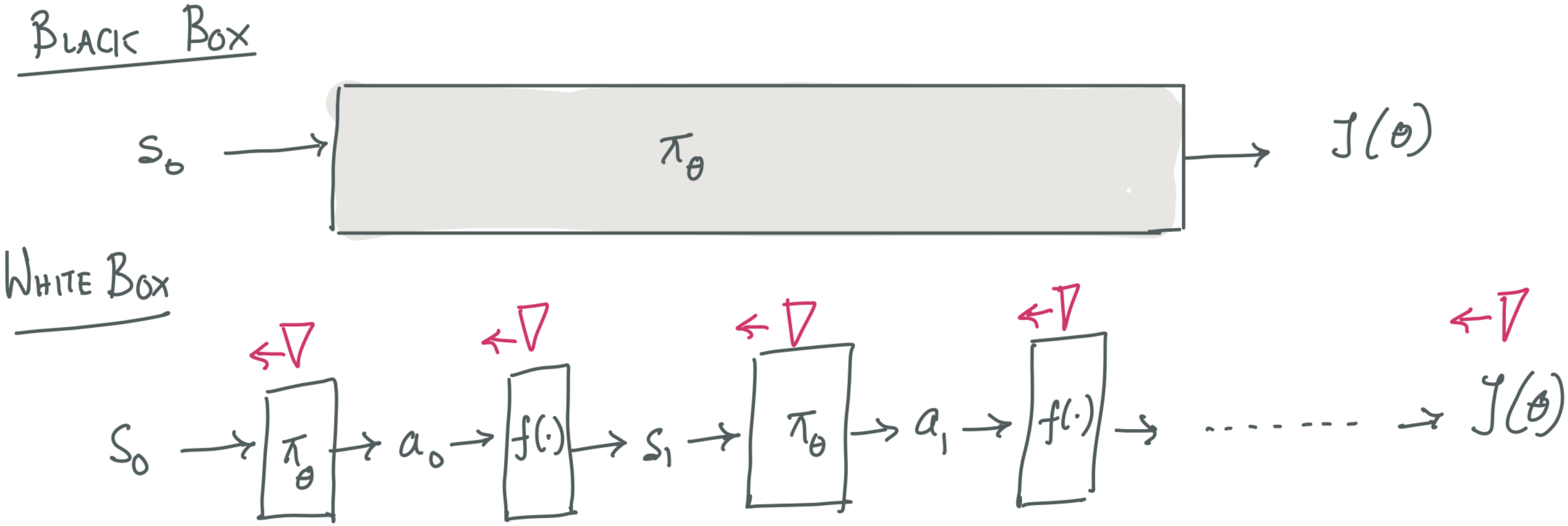
Georgia Tech Auto Rally (Byron Boots lab)


Have we redacted too much?
Form No. I
THIS CASE ORIGINATED A
REPORT MADE AT
TITLE
Mr. [redacted]
SYNOPSIS OF FACTS:
Subject
employment as a worke
chief [redacted] for t
DETAILS:
BACKGROUND
Birth
SUBJECT: [redacted]
1. [redacted] of [redacted] is being s[redacted] [redacted] and biological study at [redacted] in the areas of [redacted] work until [redacted], [redacted]. The pr[redacted]
2. This [redacted] into the [redacted] areas, [redacted] effects of [redacted] continue to be D[redacted] load [redacted] ors will [redacted] and [redacted]
3. The [redacted] The [redacted] estimated to in the [redacted] equired, under [redacted]

Black-box vs White-box vs Gray-box



Black-box vs White-box vs Gray-box



How can we take gradients if we don't know the dynamics?

The Likelihood Ratio Trick!

REINFORCE

Algorithm 20: The REINFORCE algorithm.

Start with an arbitrary initial policy π_θ

while *not converged* **do**

  Run simulator with π_θ to collect $\{\xi^{(i)}\}_{i=1}^{N}$

  Compute estimated gradient

$$\widetilde{\nabla}_\theta J = \frac{1}{N} \sum_{i=1}^{N} \left[\left(\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta \left(a_t^{(i)} | s_t^{(i)} \right) \right) R(\xi^{(i)}) \right]$$

  Update parameters $\theta \leftarrow \theta + \alpha \, \widetilde{\nabla}_\theta J$

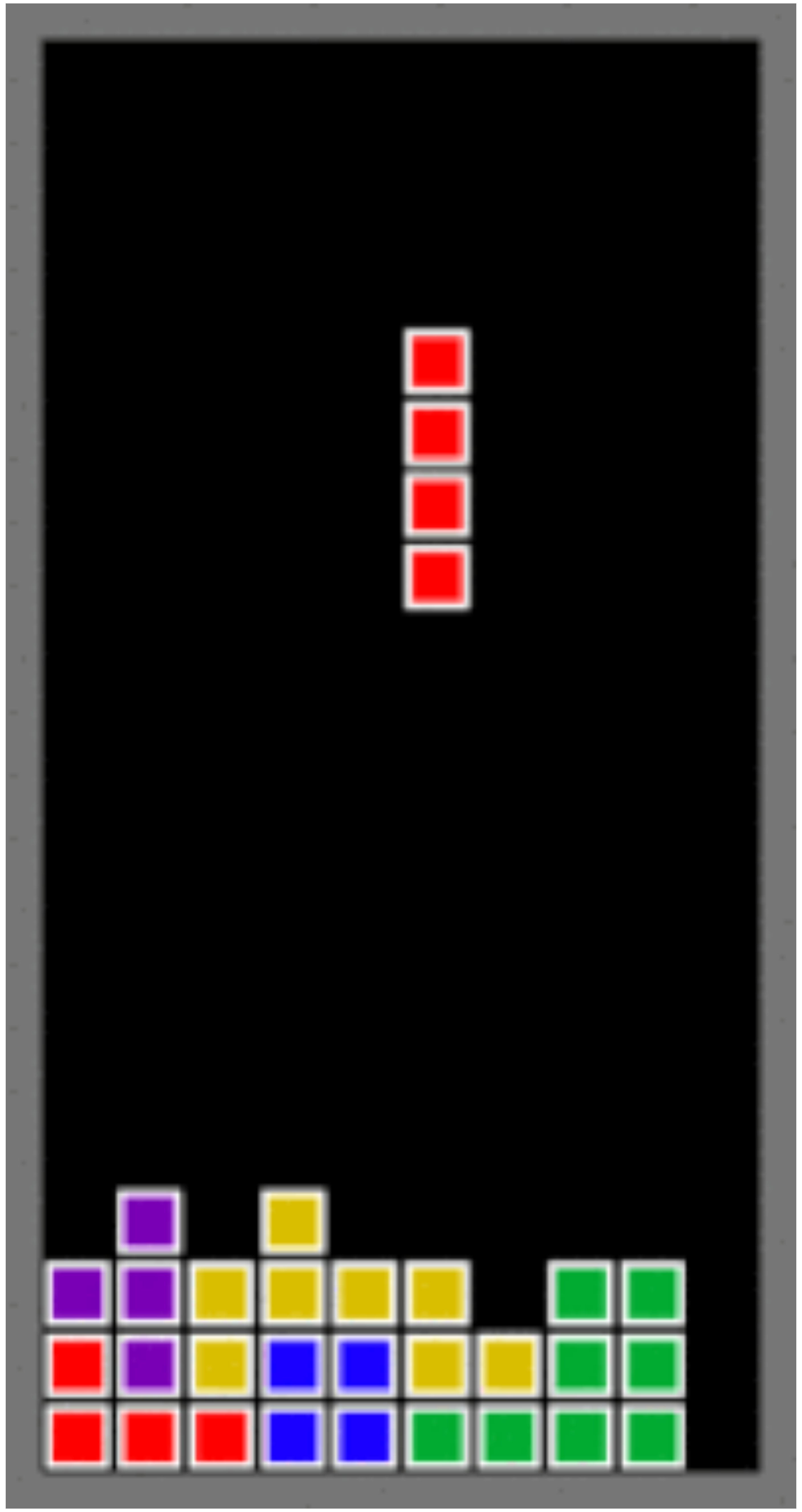
return π_θ

Tetris Policy

$$\pi_\theta(a|s) = \frac{\exp\left(\theta^\top f(s,a)\right)}{\sum_{a'} \exp\left(\theta^\top f(s,a')\right)}$$

$f_1(s,a) = $ # number of holes

$f_2(s,a) = $ # max height

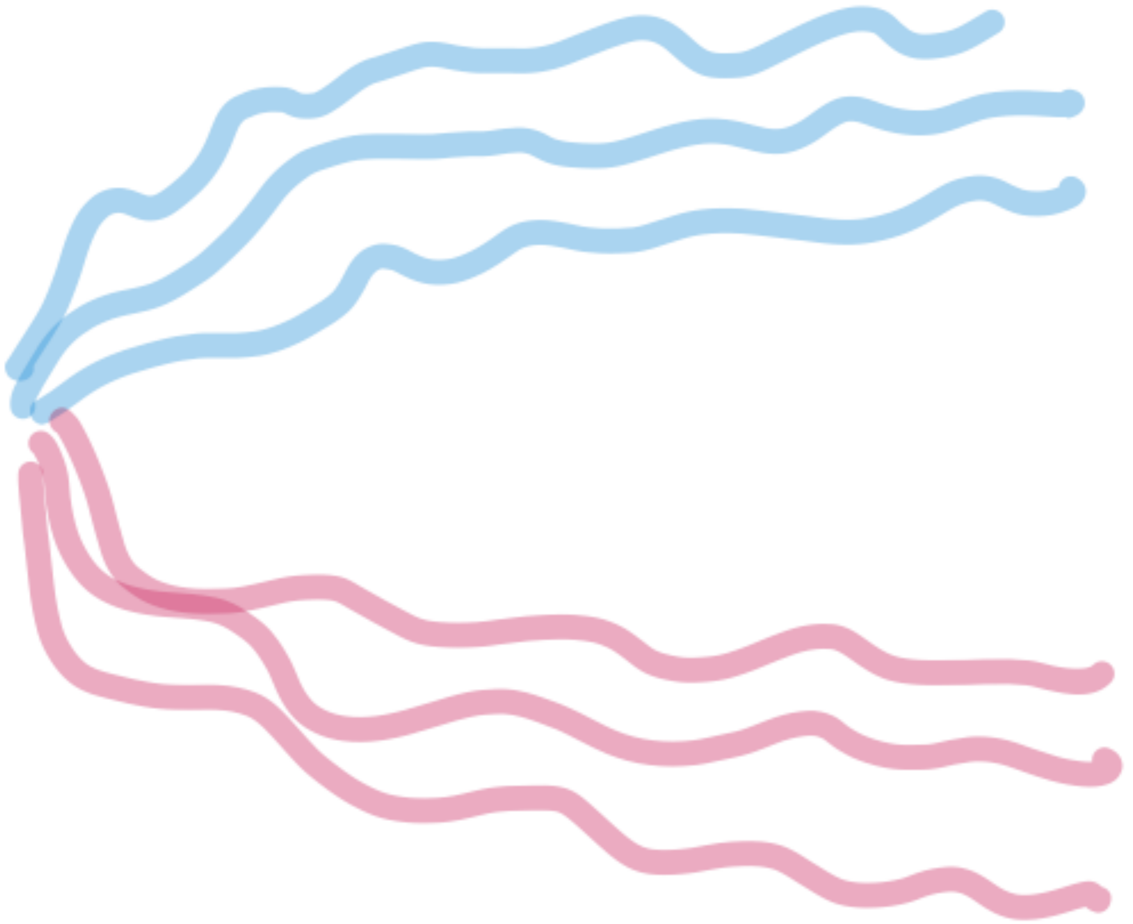


Chugging through the gradient ..

$$
\begin{aligned}
\nabla_\theta \log \pi_\theta(a|s) &= \nabla_\theta \left[\theta^\top f(s,a) - \log \sum_{a'} \exp\left(\theta^\top f(s,a')\right)\right] \\
&= f(s,a) - \frac{\sum_{a'} f(s,a') \exp\left(\theta^\top f(s,a')\right)}{\sum_{a'} \exp\left(\theta^\top f(s,a')\right)} \\
&= f(s,a) - \sum_{a'} f(s,a')\, \pi_\theta\left(a'|s\right) \\
&= f(s,a) - E_{\pi_\theta(a'|s)}\left[f(s,a')\right]
\end{aligned}
$$

Understanding the REINFORCE update

Let $f_1(s,a) = \#\text{holes}.$



R = +1
R = +1
R = +1

$$\Rightarrow \sum_{(s,a)\in\tau} \underbrace{f_1(s,a)}_{\text{Feature}} - \underbrace{\mathop{E}_{a'\sim\pi_\theta} f_1(s,a')}_{\text{Average feature}} = -5$$

R = −1
R = −1
R = −1

$$\Rightarrow \sum_{(s,a)\in\tau} \underbrace{f_1(s,a)}_{\text{Feature}} - \underbrace{\mathop{E}_{a'\sim\pi_\theta} f_1(s,a')}_{\text{Average feature}} = +3$$

$$\theta_1 = \theta_1 + \sum_{(s,a)\in\tau}\left(\nabla_\theta \log \pi_\theta(a|s)\right) R(\tau) = \theta_1 + \alpha\left(-5\times(+1) + 3(-1)\right)$$

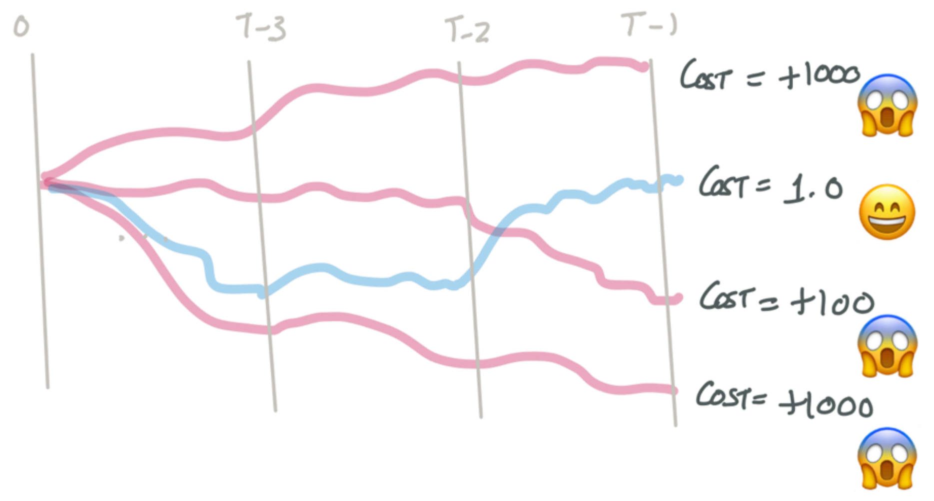
$$= \theta_1 - \alpha\, 8 \quad \text{(Bump down this feature)}$$

tl;dr

The Goal of Policy Optimization

$$\pi_\theta(s) = \arg\min_a \theta^T f(s,a)$$

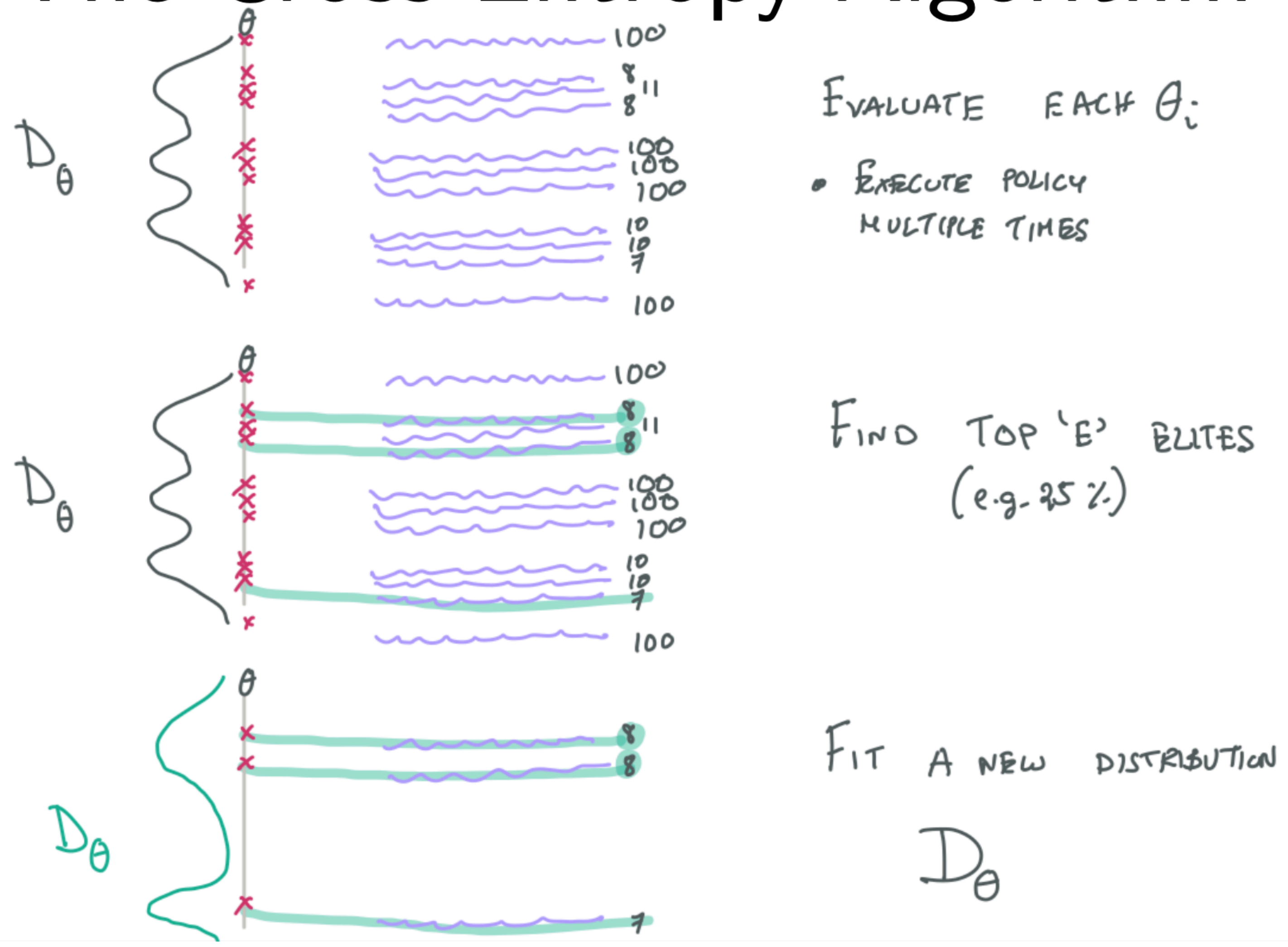
$$\min_\theta J(\theta) = \sum_{t=0}^{T-1} \mathbb{E}_{\pi_\theta} c(s_t, a_t)$$



Black-box

Gray-box

The Cross Entropy Algorithm



30

REINFORCE

Algorithm 20: The REINFORCE algorithm.

Start with an arbitrary initial policy π_θ
while *not converged* **do**
  Run simulator with π_θ to collect $\{\xi^{(i)}\}_{i=1}^N$
  Compute estimated gradient

$$\tilde{\nabla}_\theta J = \frac{1}{N}\sum_{i=1}^{N}\left[\left(\sum_{t=0}^{T-1}\nabla_\theta \log \pi_\theta\left(a_t^{(i)}|s_t^{(i)}\right)\right)R(\xi^{(i)})\right]$$

  Update parameters $\theta \leftarrow \theta + \alpha \tilde{\nabla}_\theta J$
return π_θ