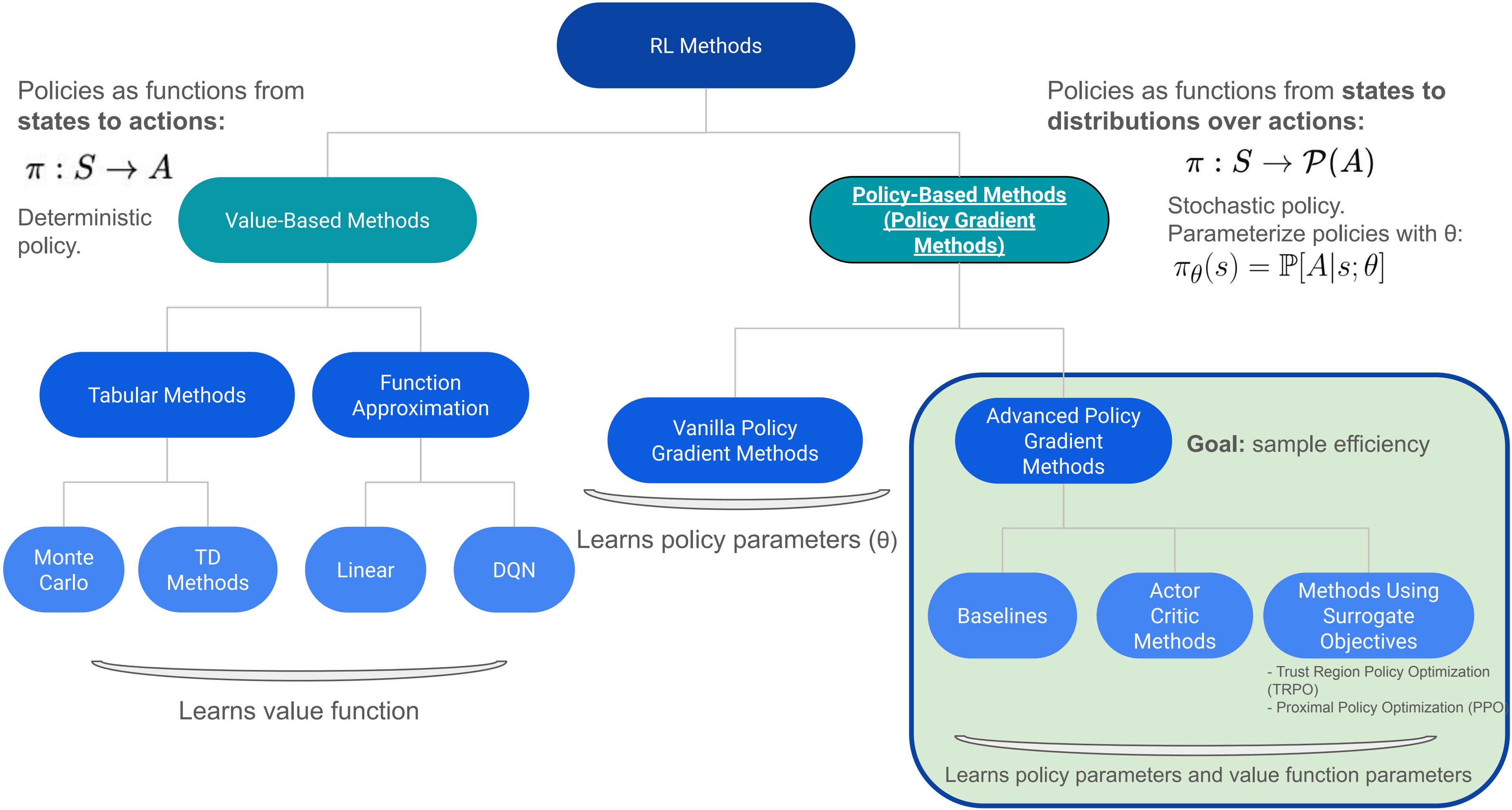
Effective Reinforcement Learning through Evolutionary Surrogate-Assisted Prescription

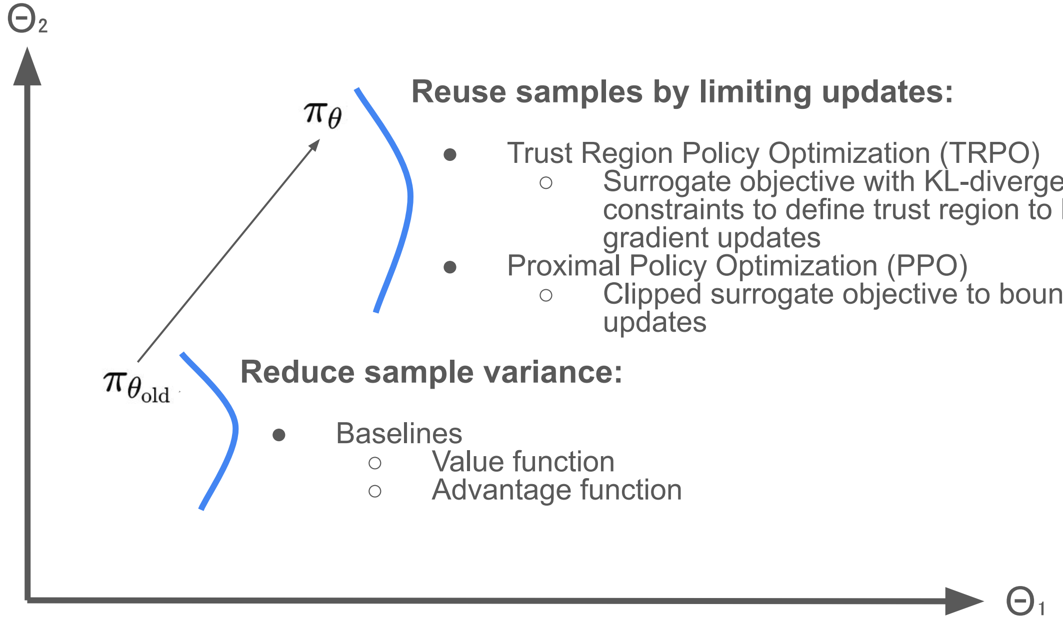
Olivier Francon, Santiago Gonzalez, Babak Hodjat, Elliot Meyerson, Risto Miikkulainen, Xin Qiu and Hormoz Shahrzad (2020)



RL Methods
Policies as functions from **states to actions:**
$\pi : S \to A$
Deterministic policy.
Value-Based Methods
Tabular Methods
Function Approximation
Monte Carlo
TD Methods
Linear
DQN
Learns value function
Policies as functions from **states to distributions over actions:**
$\pi : S \to \mathcal{P}(A)$
Stochastic policy.
Parameterize policies with θ:
$\pi_\theta(s) = \mathbb{P}[A|s;\theta]$
Policy-Based Methods (Policy Gradient Methods)
Vanilla Policy Gradient Methods
Learns policy parameters (θ)
Advanced Policy Gradient Methods
Goal: sample efficiency
Baselines
Actor Critic Methods
Methods Using Surrogate Objectives
- Trust Region Policy Optimization (TRPO)
- Proximal Policy Optimization (PPO)
Learns policy parameters and value function parameters

Recap: Advanced Policy Gradient Methods

- Methods for **direct policy optimization**
- **Goal:** sample efficiency (getting more accurate gradient estimates without collecting more samples)
  a. **For given policy:** reduce variance across samples (episodes)
  b. **Across different policies:** limit policy updates to permit reuse of samples from previous policy; explore similar policies



Reuse samples by limiting updates:

- Trust Region Policy Optimization (TRPO)
  - Surrogate objective with KL-divergence constraints to define trust region to bound gradient updates
- Proximal Policy Optimization (PPO)
  - Clipped surrogate objective to bound gradient updates

Reduce sample variance:

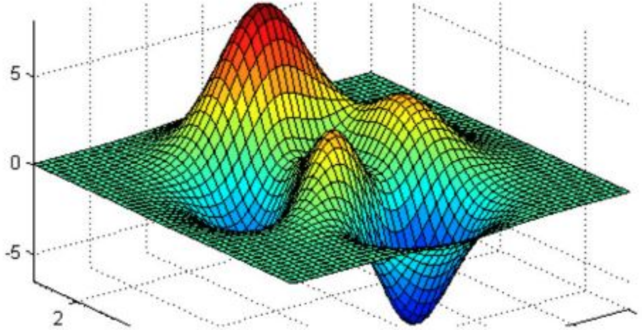
- Baselines
  - Value function
  - Advantage function

Recap: Policy Optimization

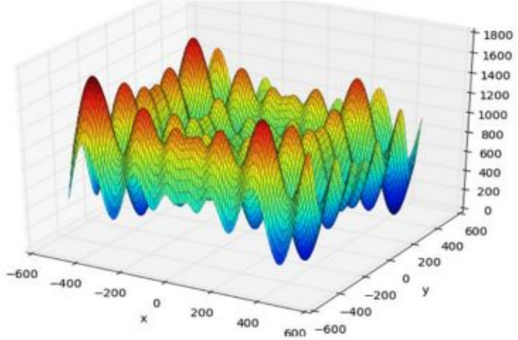
- **Intuition behind TRPO/PPO:** explore similar policies through safe, gradual gradient-based updates
- Challenges:
  - Not suitable for deceptive, high-dimensional search spaces
  - Policies need to be evaluated through interactions with the environment
- **Alternative:** Evolutionary Surrogate-Assisted Prescription (ESP)
  - Policy optimization with **population-based search (evolution)** + **surrogate optimization**
  - **Key ideas:**
    - Big updates instead of small updates to escape local minima
    - Reward different policies to encourage exploration
    - Use surrogate to evaluate policies without direct interaction with environment

Population-Based Search

- **Traditional AI:** modeling using **hill-climbing based approaches**
  - Methods based on adjusting networks based on gradients computed based on examples of desired behavior such as deep learning (DL) and traditional reinforcement learning (RL)
  - Search around single point (potential solution)
- **Creative AI (Machine Creativity):** discover correct/good solutions
  - Search space is large, high-dimensional and deceptive - NOT amenable to **hill-climbing**
  - Incremental improvement does not work
- **Solution: population-based search** methods
  - Fundamental difference between gradient-based methods is **extensive exploration**
  - Search a set (population) of solutions



(a) Search Space Appropriate for Hill Climbing



(b) Search Space in a Creative Domain

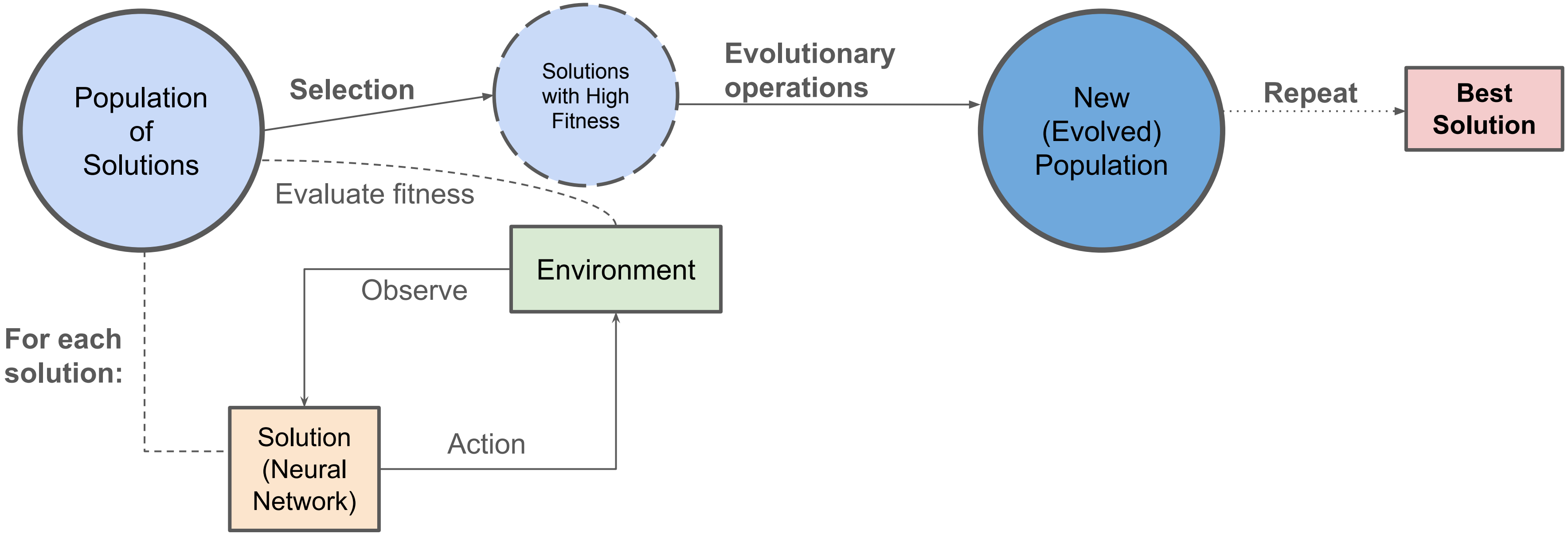
Source: Creative AI Through Evolutionary Computation: Principles and Examples

Population-Based Search

- **Exploration methods:**
  - **Parallel searches, with partial solutions shared across searches**
    - Partial solutions act as building blocks that can be combined to find better solutions
  - **Multiple objectives to search across multiple dimensions**
    - **Recombination** of solutions across the population to optimize multiple variables at once
  - **Novelty search**
    - Reward solutions that are different from current solution (less rigid; more exploration)
- Population-based search methods include **evolutionary computation**

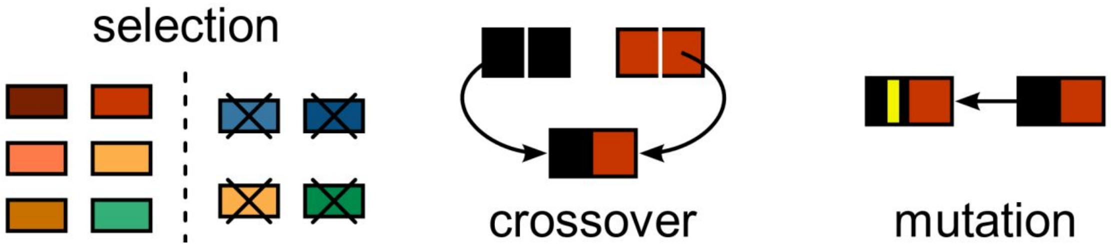
Evolutionary Computation

- Each solution (policy) represented as neural network (**neuroevolution**)
    - **Objective:** find solution with highest fitness value iteratively by applying **evolutionary operations**
    - **Fitness function** measures quality of solutions



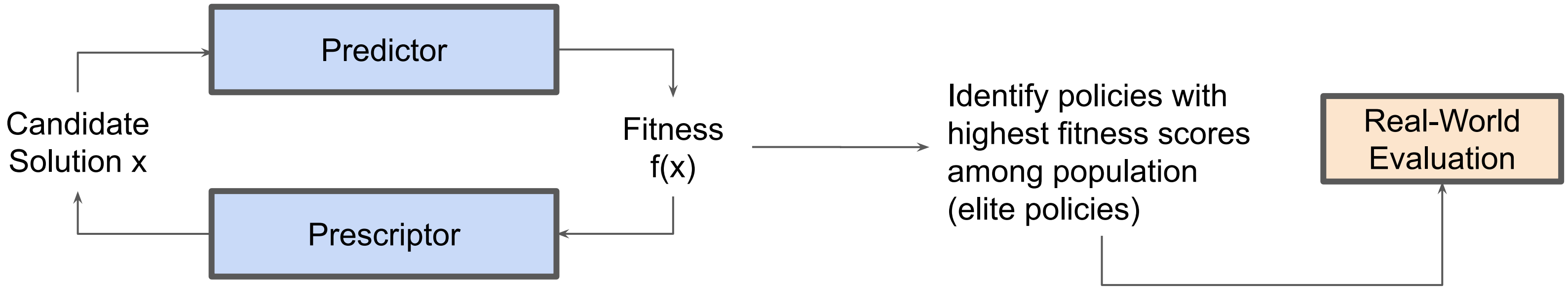
Evolutionary Computation

- **Evolution process:**
    - Explore multiple areas of search space at once
    - Use **partial solutions** across different searches
        - Different searches interact unlike PPO
    - Generate next generation of solutions by combining parent solutions using **evolutionary operations:**
        - Explore and exploit by introducing variation to known solutions
        - **Tournament selection:** select best-performing policies to serve as parents
        - **Crossover:** combine building blocks from two parents
        - **Mutation:** create new building blocks



Evolutionary Surrogate-Assisted Prescription (ESP)

- Surrogate-assisted, population-based search
  - RL method for policy search following the actor-critic framework
  - Population-based search instead of gradient-based incremental improvement
- Train **Predictor (critic)** and **Prescriptor (actor)** at the same time
  - **Predictor:** approximates fitness function (any supervised neural network)
  - **Prescriptor:** decides what actions to perform at each time step given context/state (neural network or rule-set representation)
    - Prescriptor uses Predictor as surrogate model to evaluate fitness during the evolution process
    - **Elite policies** with high fitness scores are evaluated in the real world
  - Extendable to multi-objective settings: multiple prescriptors



Evolutionary Surrogate-Assisted Prescription (ESP)

1) Train critic **Predictor**

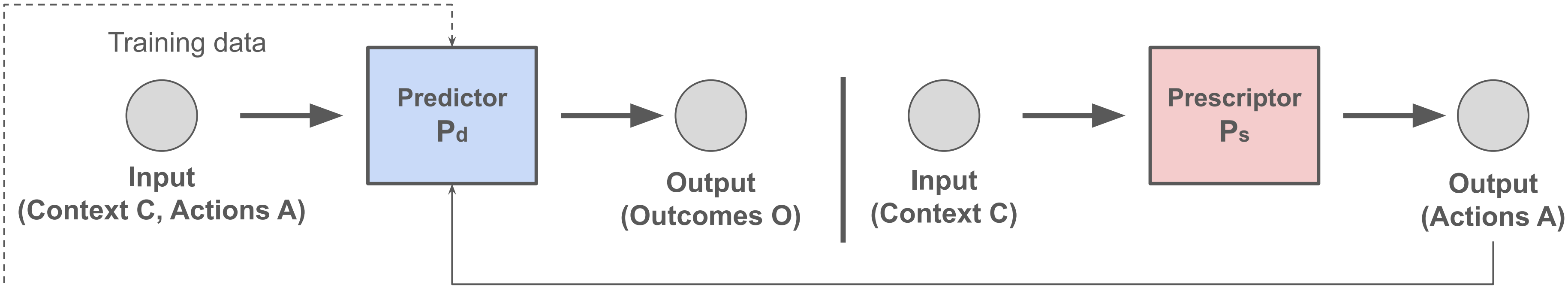
- **Training objective:** minimize loss

$$P_d(C, A) = O' \quad \text{such that} \quad \min\left(\sum_j L(O_j, O'_j)\right) \quad \text{across all dimensions } j \text{ of } O$$

2) Train actor **Prescriptor** simultaneously

- **Training objective:** maximize outcomes

$$P_s(C) = A \quad \text{such that} \quad \max\left(\sum_{i,j} O'_j(C_i, A_i)\right) \quad \text{over all possible contexts } i$$



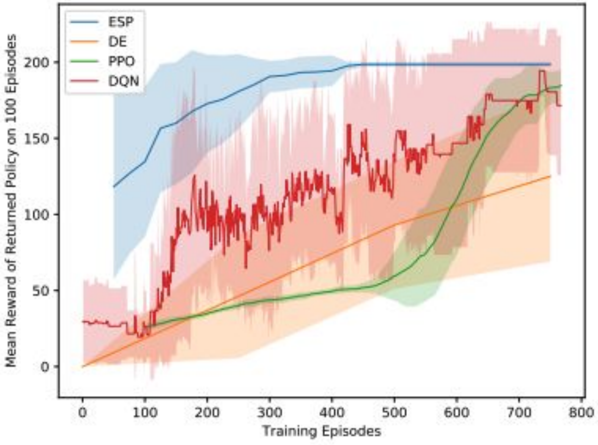
3) Use Predictor as surrogate to evaluate and evolve Prescriptor until convergence

3) Apply best Prescriptors (elite policies) in real world and observe outcomes

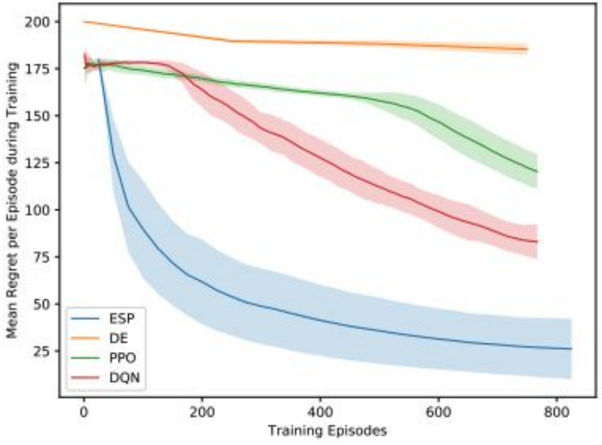
4) Use obtained (Context, Actions, Outcomes) tuples as training data for Predictor

Performance Comparison on RL Benchmarks

- **Task: Cart-Pole –** standard RL benchmark; single pole moves left/right and a reward is given for each time step the pole stays near vertical and the cart stays near the center of the track
- **Baselines:** Proximal Policy Optimization (PPO), Direct Evolution (DE), double Deep Q Networks (DQN)
    - Direct Evolution (DE): evolution without surrogate - evolution process is ran directly against real function instead of Predictor



(a) True Performance

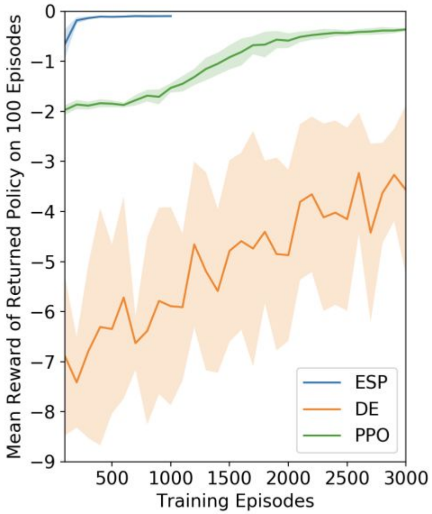


(b) Regret

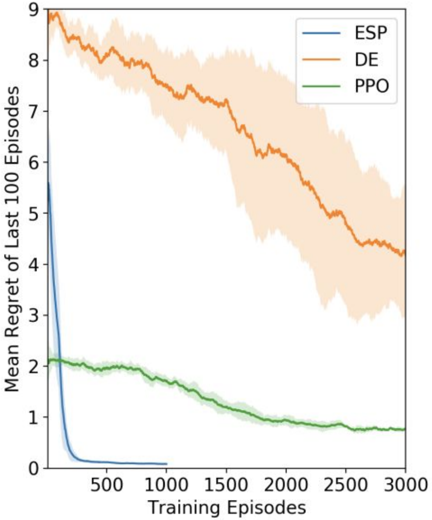
- ESP converges significantly faster
- Lower variance for ESP even after early stages
- ESP shows lower regret (reward difference between optimal and current policies)
    - More reliable

Performance Comparison on Function Approximation

- **Task:** function approximation (continuous)
- **Baselines:** Proximal Policy Optimization (PPO), Direct Evolution (DE)



(a) True Performance



(b) Regret

- Neither PPO or DE converged near after 1000 episodes
- ESP converged almost exactly to the optimal within 125 episodes
- ESP shows lower regret

Conclusion

- **Evolutionary-Surrogate Assisted Prescription (ESP):** RL as surrogate-assisted population-based search
- **Goal:** sample efficiency, like TRPO/PPO
  - **Different methods:**
    - Population-based search – big updates, not small updates
    - Novelty search – high KL divergence rewarded instead of low KL divergence
- Ideal for domains where real-world evaluation is costly
- Faster convergence and lower variance and regret on benchmarks compared to PPO