Data Efficient Reinforcement Learning
Reinforcement Learning
Reinforcement Learning

**PROS**
- Control
- No tagging data
- Sparse rewards

**CONS**
- Sample Inefficient
- Big models
- Local Optima

![Diagram of Reinforcement Learning](image)
Real World RL

• Sample efficiency:
  – Build a model of the real world

• Big Networks:
  – Try to make policy a smaller separate network

• Local Optima:
  – Train using more robust technique
Data Efficient RL

Uncertain Dynamic Model

Policy

Environment

State $s_t$, Reward $r_t$, Action $a_t$, State $s_{t+1}$, Reward $r_{t+1}$
BlackDROPS Components

- UDM → Gaussian Processes
- Policy → Simple Neural Net
- Training → CMA-ES
Gaussian Process Review
Gaussian Process Review

**PROS**
- Easy to build model
- Understands uncertainty
- Predictive
- Custom Kernels

**CONS**
- Need alternate past 10K points
- Takes time to predict values
- Doesn’t expand to high dimensional space well
How BlackDROPS uses GPs

- One GP per dimension in $S$
- $s_t \times a_t \rightarrow \Delta s_t$, why?
- Reward GP isn’t needed, but enables new possibilities
- Not sure the $S \rightarrow R$ is optimal

Uncertain Dynamic Model

$s_t \times a_t \rightarrow \Delta s_t = s_{t+1} - s_t$

$s_t \rightarrow r(s_t)$

- Always be recording
CMA-ES Review

- Optimizer
- Blackbox
- Robust
  - Noise
  - Discontinuities
  - Sharp edges

CMA-ES Review

- Evolutionary
- Blue - Offspring
- Red - Best
- Green - Elite

Images reused with permission from otoro.net 大トロ
BlackDROPS

- UDM → GP
- Policy → NN
- Training → CMS-ES
BlackDROPS Algorithm (1)

Algorithm 1 Black-DROPS

1: procedure BLACK-DROPS
2: Define policy $\pi : x \times \theta \rightarrow u$
3: $D = \emptyset$
4: for $i = 1 \rightarrow N_R$ do \hspace{1cm} $\triangleright N_R$ random episodes
5: Set robot to initial state $x_0$
6: for $j = 0 \rightarrow T - 1$ do \hspace{1cm} $\triangleright$ perform the episode
7: $u_j = \text{random\_action()}$
8: $x_{j+1}, r(x_{j+1}) = \text{execute\_on\_robot}(u_j)$
9: $D = D \cup \{\tilde{x}_j \rightarrow \Delta x_j\}$
10: $R = R \cup \{x_{j+1} \rightarrow r(x_{j+1})\}$
11: end for
12: end for
BlackDROPS Algorithm (2)

13: \[ \textbf{while} \ task \neq \textbf{solved} \ \textbf{do} \]
14: \hspace{1em} \text{Model learning: train } E \text{ GPs given data } D \]
15: \hspace{1em} \text{Reward learning: train 1 GP given data } R \]
16: \hspace{1em} \theta^* = \arg\max_{\theta} \mathbb{E}[G(\theta)] \text{ using BIPOP-CMA-ES} \quad \triangleright \text{Sec. IV-D} \]
17: \hspace{1em} \text{Set robot to initial state } x_0 \]
18: \hspace{2em} \textbf{for } j = 0 \rightarrow T - 1 \ \textbf{do} \quad \triangleright \text{perform the episode} \]
19: \hspace{3em} u_j = \pi(x_j | \theta^*) \]
20: \hspace{3em} x_{j+1}, r(x_{j+1}) = \text{execute\_on\_robot}(u_j) \]
21: \hspace{3em} D = D \cup \{\tilde{x}_j \rightarrow \Delta x_j\} \]
22: \hspace{3em} R = R \cup \{x_{j+1} \rightarrow r(x_{j+1})\} \]
23: \hspace{2em} \textbf{end for} \]
24: \hspace{1em} \textbf{end while} \]
25: \textbf{end procedure}
Training with CMA-ES (1)

- Select random $W_1 = \{\omega_{1,1}, \omega_{1,2}, \ldots, \omega_{1,p}\}$ for $\pi_1$
- Start at $s_0$
- Roll out policy
  
  $a_0 = \pi_1(s_0), s_1 = s_0 + GP_s(s_0, a_0): a_1 = \pi_1(s_1), s_2 = s_1 + GP_s(s_1, a_1): \ldots$

- Calculate rewards $r_{1,0} = GP_r(s_0): r_{1,1} = GP_r(s_1), \ldots$
Training with CMA-ES (2)

- Select more $W_i = \{\omega_{i,1}, \omega_{i,2}, \ldots, \omega_{i,p}\}$ for $\pi_i$
- Start at $s_0$
- Roll out policy
- Calculate rewards
- Do this N times
Training with CMA-ES (3)

- Select the “top” $x\%$ based on $R_i = \sum_{j=0}^{p} r_{i,j}$
- Calculate the $\mu$ and $\text{cov}$ for top entries $W_1, W_2, \ldots, W_{xN}$
- Select another $N (1 - x)$ samples
- Repeat for $M$ times
Training with CMA-ES (4)

- Hopefully found a good reward
- Now:
  - Try the policy out in the real world
  - Update the GPs
  - Retune with CMA-ES again until done
## Examples (1)

<table>
<thead>
<tr>
<th>PILCO</th>
<th><a href="https://youtu.be/XiigTGKZfks">https://youtu.be/XiigTGKZfks</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>BlackDROPS</td>
<td><a href="https://youtu.be/kTEyYiIFGPM">https://youtu.be/kTEyYiIFGPM</a></td>
</tr>
<tr>
<td>BlackDROPS</td>
<td><a href="https://youtu.be/HFkZkhGGzTo">https://youtu.be/HFkZkhGGzTo</a></td>
</tr>
</tbody>
</table>
Replication
BlackDROPS

- Exploitation vs Exploration
- Achieving better results than PILCO
- NN better than GP
- Fast learning

Fig. 4. Results for the cart-pole task (120 replicates): (A) Best reward found per episode. Black-DROPS converges to higher quality solutions in about the same number of episodes as PILCO and has less variance. (B) Best reward after 8 episodes. Our approach outperforms PILCO in the quality of the controllers found. See Fig. 2 for legend.
Training with CMA-ES (5)

• How reliable is selecting the “top” $x\%$
• Noise of sample > difference in ranking
• Expand the covariance matrix to increase signal
Observations about BlackDROPS

- Sensitive to initial conditions
- Compute power vs learning speed not optimize for real world
- Time not fully implemented
- Heavy memory requirements needed for compilation

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Success Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Pendulum</td>
</tr>
<tr>
<td>Black-DROPS (NN)</td>
<td>100%</td>
</tr>
<tr>
<td>Black-DROPS (GP)</td>
<td>100%</td>
</tr>
<tr>
<td>No Var (NN)</td>
<td>100%</td>
</tr>
<tr>
<td>No Var (GP)</td>
<td>100%</td>
</tr>
<tr>
<td>PILCO</td>
<td>89.16%</td>
</tr>
</tbody>
</table>
IoT and RL

- Move Policy to edge
- Keep UDM / Training in cloud
Enabling New Applications

• Tune difficult control system
  – Dead band, hysteresis, linearization, ...

• Allows system to compensate for errors
  – E.g. Use 3D camera to move end effector to target

• Allows optimization for hard to model features
  – E.g. Use accelerometer on end effector
Enabling New Applications

- Self healing systems
  - Adapt to aging unit, (cloud vs edge device)
- Customization on unit by unit basis
  - E.g. Understand stack performance in BMS
- Can optimize based on human feedback
  - E.g. HVAC with cellphone input
Reward Function in GP

• Do we want to map from $S \rightarrow R$
  - Do we want historical info?
  - Do we want current trajectories?
Ready to Deploy

- Practical way of deploying small policies to replace traditional control systems
- Ability to learn faster than programmer coding time to control complex systems
- Concept of splitting model into policy and UDM is effective for IoT type devices
Questions

Oliver King-Smith PhD

https://smartr.ai
oliverks@smartr.ai
831-251-8990