Trust Region Policy Optimization Demo

Rishikesh Vaishnav

July 11, 2018

Code

- The code for this project is available at: .

Implementation Details

Single Path Executable Pseudocode:

```
- Initialize policy parameter $\theta$.
- Iterate until convergence:
    - Initialize/clear list $S$ of $\{\frac{G_\theta(s,a)}{\pi_\theta(s,a)}, s, a\}$.
    - Generate $N_\tau$ trajectories $\{\tau\}$.
    - For each trajectory $\tau \in \{\tau\}$:
        - For each $\{s, a\} \in \tau$:
            - Calculate discounted return $G_\theta(s, a)$ from this time to end of episode.
            - Calculate $\pi_\theta(s, a)$ at this $(s, a)$.
            - Store $\{\frac{G_\theta(s,a)}{\pi_\theta(s,a)}, s, a\}$ in $S$.
    - Use constraint optimizer to yield $\theta'$ by solving the problem:
        - Objective (to maximize): objective($S$, $\theta$).
        - Constraint: constraint($S$, $\theta$).
    - $\theta = \theta'$.
```

objective(S, θ') Pseudocode:

```
- Initialize $L = 0$.
- For $\{\frac{G_\theta(s,a)}{\pi_\theta(s,a)}, s, a\} \in S$:
    - Add $\pi_{\theta'}(s, a)\frac{G_\theta(s,a)}{\pi_\theta(s,a)}$ to $L$.
- Return $L$.
```

constraint(S, θ') Pseudocode:

```
- Initialize $D = 0$.
- For $s \in S$:
```

  - Add $D_{KL}(\pi_\theta(\cdot|s)||\pi_{\theta'}(\cdot|s))$ to D.

- Return $\frac{D}{|S|}$.

Parameter Settings:

- $N_\tau = 5$ (adjusted to balance between empirical runtime and performance)
- $\gamma = 1$ (adjusted to maximize empirical performance)
- $\delta = 0.01$ (following Schulman et. al.)

Policy Function Encoding:

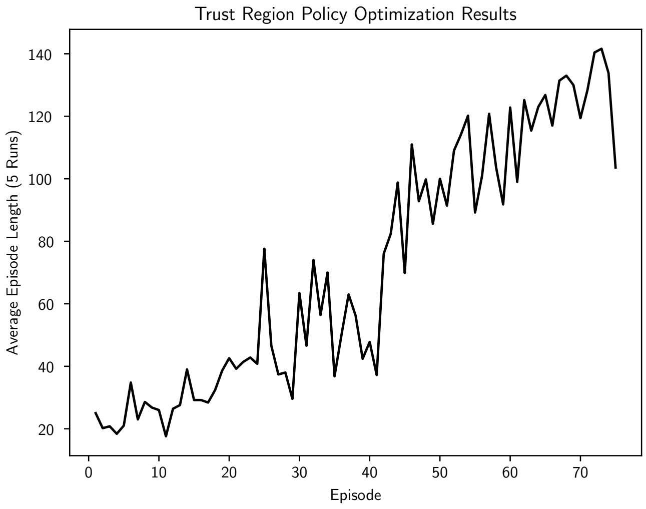
- Each state-action pair is converted to the feature vector $x(s, a)$. Letting S_{obs} and S_{act} be the size of the observation and action spaces, respectively, the size of the vector is $S_{obs} \times S_{act}$, where all features are 0 except for the S_{obs} features starting at index $S_{obs} \times a$, which are set to the environment's parameterization of s.
  - In this case, $S_{obs} = 4$ and $S_{act} = 2$.
- The policy function $\pi(a|s, \theta)$ performs the softmax on a parameterized linear mapping of feature vectors:

$$\pi(a|s,\theta) = \frac{e^{\theta^T x(s,a)}}{\sum_b e^{\theta^T x(s,b)}}$$

Constraint Optimization Method:

- I used scipy's optimize.minimize function, with the "trust-constr" method. All gradients and hessians were automatically calculated. A tolerance of 1×10^{-2} and a maximum iteration limit of 500 were used.

Results



Note: It seems possible that these results would improve given more iterations. However, due to the large amount of time it takes for my computer to run the code, I decided not to run any more.

- These results are promising, and similar to the results I got for policy iteration. However, unlike the optimization in policy gradient that exactly followed the policy gradient algorithm, the optimization here was potentially imperfect due to the approximated constraint optimization. Limiting factors include the tolerance and the maximum iteration limit. If optimization were more exact, it seems probable that this algorithm would outperform policy gradient in terms of sample efficiency.