5 VALUE FUNCTION ESTIMATION

A variety of different methods can be used to estimate the value function (see, e.g., Bertsekas (2012)). When using a nonlinear function approximator to represent the value function, the simplest approach is to solve a nonlinear regression problem:

$$\underset{\phi}{\text{minimize}} \sum_{n=1}^{N} \|V_\phi(s_n) - \hat{V}_n\|^2, \tag{28}$$

where $\hat{V}_t = \sum_{l=0}^{\infty} \gamma^l r_{t+l}$ is the discounted sum of rewards, and n indexes over all timesteps in the batch of trajectories. This is sometimes called the Monte Carlo or TD(1) approach for estimating the value function (Sutton & Barto, 1998).[2]

For the experiments in this work, we used a trust region method to optimize the value function in each iteration of a batch optimization procedure. The trust region helps us to avoid overfitting to the most recent batch of data. To formulate the trust region problem, we first compute $\sigma^2 = \frac{1}{N}\sum_{n=1}^{N} \|V_{\phi_{\text{old}}}(s_n) - \hat{V}_n\|^2$, where ϕ_{old} is the parameter vector before optimization. Then we solve the following constrained optimization problem:

$$\begin{aligned} &\underset{\phi}{\text{minimize}} && \sum_{n=1}^{N} \|V_\phi(s_n) - \hat{V}_n\|^2 \\ &\text{subject to} && \frac{1}{N}\sum_{n=1}^{N} \frac{\|V_\phi(s_n) - V_{\phi_{\text{old}}}(s_n)\|^2}{2\sigma^2} \le \epsilon. \end{aligned} \tag{29}$$

This constraint is equivalent to constraining the average KL divergence between the previous value function and the new value function to be smaller than ϵ, where the value function is taken to parameterize a conditional Gaussian distribution with mean $V_\phi(s)$ and variance σ^2.

We compute an approximate solution to the trust region problem using the conjugate gradient algorithm (Wright & Nocedal, 1999). Specifically, we are solving the quadratic program

$$\begin{aligned} &\underset{\phi}{\text{minimize}} && g^T(\phi - \phi_{\text{old}}) \\ &\text{subject to} && \frac{1}{N}\sum_{n=1}^{N} (\phi - \phi_{\text{old}})^T H (\phi - \phi_{\text{old}}) \le \epsilon. \end{aligned} \tag{30}$$

where g is the gradient of the objective, and $H = \frac{1}{N}\sum_n j_n j_n^T$, where $j_n = \nabla_\phi V_\phi(s_n)$. Note that H is the "Gauss-Newton" approximation of the Hessian of the objective, and it is (up to a σ^2 factor) the Fisher information matrix when interpreting the value function as a conditional probability distribution. Using matrix-vector products $v \to Hv$ to implement the conjugate gradient algorithm, we compute a step direction $s \approx -H^{-1}g$. Then we rescale $s \to \alpha s$ such that $\frac{1}{2}(\alpha s)^T H(\alpha s) = \epsilon$ and take $\phi = \phi_{\text{old}} + \alpha s$. This procedure is analogous to the procedure we use for updating the policy, which is described further in Section 6 and based on Schulman et al. (2015).

6 EXPERIMENTS

We designed a set of experiments to investigate the following questions:

1. What is the empirical effect of varying $\lambda \in [0, 1]$ and $\gamma \in [0, 1]$ when optimizing episodic total reward using generalized advantage estimation?
2. Can generalized advantage estimation, along with trust region algorithms for policy and value function optimization, be used to optimize large neural network policies for challenging control problems?

[2] Another natural choice is to compute target values with an estimator based on the TD(λ) backup (Bertsekas, 2012; Sutton & Barto, 1998), mirroring the expression we use for policy gradient estimation: $\hat{V}_t^\lambda = V_{\phi_{\text{old}}}(s_n) + \sum_{l=0}^{\infty}(\gamma\lambda)^l \delta_{t+l}$. While we experimented with this choice, we did not notice a difference in performance from the $\lambda = 1$ estimator in Equation (28).