0.1 Guided Policy Search

- *Algorithm:* Guided Policy Search (algo. 1)
- *Input:* Environment and Immediate Cost (Reward) function: $c(x_t, u_t)$
- *Complexity:* Subject to different cases
- *Data structure compatibility:* N/A
- *Common applications:* Do trajectory optimizaiton in dynamic systems, such as simulated robots in the swimming, hopping and walking tasks[**paper1**].

Problem. Guided Policy Search

Given the environment and immediate cost (reward) function, GPS do the interaction between controller and environment. The algorithm will gain a final policy that could "know" the optimal control through additional parameter added.

Description

As a popular algorithm in reinforcement learning, Guided Policy Search (GPS) has some special properties compared to other algorithms. It build model, or dynamics to the environment, so it could not only record, but also calculate. A better sample efficiency could be obtained compared to Model-Free algorithm. GPS will final give a parametric policy, such as neural network, so no online optimization needed when testing.

Problem Formulation

To formulate the problem, introduce the following formulation[**blog**]

- setting: fixed time length task
- assumption: deterministic dynamics: $x_t = f(x_{t-1}, u_{t-1})$
- input: environment and immediate cost (reward) function: $c(x_t, u_t)$
- output: parametric policy

With the above settings, the output parametric policy will be specified as deterministic case in this project with frame work such that: Collect Data, Fit Dynamics, Optimization and Next Iteration.

Deterministic Policy Case

1. Collect data: Use controller to interactive with environment and get the trajectory dataset D $\mathcal{D} = \tau_i$, where the trajectory is represented through $\tau_i = x_{1i}, u_{1i}, \dots, x_{1T}, u_{1T}$.
2. Fit Dynamics. Use the dataset D to fit one linear model that would use different model for different input from different time, such that $x_{t+1} = f(x_t, u_t) = A_t x_t + B_t u_t + c_t$. The model is obtained through the τ_i data $(x_{t1}, u_{t1}, x_{t+1,1}), \dots, (x_{ti}, u_{ti}, x_{t+1,i})$ from one specific time t using linear regression.
3. Optimization.

   Controller As our goal is to minimize the cost while satisfying the dynamics constraint, the problem could be summarized as the equation

$$\min_{x_1, u_1, \dots, x_T, u_T} \sum_{t=1}^{T} c(x_t, u_t)$$
$$\text{s.t. } x_t = f(x_{t-1}, u_{t-1}) \quad t = 1, \dots, T$$

To solve this equation, when the cost is a quadratic mode, there is one optimization method called *Linear Quadratic Regulator (LQR)*. LQR could take the input:linear model(F_t, f_t)and quadratic cost(C_t, c_t) with the output K_t, k_t. Obtain the optimal control, which is the solution to the previous equation $u_t = K_t x_t + k_t$, $x_{t+1} = f(x_t, u_t)$.

If the cost function is not a quadratic model, we could use iterative LQR(iLQR) to get the solution, which will use linear and quadratic expansion then apply LQR.

Policy Then it comes to the special part of GPS, as one policy will be obtained to "know" how to do the optimal control. So one optimization variable θ and constraint $u_t = \pi_\theta(x_t)$ is added

$$\begin{aligned} \min_{x_1, u_1, \ldots, x_T, u_T, \theta} & \sum_{t=1}^{T} c(x_t, u_t) \\ \text{s.t. } & u_t = \pi_\theta(x_t) \quad t = 1, \ldots, T \\ & x_t = f(x_{t-1}, u_{t-1}) \quad t = 1, \ldots, T. \end{aligned}$$

The solution for the above equation using *Dual Gradient Descent(DGD)*, the brief procedure is that

- Write the corresponding Lagrangian function $\mathcal{L}(x, \lambda) = f(x) + \lambda C(x)$
- Find the x such that minimize the function $x^* = \text{argmin}_{\ x} \mathcal{L}(x, \lambda)$
- Use x^* into $\mathcal{L}(x, \lambda)$ to get the lower bound of the original question $g(\lambda) = \mathcal{L}(x^*, \lambda)$
- Update the lambda $\lambda = \lambda + \alpha \frac{\partial g}{\partial \lambda}$
- Return to step 2 to do iteration.

4. Next Iteration. Return to Collect Data step, which is that the new controller will interactive to the environment again and do the iteration.

Algorithm 1: Guided Policy Search [**paper1**]

Generate DDP solutions $\pi_{\varsigma_1}, \ldots, \pi_{\varsigma_n}$
Sample $\zeta_q, \ldots, \zeta_m$ from $q(\zeta) = \frac{1}{n} \sum_i \pi_{\varsigma_i}(\zeta)$
Initialize $\theta^* \leftarrow \arg\max_\theta \sum_i \pi_{\varsigma_i}(\zeta)$
Build initial sample set S from $\pi_{\varsigma_1}, \ldots, \pi_{\varsigma_n}, \pi_{\theta^*}$
for *iteration* $k = 1$ **to** K **do**
    Choose current sample set $\S_k \subset S$
    Optimize $\theta_k \leftarrow \arg \text{msx}_\theta \Phi_{S_k}(\theta)$
    Append samples from π_{θ_k} to S_k and S
    Optionally generate adaptive guiding samples
    Estimate the values of π_{θ_k} and π_{θ^*} using S_k
    if π_{θ_k} *is better than* π_{θ^*} **then**
        $\theta^* \leftarrow \theta_k$
        Decrease w_r
    end if
    else
        Increase w_r
        Optionally resample from π_θ
    end if
end for
return *the best policy* π_{θ^*}