Parametrize a stochastic policy to generate data to construct replay buffer for training the deterministic policy
Stochastic policy acts like a teacher that gathers high-quality trajectories for the deterministic policy
Policy gradient of stochastic policy is evaluated using rewards as the policy improvement of the deterministic policy

## Difference from related work

Instead of traditional meta-learning where hyperparameters are optimized, this work tries to generate high quality data to better train RL agents

In DDPG, we have $\pi$ (actor policy) and $\pi_e$ (exploration policy). Generally $\pi_e$ is constructed heuristically by adding noise (eg: OU-Noise).
An assumption with $\pi_e$ is that it should be close to $\pi$. But this is not true as we need $\pi_e$ to explore states not seen before
Because DDPG is off-policy algo, we can decouple exploration policy and actor policy.

$$J(\pi_e) = \mathbb{E}_{D_0 \sim \pi_e}\left[R(\pi, D_0)\right]$$

$$= \mathbb{E}_{D_0 \sim \pi_e}\left[R_{\pi'} - R_\pi\right]$$

policy obtained after one or few updates

Here $R(\pi, D_0)$ [called meta-reward]
denotes how much the teacher ($\pi_e$) helped
the student ($\pi$)

$$\nabla_{\theta^{\pi_e}} J = \mathop{E}_{D_0 \sim \pi_e}\left[ R(\pi, D_0)\, \nabla_{\theta^{\pi_e}} \log \underbrace{P(D|\pi_e)}_{} \right]$$

Probability of generating transition $D_0 := \{s_t, a_t, r_t\}_{t=1}^{T}$ given $\pi_e$

$$P(D_0|\pi_e) = p(s_1) \prod_{t=1}^{T} \pi_e(a_t|s_t)\, p(s_{t+1}|s_t, a_t)$$

$$\nabla_{\theta^{\pi_e}} \log P(D_0|\pi_e) = \sum_{t=1}^{T} \nabla_{\theta^{\pi_e}} \log(\pi_e(a_t|s_t))$$

To estimate $R(\pi, D_0)$, run DDPG ahead for one or a small no. of steps:

→ calculate new actor policy $\pi' = DDPG(\pi, D_0)$ by running it on $D_0$.

→ Simulate $\pi'$ to get $D_1$ and use $D_1$ to get estimation $\hat{R}_{\pi'}$ of the reward of $\pi'$

$$\hat{R}(\pi, D_0) = \hat{R}_{\pi'} - \hat{R}_{\pi}$$

→ $\theta^{\pi_e} \leftarrow \theta^{\pi_e} + \eta \hat{R}(\pi, D_0) \sum_{t=1}^{T} \nabla_{\theta^{\pi_e}} \log(\pi_e(a_t|s_t))$

→ $B \leftarrow B \cup D_0 \cup D_1$

→ update $\pi$ based on $B$, $\pi \leftarrow DDPG(\pi, B)$

Types

(i) Meta(variance) :- $\pi_e$ equal to actor policy + Gaussian noise whose variance is trained adaptively
$\pi_e = \mathcal{N}(\mu(s, \theta^{\pi}), \sigma^2 I)$, $\sigma$ is parameter

of $\pi_e$

2.) Meta:- $\pi_e$ is another Gaussian

$$\pi_e = \mathcal{N}(f(s,\theta^f), \sigma^2 I) \; ; \; \theta^{\pi_e} := [\theta^f, \sigma]$$