SUPPLEMENTARY MATERIAL

A CALCULATING MARGINAL ACTION PROBABILITY GIVEN THE HISTORY FOR ON-POLICY COHERENT EXPLORATION

As discussed, forward message $\alpha(\mathbf{w}_t)$ is used to compute the marginal action probability given the history at step t for the final learning objective. Suppose we have the Gaussian policy represented as:

$$\pi_{\mathbf{w}_t,\boldsymbol{\theta}}(\mathbf{a}_t|\mathbf{s}_t) = \mathcal{N}(\mathbf{W}_t\mathbf{x}_t + \mathbf{b}_t, \boldsymbol{\Lambda}_a^{-1}), \tag{1}$$

where $\mathbf{a}_t \in \mathbb{R}^p$, $\mathbf{x}_t = f_{\boldsymbol{\theta}}(\mathbf{s}_t) \in \mathbb{R}^q$, $\mathbf{W}_t \in \mathbb{R}^{p\times q}$ is the coefficient matrix, $\mathbf{b}_t \in \mathbb{R}^p$ is the bias vector, and $\boldsymbol{\Lambda}_a$ is a constant precision matrix. It's helpful to introduce $\mathbf{w}_t \in \mathbb{R}^{pq+p}$ by flattening $\mathbf{W}_t$ and combining $\mathbf{b}_t$:

$$\mathbf{w}_t = \begin{pmatrix} w_{11} \\ \vdots \\ w_{1q} \\ \vdots \\ w_{p1} \\ \vdots \\ w_{pq} \\ b_1 \\ \vdots \\ b_p \end{pmatrix}, \tag{2}$$

and correspondingly stack $\mathbf{x}_t$ into $\mathbf{X}_t \in \mathbb{R}^{p\times(pq+p)}$:

$$\mathbf{X}_t = \begin{pmatrix} \mathbf{x}_t^T & \mathbf{0}_{q,1}^T & \dots & \mathbf{0}_{q,1}^T & \mathbf{0}_{q,1}^T & 1 & 0 & \dots & 0 & 0 \\ \mathbf{0}_{q,1}^T & \mathbf{x}_t^T & \dots & \mathbf{0}_{q,1}^T & \mathbf{0}_{q,1}^T & 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{0}_{q,1}^T & \mathbf{0}_{q,1}^T & \dots & \mathbf{x}_t^T & \mathbf{0}_{q,1}^T & 0 & 0 & \dots & 1 & 0 \\ \mathbf{0}_{q,1}^T & \mathbf{0}_{q,1}^T & \dots & \mathbf{0}_{q,1}^T & \mathbf{x}_t^T & 0 & 0 & \dots & 0 & 1 \end{pmatrix}, \tag{3}$$

where $\mathbf{0}_{q,1}$ is a q-dimension zero column vector. After this transformation, the Gaussian policy is represented equivalently as:

$$\pi_{\mathbf{w}_t,\boldsymbol{\theta}}(\mathbf{a}_t|\mathbf{s}_t) = \mathcal{N}(\mathbf{X}_t\mathbf{w}_t, \boldsymbol{\Lambda}_a^{-1}). \tag{4}$$

A.1 BASE CASE

For the base case $t = 0$, forward message $\alpha(\mathbf{w}_0)$ and the initial transition probability of $\mathbf{w}_0$ is identical by definition:

$$\alpha(\mathbf{w}_0) = p_0(\mathbf{w}_0; \boldsymbol{\mu}, \boldsymbol{\Lambda}) = \mathcal{N}\left(\boldsymbol{\mu}, \boldsymbol{\Lambda}^{-1}\right). \tag{5}$$

Additionally, the action probability is given by:

$$\pi_{\mathbf{w}_0,\boldsymbol{\theta}}(\mathbf{a}_0|\mathbf{s}_0) = \mathcal{N}(\mathbf{X}_0\mathbf{w}_0, \boldsymbol{\Lambda}_a^{-1}). \tag{6}$$

The marginal action probability given the history at $t = 0$ is given by:

$$p(\mathbf{a}_0|\mathbf{s}_0, \boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\theta}) = \int \pi_{\mathbf{w}_0,\boldsymbol{\theta}}(\mathbf{a}_0|\mathbf{s}_0)\alpha(\mathbf{w}_0)d\mathbf{w}_0 \tag{7}$$

$$= \mathcal{N}(\mathbf{X}_0\boldsymbol{\mu}, \boldsymbol{\Lambda}_a^{-1} + \mathbf{X}_0\boldsymbol{\Lambda}^{-1}\mathbf{X}_0^T). \tag{8}$$

A.2 General Case

For the general case of step $t > 0$, we need the state $\mathbf{s}_{t-1}$, action $\mathbf{a}_{t-1}$ as well as mean and covariance of forward message $\alpha(\mathbf{w}_{t-1})$ stored from previous step. Suppose we have $\alpha(\mathbf{w}_{t-1})$ as:

$$\alpha(\mathbf{w}_{t-1}) = \mathcal{N}(\mathbf{v}_{t-1}, \mathbf{L}_{t-1}^{-1}), \tag{9}$$

and the action probability from the previous step is given by:

$$\pi_{\mathbf{w}_{t-1},\boldsymbol{\theta}}(\mathbf{a}_{t-1}|\mathbf{s}_{t-1}) = \mathcal{N}\left(\mathbf{X}_{t-1}\mathbf{w}_{t-1}, \boldsymbol{\Lambda}_a^{-1}\right). \tag{10}$$

We have directly:

$$p(\mathbf{w}_{t-1}|\mathbf{s}_{[0:t-1]}, \mathbf{a}_{[0:t-1]}, \boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\theta}) = \mathcal{N}\left(\mathbf{u}_{t-1}, \boldsymbol{\Sigma}_{t-1}\right), \tag{11}$$

with

$$\mathbf{u}_{t-1} = \boldsymbol{\Sigma}_{t-1}\left(\mathbf{X}_{t-1}^T\boldsymbol{\Lambda}_a\mathbf{a}_{t-1} + \mathbf{L}_{t-1}\mathbf{v}_{t-1}\right) \tag{12}$$

$$\boldsymbol{\Sigma}_{t-1} = \left(\mathbf{L}_{t-1} + \mathbf{X}_{t-1}^T\boldsymbol{\Lambda}_a\mathbf{X}_{t-1}\right)^{-1}. \tag{13}$$

Combining the transition probability of $\mathbf{w}_t$:

$$p(\mathbf{w}_t|\mathbf{w}_{t-1}; \boldsymbol{\mu}, \boldsymbol{\Lambda}) = \mathcal{N}\left((1-\beta)\mathbf{w}_{t-1} + \beta\boldsymbol{\mu}, (2\beta - \beta^2)\boldsymbol{\Lambda}^{-1}\right), \tag{14}$$

we obtain the forward message $\alpha(\mathbf{w}_t)$:

$$\alpha\left(\mathbf{w}_t\right) = \mathcal{N}\left(\mathbf{v}_t, \mathbf{L}_t^{-1}\right), \tag{15}$$

where

$$\mathbf{v}_t = (1-\beta)\mathbf{u}_{t-1} + \beta\boldsymbol{\mu} \tag{16}$$

$$\mathbf{L}_t^{-1} = (2\beta - \beta^2)\boldsymbol{\Lambda}^{-1} + (1-\beta)^2\boldsymbol{\Sigma}_{t-1}. \tag{17}$$

Here, $\mathbf{v}_t$ and $\mathbf{L}_t^{-1}$ should be stored and used for exact inference of $\alpha(\mathbf{w}_{t+1})$ at the next step. Finally, the marginal action probability given the history at step $t > 0$ is given by:

$$p(\mathbf{a}_t|\mathbf{s}_{[0:t]}, \mathbf{a}_{[0:t-1]}, \boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\theta}) = \int \pi_{\mathbf{w}_t,\boldsymbol{\theta}}(\mathbf{a}_t|\mathbf{s}_t)\alpha(\mathbf{w}_t)d\mathbf{w}_t \tag{18}$$

$$= \mathcal{N}(\mathbf{X}_t\mathbf{v}_t, \boldsymbol{\Lambda}_a^{-1} + \mathbf{X}_t\mathbf{L}_t^{-1}\mathbf{X}_t^T). \tag{19}$$

B Deep Coherent Reinforcement Learning

In this section we provide more detailed recipes of applying deep coherent exploration for A2C, PPO, and SAC. Respectively, we call them Coherent-A2C, Coherent-PPO, and Coherent-SAC.

B.1 Coherent Advantage Actor-Critic (Coherent-A2C)

Coherent-A2C is straightforward to implement. To do that, one could just replace the regular $p(\mathbf{a}_t|\mathbf{s}_t;\boldsymbol{\theta})$ term in A2C objective with the $p(\mathbf{a}_t|\mathbf{s}_{[0:t]}, \mathbf{a}_{[0:t-1]}, \boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\theta})$ term. The pseudo-code of single-worker Coherent-A2C is shown in Algorithm 1.

B.2 Coherent Proximal Policy Optimization (Coherent-PPO)

Coherent-PPO can be implemented in a similar way as Coherent-A2C. As in Coherent-A2C, we substitute the regular $p(\mathbf{a}_t|\mathbf{s}_t;\boldsymbol{\theta})$ term with the $p(\mathbf{a}_t|\mathbf{s}_{[0:t]}, \mathbf{a}_{[0:t-1]}, \boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\theta})$ term. Here, after each step of policy update, marginal action probability given the history $p(\mathbf{a}_t|\mathbf{s}_{[0:t]}, \mathbf{a}_{[0:t-1]}, \boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\theta})$ from the newly updated policy should be evaluated on the most recent trajectory τ_k for next policy update

Algorithm 1: Coherent-A2C

Input: Initial policy parameters $\boldsymbol{\mu}_0, \boldsymbol{\Lambda}_0, \boldsymbol{\theta}_0$ and initial value function parameters $\boldsymbol{\phi}_0$.

for *each iteration k* **do**
  for *each environmental step t* **do**
    if *t=0* **then**
      $\mathbf{w}_0 \sim p_0(\mathbf{w}_0; \boldsymbol{\mu}_k, \boldsymbol{\Lambda}_k)$
    else
      $\mathbf{w}_t \sim p(\mathbf{w}_t | \mathbf{w}_{t-1}; \boldsymbol{\mu}_k, \boldsymbol{\Lambda}_k)$
    Compute $p(\mathbf{a}_t | \mathbf{s}_{[0:t]}, \mathbf{a}_{[0:t-1]}, \boldsymbol{\mu}_k, \boldsymbol{\Lambda}_k, \boldsymbol{\theta}_k)$
  Compute the Coherent-A2C objective and update the policy by performing a gradient step:

$$\boldsymbol{\mu}_{k+1} \leftarrow \boldsymbol{\mu}_k + \alpha_{\boldsymbol{\mu}} \hat{\nabla}_{\boldsymbol{\mu}} J(\boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\theta})$$
$$\boldsymbol{\Lambda}_{k+1} \leftarrow \boldsymbol{\Lambda}_k + \alpha_{\boldsymbol{\Lambda}} \hat{\nabla}_{\boldsymbol{\Lambda}} J(\boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\theta})$$
$$\boldsymbol{\theta}_{k+1} \leftarrow \boldsymbol{\theta}_k + \alpha_{\boldsymbol{\theta}} \hat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\theta})$$

  Learn value function by performing one or multiple gradient steps:

$$\boldsymbol{\phi}_{k+1} \leftarrow \boldsymbol{\phi}_k + \alpha_{\boldsymbol{\phi}} \hat{\nabla}_{\boldsymbol{\phi}} L(\boldsymbol{\phi})$$

and approximated Kullback–Leibler (KL) constraint. The pseudo-code of single-worker Coherent-PPO is shown in Algorithm 2.

Algorithm 2: Coherent-PPO

Input: Initial policy parameters $\boldsymbol{\mu}_0, \boldsymbol{\Lambda}_0, \boldsymbol{\theta}_0$ and initial value function parameters $\boldsymbol{\phi}_0$.

for *each iteration k* **do**
  for *each environmental step t* **do**
    if *t=0* **then**
      $\mathbf{w}_0 \sim p_0(\mathbf{w}_0; \boldsymbol{\mu}_k, \boldsymbol{\Lambda}_k)$
    else
      $\mathbf{w}_t \sim p(\mathbf{w}_t | \mathbf{w}_{t-1}; \boldsymbol{\mu}_k, \boldsymbol{\Lambda}_k)$
    Compute $p(\mathbf{a}_t | \mathbf{s}_{[0:t]}, \mathbf{a}_{[0:t-1]}, \boldsymbol{\mu}_k, \boldsymbol{\Lambda}_k, \boldsymbol{\theta}_k)$
  Compute the Coherent-PPO objective and update the policy by performing multiple gradient steps until the approximated KL constraint is met:

$$\boldsymbol{\mu}_{k+1} \leftarrow \boldsymbol{\mu}_k + \alpha_{\boldsymbol{\mu}} \hat{\nabla}_{\boldsymbol{\mu}} J(\boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\theta})$$
$$\boldsymbol{\Lambda}_{k+1} \leftarrow \boldsymbol{\Lambda}_k + \alpha_{\boldsymbol{\Lambda}} \hat{\nabla}_{\boldsymbol{\Lambda}} J(\boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\theta})$$
$$\boldsymbol{\theta}_{k+1} \leftarrow \boldsymbol{\theta}_k + \alpha_{\boldsymbol{\theta}} \hat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\theta})$$

  Learn value function by performing one or multiple gradient steps:

$$\boldsymbol{\phi}_{k+1} \leftarrow \boldsymbol{\phi}_k + \alpha_{\boldsymbol{\phi}} \hat{\nabla}_{\boldsymbol{\phi}} L(\boldsymbol{\phi})$$

B.3 Coherent Soft Actor-Critic (Coherent-SAC)

For Coherent-SAC, the behavior policy is the same as on-policy coherent exploration, where the last layer parameters of the policy network is sampled per step for exploration. Then, for policy updates,

the target policy is the marginal policy $p\left(\mathbf{a}_t|\mathbf{s}_t, \boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\theta}\right)$ instead of the policy conditioned on the sampled parameters $\mathbf{w}$. The pseudo-code of single-worker Coherent-SAC is shown in Algorithm 3.

Algorithm 3: Coherent-SAC

Input: Initial policy parameters $\boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\theta}$ and initial Q-function parameters $\boldsymbol{\phi}_1$,$\boldsymbol{\phi}_2$, empty replay buffer $\mathcal{D}$.

Set target parameters equal to main parameters $\boldsymbol{\phi}_{\text{targ},1} \leftarrow \boldsymbol{\phi}_1, \boldsymbol{\phi}_{\text{targ},2} \leftarrow \boldsymbol{\phi}_2$.

for *each environmental step* **do**

  if *just updated* **then**

    $\mathbf{w} \sim p_0(\mathbf{w}; \boldsymbol{\mu}, \boldsymbol{\Lambda})$

  else

    $\mathbf{w} \sim p(\mathbf{w}|\mathbf{w}_{\text{prev}}; \boldsymbol{\mu}, \boldsymbol{\Lambda})$

  if *it's time to update* **then**

    for *j in range(number of updates)* **do**

      Randomly sample a batch of transitions $\mathcal{B} = \{(\mathbf{s}, \mathbf{a}, r, \mathbf{s}', d)\}$

      Compute targets for Q-functions:

$$y\left(r, \mathbf{s}', d\right) = r + \gamma(1-d)\left(\min_{i=1,2} Q_{\boldsymbol{\phi}_{\text{targ},i}}\left(\mathbf{s}', \tilde{\mathbf{a}}'\right) - \alpha \log p\left(\tilde{\mathbf{a}}'|\mathbf{s}', \boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\theta}\right)\right),$$

      where $\tilde{\mathbf{a}}' \sim p\left(\tilde{\mathbf{a}}'|\mathbf{s}', \boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\theta}\right)$

      Update Q-functions by one step of gradient descent using:

$$\nabla_{\boldsymbol{\phi}_i} \frac{1}{|\mathcal{B}|} \sum_{(\mathbf{s}, \mathbf{a}, r, \mathbf{s}', d) \in \mathcal{B}} \left(Q_{\boldsymbol{\phi}_i}(\mathbf{s}, \mathbf{a}) - y\left(r, \mathbf{s}', d\right)\right)^2, \quad \text{for } i = 1, 2$$

      Update policy by one step of gradient ascent using:

$$\nabla_{\boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\theta}} \frac{1}{|\mathcal{B}|} \sum_{\mathbf{s} \in \mathcal{B}} \left[\min_{i=1,2} Q_{\boldsymbol{\phi}_i}\left(\mathbf{s}, \tilde{\mathbf{a}}\right) - \alpha \log p\left(\tilde{\mathbf{a}}|\mathbf{s}, \boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\theta}\right)\right],$$

      where $\tilde{\mathbf{a}}$ is a sample from $p\left(\tilde{\mathbf{a}}|\mathbf{s}, \boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\theta}\right)$ which is differentiable w.r.t. $\boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\theta}$ via the reparameterization trick

      Update target networks with:

$$\boldsymbol{\phi}_{\text{targ},i} \leftarrow \rho \boldsymbol{\phi}_{\text{targ},i} + (1-\rho)\boldsymbol{\phi}_i \quad \text{for } i = 1, 2$$

  else

    Continue

C ADDITIONAL RESULTS

In this section, we provide additional results of comparative evaluation for A2C and PPO on Hopper-v2, Reacher-v2, and InvertedDoublePendulum-v2. The results for A2C are shown in Figure 1 and the results for PPO are shown in Figure 2.

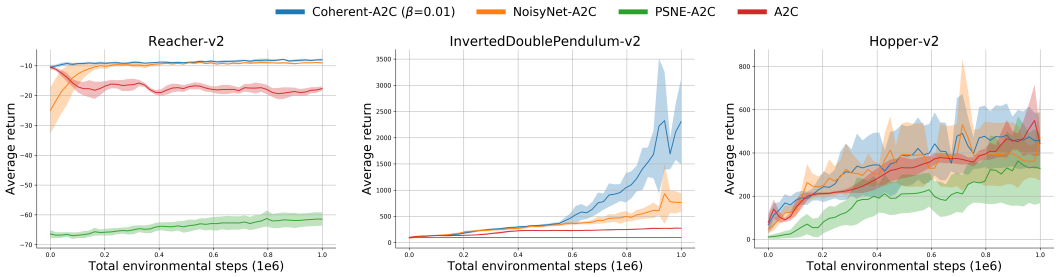

Figure 1: Learning curves for A2C with different exploration strategies on OpenAI MuJoCo continuous control tasks. The solid curves correspond to the mean, and the shaped region represents two times the standard error of the average return over 10 random seeds.

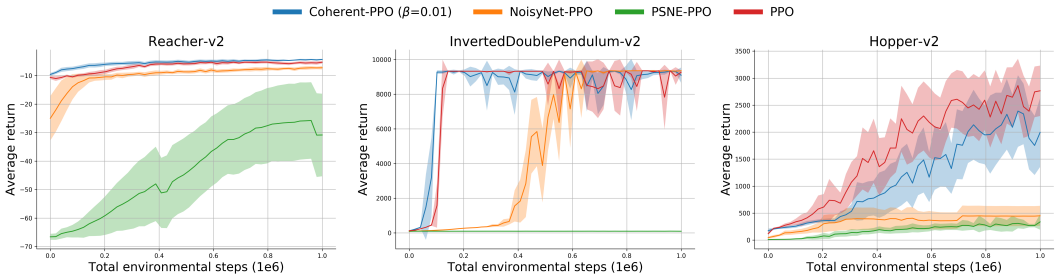

Figure 2: Learning curves for PPO with different exploration strategies on OpenAI MuJoCo continuous control tasks. The solid curves correspond to the mean, and the shaped region represents two times the standard error of the average return over 10 random seeds.