

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:

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

where $\mathbf{a}_t \in \mathbb{R}^p$, $\mathbf{x}_t = f_\theta(\mathbf{s}_t) \in \mathbb{R}^q$, $\mathbf{W}_t \in \mathbb{R}^{p \times q}$ is the coefficient matrix, $\mathbf{b}_t \in \mathbb{R}^p$ is the bias vector, and $\mathbf{\Lambda}_a$ 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 :

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

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

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

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

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

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:

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

Additionally, the action probability is given by:

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

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

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

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

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:

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

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

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

We have directly:

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

with

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

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

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

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

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

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

where

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

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

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:

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

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

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 $\mu_0, \Lambda_0, \theta_0$ and initial value function parameters ϕ_0 .

```

1 for each iteration  $k$  do
2   for each environmental step  $t$  do
3     if  $t=0$  then
4        $\mathbf{w}_0 \sim p_0(\mathbf{w}_0; \mu_k, \Lambda_k)$ 
5     else
6        $\mathbf{w}_t \sim p(\mathbf{w}_t | \mathbf{w}_{t-1}; \mu_k, \Lambda_k)$ 
7      $p(\mathbf{a}_t | \mathbf{s}_{[0:t]}, \mathbf{a}_{[0:t-1]}; \mu_k, \Lambda_k, \theta_k)$ 
8     Compute the Coherent-A2C objective and update the policy by performing a gradient step:
           
$$\mu_{k+1} \leftarrow \mu_k + \alpha_\mu \hat{\nabla}_\mu J(\mu, \Lambda, \theta)$$

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

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

9     Learn value function by performing one or multiple gradient steps:
           
$$\phi_{k+1} \leftarrow \phi_k + \alpha_\phi \hat{\nabla}_\phi L(\phi)$$


```

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 $\mu_0, \Lambda_0, \theta_0$ and initial value function parameters ϕ_0 .

```

1 for each iteration  $k$  do
2   for each environmental step  $t$  do
3     if  $t=0$  then
4        $\mathbf{w}_0 \sim p_0(\mathbf{w}_0; \mu_k, \Lambda_k)$ 
5     else
6        $\mathbf{w}_t \sim p(\mathbf{w}_t | \mathbf{w}_{t-1}; \mu_k, \Lambda_k)$ 
7      $p(\mathbf{a}_t | \mathbf{s}_{[0:t]}, \mathbf{a}_{[0:t-1]}; \mu_k, \Lambda_k, \theta_k)$ 
8     Compute the Coherent-PPO objective and update the policy by performing multiple
       gradient steps until the approximated KL constraint is met:
           
$$\mu_{k+1} \leftarrow \mu_k + \alpha_\mu \hat{\nabla}_\mu J(\mu, \Lambda, \theta)$$

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

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

9     Learn value function by performing one or multiple gradient steps:
           
$$\phi_{k+1} \leftarrow \phi_k + \alpha_\phi \hat{\nabla}_\phi L(\phi)$$


```

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 $p(\mathbf{a}_t | \mathbf{s}_t, \boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\theta})$ 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 ϕ_1, ϕ_2 , empty replay buffer \mathcal{D} .

```

1 Set target parameters equal to main parameters  $\phi_{\text{target},1} \leftarrow \phi_1, \phi_{\text{target},2} \leftarrow \phi_2$ .
2 for each environmental step do
3   if just updated then
4      $\mathbf{w} \sim p_0(\mathbf{w}; \boldsymbol{\mu}, \boldsymbol{\Lambda})$ 
5   else
6      $\mathbf{w} \sim p(\mathbf{w} | \mathbf{w}_{\text{prev}}; \boldsymbol{\mu}, \boldsymbol{\Lambda})$ 
7   if it's time to update then
8     for  $j$  in range(number of updates) do
9       Randomly sample a batch of transitions  $\mathcal{B} = \{(\mathbf{s}, \mathbf{a}, r, \mathbf{s}', d)\}$ 
10      Compute targets for  $Q$ -functions:
11
12      
$$y(r, \mathbf{s}', d) = r + \gamma(1 - d) \left( \min_{i=1,2} Q_{\phi_{\text{target},i}}(\mathbf{s}', \tilde{\mathbf{a}}') - \alpha \log p(\tilde{\mathbf{a}}' | \mathbf{s}', \boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\theta}) \right),$$

13      where  $\tilde{\mathbf{a}}' \sim p(\tilde{\mathbf{a}}' | \mathbf{s}', \boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\theta})$ 
14      Update  $Q$ -functions by one step of gradient descent using:
15
16      
$$\nabla_{\phi_i} \frac{1}{|\mathcal{B}|} \sum_{(\mathbf{s}, \mathbf{a}, r, \mathbf{s}', d) \in \mathcal{B}} (Q_{\phi_i}(\mathbf{s}, \mathbf{a}) - y(r, \mathbf{s}', d))^2, \quad \text{for } i = 1, 2$$

17
18      Update policy by one step of gradient ascent using:
19
20      
$$\nabla_{\boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\theta}} \frac{1}{|\mathcal{B}|} \sum_{\mathbf{s} \in \mathcal{B}} \left[ \min_{i=1,2} Q_{\phi_i}(\mathbf{s}, \tilde{\mathbf{a}}) - \alpha \log p(\tilde{\mathbf{a}} | \mathbf{s}, \boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\theta}) \right],$$

21      where  $\tilde{\mathbf{a}}$  is a sample from  $p(\tilde{\mathbf{a}} | \mathbf{s}, \boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\theta})$  which is differentiable w.r.t.  $\boldsymbol{\mu}, \boldsymbol{\Lambda}, \boldsymbol{\theta}$  via
22      the reparameterization trick
23      Update target networks with:
24
25      
$$\phi_{\text{target},i} \leftarrow \rho \phi_{\text{target},i} + (1 - \rho) \phi_i \quad \text{for } i = 1, 2$$

26   else
27     Continue
    
```
