Algorithm 1: PAC-Bayesian Actor-Critic (PBAC)

Input: Polyak coefficient $\tau \in (0,1)$, batch size n, bootstrap rate κ, posterior sampling rate `PSR`, prior variance σ_0^2, ensemble size K

Initialization: Replay buffer $\mathcal{D} \leftarrow \emptyset$; critic parameters $\{\theta_k\}_{k=1}^K$ and targets $\bar{\theta}_k \leftarrow \theta_k$; actor trunk g and heads $h_1, \ldots, h_K$

Initialize state $s \leftarrow$ `env.reset()`, interaction counter $e \leftarrow 0$;

while *training* **do**

  if e mod *PSR* $= 0$ **then**

    Sample index $j \sim \mathcal{U}(\{1, \ldots, K\})$;

    Set active policy: $\pi \leftarrow \pi_{g \circ h_j}$;

  end

  Sample action $a \sim \pi(s)$;

  Execute a in environment, observe reward r, next state s';

  Store transition: $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s, a, r, s')\}$;

  $e \leftarrow e + 1$;

  Sample batch $\{(s_i, a_i, r_i, s_i')\}_{i=1}^n \sim \mathcal{D}$;

  Sample bootstrap mask: $b_{ik} \sim \text{Bernoulli}(1-\kappa)$ for all $i \in [n],\ k \in [K]$;

  foreach (s_i, a_i, r_i, s_i') *in batch* **do**

$$\bar{\mu}_\pi(s_i') \leftarrow \frac{1}{K} \sum_{k=1}^K b_{ik} \cdot \bar{X}_k(s_i', \pi(s_i'))$$

$$\mu_\pi(s_i) \leftarrow \frac{1}{K} \sum_{k=1}^K b_{ik} \cdot X_k(s_i, \pi(s_i))$$

$$\sigma_\pi^2(s_i) \leftarrow \frac{1}{K-1} \sum_{k=1}^K b_{ik} \cdot \left(X_k(s_i, \pi(s_i)) - \mu_\pi(s_i)\right)^2$$

  end

  Update critic parameters $\{\theta_k\}$ by minimizing:

$$\frac{1}{nK} \sum_{i=1}^n \sum_{k=1}^K b_{ik} \left(r_i + \gamma \bar{X}_k(s_i', \pi(s_i')) - X_k(s_i, \pi(s_i))\right)^2$$

$$+ \frac{1}{nK} \sum_{i=1}^n \sum_{k=1}^K \frac{b_{ik} \left(r_i + \gamma \bar{\mu}_\pi(s_i') - X_k(s_i, \pi(s_i))\right)^2}{2\gamma^2 \sigma_0^2} - \frac{\gamma^2 + 1/2}{n} \sum_{i=1}^n \log \sigma_\pi^2(s_i)$$

  Update actor:

$$(g, h_1, \ldots, h_K) \leftarrow \arg \max_{g, h_1, \ldots, h_K} \frac{1}{nK} \sum_{i=1}^n \sum_{k=1}^K X_k(s_i, \pi_{g \circ h_k}(s_i))$$

  Update target critics:

$$\bar{\theta}_k \leftarrow \tau \theta_k + (1 - \tau) \bar{\theta}_k, \quad \text{for all } k \in [K]$$

  if *episode ended* **then**

    $s \leftarrow$ `env.reset()`;

  else

    $s \leftarrow s'$;

  end

end