We thank the reviewers for their valuable comments and suggestions, which will help us to improve the paper. We first respond to **R1**: **1)** While "twin-delay" is well-established to help estimate $Q(\boldsymbol{s}, \boldsymbol{a})$, IDAC advances it to a distributional RL setting for continuous actions to estimate the distribution of the discounted cumulative return $Z(\boldsymbol{s}, \boldsymbol{a})$, whose expectation is $Q(\boldsymbol{s}, \boldsymbol{a})$. While the usual "twin-delay" only involves minimization of two scalars, the one in IDAC is distinct in involving *element-wise* minimization of two *sorted* vectors, whose elements are iid sampled from two different DGNs. Besides, we introduce SIA and an asymptotic lower bound for entropy estimation. Both new techniques are helpful for IDAC to outperform the SOTA algorithms (please refer to Fig. 3). **2)** We will add the suggested ablation studies. **3)** Similar to Fig. 2(a)(b) illustrating SIA at an early training stage, we have examined SIA at a late stage and found that its differences from a diagonal Gaussian remain evident, as verified with both a normality test on the marginal of each dimension of the SIA policy and the Pearson correlation test on many randomly selected action dimension pairs. In addition, please see a related response to R3 (Lines 28-39) that strengthens the claim that SIA can represent more complicated policy distributions. These details will be added. **4)** As suggested, we will draw more explicit distinctions, such as the unique operation of sorting followed by element-wise minimization, and expand related work.

R2: **1)** We will clarify sorting is applied to each individual vector. We denote $\overrightarrow{x}_i$, which is a scalar, as the i-th element after sorting vector $\boldsymbol{x}$. **2)** We will better discuss related work to help avoid confusions and highlight our contributions.

R3: **1)** We are puzzled why IDAC is considered by R3 to be not sufficiently compared against other similar approaches; we would appreciate R3 pointing out these missing baselines. First, from the actor-critic perspective, we have already compared IDAC to SOTAs: PPO, SAC, & TD3. Second, from the distributional RL perspective, while there exist well-known algorithms (e.g., C51, QR-DQN, and IQN) for discrete actions, we find D4PG to be the only published one designed for continuous controls. Note we did not add a direct comparison to D4PG (as well as SDPG) as its implementation has notable differences from the Stable-baselines used in this paper. In particular, D4PG uses distributed agents to do distributed sampling for the replay buffer, allowing an implied advantage in observing more state-action pairs given the same number of policy gradient update steps. This is the main reason that we have adapted SDGP, which improves over D4PG, into Stable-baselines and used it for ablation study. To help address R3's concern, we will add these SDGP results (clearly worse than IDAC) into Fig. 1. Moreover, we have run the original D4PG code in hope to further eliminate this concern, and found that it consistently underperforms IDAC given the same number of policy gradient update steps. E.g., the max average returns of [D4PG, IDAC] after 10^6 policy gradient steps are $[9776 \pm 739, \mathbf{12222 \pm 157}]$ on HalfCheetah-v2, and $[4742 \pm 1320, \mathbf{5386 \pm 335}]$ on Walker2d-v2. **2)** *Verifying that SIA would improve exploration*: We note good exploration by SIA can be implied from Fig. 3, by better empirical performance; as the reward mechanism in continuous control RL tasks is complicated, such empirical comparison is a common way to indirectly evaluate whether better explorations have been achieved for these tasks (e.g. Sec. 4.4 of Hong et al. [2018], arXiv:1802.04564). To more directly verify that SIA does improve exploration, we mimic Sec. 5.1 of Haarnoja et al. [2017], arXiv:1702.08165 to introduce a Multigoal Environment with four equally-spaced and well-separated destinations in a 2D map that provide large rewards of [200, 200, 200, 600], with spatial location-dependent negative rewards elsewhere. This task requires extensive exploration to reach the optimal solution. We combine a Gaussian actor or a SIA with an Actor-Critic (A2C) algorithm; each algorithm is trained with 10 independent runs (10^5 episodes, one evaluation). In a single run, Gaussian often only explores no more than two (sub-optimal) destinations, achieving average cumulative reward as 123.17 ± 156.86, while SIA in general successfully explores all four destinations, achieving $\mathbf{336.84 \pm 26.9}$. These details verify that SIA does help exploration. **3)** We'd like to point out that IDAC uses Huber loss, and it is unclear whether additional theoretical justifications are needed to combine Huber loss with empirical Wasserstein distance. As quantile regression Huber loss is successfully used by QR-DQN for discrete actions, avoiding the potential issue of having biased gradients, we made the safe choice of using the same loss in IDAC. We agree it would be an interesting future work to investigate the use of empirical Wasserstein distance (with Huber loss) under IDAC. **4)** We will add more textual descriptions to improve our pseudo-code in the Appendix.

R4: **1)** We integrate DGN and SIA to allow them to help each other: *(i)* Modeling G exploits distributional information to help better estimate its mean Q (note C51, which outperforms DQN by exploiting distributional information, also conducts its argmax operation on Q); *(ii)* A more flexible policy may become more necessary given a better estimated Q. We have followed your suggestions to conducted additional ablation studies, which show removing either SIA or DGN from IDAC in general negatively impacts its performance. These results will be added into revision. **2)** Quantile regression loss can be applied to both explicit (as in QR-DQN) and implicit distribution (as in DGN). We favor DGN because it not only is more flexible, but also avoids a potential pitfall: QR-DQN fixes the quantile locations and feeds each $(\boldsymbol{s}, \boldsymbol{a})$ to a deep neural network (NN) to estimate their corresponding values; A clear concern, however, is that given input $(\boldsymbol{s}, \boldsymbol{a})$, the NN output at a designated higher quantile is not guaranteed to be larger than that at a lower quantile. By contrast, there is no such concern in DGN, which simply sorts its iid sampled values. **3)** We clarify that the SAC results were obtained by using the modern version [41] with the learned temperature. **4)** In Table 1 caption, we will make the change to use "max average returns", obtained by taking the maximal of averaged evaluation score (over 4 random seeds). **5)** The normalizing integral does not contribute to the gradient of $\boldsymbol{\theta}$ and hence omitted. In Eq(15), it is $\pi_\theta(\boldsymbol{a}^{(1)}|\boldsymbol{s}, \xi^{(\ell)})$ rather than $\pi_\theta(\boldsymbol{a}^{(\ell)}|\boldsymbol{s}, \xi^{(\ell)})$, as indicated in Eq(12) multiple auxiliary variables ξ are needed for each $\boldsymbol{a}$.