A. Maximum Entropy Objective

The exact definition of the discounted maximum entropy objective is complicated by the fact that, when using a discount factor for policy gradient methods, we typically do not discount the state distribution, only the rewards. In that sense, discounted policy gradients typically do not optimize the true discounted objective. Instead, they optimize average reward, with the discount serving to reduce variance, as discussed by Thomas (2014). However, we can define the objective that *is* optimized under a discount factor as

$$J(\pi) = \sum_{t=0}^{\infty} \mathbb{E}_{(\mathbf{s}_t, \mathbf{a}_t)\sim\rho_\pi} \left[\sum_{l=t}^{\infty} \gamma^{l-t}\, \mathbb{E}_{\mathbf{s}_l\sim p, \mathbf{a}_l\sim\pi} \left[r(\mathbf{s}_t, \mathbf{a}_t) + \alpha \mathcal{H}(\pi(\,\cdot\,|\mathbf{s}_t)) | \mathbf{s}_t, \mathbf{a}_t \right] \right]. \tag{14}$$

This objective corresponds to maximizing the discounted expected reward and entropy for future states originating from every state-action tuple $(\mathbf{s}_t, \mathbf{a}_t)$ weighted by its probability ρ_π under the current policy.

B. Proofs

B.1. Lemma 1

Lemma 1 (Soft Policy Evaluation). *Consider the soft Bellman backup operator $\mathcal{T}^\pi$ in Equation 2 and a mapping $Q^0 : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ with $|\mathcal{A}| < \infty$, and define $Q^{k+1} = \mathcal{T}^\pi Q^k$. Then the sequence Q^k will converge to the soft Q-value of π as $k \to \infty$.*

Proof. Define the entropy augmented reward as $r_\pi(\mathbf{s}_t, \mathbf{a}_t) \triangleq r(\mathbf{s}_t, \mathbf{a}_t) + \mathbb{E}_{\mathbf{s}_{t+1}\sim p}\left[\mathcal{H}\left(\pi(\,\cdot\,|\mathbf{s}_{t+1})\right)\right]$ and rewrite the update rule as

$$Q(\mathbf{s}_t, \mathbf{a}_t) \leftarrow r_\pi(\mathbf{s}_t, \mathbf{a}_t) + \gamma\, \mathbb{E}_{\mathbf{s}_{t+1}\sim p, \mathbf{a}_{t+1}\sim\pi}\left[Q(\mathbf{s}_{t+1}, \mathbf{a}_{t+1})\right] \tag{15}$$

and apply the standard convergence results for policy evaluation (Sutton & Barto, 1998). The assumption $|\mathcal{A}| < \infty$ is required to guarantee that the entropy augmented reward is bounded. □

B.2. Lemma 2

Lemma 2 (Soft Policy Improvement). *Let $\pi_{\text{old}} \in \Pi$ and let π_{new} be the optimizer of the minimization problem defined in Equation 4. Then $Q^{\pi_{\text{new}}}(\mathbf{s}_t, \mathbf{a}_t) \geq Q^{\pi_{\text{old}}}(\mathbf{s}_t, \mathbf{a}_t)$ for all $(\mathbf{s}_t, \mathbf{a}_t) \in \mathcal{S} \times \mathcal{A}$ with $|\mathcal{A}| < \infty$.*

Proof. Let $\pi_{\text{old}} \in \Pi$ and let $Q^{\pi_{\text{old}}}$ and $V^{\pi_{\text{old}}}$ be the corresponding soft state-action value and soft state value, and let π_{new} be defined as

$$\begin{aligned} \pi_{\text{new}}(\,\cdot\,|\mathbf{s}_t) &= \arg\min_{\pi'\in\Pi} \mathrm{D_{KL}}\left(\pi'(\,\cdot\,|\mathbf{s}_t)\;\|\;\exp\left(Q^{\pi_{\text{old}}}(\mathbf{s}_t,\,\cdot\,) - \log Z^{\pi_{\text{old}}}(\mathbf{s}_t)\right)\right) \\ &= \arg\min_{\pi'\in\Pi} J_{\pi_{\text{old}}}(\pi'(\,\cdot\,|\mathbf{s}_t)). \end{aligned} \tag{16}$$

It must be the case that $J_{\pi_{\text{old}}}(\pi_{\text{new}}(\,\cdot\,|\mathbf{s}_t)) \leq J_{\pi_{\text{old}}}(\pi_{\text{old}}(\,\cdot\,|\mathbf{s}_t))$, since we can always choose $\pi_{\text{new}} = \pi_{\text{old}} \in \Pi$. Hence

$$\mathbb{E}_{\mathbf{a}_t\sim\pi_{\text{new}}}\left[\log \pi_{\text{new}}(\mathbf{a}_t|\mathbf{s}_t) - Q^{\pi_{\text{old}}}(\mathbf{s}_t, \mathbf{a}_t) + \log Z^{\pi_{\text{old}}}(\mathbf{s}_t)\right] \leq \mathbb{E}_{\mathbf{a}_t\sim\pi_{\text{old}}}\left[\log \pi_{\text{old}}(\mathbf{a}_t|\mathbf{s}_t) - Q^{\pi_{\text{old}}}(\mathbf{s}_t, \mathbf{a}_t) + \log Z^{\pi_{\text{old}}}(\mathbf{s}_t)\right], \tag{17}$$

and since partition function $Z^{\pi_{\text{old}}}$ depends only on the state, the inequality reduces to

$$\mathbb{E}_{\mathbf{a}_t\sim\pi_{\text{new}}}\left[Q^{\pi_{\text{old}}}(\mathbf{s}_t, \mathbf{a}_t) - \log \pi_{\text{new}}(\mathbf{a}_t|\mathbf{s}_t)\right] \geq V^{\pi_{\text{old}}}(\mathbf{s}_t). \tag{18}$$

Next, consider the soft Bellman equation:

$$\begin{aligned} Q^{\pi_{\text{old}}}(\mathbf{s}_t, \mathbf{a}_t) &= r(\mathbf{s}_t, \mathbf{a}_t) + \gamma\, \mathbb{E}_{\mathbf{s}_{t+1}\sim p}\left[V^{\pi_{\text{old}}}(\mathbf{s}_{t+1})\right] \\ &\leq r(\mathbf{s}_t, \mathbf{a}_t) + \gamma\, \mathbb{E}_{\mathbf{s}_{t+1}\sim p}\left[\mathbb{E}_{\mathbf{a}_{t+1}\sim\pi_{\text{new}}}\left[Q^{\pi_{\text{old}}}(\mathbf{s}_{t+1}, \mathbf{a}_{t+1}) - \log \pi_{\text{new}}(\mathbf{a}_{t+1}|\mathbf{s}_{t+1})\right]\right] \\ &\;\;\vdots \\ &\leq Q^{\pi_{\text{new}}}(\mathbf{s}_t, \mathbf{a}_t), \end{aligned} \tag{19}$$

where we have repeatedly expanded $Q^{\pi_{\text{old}}}$ on the RHS by applying the soft Bellman equation and the bound in Equation 18. Convergence to $Q^{\pi_{\text{new}}}$ follows from Lemma 1. □

B.3. Theorem 1

Theorem 1 (Soft Policy Iteration). *Repeated application of soft policy evaluation and soft policy improvement to any* $\pi \in \Pi$ *converges to a policy* π^* *such that* $Q^{\pi^*}(\mathbf{s}_t, \mathbf{a}_t) \geq Q^{\pi}(\mathbf{s}_t, \mathbf{a}_t)$ *for all* $\pi \in \Pi$ *and* $(\mathbf{s}_t, \mathbf{a}_t) \in \mathcal{S} \times \mathcal{A}$*, assuming* $|\mathcal{A}| < \infty$*.*

Proof. Let π_i be the policy at iteration i. By Lemma 2, the sequence Q^{π_i} is monotonically increasing. Since Q^{π} is bounded above for $\pi \in \Pi$ (both the reward and entropy are bounded), the sequence converges to some π^*. We will still need to show that π^* is indeed optimal. At convergence, it must be case that $J_{\pi^*}(\pi^*(\,\cdot\,|\mathbf{s}_t)) < J_{\pi^*}(\pi(\,\cdot\,|\mathbf{s}_t))$ for all $\pi \in \Pi$, $\pi \neq \pi^*$. Using the same iterative argument as in the proof of Lemma 2, we get $Q^{\pi^*}(\mathbf{s}_t, \mathbf{a}_t) > Q^{\pi}(\mathbf{s}_t, \mathbf{a}_t)$ for all $(\mathbf{s}_t, \mathbf{a}_t) \in \mathcal{S} \times \mathcal{A}$, that is, the soft value of any other policy in Π is lower than that of the converged policy. Hence π^* is optimal in Π. $\square$

C. Enforcing Action Bounds

We use an unbounded Gaussian as the action distribution. However, in practice, the actions needs to be bounded to a finite interval. To that end, we apply an invertible squashing function ($\tanh$) to the Gaussian samples, and employ the change of variables formula to compute the likelihoods of the bounded actions. In the other words, let $\mathbf{u} \in \mathbb{R}^D$ be a random variable and $\mu(\mathbf{u}|\mathbf{s})$ the corresponding density with infinite support. Then $\mathbf{a} = \tanh(\mathbf{u})$, where $\tanh$ is applied elementwise, is a random variable with support in $(-1, 1)$ with a density given by

$$\pi(\mathbf{a}|\mathbf{s}) = \mu(\mathbf{u}|\mathbf{s}) \left| \det\left(\frac{\mathrm{d}\mathbf{a}}{\mathrm{d}\mathbf{u}} \right) \right|^{-1}. \tag{20}$$

Since the Jacobian ${}^{\mathrm{d}\mathbf{a}}/_{\mathrm{d}\mathbf{u}} = \mathrm{diag}(1 - \tanh^2(\mathbf{u}))$ is diagonal, the log-likelihood has a simple form

$$\log \pi(\mathbf{a}|\mathbf{s}) = \log \mu(\mathbf{u}|\mathbf{s}) - \sum_{i=1}^{D} \log \left(1 - \tanh^2(u_i)\right), \tag{21}$$

where u_i is the i^{th} element of $\mathbf{u}$.

D. Hyperparameters

Table 1 lists the common SAC parameters used in the comparative evaluation in Figure 1 and Figure 4. Table 2 lists the reward scale parameter that was tuned for each environment.

Table 1. SAC Hyperparameters

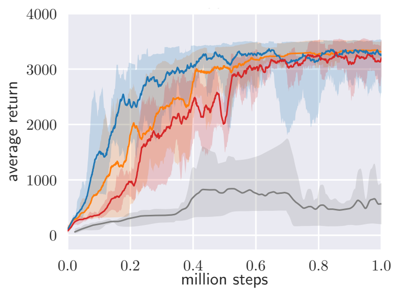
Parameter	Value
Shared	
optimizer	Adam (Kingma & Ba, 2015)
learning rate	$3 \cdot 10^{-4}$
discount (γ)	0.99
replay buffer size	10^6
number of hidden layers (all networks)	2
number of hidden units per layer	256
number of samples per minibatch	256
nonlinearity	ReLU
SAC	
target smoothing coefficient (τ)	0.005
target update interval	1
gradient steps	1
SAC (hard target update)	
target smoothing coefficient (τ)	1
target update interval	1000
gradient steps (except humanoids)	4
gradient steps (humanoids)	1

Table 2. SAC Environment Specific Parameters

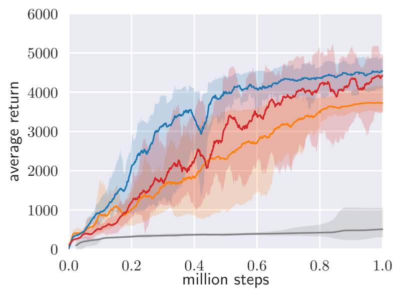
Environment	Action Dimensions	Reward Scale
Hopper-v1	3	5
Walker2d-v1	6	5
HalfCheetah-v1	6	5
Ant-v1	8	5
Humanoid-v1	17	20
Humanoid (rllab)	21	10

E. Additional Baseline Results

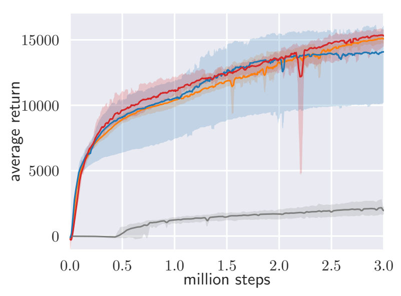
Figure 4 compares SAC to Trust-PCL (Figure 4. Trust-PC fails to solve most of the task within the given number of environment steps, although it can eventually solve the easier tasks (Nachum et al., 2017b) if ran longer. The figure also includes two variants of SAC: a variant that periodically copies the target value network weights directly instead of using exponentially moving average, and a deterministic ablation which assumes a deterministic policy in the value update (Equation 6) and the policy update (Equation 13), and thus strongly resembles DDPG with the exception of having two Q-functions, using hard target updates, not having a separate target actor, and using fixed exploration noise rather than learned. Both of these methods can learn all of the tasks and they perform comparably to SAC on all but Humanoid (rllab) task, on which SAC is the fastest.



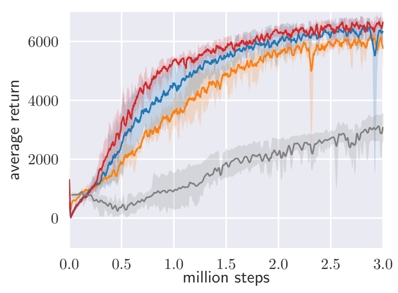
(a) Hopper-v1



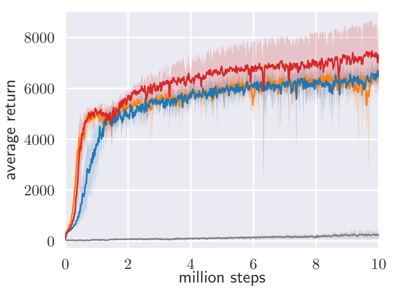
(b) Walker2d-v1



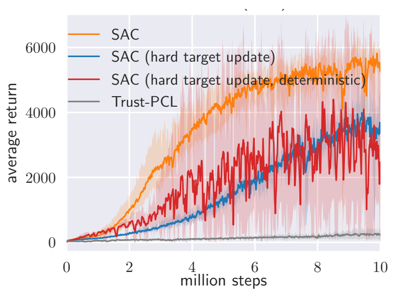
(c) HalfCheetah-v1



(d) Ant-v1



(e) Humanoid-v1



(f) Humanoid (rllab)

Figure 4. Training curves for additional baseline (Trust-PCL) and for two SAC variants. Soft actor-critic with hard target update (blue) differs from standard SAC in that it copies the value function network weights directly every 1000 iterations, instead of using exponentially smoothed average of the weights. The deterministic ablation (red) uses a deterministic policy with fixed Gaussian exploration noise, does not use a value function, drops the entropy terms in the actor and critic function updates, and uses hard target updates for the target Q-functions. It is equivalent to DDPG that uses two Q-functions, hard target updates, and removes the target actor.