We thank the reviewers for their comments. We will fix the typos in the final version.

Comparison with [34]: In general solving the safe RL problem is a harder problem. In order to apply to Reinforcement learning, we need to verify that the Lipschitz condition is satisfied, and also the policy gradient has to be estimated (instead of directly evaluated as in a standard optimization problem). The usage of Actor-Critic algorithm reduces the variance of the sampling a lot (see next paragraph), which is unique to Reinforcement learning.

Actor-Critic improves performance and reduces variance: In general, an actor-critic method helps reduce variance by using a value function instead of Monte-Carlo sampling. Specifically, in Algorithm 1 we need to obtain a sample trajectory and calculate $J^*(\theta)$ and $\nabla_\theta J^*(\theta)$ by Monte-Carlo sampling. This step has a high variance since we need to sample a potentially long trajectory and sum up a lot of random rewards. In contrast, in Algorithm 2, this step is replaced by a value function $V_w^J(s)$, which reduces the variance. We will clarify this in the final version.

About safe exploration: Our algorithm does not guarantee safe exploration during the training phase. Ensuring safety during learning is a more challenging problem. Sometimes even finding a feasible point is not straightforward, otherwise Assumption 3 is not necessary. We will clarify this in final version, and leave safety training to future work.

Reviewer 1: Experiments: we run additional multiple trials to compare our method with Lagrangian method. The following table reports the averaged results with mean and standard deviation. In columns 2 and 3, we compare the minimum value and number of iterations to achieve it. Since for both the methods, the constraint values oscillate above and below D_0 (as shown in Figure 2cd), we also consider an approximate version, where we are satisfied with the result if the objective value exceeds less than 0.2% of the minimum value. Columns 4, 5 of the table report the averaged results for this approximate version. We can see that both methods achieve similar minimum values, but ours requires less number of policy updates, for both minimum and approximate minimum version. We will include this experiment result, and move all the experiment parts to the main text in the final version, since we have an additional page.

	minimum value	# iterations	approximate minimum value	approximate # iterations
Our method	30.689 ± 0.114	2001 ± 1172	30.694 ± 0.114	604.3 ± 722.4
Lagrangian	30.693 ± 0.113	7492 ± 1780	30.699 ± 0.113	5464 ± 2116

Other comments: **(i).** Line 201: yes, it should be “and their derivatives are". **(ii).** Figure 1a/b and Figure 2a/b are two realizations under the same setting. It is safe to ignore Figure 1a/b. **(iii).** Note that we are dealing with nonconvex optimization with a nonconvex constraint. When adding random noise to the iterates, we could escape saddle points and achieve a second-order stationary point. **(iv).** Line 152 closed form: Solving (9) is effectively finding a minimum value of $\overline{D}^{(k)}(\theta)$, and $\alpha = \min_\theta \overline{D}^{(k)}(\theta) - D_0$. Looking at (6) and (8), $\overline{D}^{(k)}(\theta)$ is quadratic and decomposable to each component of θ. Therefore it is minimizing several quadratic functions, and we have closed form solution. **(v).** Section 5 provides an example of concrete application of our approach. The settings are used in experiments. **(vi).** To verify the constraint is satisfied, we could use Monte-Carlo sampling which could be computationally heavy. **(vii).** Even for simple problems, modeling the constraints as negative rewards will lead to very conservative or risky behavior. See [Undurti, Aditya. Planning under uncertainty and constraints for teams of autonomous agents. 2011]. **(viii).** Intuition for the proof: we first show that the surrogate functions converge (Line 218), and then show that the iterates converge to feasible region (Line 545). Using Slater's condition completes the proof.

As defined in Section 5, LQR is a control problem, and hence naturally fits into RL. Standard LQR has closed form solution. One research direction is on algorithm part, to modify the problem so that it no longer has a closed form solution. Then RL algorithm is useful to solve this modified problem, for example in this paper with safety constraint. Another is to understand the convergence of RL algorithms on LQR, see [22] for policy gradient, and [7] for actor-critic.

Reviewer 2: **(i).** L119: we will provide more intuitions and references on the surrogate functions. **(ii).** L137: the definition and reference of policy gradient algorithm is given in L92-93 in background section. We will provide a reference to equation (3) here in the final version. **(iii).** In (5) and (6), τ can be any positive constant and it does not affect the theoretical convergence. **(iv).** Expression for the $J_{truncate}$: if γ is close to 1, the random variable T would be large. Therefore the summation of the reward $\sum_{t=0}^{T} r(s_t, a_t)$ would be large. The term $(1-\gamma)$ is like a normalizer. The intuition is that, if γ is close to 1, then future rewards are important so we need a large T. This is captured by the definition of T in L172. **(v).** Experiments: we will release codes on github and evaluate on other general set of problem in the final version. The paper on cross entropy method [59] does not have codes so we only compare with the standard (and easy to implemented) Lagrangian method.

Assumption 3 is indeed a relatively strong assumption. As discussed in line 222, it is not necessary if we initialize with a feasible point. Moreover, if (in practice) we reach a feasible point in the iterates, then we could view it as an initializer as again Assumption 3 is not necessary. If we could not find a feasible point, then the iterates may converge to an infeasible stationary point of (12). As far as we know, without Assumption 3 we can not rule out this case. In practice, we could initialize with multiple start points, and the convergence is then guaranteed as long as we reach a single feasible point for one of these iterates. For our experiments on LQR, for every single replicate, we could reach a feasible point, and therefore Assumption 3 is not necessary.