We thank all the reviewers for their constructive feedback. Our revision will incorporate all the points detailed below.

R#1: Theoretical advantage over comp-SCGD [8]. In the table below, we compare our bound to [8], which shows that we have better dependency on κ (see our reply to reviewer #6 for how to derive the complexity in special cases).

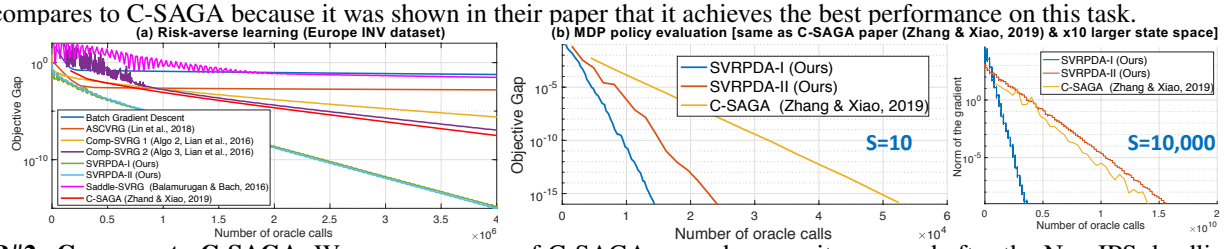
R#2: Bounded gradient assumption. Bounded gradient is commonly assumed for the interior function f_θ in existing compositional optimization works [8, 9, 17, 18] and C-SAGA. The key intuition can be seen from (8): the dual gradients, which are linear combinations of f_θ, would be Lipschitz if f_θ is Lipschitz (guaranteed by bounded f'_θ).

R#2: Novelty of SVRG + primal-dual w.r.t. [1,5]. We have discussed [1,5] in Sec. 6 (see lines 249-250, 252-253). The algorithms in [1] are not for composition optimization but for general saddle-point problems. Applying their algorithms to our saddle-point problem (8) will fail to capture its inherent special structures (e.g., dual decomposition). In contrast, our algorithm fully exploits the dual decomposition and coordinate ascent structures in (8) to develop a much better control variate compared to regular SVRG (see our discussion in Sec. 3.3.). Our new experiments (see Fig-(a) below) shows the superiority of our algorithms over [1] (*Saddle-SVRG*). That is, our work is not a simple combination of SVRG with primal-dual formulation but a novel algorithm that is carefully designed to solve (1). In addition, we believe it is the first work to solve the more general compositional optimization (1) along this direction. Regarding the work [5], it only considers MSPBE cost in policy evaluation, which can be viewed as a special case of (2) with quadratic ϕ_i and linear f_θ. The table below shows that our method has better dependency on κ in this special case.

R#2: Compare to other composition optimization methods. The following table lists their complexity bounds (in number of oracle calls) using our notation for different problem settings. First, ASCVRG [9] does not assume strong convexity and thus cannot achieve linear convergence rate. (Its dependency on κ has been dropped as it was not reported in [9].) Second, none of the algorithms consider the general problem (1) as we did. Instead, they consider its special case (2). Even in the special cases, our algorithm still have better (or comparable) complexity bound than other methods. Note that our complexity for problem (2) is lower than (1) due to its structure (see our reply to reviewer #6).

Methods	SVRPDA-I (Ours)	Comp-SVRG [8]	C-SAGA (minibatch: $\alpha=\frac{2}{3}$; batch-size=1: $\alpha=1$)	MSPBE-SVRG/SAGA [5]	ASCVRG [9]
General: problem (1)	$(n_X n_Y + n_X\kappa)\ln\frac{1}{\epsilon}$				
Special: problem (2)	$(n_X+n_Y+n_X\kappa)\ln\frac{1}{\epsilon}$	$(n_X+n_Y+\kappa^3)\ln\frac{1}{\epsilon}$	$(n_X+n_Y+(n_X+n_Y)^\alpha\kappa)\ln\frac{1}{\epsilon}$		$(n_X+n_Y)\ln\frac{1}{\epsilon}+\frac{1}{\epsilon^3}$
Special: (2) & $n_X=1$	$(n_Y+\kappa)\ln\frac{1}{\epsilon}$	$(n_Y+\kappa^3)\ln\frac{1}{\epsilon}$	$(n_Y+n_Y^\alpha\kappa)\ln\frac{1}{\epsilon}$	$(n_Y+\kappa^2)\ln\frac{1}{\epsilon}$	$n_Y\ln\frac{1}{\epsilon}+\frac{1}{\epsilon^3}$

R#2: Large-scale experiments/more applications. First, risk-averse learning is a real-world benchmark that was widely used by existing compositional optimization works [8, 9, 17, 18]. As per your request, we add a policy evaluation task with state-space size $S = 10$ (used in the C-SAGA paper) and $S = 10^4$ (10 times larger than the biggest one experimented in C-SAGA). All the new results (figures below) further demonstrate the superiority of our algorithm (even over the most recent C-SAGA work), where each baseline has been tuned to its best performance. Fig.(b) only compares to C-SAGA because it was shown in their paper that it achieves the best performance on this task.



R#2: Compare to C-SAGA. We were unaware of C-SAGA paper because it appeared after the NeurIPS deadline. However, we will be happy to include the comparison (Fig.(a)-(b) and the table above). C-SAGA mainly considers the special case of $n_X = 1$ and extends it to (2). It seems that in the case (2), the comparisons would depend on n_X and n_Y. However, since we have derived bounds for a more general problem (1) and then specialize it, we would like to highlight the technical difficulties that may arise in deriving our bounds, and the fact that it may not specialize to the best bound for the simpler problem. We observe in our experiments that despite having much smaller memory requirement, SVRPDA-II has a comparable/better performance with C-SAGA, while SVRPDA-I clearly beats C-SAGA.

R#6: Compare Theorems 1 & 2 to [8]'s bound. Our complexity bound $O((n_X n_Y + n_X\kappa)\ln(\frac{1}{\epsilon}))$ is for solving the general problem (1), while [8] is only considering its special case (2). When applying our algorithm to (2), our complexity could be reduced to $O((n_X + n_Y + n_X\kappa)\ln(\frac{1}{\epsilon}))$ (see the above table). This is because the complexity for evaluating the batch quantities in (11) (see Algorithm 1) can be reduced from $O(n_X n_Y)$ in the general case (1) to $O(n_X + n_Y)$ in the special case (2). To see this, note that f_θ and $n_{Y_i} = n_Y$ become independent of i in (2) and (11). This means that we can factor U_0 in (11) as $U_0 = \frac{1}{n_X n_Y}\sum_{j=0}^{n_Y-1} f'_{\tilde\theta}(y_j)\sum_{i=0}^{n_X} w_i^{(0)}$, where the two sums can be evaluated independently with complexity $O(n_Y)$ and $O(n_X)$, respectively. The other two quantities in (11) need only $O(n_Y)$ due to their independence of i. In this case, our bound is better than that in [8] since $\kappa^2 > n_X$ generally holds.

R#6: Strong convexity & reference by (Du & Hu). We indeed added a small (10^{-6}) L_2-regularizaton in our experiments. But we did observe that the algorithm can still converge linearly even without the strongly convex regularization, similar to what is observed in (Du & Hu) for linear coupling. We will clarify this and add the reference.