Rebuttal Response

Anonymous Authors[1]

1. Reviewer g3Q2 (rating 7 confidence 3)

Weakness 1. The proposed algorithm is very similar to the one from (Hao et al., 2024). Basically, it is simply replacing the inner loop per a SGD step (setting $I = N = 1$ in the `UpdateLower` algorithm). I find that giving a novel name to the algorithm make it hard to navigate the literature. As the contribution is mostly theoretical, I suggest to explicitly stating that this is the same algorithm with $N = I = 1$, and recommend using a similar name (even though it might be hard with the Periodic in the acronym).

A1. Thank you for your insightful question. We would like to emphasize that the SLIP algorithm proposed in this paper is different from the BO-REP algorithm in (Hao et al., 2024) among the following two aspects. First, SLIP only runs SGD for $T_0 = \widetilde{O}(1)$ steps in the warm-start phase, while BO-REP runs epoch-SGD for $T_0 = \widetilde{O}(1/\epsilon^2)$ steps for any given $\epsilon > 0$ in the initialization refinement phase. This indicates that simply setting $I = N = 1$ in the UpdateLower algorithm from (Hao et al., 2024) could not entirely recover the BO-REP algorithm. Second, SLIP is designed in a single loop style, that is, we update the lower-level variable y_t only once per iteration, while BO-REP update y_t for $\widetilde{O}(1/\epsilon^2)$ steps periodically and it is a double loop algorithm. Therefore, we use a different name to highlight the simple and single loop nature of our proposed algorithm.

Weakness 2. The algorithm is said to be optimal, but with respect to the lower bound for single-level problems with unbounded smoothness. While it is highly plausible that the bound is the same for bilevel problems, it is not explicitly shown. In particular, any bilevel problem is associated with a single-level problem for the value function, so the class of single-level problem is larger than the one of bilevel problem, and thus the lower bound for bilevel problem might be smaller. The only work that reported lower bounds for bilevel problems are (Ji & Liang, 2023; Dagréou et al., 2023) but for smooth problems. I think this statement should be toned down unless a lower bound is proven for a class of algorithm matching the one used in this setting.

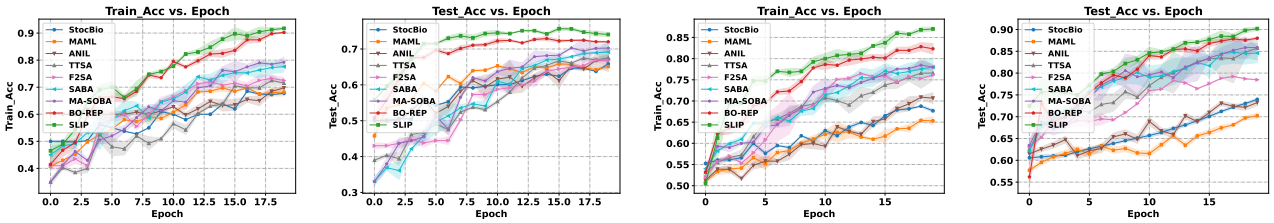
A2. Thanks for your insightful question. Please note that lower bounds in (Ji & Liang, 2023) only work for deterministic and convex/strongly-convex upper-level problems, and the lower bounds in (Dagréou et al., 2023) only work for nonconvex smooth finite-sum objective in stochastic setting, respectively. In contrast, we study general expectation form in stochastic setting, where the upper-level problem is nonconvex and unbounded smooth. Therefore their lower bounds can not be applied in our problem setting.

Now we will establish a $\Omega(1/\epsilon^4)$ lower bound via a simple reduction to the single-level stochastic nonconvex smooth optimization, when the stochastic gradient oracle does not have mean-squared smoothness.

It is shown in (Arjevani et al., 2023) that without mean-squared smoothness assumption, $\Omega(1/\epsilon^4)$ complexity is necessary for finding an ϵ stationary point when using stochastic first-order methods for single-level stochastic nonconvex optimization problems. Note that our problem class is more expressive than the function class considered in (Arjevani et al., 2023) and hence our problem is harder. This is because standard smoothness is a special case of relaxed smoothness and single-level optimization is a special case of bilevel optimization. For example, if we consider an easy case where the upper-level function does not depend on the lower-level problem (e.g., $\Phi(x) = f(x)$ such that Φ is independent of $y^*(x)$) and does not have mean-squared smoothness, then the $\Omega(1/\epsilon^4)$ lower bound in (Arjevani et al., 2023) can be applied in our setting. Therefore the $\widetilde{O}(1/\epsilon^4)$ complexity achieved in this paper is already optimal up to logarithmic factors. For more detailed

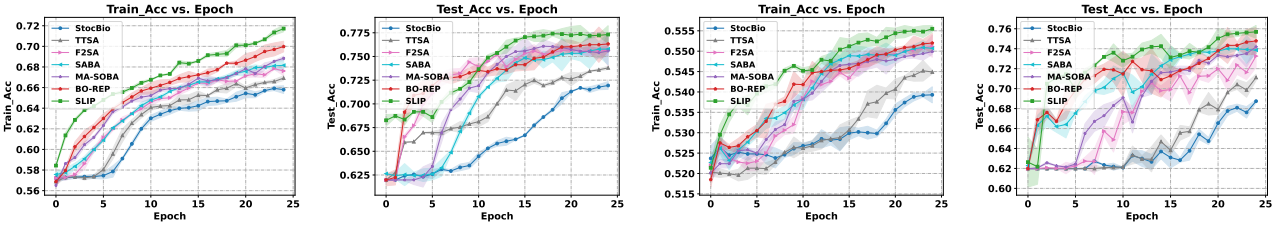
[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.



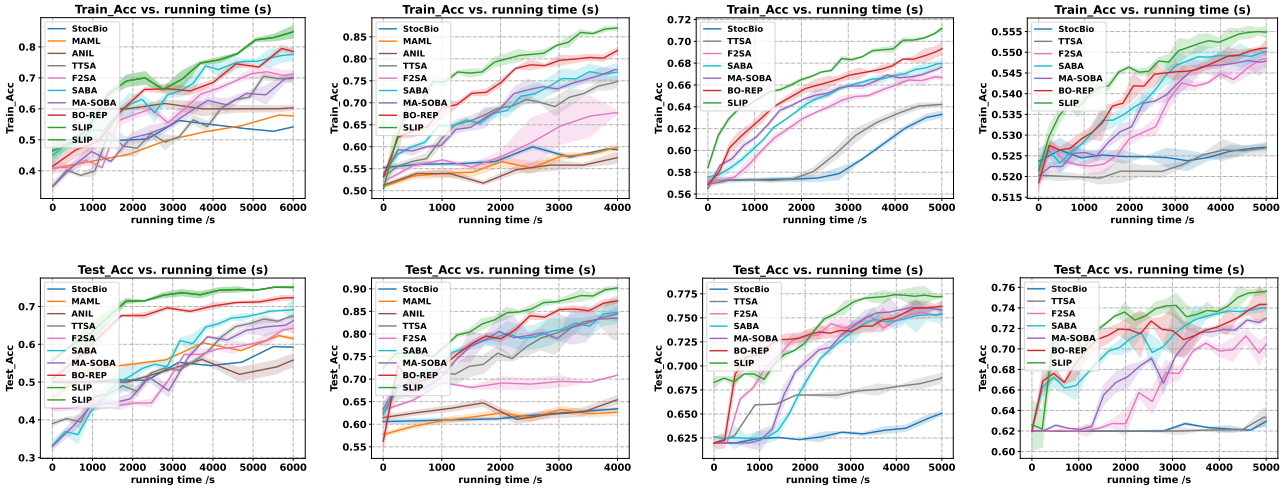
(a) Training accuracy on SNLI (b) Test accuracy on SNLI (c) Training accuracy on ARD (d) Test accuracy on ARD

Figure 1. Comparison with bilevel optimization baselines on Hyper-representation. Figure (a) and (b) are the results in the SNLI dataset. Figures (c) and (d) are the results of the Amazon Review Dataset (ARD).



(a) Training acc with $p = 0.2$ (b) Test acc with $p = 0.2$ (c) Training acc with $p = 0.4$ (d) Test acc with $p = 0.4$

Figure 2. Comparison with bilevel optimization baselinses on data hyper-cleaning. Figure (a), (b) are the results with the corruption rate $p = 0.2$. Figure (c), (d) are the results with the corruption rate $p = 0.4$.



(a) Hyper-representation (SNLI) (b) Hyper-representation (ARD) (c) Data Hyper-cleaning (p=0.2) (d) Data Hyper-cleaning (p=0.4)

Figure 3. Comparision on running time. (a) Results of Hyper-representation on SNLI dataset. (b) Results of Hyper-representation on Amazon Review Dataset (ARD). (c), (d) Results of data Hyper-cleaning on Sentiment140 with corruption rate $p = 0.2$ and $p = 0.4$.

discussions, we refer the reviewer to Appendix L in (Hao et al., 2024).

Weakness 3. For Figures 1, 2 and 3, the authors do not report the error bar averaged over multiple repetitions, which is standard practice when evaluating stochastic algorithms. From the look of it, the performance in iteration and in time from this algorithm and `BO-REP` are similar?

A3. Thanks for your kind reminder. We rerun all experiments for 5 times using random seeds $\{0, 1, 2, 3, 4\}$, respectively. We present the average accuracy curves with standard deviation for both the training and test sets in Figures 1, 2, and 3. In the hyper-representation task, SLIP consistently exhibits low standard deviation and outperforms other baselines through the training and test process significantly. In the data hyper-cleaning task, the standard deviation for all algorithms increases as the corruption rate p rises (i.e., from $p = 0.2$ to $p = 0.4$). Despite this, SLIP still performs better than the others in

terms of average training and test accuracy. As can be seen from the figures, the averaged performance of SLIP surpasses BO-REP both in epoch and in time, instead of being similar. Specifically, SLIP outperforms BO-REP in both training and test accuracy within the same number of epochs, and also achieves specified accuracy level faster than BO-REP.

Minor issue 1. For the second inequality of Assumption 3.1, the Lipschitz constant should be with y.

A4. Thank you for catching this! It is indeed a typo in our original manuscript. We will fix this issue (replace $L_{x,0}, L_{x,1}$ with $L_{y,0}, L_{y,1}$ for the second inequality in Assumption 3.1) in the new version.

Minor issue 2. The authors in SABA (Dagréou et al., 2022) explicitly suggest to use the same learning rate for inner problem and for the linear system, which is not what is reported in Appendix E.1.

A5. Thanks for your suggestion and we've fixed this issue. For hyper-representation on ARD dataset, we reset the learning rate to $\gamma = 0.05$ for linear system updates, which is adopted for the lower-level problem. For data hyper-cleaning on Sentiment140 dataset, we reset $\gamma = 0.1$, which is also the same as the lower-level problem. We rerun SABA on ARD dataset of hyper-representation and Sentiment140 of data hyper-cleaning, the training and test accuracy results are shown in Figure 1, 2, 3. The new run shows only a slight increase on accuracy, with improvements of 1.70% and 1.10% in averaged final training accuracy and test accuracy on ARD dataset, 0.14% and 0.31% on Sentiment140 when $p = 0.2$, 0.05% and 0.10% on Sentiment140 when $p = 0.4$, respectively.

Question 1. It is unclear why single loop algorithms require more tuning efforts than double loop ones? In the numerical experiments, there is not really more parameters for the double loops algorithms compared to the single loop ones. Could the author comment on this?

A6. Thanks for your insightful question. For double loop algorithm like BO-REP in (Hao et al., 2024), they need to tune several additional hyperparameters: the radius R of the projection ball, the update period I and the number of iterations N in the UpdateLower subroutine (Algorithm 2 in (Hao et al., 2024)) for lower-level variable y_t updates. However, none of these hyperparameters appear in our proposed single loop algorithm SLIP, which saves significant tuning efforts.

2. Reviewer P6Br (rating 4 confidence 3)

Minor issues.

A1. Thanks a lot for catching these issues! We will fix those issues in the new version of the submission.

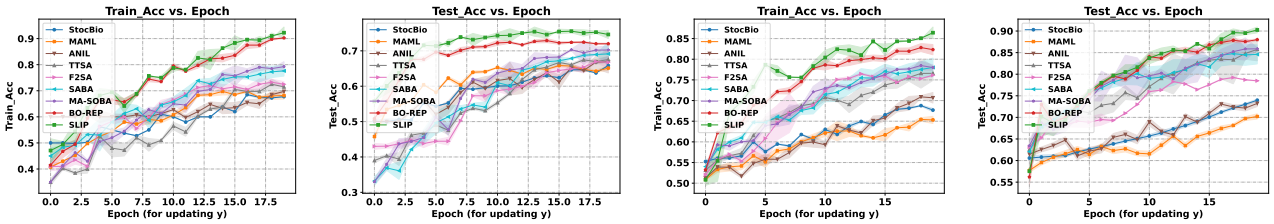
Code issue 1. In the provided code, the implementation of the competitor methods is missing (I only found the implementation of SLIP). This is an issue for reproducibility purposes.

A2. Thank you for pointing out this issue. We have added the implementation of other baseline methods. Please refer to `https://anonymous.4open.science/r/SLIP` for the complete code (see "methods" folder).

Code issue 2. For the data-cleaning experiment, the default parameter for `inner_update_step` is set to 64. If I understand correctly the code, it corresponds to the number of inner iterations. However, the SLIP algorithm is a single-loop method and, in the README, it seems to be launched using this default parameter.

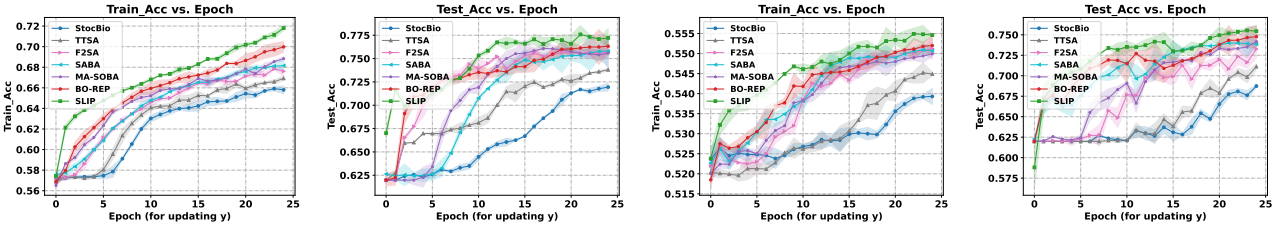
A3. We are sorry for the ambiguity. In our implementation, the hyperparameter "inner_update_step" needs to be specified before running the experiments, and for the SLIP algorithm, it was set to 1 in our experiments in the submission. To make it more clear, we specify the hyperparameters setting clearly in the new version of the code, please refer to README at `https://anonymous.4open.science/r/SLIP` for more details.

Question 1. In bilevel optimization, the notion of "epoch" is a bit fuzzy since there are the training set and the validation set in which we sample independently from each other. In the plotted results for hyper-representation learning, what is an epoch? A full pass over the training set? A full pass over the validation set? Also, for the results of SLIP, is the warm-start step included in the first epoch? I think these points should be clarified in the paper.



(a) Training accuracy on SNLI (b) Test accuracy on SNLI (c) Training accuracy on ARD (d) Test accuracy on ARD

Figure 4. Comparison with bilevel optimization baselines on Hyper-representation. Figure (a) and (b) are the results in the SNLI dataset. Figures (c) and (d) are the results of the Amazon Review Dataset (ARD).



(a) Training acc with $p = 0.2$ (b) Test acc with $p = 0.2$ (c) Training acc with $p = 0.4$ (d) Test acc with $p = 0.4$

Figure 5. Comparison with bilevel optimization baselinses on data hyper-cleaning. Figure (a), (b) are the results with the corruption rate $p = 0.2$. Figure (c), (d) are the results with the corruption rate $p = 0.4$.

A4. Thank you for your insightful question. In the hyper-representation task, an epoch means a full pass over the validation set. We actually partition the original training set equally into the training set and the validation set, where the validation set is for upper-level variable (x) update and the training set is for lower-level variable (y) update. For the SLIP algorithm, the warm-start phase is implemented only in the first epoch, which means that there are 3 more iterations for y updates than x updates in the first epoch when passing over the whole validation set.

To provide a comprehensive comparison, we rerun SLIP algorithm where an epoch means a full pass over the training set (i.e., updates for y), then there are 3 fewer iterations of the x updates than the original run in our paper (to keep the number of y updates the same as other single-loop baselines). We report the new results in Figure 4 and 5, where x-axis means "epoch for updating y". The performance of SLIP shows negligible differences between the original runs (Figure 1, 2) and the new runs (Figure 4, 5). It indicates that our algorithm SLIP is still empirically better than other baselines. We will add these comparisons in the revised version.

Question 2. What is the number of warm-start iterations in the experiments?

A5. We set the number of warm-start iterations $T_0 = 3$ in the experiments. We actually included this detail in Appendix E.1 (line 1961-1962), "For SLIP, the number of initial steps (i.e., number of warm-start iterations) for lower-level updates is set to 3".

3. Reviewer F4oW (rating 5 confidence 3)

Weakness 1. It would be better if the authors provided a table summarizing the theoretical results of all the algorithms in the experiments.

A1. Thanks for your suggestion. We've added a table to summarize the theoretical results of popular and commonly used bilevel optimization methods. Please see Table 1 for details.

Weakness 2. The novelty is insignificant as the proposed algorithm closely follows the algorithm in (Hao et al., 2024) but only changes the update rule of y_t.

Table 1. Comparison of stochastic bilevel optimization algorithms in the nonconvex-strongly-convex setting under different smoothness assumptions on f and g. The oracle complexity stands for the number of oracle calls to stochastic gradients and stochastic Hessian/Jacobian-vector products to find an ϵ-stationary point. $\mathcal{C}_L^{a,k}$ denotes a-times differentiability with Lipschitz k-th order derivatives. "SC" means "strongly-convex". $\widetilde{O}(\cdot)$ compresses logarithmic factors of $1/\epsilon$ and $1/\delta$, where $\delta \in (0,1)$ denotes the failure probability.

Method [1]	Loop Style	Stochastic Setting	Oracle Complexity	Upper-Level f	Lower-Level g	Batch Size
BSA (Ghadimi & Wang, 2018)	Double	General expectation	$\widetilde{O}(\epsilon^{-6})$	$\mathcal{C}_L^{1,1}$	SC and $\mathcal{C}_L^{2,2}$	$\widetilde{O}(1)$
StocBio (Ji et al., 2021)	Double	General expectation	$\widetilde{O}(\epsilon^{-4})$	$\mathcal{C}_L^{1,1}$	SC and $\mathcal{C}_L^{2,2}$	$\widetilde{O}(\epsilon^{-2})$
AmIGO (Arbel & Mairal, 2021)	Double	General expectation	$\widetilde{O}(\epsilon^{-4})$	$\mathcal{C}_L^{1,1}$	SC and $\mathcal{C}_L^{2,2}$	$O(\epsilon^{-2})$
ALSET (Chen et al., 2021)	Double / Single [2]	General expectation	$O(\epsilon^{-4})$	$\mathcal{C}_L^{1,1}$	SC and $\mathcal{C}_L^{2,2}$	$O(1)$
TTSA (Hong et al., 2023)	Single	General expectation	$\widetilde{O}(\epsilon^{-5})$	$\mathcal{C}_L^{1,1}$	SC and $\mathcal{C}_L^{2,2}$	$\widetilde{O}(1)$
F^{2}SA (Kwon et al., 2023)	Single	General expectation	$O(\epsilon^{-7})$	$\mathcal{C}_L^{1,1}$	SC and $\mathcal{C}_L^{2,2}$	$O(1)$
SOBA (Dagréou et al., 2022)	Single	Finite sum	$O(\epsilon^{-4})$	$\mathcal{C}_L^{2,2}$	SC and $\mathcal{C}_L^{3,3}$	$O(1)$
SABA (Dagréou et al., 2022)	Single	Finite sum	$O(N^{4/3}\epsilon^{-2})$ [3]	$\mathcal{C}_L^{2,2}$	SC and $\mathcal{C}_L^{3,3}$	$O(1)$
MA-SOBA (Chen et al., 2023)	Single	General expectation	$O(\epsilon^{-4})$	$\mathcal{C}_L^{1,1}$	SC and $\mathcal{C}_L^{2,2}$	$O(1)$
BO-REP (Hao et al., 2024)	Double	General expectation	$\widetilde{O}(\epsilon^{-4})$	$(L_{x,0}, L_{x,1}, L_{y,0}, L_{y,1})$-smooth	SC and $\mathcal{C}_L^{2,2}$	$O(1)$
SLIP (This work)	Single	General expectation	$\widetilde{O}(\epsilon^{-4})$	$(L_{x,0}, L_{x,1}, L_{y,0}, L_{y,1})$-smooth	SC and $\mathcal{C}_L^{2,2}$	$O(1)$

A2. Thank you for your insightful question. We would like to emphasize that the SLIP algorithm proposed in this paper is different from the BO-REP algorithm in (Hao et al., 2024) among multiple aspects. First, SLIP only runs SGD for $T_0 = \widetilde{O}(1)$ steps in the warm-start phase, while BO-REP runs epoch-SGD for $T_0 = \widetilde{O}(1/\epsilon^2)$ steps for any given small $\epsilon > 0$ in the initialization refinement phase. Second, SLIP is designed in a single loop style, that is, we update the lower-level variable y_t only once per iteration, while BO-REP update y_t for $\widetilde{O}(1/\epsilon^2)$ steps periodically and it is a double loop algorithm. Third, the theoretical analysis is more challenging due to the single loop nature of the proposed algorithm, since it typically makes small progress on each step and cannot guarantee good estimates at every iteration, and thus a refined characterization and control for the lower-level variable y_t is needed. In contrast, the key idea in (Hao et al., 2024) is to obtain an accurate estimator for the optimal lower-level solution at every iteration, which provides much more convenience for convergence analysis.

Question 1. Assumption 4.2 is very strong. The gradient and hessian noise are not uniformly bounded in most applications. It would be better to make it milder.

A3. Assumption 4.2 is a common technical assumption to establish the high probability guarantee. In fact, almost surely bounded noise assumption (i.e., the noise is bounded with probability 1) is widely used in single-level optimization literature, see e.g., Assumption 5 in (Zhang et al., 2020b), Assumption 2.4 in (Zhang et al., 2020a), Assumption 1(iii) in (Liu et al., 2022), Assumption 3 in (Crawshaw et al., 2022); and sub-Gaussian noise assumption is also made in (Cutler et al., 2023) for minimizing strongly convex functions under distributional drift. Our problem setting is more challenging because their

[1] We omit the comparison with variance reduction-based methods (except for SABA (Dagréou et al., 2022)) that may achieve $\widetilde{O}(\epsilon^{-3})$ complexity under additional mean-squared smoothness assumptions on both upper-level and lower-level problems, e.g., VRBO, MRBO (Yang et al., 2021); SUSTAIN (Khanduri et al., 2021); SVRB (Guo et al., 2021); or under the finite sum setting, e.g., SRBA (Dagréou et al., 2023).

[2] ALSET can converge without the need for double loops, but at the cost of a worse dependence on $\kappa := l_{g,1}/\mu$ in oracle complexity.

[3] SABA (Dagréou et al., 2022) studies finite-sum problem and adapts variance reduction technique SAGA (Defazio et al., 2014), here $N = m + n$ denotes the total number of samples.

goal is to optimize a single-level (relaxed smooth) function, while our work is for bilevel optimization under unbounded smoothness.

Specifically, Assumption 4.2 is crucial for deriving Lemma D.6 in our current analysis, as it necessitates the application of Lemma D.5, which in turn requires the almost surely bounded noise condition.

Question 2. The authors stated Algorithm 1 as SGD with distribution drift. I am wondering what differentiates it from SGD.

A4. Thanks for your insightful question. The main difference comes from whether the objective function is independent of time t or not, although they both perform SGD updates. Typically, the objection function we aim to minimize is fixed and independent of time t, while for single loop algorithms in the bilevel optimization problem, the lower-level objective function is actually evolving in time (in other words, the function is indexed by time t). To be more specific, at time t, the lower-level problem can be written as

$$y_t^* = \arg\min_{y \in \mathbb{R}^{d_y}} \psi_t(y) \coloneqq g(x_t, y),$$

which is so called stochastic optimization under distribution drift per (Cutler et al., 2023). For Algorithm 1, we state it as SGD with distribution drift since $\tilde{x}_t$ may change for each iteration t and thus the objective function is indexed by time t. It reduces to SGD if $\tilde{x}_t = x_0$ for any t, where x_0 is any fixed point in $\mathbb{R}^{d_x}$. Moreover, the convergence analysis for SGD (where the objective is time-independent) and SGD with distribution drift (where the objective is time-dependent) under the stochastic optimization setting is quite different.

References

Arbel, M. and Mairal, J. Amortized implicit differentiation for stochastic bilevel optimization. *arXiv preprint arXiv:2111.14580*, 2021.

Arjevani, Y., Carmon, Y., Duchi, J. C., Foster, D. J., Srebro, N., and Woodworth, B. Lower bounds for non-convex stochastic optimization. *Mathematical Programming*, 199(1-2):165–214, 2023.

Chen, T., Sun, Y., and Yin, W. Closing the gap: Tighter analysis of alternating stochastic gradient methods for bilevel problems. *Advances in Neural Information Processing Systems*, 34:25294–25307, 2021.

Chen, X., Xiao, T., and Balasubramanian, K. Optimal algorithms for stochastic bilevel optimization under relaxed smoothness conditions. *arXiv preprint arXiv:2306.12067*, 2023.

Crawshaw, M., Liu, M., Orabona, F., Zhang, W., and Zhuang, Z. Robustness to unbounded smoothness of generalized signsgd. *Advances in neural information processing systems*, 2022.

Cutler, J., Drusvyatskiy, D., and Harchaoui, Z. Stochastic optimization under distributional drift. *Journal of Machine Learning Research*, 24(147):1–56, 2023.

Dagréou, M., Ablin, P., Vaiter, S., and Moreau, T. A framework for bilevel optimization that enables stochastic and global variance reduction algorithms. *Advances in Neural Information Processing Systems*, 35:26698–26710, 2022.

Dagréou, M., Moreau, T., Vaiter, S., and Ablin, P. A lower bound and a near-optimal algorithm for bilevel empirical risk minimization. *arXiv preprint arXiv:2302.08766*, 2023.

Defazio, A., Bach, F., and Lacoste-Julien, S. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in Neural Information Processing Systems*, pp. 1646–1654, 2014.

Ghadimi, S. and Wang, M. Approximation methods for bilevel programming. *arXiv preprint arXiv:1802.02246*, 2018.

Guo, Z., Hu, Q., Zhang, L., and Yang, T. Randomized stochastic variance-reduced methods for multi-task stochastic bilevel optimization. *arXiv preprint arXiv:2105.02266*, 2021.

Hao, J., Gong, X., and Liu, M. Bilevel optimization under unbounded smoothness: A new algorithm and convergence analysis. In *The Twelfth International Conference on Learning Representations*, 2024.

Hong, M., Wai, H.-T., Wang, Z., and Yang, Z. A two-timescale stochastic algorithm framework for bilevel optimization: Complexity analysis and application to actor-critic. *SIAM Journal on Optimization*, 33(1):147–180, 2023.

Ji, K. and Liang, Y. Lower bounds and accelerated algorithms for bilevel optimization. *Journal of Machine Learning Research*, 24(22):1–56, 2023. URL http://jmlr.org/papers/v24/21-0949.html.

Ji, K., Yang, J., and Liang, Y. Bilevel optimization: Convergence analysis and enhanced design. In *International conference on machine learning*, pp. 4882–4892. PMLR, 2021.

Khanduri, P., Zeng, S., Hong, M., Wai, H.-T., Wang, Z., and Yang, Z. A near-optimal algorithm for stochastic bilevel optimization via double-momentum. *Advances in neural information processing systems*, 34:30271–30283, 2021.

Kwon, J., Kwon, D., Wright, S., and Nowak, R. D. A fully first-order method for stochastic bilevel optimization. In *International Conference on Machine Learning*, pp. 18083–18113. PMLR, 2023.

Liu, M., Zhuang, Z., Lei, Y., and Liao, C. A communication-efficient distributed gradient clipping algorithm for training deep neural networks. *Advances in Neural Information Processing Systems*, 35:26204–26217, 2022.

Yang, J., Ji, K., and Liang, Y. Provably faster algorithms for bilevel optimization. *Advances in Neural Information Processing Systems*, 34:13670–13682, 2021.

Zhang, B., Jin, J., Fang, C., and Wang, L. Improved analysis of clipping algorithms for non-convex optimization. *Advances in Neural Information Processing Systems*, 2020a.

Zhang, J., He, T., Sra, S., and Jadbabaie, A. Why gradient clipping accelerates training: A theoretical justification for adaptivity. In *International Conference on Learning Representations*, 2020b.