
Rebuttal Response

Anonymous Authors¹

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 $T_0 = \tilde{O}(1)$ steps in the warm-start phase, while BO-REP runs epoch-SGD for $T_0 = \tilde{O}(1/\epsilon^2)$ 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 $\tilde{O}(1/\epsilon^2)$ 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.

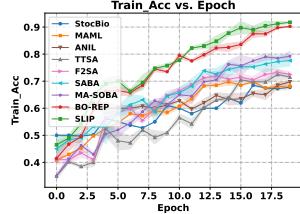
Weakness 2. The algorithm is said to be optimal, but with respect to the lower bound for single-level problems with unbounded smoothness. While it is highly plausible that the bound is the same for bilevel problems, it is not explicitly shown. In particular, any bilevel problem is associated with a single-level 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.

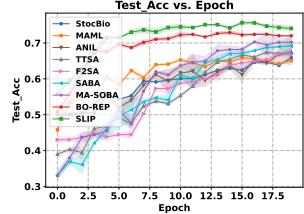
Now we will establish a $\Omega(1/\epsilon^4)$ lower bound via a simple reduction to the single-level stochastic nonconvex smooth optimization, when the stochastic gradient oracle does not have mean-squared smoothness.

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 $\tilde{O}(1/\epsilon^4)$ complexity achieved in this paper is already optimal up to logarithmic factors. For more detailed

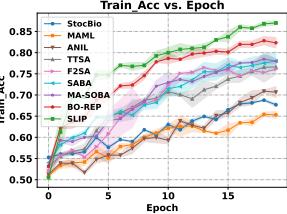
¹Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.



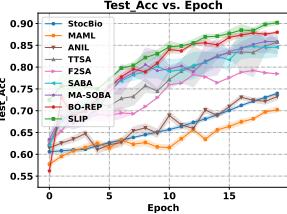
(a) Training accuracy on SNLI



(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).

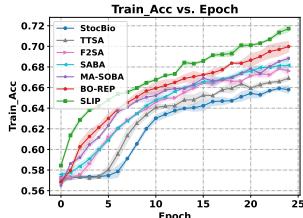
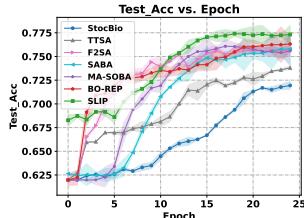
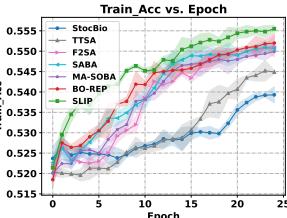
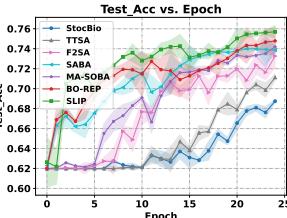
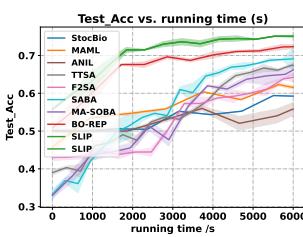
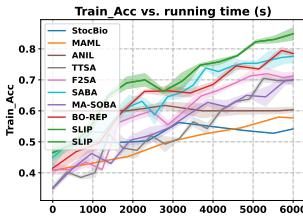
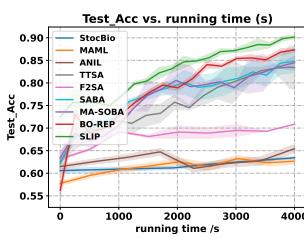
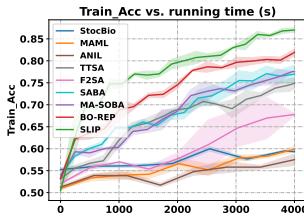
(a) Training acc with $p = 0.2$ (b) Test acc with $p = 0.2$ (c) Training acc with $p = 0.4$ (d) Test acc with $p = 0.4$

Figure 2. Comparison with bilevel optimization baselines 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)

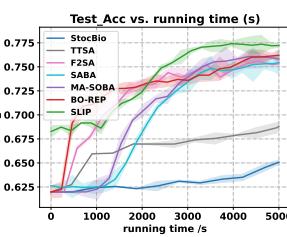
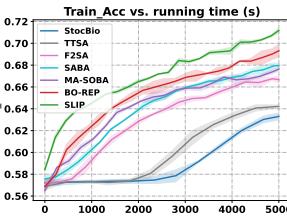
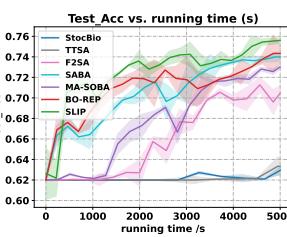
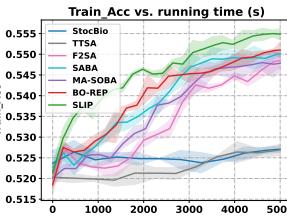
(c) Data Hyper-cleaning ($p=0.2$)(d) Data Hyper-cleaning ($p=0.4$)

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

110 terms of average training and test accuracy. As can be seen from the figures, the averaged performance of SLIP surpasses
 111 BO-REP both in epoch and in time, instead of being similar. Specifically, SLIP outperforms BO-REP in both training and
 112 test accuracy within the same number of epochs, and also achieves specified accuracy level faster than BO-REP.
 113

114 **Minor issue 1.** For the second inequality of Assumption 3.1, the Lipschitz constant should be with y .
 115

116 **A4.** Thank you for catching this! It is indeed a typo in our original manuscript. We will fix this issue (replace $L_{x,0}, L_{x,1}$
 117 with $L_{y,0}, L_{y,1}$ for the second inequality in Assumption 3.1) in the new version.
 118

119 **Minor issue 2.** The authors in SABA (Dagréou et al., 2022) explicitly suggest to use the same learning rate for inner
 120 problem and for the linear system, which is not what is reported in Appendix E.1.
 121

122 **A5.** Thanks for your suggestion and we've fixed this issue. For hyper-representation on ARD dataset, we reset the learning
 123 rate to $\gamma = 0.05$ for linear system updates, which is adopted for the lower-level problem. For data hyper-cleaning on
 124 Sentiment140 dataset, we reset $\gamma = 0.1$, which is also the same as the lower-level problem. We rerun SABA on ARD
 125 dataset of hyper-representation and Sentiment140 of data hyper-cleaning, the training and test accuracy results are shown
 126 in Figures 1, 2, and 3. The new run shows only a slight increase on accuracy, with improvements of 1.70% and 1.10%
 127 in averaged final training accuracy and test accuracy on ARD dataset, 0.14% and 0.31% on Sentiment140 when $p = 0.2$,
 128 0.05% and 0.10% on Sentiment140 when $p = 0.4$, respectively.
 129

130 **Question 1.** It is unclear why single loop algorithms require more tuning efforts than double loop ones? In the numerical
 131 experiments, there is not really more parameters for the double loops algorithms compared to the single loop ones. Could
 132 the author comment on this?
 133

134 **A6.** Thanks for your insightful question. For double loop algorithm like BO-REP in (Hao et al., 2024), they need to tune
 135 several additional hyperparameters: the radius R of the projection ball, the update period I and the number of iterations N
 136 in the UpdateLower subroutine (Algorithm 2 in (Hao et al., 2024)) for lower-level variable y_t updates. However, none of
 137 these hyperparameters appear in our proposed single loop algorithm SLIP, which saves significant tuning efforts.
 138

140 2. Reviewer P6Br (rating 4 confidence 3)

141 **Presentation and minor issues.**

143 **A1.** Thanks a lot for catching these issues! We will rearrange equation layout to make the format neater, reorganize Lemma
 144 4.7 to improve its readability and fix other minor issues in the new version of the submission.
 145

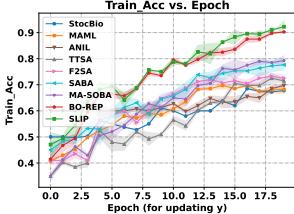
147 **Code issue 1.** In the provided code, the implementation of the competitor methods is missing (I only found the implemen-
 148 tation of SLIP). This is an issue for reproducibility purposes.
 149

150 **A2.** Thank you for pointing out this issue. We have added the implementation of other baseline methods. Please refer to
 151 <https://anonymous.4open.science/r/SLIP> for the complete code (see “methods” folder).
 152

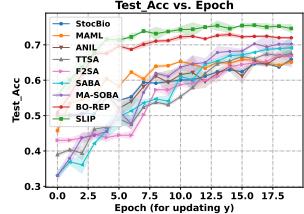
153 **Code issue 2.** For the data-cleaning experiment, the default parameter for `inner_update_step` is set to 64. If I
 154 understand correctly the code, it corresponds to the number of inner iterations. However, the SLIP algorithm is a single-loop
 155 method and, in the README, it seems to be launched using this default parameter.
 156

157 **A3.** We are sorry for the ambiguity. In our implementation, the hyperparameter “`inner_update_step`” needs to be specified
 158 before running the experiments, and for the SLIP algorithm, it was set to 1 in our experiments in the submission. To make
 159 it more clear, we specify the hyperparameters setting clearly in the new version of the code, please refer to README at
 160 <https://anonymous.4open.science/r/SLIP> for more details.
 161

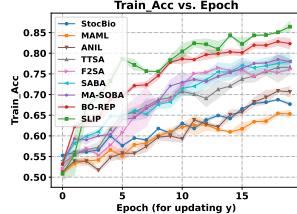
162 **Question 1.** In bilevel optimization, the notion of “epoch” is a bit fuzzy since there are the training set and the validation
 163 set in which we sample independently from each other. In the plotted results for hyper-representation learning, what is an
 164



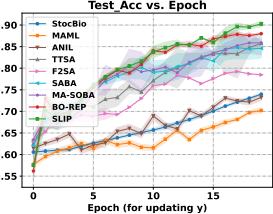
(a) Training accuracy on SNLI



(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).

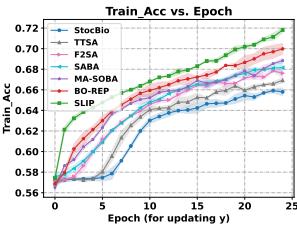
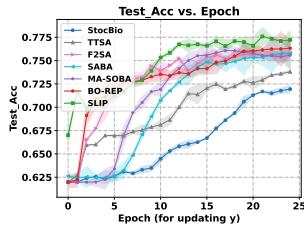
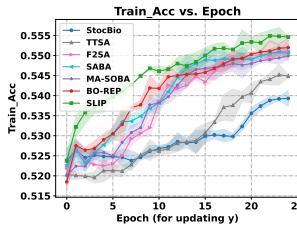
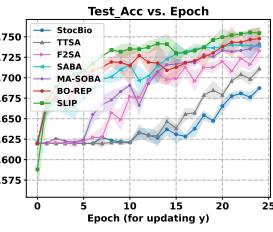
(a) Training acc with $p = 0.2$ (b) Test acc with $p = 0.2$ (c) Training acc with $p = 0.4$ (d) Test acc with $p = 0.4$

Figure 5. Comparison with bilevel optimization baselines 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 $p = 0.4$.

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.

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.

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”. $\tilde{O}(\cdot)$ 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 | $\tilde{O}(\epsilon^{-6})$ | $\mathcal{C}_L^{1,1}$ | SC and $\mathcal{C}_L^{2,2}$ | $\tilde{O}(1)$ |
| StocBio (Ji et al., 2021) | Double | General expectation | $\tilde{O}(\epsilon^{-4})$ | $\mathcal{C}_L^{1,1}$ | SC and $\mathcal{C}_L^{2,2}$ | $\tilde{O}(\epsilon^{-2})$ |
| AmIGO (Arbel & Mairal, 2021) | Double | General expectation | $\tilde{O}(\epsilon^{-4})$ | $\mathcal{C}_L^{1,1}$ | 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 | $\tilde{O}(\epsilon^{-5})$ | $\mathcal{C}_L^{1,1}$ | SC and $\mathcal{C}_L^{2,2}$ | $\tilde{O}(1)$ |
| 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 | $\tilde{O}(\epsilon^{-4})$ | $(L_{x,0}, L_{x,1}, L_{y,0}, L_{y,1})$ -smooth | SC and $\mathcal{C}_L^{2,2}$ | $O(1)$ |
| SLIP (This work) | Single | General expectation | $\tilde{O}(\epsilon^{-4})$ | $(L_{x,0}, L_{x,1}, L_{y,0}, L_{y,1})$ -smooth | SC and $\mathcal{C}_L^{2,2}$ | $O(1)$ |

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 .

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 $T_0 = \tilde{O}(1)$ steps in the warm-start phase, while BO-REP runs epoch-SGD for $T_0 = \tilde{O}(1/\epsilon^2)$ 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 $\tilde{O}(1/\epsilon^2)$ 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,

¹We omit the comparison with variance reduction-based methods (except for SABA (Dagréou et al., 2022)) that may achieve $\tilde{O}(\epsilon^{-3})$ 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).

²ALSET 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.

³SABA (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.

275 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.,
 276 2022), Assumption 3 in (Crawshaw et al., 2022); and sub-Gaussian noise assumption is also made in (Cutler et al., 2023)
 277 for minimizing strongly convex functions under distributional drift. Our problem setting is more challenging because their
 278 goal is to optimize a single-level (relaxed smooth) function, while our work is for bilevel optimization under unbounded
 279 smoothness.

280 Specifically, Assumption 4.2 is crucial for deriving Lemma D.6 in our current analysis, as it necessitates the application of
 281 Lemma D.5, which in turn requires the almost surely bounded noise condition.
 282

283 **Question 2.** The authors stated Algorithm 1 as SGD with distribution drift. I am wondering what differentiates it from
 284 SGD.
 285

286 **A4.** Thanks for your insightful question. The main difference comes from whether the objective function is independent
 287 of time t or not, although they both perform SGD updates. Typically, the objection function we aim to minimize is fixed
 288 and independent of time t , while for single loop algorithms in the bilevel optimization problem, the lower-level objective
 289 function is actually evolving in time (in other words, the function is indexed by time t). To be more specific, at time t , the
 290 lower-level problem can be written as
 291

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

292 which is so called stochastic optimization under distribution drift per (Cutler et al., 2023). For Algorithm 1, we state it as
 293 SGD with distribution drift since \tilde{x}_t may change for each iteration t and thus the objective function is indexed by time t .
 294 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
 295 (where the objective is time-independent) and SGD with distribution drift (where the objective is time-dependent) under the
 296 stochastic optimization setting is quite different.
 297

298
 299
 300
 301
 302
 303
 304
 305
 306
 307
 308
 309
 310
 311
 312
 313
 314
 315
 316
 317
 318
 319
 320
 321
 322
 323
 324
 325
 326
 327
 328
 329

References

- 330 Arbel, M. and Mairal, J. Amortized implicit differentiation for stochastic bilevel optimization. *arXiv preprint*
 331 *arXiv:2111.14580*, 2021.
- 332 Arjevani, Y., Carmon, Y., Duchi, J. C., Foster, D. J., Srebro, N., and Woodworth, B. Lower bounds for non-convex stochastic
 333 optimization. *Mathematical Programming*, 199(1-2):165–214, 2023.
- 334 Chen, T., Sun, Y., and Yin, W. Closing the gap: Tighter analysis of alternating stochastic gradient methods for bilevel
 335 problems. *Advances in Neural Information Processing Systems*, 34:25294–25307, 2021.
- 336 Chen, X., Xiao, T., and Balasubramanian, K. Optimal algorithms for stochastic bilevel optimization under relaxed
 337 smoothness conditions. *arXiv preprint arXiv:2306.12067*, 2023.
- 338 Crawshaw, M., Liu, M., Orabona, F., Zhang, W., and Zhuang, Z. Robustness to unbounded smoothness of generalized
 339 signsgd. *Advances in neural information processing systems*, 2022.
- 340 Cutler, J., Drusvyatskiy, D., and Harchaoui, Z. Stochastic optimization under distributional drift. *Journal of Machine*
 341 *Learning Research*, 24(147):1–56, 2023.
- 342 Dagréou, M., Ablin, P., Vaiter, S., and Moreau, T. A framework for bilevel optimization that enables stochastic and global
 343 variance reduction algorithms. *Advances in Neural Information Processing Systems*, 35:26698–26710, 2022.
- 344 Dagréou, M., Moreau, T., Vaiter, S., and Ablin, P. A lower bound and a near-optimal algorithm for bilevel empirical risk
 345 minimization. *arXiv preprint arXiv:2302.08766*, 2023.
- 346 Defazio, A., Bach, F., and Lacoste-Julien, S. Saga: A fast incremental gradient method with support for non-strongly convex
 347 composite objectives. In *Advances in Neural Information Processing Systems*, pp. 1646–1654, 2014.
- 348 Ghadimi, S. and Wang, M. Approximation methods for bilevel programming. *arXiv preprint arXiv:1802.02246*, 2018.
- 349 Guo, Z., Hu, Q., Zhang, L., and Yang, T. Randomized stochastic variance-reduced methods for multi-task stochastic bilevel
 350 optimization. *arXiv preprint arXiv:2105.02266*, 2021.
- 351 Hao, J., Gong, X., and Liu, M. Bilevel optimization under unbounded smoothness: A new algorithm and convergence
 352 analysis. In *The Twelfth International Conference on Learning Representations*, 2024.
- 353 Hong, M., Wai, H.-T., Wang, Z., and Yang, Z. A two-timescale stochastic algorithm framework for bilevel optimization:
 354 Complexity analysis and application to actor-critic. *SIAM Journal on Optimization*, 33(1):147–180, 2023.
- 355 Ji, K. and Liang, Y. Lower bounds and accelerated algorithms for bilevel optimization. *Journal of Machine Learning*
 356 *Research*, 24(22):1–56, 2023. URL <http://jmlr.org/papers/v24/21-0949.html>.
- 357 Ji, K., Yang, J., and Liang, Y. Bilevel optimization: Convergence analysis and enhanced design. In *International conference*
 358 *on machine learning*, pp. 4882–4892. PMLR, 2021.
- 359 Khanduri, P., Zeng, S., Hong, M., Wai, H.-T., Wang, Z., and Yang, Z. A near-optimal algorithm for stochastic bilevel
 360 optimization via double-momentum. *Advances in neural information processing systems*, 34:30271–30283, 2021.
- 361 Kwon, J., Kwon, D., Wright, S., and Nowak, R. D. A fully first-order method for stochastic bilevel optimization. In
 362 *International Conference on Machine Learning*, pp. 18083–18113. PMLR, 2023.
- 363 Liu, M., Zhuang, Z., Lei, Y., and Liao, C. A communication-efficient distributed gradient clipping algorithm for training
 364 deep neural networks. *Advances in Neural Information Processing Systems*, 35:26204–26217, 2022.
- 365 Yang, J., Ji, K., and Liang, Y. Provably faster algorithms for bilevel optimization. *Advances in Neural Information*
 366 *Processing Systems*, 34:13670–13682, 2021.
- 367 Zhang, B., Jin, J., Fang, C., and Wang, L. Improved analysis of clipping algorithms for non-convex optimization. *Advances*
 368 *in Neural Information Processing Systems*, 2020a.
- 369 Zhang, J., He, T., Sra, S., and Jadbabaie, A. Why gradient clipping accelerates training: A theoretical justification for
 370 adaptivity. In *International Conference on Learning Representations*, 2020b.