Generalized Frank-Wolfe Algorithm for Bilevel Optimization

Ruichen Jiang∗ Nazanin Abolfazli† Aryan Mokhtari∗ Erfan Yazdandoost Hamedani†

June 20, 2022

Abstract

In this paper, we study a class of bilevel optimization problems, also known as simple bilevel optimization, where we minimize a smooth objective function over the optimal solution set of another convex constrained optimization problem. Several iterative methods have been developed for tackling this class of problems. Alas, their convergence guarantees are not satisfactory as they are either asymptotic for the upper-level objective, or the convergence rates are slow and sub-optimal. To address this issue, in this paper, we introduce a generalization of the Frank-Wolfe (FW) method to solve the considered problem. The main idea of our method is to locally approximate the solution set of the lower-level problem via a cutting plane, and then run a FW-type update to decrease the upper-level objective. When the upper-level objective is convex, we show that our method requires $O(\max\{1/\epsilon_f, 1/\epsilon_g\})$ iterations to find a solution that is $\epsilon_f$-optimal for the upper-level objective and $\epsilon_g$-optimal for the lower-level objective. Moreover, when the upper-level objective is non-convex, our method requires $O(\max\{1/\epsilon_f^2, 1/(\epsilon_f \epsilon_g)\})$ iterations to find an $(\epsilon_f, \epsilon_g)$-optimal solution. We further prove stronger convergence guarantees under the Hölderian error bound assumption on the lower-level problem. To the best of our knowledge, our method achieves the best-known iteration complexity for the considered bilevel problem. We also present numerical experiments to showcase the superior performance of our method compared with state-of-the-art methods.

∗Department of Electrical and Computer Engineering, The University of Texas at Austin, Austin, TX, USA
{rjiang@utexas.edu, mokhtari@austin.utexas.edu}
†Department of Systems and Industrial Engineering, The University of Arizona, Tucson, AZ, USA
{nazaninabolfazli@email.arizona.edu, erfany@arizona.edu}
1 Introduction

Bilevel optimization is a form of optimization where one problem is embedded within another. It captures a hierarchical structure, where an upper-level function is minimized over the solution set of a lower-level problem. This class of problems has attracted great attention due to their applications in hyper-parameter optimization [FFSGP18; SCHB19], meta-learning [RFKL19; BHTV19], and reinforcement learning [KT00; HWWY20], to name a few. In this paper, we focus on a specific form of bilevel optimization formally defined as

$$\min_{x \in \mathbb{R}^n} f(x) \quad \text{s.t.} \quad x \in \arg\min_{z \in Z} g(z),$$  \hspace{1cm} (1)

where $Z$ is a compact convex set and $f, g : \mathbb{R}^n \to \mathbb{R}$ are continuously differentiable functions on an open set containing $Z$. We assume that $g$ is convex but not necessarily strongly convex, and hence the lower-level problem in (1) could have multiple optimal solutions. We remark that Problem (1) is often referred to as the “simple bilevel problem” in the literature [DDD10; DP20; SVZ21] to differentiate it from the more general settings where the lower-level problem is parameterized by some upper-level variables. This class of bilevel problems appears in several settings and we review a few of them in Section 2.

The key challenge to solve Problem (1) stems from the fact that its feasible set—the solution set of the lower-level problem—does not admit a simple characterization and is not explicitly given. This rules out the possibility of applying projection-based methods as well as the Frank-Wolfe (FW) method, since projection onto or minimizing a linear objective over the feasible set is intractable. An alternative scheme is reformulating Problem (1) as a constrained optimization problem with functional constraints and applying primal-dual methods. Specifically, Problem (1) can be written as

$$\min_{x \in \mathbb{R}^n} f(x) \quad \text{s.t.} \quad x \in Z, \quad g(x) \leq g^*,$$  \hspace{1cm} (2)

where $g^*$ is the optimal value of the lower-level problem. However, a critical issue is that Problem (2) does not satisfy strict feasibility and hence the Slater’s condition fails, which is required for most primal-dual methods. Even relaxing the constraint ($g(x) \leq g^* + \epsilon$) to ensure strict feasibility would inevitably lead to numerical issues. In fact, as $\epsilon$ approaches zero and the problem becomes nearly degenerate, the dual optimal variable may tend to infinity, which slows down the convergence and may lead to numerical instability [BS13] (see more discussions in Section 5.1 and Appendix E).

Therefore, Problem (1) cannot be simply treated as a classic constrained optimization problem and calls for new theories and algorithms tailored to its hierarchical structure. A common approach to solve Problem (1) is the Tikhonov-type regularization [TA77], where the objective functions of both levels are combined using a regularization parameter $\sigma > 0$ to form a single-level problem. It is known that as $\sigma \to 0$, any cluster point of the solutions of the regularized single-level problem is a solution to the original bilevel problem in (1). Moreover, under certain assumptions [FT08; DDDDP21], the solution set of Problem (1) exactly matches with the regularized problem for $\sigma$ small enough. However, checking such conditions and finding the threshold are often difficult in practice. To avoid this issue, [Cab05; Sol07] proposed adjusting the regularization parameter $\sigma$ dynamically and proved an asymptotic convergence guarantee. Along another line of research, several works [Yam01; Xu04] have studied the more general problem of solving a variational inequality over the fixed-point set of a nonexpansive mapping, to which the bilevel problem in (1) is a special case. In particular, the hybrid steepest descent method in [Yam01] and the sequential averaging method (SAM) in [Xu04] converge asymptotically to the optimal solution when the parameters are properly chosen.
Table 1: Summary of bilevel optimization algorithms. The abbreviations “SC”, “C”, and “non-C” stand for “strongly convex”, “convex”, and “non-convex”, respectively.

| References          | Upper level | Lower level | Convergence       | Projection free? |
|---------------------|-------------|-------------|-------------------|-----------------|
|                     | Objective   | Objective   | Feasible set $Z$ | Upper level     | Lower level     |                  |
| MNG [BS14]          | SC, differentiable | C, smooth | Closed | Asymptotic | $O(1/\epsilon^2)$ | ✗ |
| BiG-SAM [SS17]      | SC, smooth | C, composite | Closed | Asymptotic | $O(1/\epsilon)$ | ✗ |
| Tseng’s method [Mal17] | C, composite | C, composite | Closed | Asymptotic | $o(1/\epsilon)$ | ✗ |
| a-IRG [KY21a]       | C, Lipschitz | C, Lipschitz | Closed | $O(\max\{1/\epsilon_f^4, 1/\epsilon_g^4\})$ | ✗ |
| Ours                | C, smooth | C, smooth | Compact | $O(\max\{1/\epsilon_f, 1/\epsilon_g\})$ | ✓ |
| Ours                | Non-C, smooth | C, smooth | Compact | $O(\max\{1/\epsilon_f^2, 1/(\epsilon_f \epsilon_g)\})$ | ✓ |

More recently, there has been a surge of interest in establishing non-asymptotic convergence rates for Problem (1). One of the first methods of this kind is the minimal norm gradient (MNG) method proposed in [BS14]. When the upper-level function $f$ is strongly-convex and the lower-level function $g$ is convex and smooth, it is shown that MNG converges asymptotically to the optimal solution and achieves a complexity bound of $O(1/\epsilon^2)$ in terms of the lower-level objective value. Subsequently, built upon the SAM framework, the BiG-SAM method was proposed in [SS17] and it was shown that it achieves a complexity of $O(1/\epsilon)$ for the lower-level problem; see also [SVZ21] for a related method. In [Mal17], a version of Tseng’s accelerated gradient method was studied that obtains a convergence rate of $o(1/k)$ for the lower-level problem. When $f$ and $g$ are convex and Lipschitz, [KY21a] studied iterative regularization and showed a convergence rate of $O(1/k^{0.5-b})$ for the upper-level objective and a rate of $O(1/k^b)$ for the lower-level, where $b \in (0, 0.5)$ is a user-defined parameter. Several works also extended this method to stochastic [AY19b; AY19a] and distributed [You21; KY21b] settings.

Contributions. As discussed, prior works only establish convergence rates for the lower-level problem, while the rate for the upper-level objective is missing. The only exception is [KY21a], but it considers a different setting where both upper-level and lower-level functions are Lipschitz and possibly non-smooth, which results in slow convergence rates; see Table 1. Moreover, existing methods require projection onto the set $Z$ and are not suitable when such projection is computationally costly. Our main contribution is presenting the generalized bilevel Frank-Wolfe (GBFW) method with tight non-asymptotic guarantees for both upper- and lower-level problems. In addition to its fast convergence, GBFW also benefits from being a projection-free method. At each iteration, GBFW uses a cutting plane to locally approximate the solution set of the lower-level problem, and then applies a FW-type update on the upper-level objective. Our theoretical guarantees for GBFW are:

- When the upper-level function $f$ is convex, we show that our GBFW method finds $\hat{x}$ that satisfies $f(\hat{x}) - f^* \leq \epsilon_f$ and $g(\hat{x}) - g^* \leq \epsilon_g$ within $O(\max\{1/\epsilon_f, 1/\epsilon_g\})$ iterations, where $f^*$ is the optimal value of Problem (1) and $g^*$ is the optimal value of the lower-level problem. This guarantees match the best-known results in terms of the lower-level objective and are optimal for bilevel projection-free methods.

- When $f$ is non-convex, GBFW finds $\hat{x}$ that satisfies $G(\hat{x}) \leq \epsilon_f$ and $g(\hat{x}) - g^* \leq \epsilon_g$ within $O(\max\{1/\epsilon_f^2, 1/(\epsilon_f \epsilon_g)\})$ iterations, where $G(\hat{x})$ is the FW gap function (cf. (7)).
With an additional \( r \)-th-order \((r \geq 1)\) Hölderian error bound assumption on the lower-level problem, GBFW finds \( \hat{x} \) with \( |f(\hat{x}) - f^*| \leq \epsilon_f \) within \( O(1/\epsilon_f^r) \) iterations in the convex case, and \( \hat{x} \) with \( |g(\hat{x})| \leq \epsilon_f \) within \( O(1/\epsilon_f^{r+1}) \) iterations in the non-convex case.

**Additional related work.** In the general form of bilevel problems, the upper-level function \( f \) may also depend on an additional variable \( w \in \mathbb{R}^m \) that in turn influences the lower-level problem:

\[
\min_{x \in \mathbb{R}^n, w \in \mathbb{R}^m} f(x, w) \quad \text{s.t.} \quad x \in \arg\min_{z \in Z} g(z, w). \tag{3}
\]

Problem (3) has been studied deeply and we refer readers to the extensive survey in [Dem20]. We can also see its close connection with the simple bilevel problem we study in this paper, as Problem (3) reduces to Problem (1) for any fixed \( w \). In recent years, gradient-based methods for Problem (3) have become increasingly popular including implicit differentiation [Dom12; Ped16; GFCACG16; JYL21] and iterative differentiation [MDA15]. However, most of the existing methods work under the assumption that the lower-level problem has a unique minimum, and hence they essentially deal with a different aspect of challenge from ours. More relevant to our work, some concurrent papers consider the case where the lower-level problem can have multiple minima [LMYZ20; LGH20; LLYYZ21; LLZZ21; SJGL22]. As they consider a more general problem than ours, their theoretical results are also weaker, providing only asymptotic convergence guarantees or slower rates.

## 2 Preliminaries

In this section, we first mention a few motivating examples for Problem (1). Additional examples are also provided in Appendix F. Then, we state the required assumptions and notions of optimality that we use for our theoretical results.

### 2.1 Motivating examples

**Example 1 (Over-parameterized regression).** Consider a constrained regression problem where we aim to find a parameter vector \( \beta \in \mathbb{R}^n \) that minimizes the loss \( \ell_{\text{tr}}(\beta) \) with respect to the training dataset \( D_{\text{tr}} \). We also constrain \( \beta \) to be in some set \( Z \subseteq \mathbb{R}^n \) representing some prior knowledge. For instance, we have \( Z = \{ \beta \in \mathbb{R}^n \mid \beta_1 \leq \ldots \leq \beta_n \} \) in isotonic regression and \( Z = \{ \|\beta\|_1 \leq \lambda \} \) for some \( \lambda > 0 \) in sparse regression. Without an explicit regularization, an over-parameterized regression over the training dataset possesses multiple global minima. In fact, while any optimization algorithm can achieve one of these many global minima, not all optimal regression coefficients perform equally. Hence, one can consider a secondary objective, such as the loss over a validation set \( D_{\text{val}} \), to select one from the minimizers of the training loss. This leads to the following bilevel problem:

\[
\min_{\beta \in \mathbb{R}^n} f(\beta) \triangleq \ell_{\text{val}}(\beta) \quad \text{s.t.} \quad \beta \in \arg\min_{z \in Z} g(z) \triangleq \ell_{\text{tr}}(z). \tag{4}
\]

In this case, both the upper-level and lower-level objectives are smooth and convex if the loss function \( \ell \) is smooth and convex.

**Example 2 (Dictionary learning).** The goal of dictionary learning is to learn a concise representation of the input data from a massive dataset. Formally, let \( A = \{a_1, \ldots, a_n\} \) denote a dataset of \( n \) points with \( a_i \in \mathbb{R}^m \) \((1 \leq i \leq n)\). We aim to find a dictionary \( D = [d_1, \ldots, d_p] \in \mathbb{R}^{m \times p} \) such
that each data point \( a_i \) can be well approximated by a linear combination of a few basis vectors in \( D \). One common approach is to formulate this as the following non-convex optimization problem \cite{KMRELS03; YBD09; Rak13; BJQS16}:

\[
\min_{D \in \mathbb{R}^{m \times p}} \min_{X \in \mathbb{R}^{p \times n}} \frac{1}{2n} \sum_{i=1}^{n} \| a_i - DX_i \|_2^2 \quad \text{s.t.} \quad \| d_j \|_2 \leq 1, j = 1, \ldots, p; \quad \| x_i \|_1 \leq \delta, i = 1, \ldots, n, \tag{5}
\]

where \( X = [x_1, \ldots, x_n] \in \mathbb{R}^{p \times n} \) denote the coefficient matrix. Also, we normalize the basis vectors to have bounded \( \ell_2 \)-norm and impose \( \ell_1 \)-norm constraints to encourage sparsity in \( \{x_i\}_{i=1}^n \).

In real applications, the data points typically arrive sequentially and the underlying representation may be gradually evolving. Thus, it is desirable to update our dictionary in a continuous manner. Suppose we already learned a dictionary \( \tilde{D} \in \mathbb{R}^{m \times p} \) and the corresponding coefficient matrix \( \tilde{X} \in \mathbb{R}^{p \times n} \) for the dataset \( A \). When a new dataset \( A' = \{a'_1, \ldots, a'_q\} \) arrives, we hope to enlarge our dictionary by learning new basis vectors from \( A' \) while retaining the learned information in \( D \). To achieve so, we aim to find the dictionary \( \tilde{D} \in \mathbb{R}^{m \times q} (q > p) \) and the coefficient matrix \( \tilde{X} \in \mathbb{R}^{q \times n'} \) for the new dataset \( A' \), and at the same time enforce \( \tilde{D} \) to perform well on the old dataset \( A \) together with the learned coefficient matrix \( \tilde{X} \). This leads to the following bilevel problem:

\[
\min_{D \in \mathbb{R}^{m \times q}} \min_{X \in \mathbb{R}^{q \times n'}} f(\tilde{D}, \tilde{X}) \quad \text{s.t.} \quad \| \tilde{x}_k \|_1 \leq \delta, k = 1, \ldots, n'; \quad \tilde{D} \in \text{argmin}_{\| d_j \|_2 \leq 1} g(\tilde{D}), \tag{6}
\]

where the upper-level objective \( f(\tilde{D}, \tilde{X}) \triangleq \frac{1}{2n'} \sum_{k=1}^{n'} \| a'_k - \tilde{D}\tilde{x}_k \|_2^2 \) is the average reconstruction error on the new dataset \( A' \), the lower-level objective \( g(\tilde{D}) \triangleq \frac{1}{2n} \sum_{i=1}^{n} \| a_i - \tilde{D}\tilde{x}_i \|_2^2 \) is the error on the old dataset \( A \), and with a slight abuse of notation we let \( \tilde{x}_i \) denote the extended vector in \( \mathbb{R}^q \) by appending zeros at the end. Note that in Problem (6), the upper-level objective is non-convex while the lower-level objective is convex with multiple minima.

2.2 Assumptions and definitions

We focus on the case where the lower-level function \( g \) is smooth and convex, while the upper-level function \( f \) is smooth but not necessarily convex. Formally, we make the following assumptions.

**Assumption 2.1.** Let \( \| \cdot \| \) be an arbitrary norm on \( \mathbb{R}^n \) and \( \| \cdot \|_* \) be its dual norm. We assume

(i) \( \mathcal{Z} \subset \mathbb{R}^n \) is convex and compact with diameter \( \mathcal{D} \), i.e., \( \| x - y \| \leq D \) for all \( x, y \in \mathcal{Z} \).

(ii) \( g \) is convex and continuously differentiable on an open set containing \( \mathcal{Z} \), and its gradient is Lipschitz with constant \( L_g \), i.e., \( \| \nabla g(x) - \nabla g(y) \|_* \leq L_g \| x - y \| \) for all \( x, y \in \mathcal{Z} \).

(iii) \( f \) is continuously differentiable and its gradient is Lipschitz with constant \( L_f \).

**Remark 2.1.** Instead of the Lipschitz gradient assumptions above, we may also assume that the functions \( f \) and \( g \) have bounded curvature constants. Such assumption is common in the analysis of the FW method and has the advantage of being affine-invariant, e.g., see \cite{Jag13; Lac16}.

Throughout the paper, we adopt the following notations. We use \( g^* \triangleq \min_{z \in \mathcal{Z}} g(z) \) and \( \mathcal{X}_g^* \triangleq \text{argmin}_{z \in \mathcal{Z}} g(z) \) to denote the optimal value and the optimal solution set of the lower-level problem, respectively. Note that by Assumption 2.1, the set \( \mathcal{X}_g^* \) is nonempty, compact and convex, but in
general not a singleton as $g$ could have multiple minima on $Z$. Moreover, we use $f^*$ to denote the optimal value and $x^*$ to denote an optimal solution of Problem (1), which are guaranteed to exist as $f$ is continuous and $X_g^*$ is compact.

For generality, we allow different target accuracies $\epsilon_f$ and $\epsilon_g$ for the upper-level and lower-level problems, respectively, and define an $(\epsilon_f, \epsilon_g)$-optimal solution as follows.

**Definition 2.1** $(\epsilon_f, \epsilon_g)$-optimal solution. When $f$ is convex, a point $\hat{x} \in Z$ is $(\epsilon_f, \epsilon_g)$-optimal for the bilevel problem in (1) if

$$f(\hat{x}) - f^* \leq \epsilon_f \quad \text{and} \quad g(\hat{x}) - g^* \leq \epsilon_g.$$  

When $f$ is non-convex, $\hat{x} \in Z$ is $(\epsilon_f, \epsilon_g)$-optimal if

$$G(\hat{x}) \leq \epsilon_f \quad \text{and} \quad g(\hat{x}) - g^* \leq \epsilon_g,$$

where $G(\hat{x})$ is the FW gap defined by

$$G(\hat{x}) \triangleq \max_{s \in X_g^*} \{ \langle \nabla f(\hat{x}), \hat{x} - s \rangle \}.$$  

(7)

### 3 Generalized FW method for bilevel optimization

Before stating our proposed method, we start by the standard FW method for solving Problem (1). Recall that $X_g^*$ denotes the solution set of the lower-level problem. If we assume $x_0 \in X_g^*$, then the update of FW at iteration $k$ is given by

$$x_{k+1} = (1 - \gamma_k) x_k + \gamma_k s_k \quad \text{where} \quad s_k = \arg\min_{s \in X_g^*} \langle \nabla f(x_k), s \rangle,$$

(8)

and $\gamma_k \in [0, 1]$ is the stepsize. However, as we discussed earlier, the main challenge here is that the solution set $X_g^*$ for the lower-level problem is not explicitly given, and hence the linear minimization required in (8) is computationally intractable. Moreover, the standard FW method needs to be initialized with a feasible point. In this case, $x_0$ has to be an optimal solution of the lower-level problem, which is hard to guarantee in general—in finite number of iterations one may not be able to find an exact optimal solution for the lower-level problem. Similar issues also hold if we try to use projection-based methods such as projected gradient descent to solve Problem (2).

Our key idea is to run the FW update over a local approximation set $X_k$ at the $k$-th iteration in place of the more complicated set $X_g^*$. To this end, we borrow the idea of cutting plane from the optimization literature [BV18] and let $X_k$ be the intersection of $Z$ and the halfspace $H_k$:

$$X_k \triangleq Z \cap H_k \quad \text{where} \quad H_k = \{ s \in Z : \langle \nabla g(x_k), s - x_k \rangle \leq g(x_0) - g(x_k) \}.$$  

(9)

We can see that $X_k$ is potentially more tractable than $X_g^*$, as the difficult nonlinear inequality $g(x) \leq g^*$ in (2) is replaced by a single linear inequality. Also, by using the convexity of $g$, we can show that the hyperplane $H_k$ eliminates those points that are known to have a larger value of than $g(x_0)$. Thus, if we initialize our algorithm such that $x_0$ is near-optimal, the linear inequality in (9) ensures improvement in terms of the lower-level function. Further, this also implies that $X_k$ contains the solution set $X_g^*$, so we are guaranteed to make progress on the upper-level objective $f$. We justify this observation in the following lemma.
Algorithm 1 Generalized bilevel Frank-Wolfe (GBFW)

1: **Input**: Target accuracies $\epsilon_f, \epsilon_g > 0$, stepsizes $\{\gamma_k\}_k$
2: **Initialization**: Initialize $x_0 \in \mathcal{Z}$ such that $0 \leq g(x_0) - g^* \leq \epsilon_g / 2$
3: for $k = 0, \ldots, K$ do
4: Compute $s_k \leftarrow \text{argmin}_{s \in \mathcal{X}_k} \langle \nabla f(x_k), s \rangle$
   where $\mathcal{X}_k \triangleq \{ s \in \mathcal{Z} : \langle \nabla g(x_k), s - x_k \rangle \leq g(x_0) - g(x_k) \}$
5: if $\langle \nabla f(x_k), x_k - s_k \rangle \leq \epsilon_f$ and $\langle \nabla g(x_k), x_k - s_k \rangle \leq \epsilon_g / 2$ then
6: Return $x_k$ and STOP
7: else
8: $x_{k+1} \leftarrow (1 - \gamma_k)x_k + \gamma ks_k$
9: end if
10: end for

Lemma 3.1. Recall $\mathcal{X}^*_g$ as the solution set for the lower-level problem in (1) and recall the definition of the set $\mathcal{X}_k$ in (9). Then, for any $k \geq 0$, we have $\mathcal{X}^*_g \subseteq \mathcal{X}_k$.

Now we are ready to state our generalized bilevel Frank-Wolfe (GBFW) method. We first initialize $x_0 \in \mathcal{Z}$ as a near-optimal solution for the lower-level problem, i.e., $g(x_0) - g^* \leq \epsilon_g / 2$ for some prescribed accuracy $\epsilon_g$. This can be achieved by running the standard FW method on the lower-level problem, which requires at most $\mathcal{O}(1/\epsilon_g)$ iterations. Once the initialization step is done, we simply run FW with respect to the approximation sets $\mathcal{X}_k$. More precisely, at the $k$-th iteration, we solve the following subproblem over the set $\mathcal{X}_k$ defined in (9):

$$ s_k = \text{argmin}_{s \in \mathcal{X}_k} \langle \nabla f(x_k), s \rangle, \quad (10) $$

and update the iterate with stepsize $\gamma_k \in [0, 1]$:

$$ x_{k+1} = (1 - \gamma_k)x_k + \gamma ks_k. $$

We note that if $\mathcal{Z}$ can be described by a system of linear inequalities, then the subproblem in (10) corresponds to a linear program and can be solved efficiently by a standard solver. We repeat the process above until we reach an accuracy of $\epsilon_f$ for the upper-level objective and an accuracy of $\epsilon_g$ for the lower-level objective. The steps of our proposed GBFW method are summarized in Algorithm 1.

4 Convergence analysis

In this section, we analyze the iteration complexity of our GBFW method. We first consider the case where the upper-level function $f$ is convex. In this case, we choose the stepsize as $\gamma_k = 2/(k + 2)$, which is a typical choice in the standard FW method [Jag13].

Theorem 4.1 (Convex upper-level). Suppose that Assumption 2.1 holds and $f$ is convex. Let $\{x_k\}_{k=0}^K$ be the sequence generated by Algorithm 1 with stepsize $\gamma_k = 2/(k + 2)$ for $k \geq 0$. Then we have

$$ f(x_K) - f^* \leq \frac{2L_f D^2}{K + 1} \quad \text{and} \quad g(x_K) - g^* \leq \frac{2L_g D^2}{K + 1} + \frac{1}{2} \epsilon_g. $$
Theorem 4.1 shows that the gap of the upper-level objective can be upper bounded by $O(1/K)$, similar to the convergence bound of standard FW. At the same time, the gap of the lower-level objective can also be controlled by a term of order $O(1/K)$ in addition to the initial error $\epsilon_g/2$. As a corollary, Algorithm 1 will return an $(\epsilon_f, \epsilon_g)$-optimal solution when the number of iterations $K$ exceeds

$$
\max \left\{ \frac{2L_f D^2}{\epsilon_f} - 1, \frac{4L_g D^2}{\epsilon_g} - 1 \right\} = O \left( \max \left\{ \frac{1}{\epsilon_f}, \frac{1}{\epsilon_g} \right\} \right).
$$

Our complexity bound improves over the result in [KY21a], which considers a different setup where both the upper-level and lower-level functions are Lipschitz but not necessarily smooth. Also, comparing with existing works in the same setup, our convergence rate is tight at least within the family of projection-free methods, since it is known that their worst-case complexity is $\Theta(1/\epsilon_f)$ even for a single-level problem [Jag13; Lan13].

Now we turn to the case where $f$ is non-convex. In this case, we choose the stepsize as a constant depending on the target accuracies as well as the problem parameters.

**Theorem 4.2 (Non-convex upper-level).** Suppose that Assumption 2.1 holds. Let $\{x_k\}_{k=0}^{K-1}$ be the sequence generated by Algorithm 1 with stepsize $\gamma_k = \min \left\{ \frac{\epsilon_f}{L_f D^2}, \frac{\epsilon_g}{L_g D^2} \right\}$ for all $k \geq 0$. Define $f = \min_{x \in \mathcal{Z}} f(x)$. Then for $K \geq \max \left\{ \frac{2L_f D^2(f(x_0) - f)}{\epsilon_f}, \frac{4L_g D^2(f(x_0) - f)}{\epsilon_g} \right\}$, there exists $k^* \in \{0, 1, \ldots, K-1\}$ such that $G(x_{k^*}) \leq \epsilon_f$ and $g(x_{k^*}) - g^* \leq \epsilon_g$.

As a corollary of Theorem 4.2, the number of iterations required to find an $(\epsilon_f, \epsilon_g)$-optimal solution can be upper bounded by $O(\max\{1/\epsilon_f^2, 1/(\epsilon_f \epsilon_g)\})$. We note that the dependence on the upper-level accuracy $\epsilon_f$ also matches that in the standard FW method for a single-level problem [Lac16; MOJ18].

### 4.1 Convergence under Hölderian error bound assumption

In Theorems 4.1 and 4.2, we measure the progress for the upper-level objective in terms of $f(x) - f^*$ (in the convex case) or $G(x)$ (in the non-convex case). However, in general they may not serve as a good performance metric: since the generated iterate $x$ may lie outside of the feasible set $\mathcal{X}_g^*$, both $f(x) - f^*$ and $G(x)$ could be negative. Thus, our convergence result will be stronger if we can instead upper bound $|f(x) - f^*|$ or $|G(x)|$.

Let $\hat{x}$ be an $(\epsilon_f, \epsilon_g)$-optimal solution as defined in Definition 2.1. Intuitively, since $\hat{x}$ is $\epsilon_g$-optimal for the lower-level function, it should be close to the optimal solution set $\mathcal{X}_g^*$ under some regularity condition on $g$. As such, we can lower bound $f(\hat{x}) - f^*$ by using the smoothness of $f$. Formally, we assume that the lower-level function satisfies the Hölderian error bound, which quantifies the growth rate of the objective value $g(x)$ as the point $x$ deviates from the optimal solution set $\mathcal{X}_g^*$.

**Assumption 4.1.** The function $g$ satisfies the Hölderian error bound for some $\alpha > 0$ and $r \geq 1$, i.e.,

$$
\frac{\alpha}{r} \cdot \text{dist}(x, \mathcal{X}_g^*)^r \leq g(x) - g^*, \quad \forall x \in \mathcal{Z},
$$

where $\text{dist}(x, \mathcal{X}_g^*) \equiv \inf_{x' \in \mathcal{X}_g^*} \|x - x'\|$.  


We note that the error bound condition in (11) is well-studied in the optimization literature (see [Pan97; BNPS17; Rd20] and the references therein) and is known to hold generally when the function $g$ is analytic and the set $Z$ is bounded [Loj59; LP94]. Two important special cases are: 1) $g$ satisfies (11) with $r = 1$, i.e., $X_g^* = \{0\}$ is a set of weak sharp minima of $g$ [BF93; BD05]; 2) $g$ satisfies (11) with $r = 2$, which can be viewed as a general notion of strong convexity.

Under Assumption 4.1, we can establish the following lower bounds on $f(\hat{x}) - f^*$ and $\mathcal{G}(\hat{x})$. Notably, the following result is an intrinsic property of Problem (1) and independent of the algorithm we use.

**Proposition 4.3.** Assume that $g$ satisfies the Hölderian error bound in Assumption 4.1, and define $M = \max_{x \in X_g^*} \| \nabla f(x) \|_*$. Then for any $\hat{x}$ that satisfies $g(\hat{x}) - g^* \leq \epsilon_g$, it holds that:

(i) If $f$ is convex, then $f(\hat{x}) - f^* \geq -M \left( \frac{r \epsilon_g}{\alpha} \right) \frac{1}{r}$.

(ii) If $f$ is non-convex and has $L_f$-Lipschitz gradient, then $\mathcal{G}(\hat{x}) \geq -M \left( \frac{r \epsilon_g}{\alpha} \right) \frac{1}{r} - L_f \left( \frac{r \epsilon_g}{\alpha} \right) \frac{2}{r}$.

By combining Theorems 4.1 and 4.2 with Proposition 4.3, we obtain the following stronger convergence guarantees for the output of our proposed method.

**Corollary 4.4.** Suppose that Assumption 2.1 holds and $g$ satisfies the Hölderian error bound in Assumption 4.1 with $\alpha > 0$ and $r \geq 1$. Let $M = \max_{x \in X_g^*} \| \nabla f(x) \|_*$. Then after $K = \mathcal{O}(1/\epsilon_f^r)$ iterations, we have $|f(x_K) - f^*| \leq \epsilon_f$ and $g(x_K) - g^* \leq \epsilon_g$.

(ii) If $f$ in Problem (1) is convex, we can set $\epsilon_g = \frac{\alpha}{r} \left( \frac{\epsilon_f}{M} \right)^r$. Then after $K = \mathcal{O}(1/\epsilon_f^r)$ iterations, there exists $k^* \in \{0, 1, \ldots, K - 1\}$ such that $|\mathcal{G}(x_{k^*})| \leq \epsilon_f$.

Corollary 4.4 shows that under the $r$-th Hölderian error bound assumption, we can find an iterate to be $\epsilon_f$-close to optimality within $\mathcal{O}(1/\epsilon_f^r)$ iterations in the convex case, and to be $\epsilon_f$-close to stationarity within $\mathcal{O}(1/\epsilon_f^{r+1})$ iterations in the non-convex case.

## 5 Numerical experiments

In this section, we test our method for solving different bilevel optimization problems. First, we consider a toy example and compare the iteration trajectory of our proposed method with accelerated primal-dual method with backtracking (APDB) proposed in [HA21]. Next, we consider the examples described in Section 2 with real and synthetic datasets and compare our method with averaging iteratively regularized gradient (a-IRG) proposed in [KY21a]. All experiments are performed on a machine running 64-bit Windows 10 with Intel i7-8650U @2.11GHz and 16GB RAM.

### 5.1 Toy example

Here we consider a simple two-dimensional example to illustrate the numerical instability of primal-dual methods applied to the relaxed problem in (23). To this consider the following problem

$$\min_{x \in \mathbb{R}^2} 0.5x_1^2 - 0.5x_1 + 0.1x_2 \quad \text{s.t.} \quad x \in \arg\min_{x \in \mathcal{Z}} \{ -z_1 - z_2 \},$$

(12)
where $Z = \{ z \in \mathbb{R}^n_+ \mid z_1 + z_2 \leq 1, 4z_1 + 6z_2 \leq 5 \}$. The lower-level problem has multiple solutions which can be described by $X^*_g = \{ x \in \mathbb{R}^2 \mid x_1 + x_2 = 1, x_1 \in [0.5, 1], x_2 \in [0, 0.5] \}$ and the optimal solution of (12) is $(x_1^*, x_2^*) = (0.6, 0.4)$. We implemented our proposed method and compared it with APDB. Figure 1 illustrates the iteration trajectory of both methods. We selected the relaxing parameter in (23) as $\epsilon = 10^{-5}$ for APDB. We also used the same accuracy for $\epsilon_g$ and $\epsilon_f$ when implementing GBFW. The primal-dual method finds an $\epsilon$-solution (dark red cross) within 193 iterations while GBFW finds an $\epsilon$-solution (green star) within 20 iterations. Furthermore, we observe a more stable numerical behavior for GBFW in comparison with APDB.

5.2 Over-parameterized regression

In this section, we consider a sparse linear regression problem on the Wikipedia Math Essential dataset [Roz+21], which consists of a data matrix $A \in \mathbb{R}^{m \times n}$ with $m = 1068$ instances and $n = 731$ attributes and an outcome vector $b \in \mathbb{R}^m$. Our goal is to find a sparse parameter $\beta \in \mathbb{R}^n$ to achieve a small prediction error $\frac{1}{2} \| A\beta - b \|_2^2$. We formulate the regression problem as the bilevel optimization problem in Example 1. Specifically, we assign 60% of the dataset as the training set $(A_{tr}, b_{tr})$, 20% as the validation set $(A_{val}, b_{val})$ and the rest as the test set $(A_{test}, b_{test})$. Then the lower-level objective in (4) is the training error $g(\beta) = \frac{1}{2} \| A_{tr}\beta - b_{tr} \|_2^2$, the upper-level objective is the validation error $f(\beta) = \frac{1}{2} \| A_{val}\beta - b_{val} \|_2^2$, and the constraint set is the $\ell_1$-ball $Z = \{ \beta : \| \beta \|_1 \leq \lambda \}$ for some $\lambda > 0$ to induce sparsity in $\beta$. We also use the test error $\frac{1}{2} \| A_{test}\beta - b_{test} \|_2^2$ as our performance metric. Note that the regression problem is over-parameterized since the number of parameters $n$ is larger than the number of data instances in the training set.

In the experiment, we implement our GBFW algorithm to solve the bilevel problem in (4) with parameter $\lambda = 1$. We set the target accuracies for the upper-level and lower-level problems to $\epsilon_f = 10^{-4}$ and $\epsilon_g = 10^{-4}$, respectively. For comparison, we also implement the a-IRG method in [KY21a] with the stepsizes suggested by the authors. For benchmarking purposes, we use CVX [GB14; GB08] to solve the lower-level problem and the constrained reformulation in (2) to obtain the optimal values $g^*$ and $f^*$, respectively.

In Fig. 2, we illustrate the numerical performance of both algorithms. From Fig. 2(a), we can see that GBFW converges at a faster rate than a-IRG in terms of the lower-level objective, which confirms our theoretical result (cf. Table 1). Fig. 2(c) and (d) also show that it is able to achieve a smaller upper-level objective value as well as a smaller test error compared with a-IRG within the same running time. Interestingly, we observe that after the initial stage, the upper-level objective $f(\beta_k)$ of GBFW actually increases, while the optimality gap $|f(\beta_k) - f^*|$ decreases (cf. Fig. 2(b)).
Figure 2: The performance of GBFW vs a-IRG on Problem (4). Plots from left to right: upper-level suboptimality, lower-level suboptimality, the value of the upper-level objective and the test error.

and (c)). This suggests that GBFW may “overshoot” at the beginning due to its relatively large stepsize. Nevertheless, as the number of iterations increases and the level of infeasibility decreases, the upper-level objective of our algorithm approaches the optimal value of the bilevel problem, which is also in line with Proposition 4.3.

5.3 Dictionary learning

We evaluate our GBFW algorithm for Problem (6) on a synthetic dataset, similar to the experiment setup in [Rak13]. We first generate the true dictionary \( \tilde{D}^* \in \mathbb{R}^{25 \times 50} \) consisting of 50 basis vectors in \( \mathbb{R}^{25} \), each of which has its entries drawn from a standard Gaussian distribution and is normalized to have unit \( \ell_2 \)-norm. We further construct the two dictionaries \( D^* \) and \( D^{*'} \) consisting of 40 and 20 basis vectors in \( \tilde{D}^* \), respectively (and hence they share 10 bases in common). The two datasets \( A = \{a_1, \ldots, a_{250}\} \) and \( A' = \{a'_{1}, \ldots, a'_{200}\} \) are then generated according to the following rules:

\[
a_i = D^* x_i + n_i, \quad i = 1, 2, \ldots, 250 \quad \text{and} \quad a'_k = D^{*'} x'_k + n'_k, \quad k = 1, 2, \ldots, 200,
\]

where \( \{x_i\}_{i=1}^{250}, \{x'_k\}_{k=1}^{1200} \) are the sparse coefficient vectors and \( \{n_i\}_{i=1}^{250}, \{n'_k\}_{k=1}^{1200} \) are the random Gaussian noise vectors. Since neither \( A \) nor \( A' \) contains the full information of the true dictionary \( \tilde{D}^* \), it is crucial for our learning algorithm to update our dictionary given the new dataset \( A' \) while retaining our knowledge from the old dataset \( A \).

In our experiment, we first solve the standard dictionary learning problem in (5) using the dataset \( A \) to obtain the initial dictionary \( \hat{D} \) and the coefficient vectors \( \{\hat{x}_i\}_{i=1}^{250} \). Then we use the reconstruction error on \( A \) with respect to \( \{\hat{x}_i\}_{i=1}^{250} \) to define the lower-level objective in Problem (6), and use the error on the new dataset \( A' \) to define the upper-level objective. Note that in this case \( \hat{D} \) serves as a near-optimal solution for the lower-level problem. We implement our GBFW algorithm as well as the a-IRG method in [KY21a]. Moreover, to demonstrate the necessity of the cutting plane in (9), we also run a baseline algorithm that performs the FW update over the set \( \mathcal{Z} \) instead of \( X_k \) (cf. the update rule in (10)). It essentially ignores the lower-level objective and may be regarded as applying the standard FW algorithm solely on the upper-level objective. In all algorithms, we initialize the variable \( \tilde{D} \) with the dictionary \( \hat{D} \) learnt from \( A \) and initialize the variable \( \tilde{X} \) randomly.

We report our results in Fig. 3. In addition to the the upper- and lower-level objective values, we also use the recovery rate of the true basis vectors as our performance metric. Specifically, we regard a basis vector \( d^*_i \) in \( \tilde{D}^* \) as successfully recovered in the output dictionary \( \tilde{D} \) if there exists \( d_j \) in \( D \) such that \( |\langle d^*_i, d_j \rangle| > 0.9 \). From Fig. 3(a) and (b), we can see that our algorithm converges much
faster compared with the a-IRG algorithm, achieving smaller errors in terms of both the upper- and lower-level objectives. On the other hand, the baseline FW algorithm only focuses on the upper-level objective and as a result incurs a much larger error on the lower-level objective. In terms of the recovery rate, Fig. 3(c) shows that our algorithm successfully recovers almost all the basis vectors in $\tilde{D}^*$ at the end of its execution. In contrast, the a-IRG algorithm only learns the dictionary from the dataset $A$ due to its slow convergence, while the baseline FW algorithm “forgets” the previously learned basis vectors and only recovers those underlying the new dataset $A'$.

6 Conclusion

In this paper, we generalized the FW algorithm to solve a class of bilevel optimization problems. We closed an important gap in the existing literature by providing a tight non-asymptotic complexity bound for the upper-level objective. Specifically, we proved that our GBFW algorithm can find an $(\epsilon_f, \epsilon_g)$-optimal solution after at most $O(\max\{1/\epsilon_f, 1/\epsilon_g\})$ iterations when the upper-level objective $f$ is convex, and after at most $O(\max\{1/\epsilon_f^2, 1/(\epsilon_f \epsilon_g)\})$ iterations when $f$ is non-convex. Furthermore, if the lower-level problem satisfies an $r$-th-order Hölderian error bound, our algorithm finds a point $\hat{x}$ with $|f(\hat{x}) - f^*| \leq \epsilon_f$ within $O(1/\epsilon_f^r)$ iterations in the convex case, and $\hat{x}$ with $|G(\hat{x})| \leq \epsilon_f$ within $O(1/\epsilon_f^{r+1})$ iterations in the non-convex case. Our numerical results also showed superior performance of our GBFW algorithm compared to existing algorithms.

Acknowledgements

The research of R. Jiang and A. Mokhtari is supported in part by NSF Grants 2127697, 2019844, and 2112471, ARO Grant W911NF2110226, the Machine Learning Lab (MLL) at UT Austin, and the Wireless Networking and Communications Group (WNCG) Industrial Affiliates Program. The research of N. Abolfazli and E. Yazdandoost Hamedani is supported by NSF Grant 2127696.
References

[AY19a] Mostafa Amini and Farzad Yousefian. “An Iterative Regularized Incremental Projected Subgradient Method for a Class of Bilevel Optimization Problems”. In: 2019 American Control Conference (ACC). IEEE. 2019, pp. 4069–4074 (page 3).

[AY19b] Mostafa Amini and Farzad Yousefian. “An iterative regularized mirror descent method for ill-posed nondifferentiable stochastic optimization”. In: arXiv preprint arXiv:1901.09506 (2019). arXiv: 1901.09506 [math.OC] (page 3).

[BJQS16] Chenglong Bao, Hui Ji, Yuhui Quan, and Zuowei Shen. “Dictionary Learning for Sparse Coding: Algorithms and Convergence Analysis”. In: IEEE Transactions on Pattern Analysis and Machine Intelligence 38.7 (2016), pp. 1356–1369 (page 5).

[BS14] Amir Beck and Shoham Sabach. “A first order method for finding minimal norm-like solutions of convex optimization problems”. In: Mathematical Programming 147.1-2 (2014), pp. 25–46 (page 3).

[BHTV19] Luca Bertinetto, Joao F. Henriques, Philip Torr, and Andrea Vedaldi. “Meta-learning with differentiable closed-form solvers”. In: International Conference on Learning Representations. 2019 (page 2).

[BNPS17] Jérôme Bolte, Trong Phong Nguyen, Juan Peypouquet, and Bruce W. Suter. “From error bounds to the complexity of first-order descent methods for convex functions”. In: Mathematical Programming 165.2 (2017), pp. 471–507 (page 9).

[BS13] J Frédéric Bonnans and Alexander Shapiro. Perturbation analysis of optimization problems. Springer Science & Business Media, 2013 (page 2).

[BV18] Stephen Boyd and Lieven Vandenberghe. Localization and Cutting-Plane Methods. 2018. url: https://web.stanford.edu/class/ee364b/lectures/localization_methods_notes.pdf (page 6).

[BF93] J. V. Burke and M. C. Ferris. “Weak Sharp Minima in Mathematical Programming”. In: SIAM Journal on Control and Optimization 31.5 (1993), pp. 1340–1359 (page 9).

[BD05] James V. Burke and Sien Deng. “Weak sharp minima revisited, part II: application to linear regularity and error bounds”. In: Mathematical Programming 104.2 (2005), pp. 235–261 (page 9).

[Cab05] Alexandre Cabot. “Proximal Point Algorithm Controlled by a Slowly Vanishing Term: Applications to Hierarchical Minimization”. In: SIAM Journal on Optimization 15.2 (2005), pp. 555–572 (page 2).

[CP16] Antonin Chambolle and Thomas Pock. “On the ergodic convergence rates of a first-order primal–dual algorithm”. In: Mathematical Programming 159.1 (2016), pp. 253–287 (page 21).

[Dem20] Stephan Dempe. “Bilevel Optimization: Theory, Algorithms, Applications and a Bibliography”. In: Bilevel Optimization: Advances and Next Challenges. Cham: Springer International Publishing, 2020, pp. 581–672 (page 4).

[DDDP21] Stephan Dempe, Nguyen Dinh, Joydeep Dutta, and Tanushree Pandit. “Simple bilevel programming and extensions”. In: Mathematical Programming 188.1 (2021), pp. 227–253 (page 2).
Stephen Dempe, Nguyen Dinh, and Joydeep Dutta. “Optimality Conditions for a Simple Convex Bilevel Programming Problem”. In: Variational Analysis and Generalized Differentiation in Optimization and Control: In Honor of Boris S. Mordukhovich. New York, NY: Springer New York, 2010, pp. 149–161 (page 2).

Justin Domke. “Generic Methods for Optimization-Based Modeling”. In: Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics. 2012, pp. 318–326 (page 4).

Joydeep Dutta and Tanushree Pandit. “Algorithms for Simple Bilevel Programming”. In: Bilevel Optimization: Advances and Next Challenges. Cham: Springer International Publishing, 2020, pp. 253–291 (page 2).

Luca Franceschi, Paolo Frasconi, Saverio Salzo, Riccardo Grazzi, and Massimiliano Ponti. “Bilevel Programming for Hyperparameter Optimization and Meta-Learning”. In: Proceedings of the 35th International Conference on Machine Learning. 2018, pp. 1568–1577 (page 2).

Michael P. Friedlander and Paul Tseng. “Exact Regularization of Convex Programs”. In: SIAM Journal on Optimization 18.4 (2008), pp. 1326–1350 (page 2).

Stephen Gould, Basura Fernando, Anoop Cherian, Peter Anderson, Rodrigo Santa Cruz, and Edison Guo. “On differentiating parameterized argmin and argmax problems with application to bi-level optimization”. In: arXiv preprint arXiv:1607.05447 (2016). arXiv: 1607.05447 [math.OC] (page 4).

Michael Grant and Stephen Boyd. CVX: Matlab Software for Disciplined Convex Programming, version 2.1. http://cvxr.com/cvx. Mar. 2014 (page 10).

Michael Grant and Stephen Boyd. “Graph implementations for nonsmooth convex programs”. In: Recent Advances in Learning and Control. Ed. by V. Blondel, S. Boyd, and H. Kimura. Lecture Notes in Control and Information Sciences. http://stanford.edu/~boyd/graph_dcp.html. Springer-Verlag Limited, 2008, pp. 95–110 (page 10).

Erfan Yazdandoost Hamedani and Necdet Serhat Aybat. “A primal-dual algorithm with line search for general convex-concave saddle point problems”. In: SIAM Journal on Optimization 31.2 (2021), pp. 1299–1329 (pages 9, 22).

Niao He, Anatoli Juditsky, and Arkadi Nemirovski. “Mirror prox algorithm for multi-term composite minimization and semi-separable problems”. In: Computational Optimization and Applications 61.2 (2015), pp. 275–319 (page 22).

Mingyi Hong, Hoi-To Wai, Zhaoran Wang, and Zhuoran Yang. “A two-timescale framework for bilevel optimization: Complexity analysis and application to actor-critic”. In: arXiv preprint arXiv:2007.05170 (2020). arXiv: 2007.05170 [math.OC] (page 2).

Martin Jaggi. “Revisiting Frank-Wolfe: Projection-Free Sparse Convex Optimization”. In: Proceedings of the 30th International Conference on Machine Learning. 2013, pp. 427–435 (pages 5, 7, 8, 19).

Kaiyi Ji, Junjie Yang, and Yingbin Liang. “Bilevel optimization: Convergence analysis and enhanced design”. In: International Conference on Machine Learning. 2021, pp. 4882–4892 (page 4).
[Pan97] Jong-Shi Pang. “Error bounds in mathematical programming”. In: *Mathematical Programming* 79.1 (1997), pp. 299–332 (page 9).

[Ped16] Fabian Pedregosa. “Hyperparameter optimization with approximate gradient”. In: *Proceedings of the 33nd International Conference on Machine Learning (ICML)*. 2016 (page 4).

[RFKL19] Aravind Rajeswaran, Chelsea Finn, Sham M Kakade, and Sergey Levine. “Meta-learning with implicit gradients”. In: *Advances in Neural Information Processing Systems*. 2019, pp. 113–124 (page 2).

[Rak13] Alain Rakotomamonjy. “Direct Optimization of the Dictionary Learning Problem”. In: *IEEE Transactions on Signal Processing* 61.22 (2013), pp. 5495–5506 (pages 5, 11).

[Rd20] Vincent Roulet and Alexandre d’Aspremont. “Sharpness, Restart, and Acceleration”. In: *SIAM Journal on Optimization* 30.1 (2020), pp. 262–289 (page 9).

[Roz+21] Benedek Rozemberczki, Paul Scherer, Yixuan He, George Panagopoulos, Alexander Riedel, Maria Asteftanoaei, Oliver Kiss, Ferenc Beres, Guzmán López, Nicolas Collignon, and Rik Sarkar. “PyTorch Geometric Temporal: Spatiotemporal Signal Processing with Neural Machine Learning Models”. In: *The 30th ACM International Conference on Information & Knowledge Management*. 2021, pp. 4564–4573 (pages 10, 23).

[SS17] Shoham Sabach and Shimrit Shtern. “A first order method for solving convex bilevel optimization problems”. In: *SIAM Journal on Optimization* 27.2 (2017), pp. 640–660 (pages 3, 8).

[SCHB19] Amirreza Shaban, Ching-An Cheng, Nathan Hatch, and Byron Boots. “Truncated back-propagation for bilevel optimization”. In: *The 22nd International Conference on Artificial Intelligence and Statistics*. 2019, pp. 1723–1732 (page 2).

[SVZ21] Yekini Shehu, Phan Tu Vuong, and Alain Zemkoho. “An inertial extrapolation method for convex simple bilevel optimization”. In: *Optimization Methods and Software* 36.1 (2021), pp. 1–19 (pages 2, 3).

[Sol07] Mikhail Solodov. “An explicit descent method for bilevel convex optimization”. In: *Journal of Convex Analysis* 14.2 (2007), p. 227 (page 2).

[SJGL22] Daouda Sow, Kaiyi Ji, Ziwei Guan, and Yingbin Liang. “A Constrained Optimization Approach to Bilevel Optimization with Multiple Inner Minima”. In: *arXiv preprint arXiv:2203.01123* (2022). arXiv: 2203.01123 [math.OC] (page 4).

[TA77] A.N. Tikhonov and V.Y. Arsenin. *Solutions of Ill-Posed Problems*. New York: Wiley, 1977 (page 2).

[Xu04] Hong-Kun Xu. “Viscosity approximation methods for nonexpansive mappings”. In: *Journal of Mathematical Analysis and Applications* 298.1 (2004), pp. 279–291 (page 2).

[Xu21] Yangyang Xu. “Iteration complexity of inexact augmented Lagrangian methods for constrained convex programming”. In: *Mathematical Programming* 185.1 (2021), pp. 199–244 (page 22).

[YBD09] Mehrdad Yaghoobi, Thomas Blumensath, and Mike E. Davies. “Dictionary Learning for Sparse Approximations With the Majorization Method”. In: *IEEE Transactions on Signal Processing* 57.6 (2009), pp. 2178–2191 (page 5).
[Yam01] Isao Yamada. “The Hybrid Steepest-Descent Method for Variational Inequality Problems over the Intersection of the Fixed-Point Sets of Nonexpansive Mappings”. In: Inherently Parallel Algorithms in Feasibility and Optimization and Their Applications. North-Holland, 2001, pp. 473–504 (page 2).

[You21] Farzad Yousefian. “Bilevel Distributed Optimization in Directed Networks”. In: 2021 American Control Conference (ACC). 2021, pp. 2230–2235 (page 3).
Appendix

A  Supporting lemmas

A.1  Proof of Lemma 3.1

Let $x^*_g$ be any point in $X^*_g$, i.e., any optimal solution of the lower-level problem. By definition, we have $g(x^*_g) = g^*$. Since $g$ is convex and $g^* \leq g(x_0)$, we have

$$g(x_0) - g(x_k) \geq g^* - g(x_k) \geq \langle \nabla g(x_k), x^*_g - x_k \rangle,$$

which implies $x^*_g \in X_k$. Hence, we conclude that $X^*_g \subseteq X_k$.

A.2  Improvement in one step

The following lemma characterizes the improvement of both the upper-level and lower-level objective values after one step of Algorithm 1.

Lemma A.1. Let $\{x_k\}_{k=0}^K$ be the sequence generated by Algorithm 1. Suppose Assumption 2.1 holds, then for any $k \geq 0$ we have

$$f(x_{k+1}) \leq f(x_k) - \gamma_k G(x_k) + \frac{1}{2} \gamma_k^2 L_f D^2,$$

$$g(x_{k+1}) \leq (1 - \gamma_k) g(x_k) + \gamma_k g(x_0) + \frac{1}{2} \gamma_k^2 L_g D^2.\quad (14)$$

Proof. Since the gradient of $f$ is $L_f$-Lipschitz and $Z$ is bounded with diameter $D$, we have

$$f(x_{k+1}) \leq f(x_k) + \langle \nabla f(x_k), x_{k+1} - x_k \rangle + \frac{1}{2} L_f \|x_{k+1} - x_k\|^2$$

$$= f(x_k) + \gamma_k \langle \nabla f(x_k), s_k - x_k \rangle + \frac{1}{2} L_f \gamma_k \|s_k - x_k\|^2$$

$$\leq f(x_k) + \gamma_k \langle \nabla f(x_k), s_k - x_k \rangle + \frac{1}{2} L_f \gamma_k^2 D^2.\quad (15)$$

Now using the definition of $s_k$ in (10), the definition fo $G(x)$ in (7) and Lemma 3.1, we obtain

$$\langle \nabla f(x_k), s_k - x_k \rangle = \min_{s \in \mathcal{X}_k} \langle \nabla f(x_k), s - x_k \rangle \leq \min_{s \in \mathcal{X}_g^*} \langle \nabla f(x_k), s - x_k \rangle = -G(x_k).\quad (16)$$

Then (13) follows from (15) and (16).

Similarly, since the gradient of $g$ is $L_g$-Lipschitz, we have

$$g(x_{k+1}) \leq g(x_k) + \gamma_k \langle \nabla g(x_k), s_k - x_k \rangle + \frac{1}{2} L_g \gamma_k^2 D^2.\quad (17)$$

Moreover, since $s_k \in \mathcal{X}_k$, from the definition of $\mathcal{X}_k$ in (10) we get $\langle \nabla g(x_k), s_k - x_k \rangle \leq g(x_0) - g(x_k)$. Combining this with (17) leads to (14).
B Proof of Theorem 4.1

We first prove the convergence rate of the upper-level objective $f$, which largely mirrors the standard analysis of the FW method [Jag13]. Since $x^* \in X_g^*$ and $f$ is convex, from the definition of $G(x_k)$ in (7) we have

$$G(x_k) = \max_{s \in X_g^*} \{ \langle \nabla f(x_k), x_k - s \rangle \} \geq \langle \nabla f(x_k), x_k - x^* \rangle \leq f(x_k) - f^*. \quad (18)$$

Subtracting $f^*$ from both sides of (13) in Lemma A.1 and using (18), we obtain that

$$f(x_{k+1}) - f^* \leq (1 - \gamma_k)(f(x_k) - f^*) + \frac{1}{2} \gamma_k^2 L_f D^2. \quad (19)$$

Now define $A_k = k(k+1)$. By substituting $\gamma_k = 2/(k+2)$ and multiplying both sides of (19) by $A_k$, we get

$$A_{k+1}(f(x_{k+1}) - f^*) \leq A_k(f(x_k) - f^*) + \frac{2(k+1)}{k+2} L_f D^2 \leq A_k(f(x_k) - f^*) + 2L_f D^2.$$

Hence, if follows from induction that

$$A_K(f(x_K) - f^*) \leq A_0(f(x_0) - f^*) + 2KL_f D^2 \Rightarrow f(x_K) - f^* \leq \frac{2KL_f D^2}{A_k} = \frac{2L_f D^2}{K+1}.$$

This completes the first part of the proof.

The proof for the lower-level problem follows from similar arguments. By subtracting $g(x_0)$ from both sides of (14) in Lemma A.1, we have

$$g(x_{k+1}) - g(x_0) \leq (1 - \gamma_k)(g(x_k) - g(x_0)) + \frac{1}{2} \gamma_k^2 L_g D^2. \quad (20)$$

By substituting $\gamma_k = 2/(k+2)$ and multiplying both sides of (20) by $A_k$, we obtain

$$A_{k+1}(g(x_{k+1}) - g(x_0)) \leq A_k(g(x_k) - g(x_0)) + 2L_g D^2.$$

Hence, if follows from induction that

$$A_K(g(x_K) - g(x_0)) \leq 2KL_g D^2 \Rightarrow g(x_K) - g(x_0) \leq \frac{2KL_g D^2}{A_k} = \frac{2L_g D^2}{K+1}.$$

Since $g(x_0) - g^* \leq \epsilon_g/2$, we obtain

$$g(x_K) - g^* \leq \frac{2L_g D^2}{K+1} + \frac{1}{2} \epsilon_g,$$

which completes the proof.

C Proof of Theorem 4.2

Since we use a fixed stepsize in Theorem 4.2, in the following we will write $\gamma_k = \gamma$. 

19
We first consider the upper-level objective $f$. The analysis here is similar to the one in [MOJ18]. By using (13) in Lemma A.1, we have

$$G(x_k) \leq \frac{f(x_k) - f(x_{k+1})}{\gamma} + \frac{1}{2} \gamma L_f D^2.$$  

Summing both sides of the above inequality from $k = 0$ to $K - 1$, we get

$$\sum_{k=0}^{K-1} G(x_k) \leq \frac{f(x_0) - f(x_K)}{\gamma} + \frac{1}{2} K \gamma L_f D^2 \leq \frac{f(x_0) - f}{\gamma} + \frac{1}{2} K \gamma L_f D^2,$$

where we used the fact that $f(x_K) \geq \frac{f}{\gamma} = \min_{x \in \mathcal{Z}} f(x)$. This further implies that

$$\min_{0 \leq k \leq K-1} G(x_k) \leq \frac{1}{K} \sum_{k=0}^{K-1} G(x_k) \leq \frac{f(x_0) - f}{\gamma K} + \frac{1}{2} \gamma L_f D^2. \quad (21)$$

To upper bound the right-hand side of (21), note that our choices of the stepsize $\gamma$ and the number of iterations $K$ satisfy

$$\gamma \leq \frac{\epsilon_f}{L_f D^2} \quad \text{and} \quad K \geq \frac{2(f(x_0) - f)}{\epsilon_f \gamma}.$$  

Thus, we have

$$\min_{0 \leq k \leq K-1} G(x_k) \leq \frac{f(x_0) - f}{\gamma K} + \frac{1}{2} \gamma L_f D^2 \leq \frac{\epsilon_f}{2} + \frac{\epsilon_f}{2} = \epsilon_f.$$  

This guarantees that $G(x_k) \leq \epsilon_f$ by choosing $k^* = \arg\min_{0 \leq k \leq K-1} G(x_k)$.

Now we move to the analysis of the lower-level objective $g$. For any $k \geq 0$, by applying induction on (14) in Lemma A.1 it follows that

$$g(x_k) - g(x_0) \leq \frac{1}{2} L_g D^2 \sum_{j=0}^{k-1} \gamma^2 (1 - \gamma)^j \leq \frac{1}{2} L_g D^2 \gamma,$$

where we used $\sum_{j=0}^{k-1} (1 - \gamma)^j \leq 1/\gamma$ in the last inequality. Furthermore, since $g(x_0) - g^* \leq \epsilon_g/2$ and $\gamma \leq \frac{\epsilon_f}{L_f D^2}$, this implies that $g(x_k) - g^* \leq \frac{1}{2} \epsilon_g + \frac{1}{2} \epsilon_g = \epsilon_g$ for any $0 \leq k \leq K - 1$. In particular, we can take $k = k^*$ and conclude that $g(x_{k^*}) - g^* \leq \epsilon_g$. This completes the proof.

## D Proofs under Hölderian error bound assumption

### D.1 Proof of Proposition 4.3

Since $X_g^*$ is closed and compact, we can let $\hat{x}^* = \arg\min_{x \in X_g^*} \|x - \hat{x}\|$ such that $\|\hat{x}^* - \hat{x}\| = \text{dist}(\hat{x}, X_g^*)$.

By Assumption 4.1, we obtain

$$\frac{\alpha}{r} \|\hat{x}^* - \hat{x}\| \leq g(\hat{x}) - g^* \leq \epsilon_g \quad \Longleftrightarrow \quad \|\hat{x}^* - \hat{x}\| \leq \left(\frac{r \epsilon_g}{\alpha}\right)^{\frac{1}{2}}.$$  

When $f$ is convex, we have

$$f(\hat{x}) - f^* = f(\hat{x}) - f(\hat{x}^*) \geq \langle \nabla f(\hat{x}^*), \hat{x} - \hat{x}^* \rangle \geq -\|\nabla f(\hat{x}^*)\| \|\hat{x} - \hat{x}^*\| \geq -M \left(\frac{r \epsilon_g}{\alpha}\right)^{\frac{1}{2}},$$

where $M$ is the strong convexity constant of $f$. For $0 < r < 1$, we can choose $\alpha = \frac{r \epsilon_g}{\sqrt{M}}$ and $\epsilon_g = \epsilon_f$ to satisfy the conditions of the proposition.
where we used the convexity of \( f \) in the first inequality. When \( f \) is non-convex, we have

\[
\mathcal{G}(\hat{x}) = \max_{s \in \mathcal{X}^*_g} \{\langle \nabla f(\hat{x}), \hat{x} - s \rangle \} \geq \langle \nabla f(\hat{x}), \hat{x} - \hat{x}^* \rangle = \langle \nabla f(\hat{x}) - \nabla f(\hat{x}^*), \hat{x} - \hat{x}^* \rangle \geq -\|\nabla f(\hat{x}) - \nabla f(\hat{x}^*)\|_* \|\hat{x} - \hat{x}^*\| - \|\nabla f(\hat{x}^*)\| \|\hat{x} - \hat{x}^*\| \geq -L_f \|\hat{x} - \hat{x}^*\|^2 - M \|\hat{x} - \hat{x}^*\| \geq -M \left( \frac{r_e \alpha}{\alpha} \right)^{1/2} - L_f \left( \frac{r_e \alpha}{\alpha} \right)^{1/2},
\]

where we used the fact that \( \nabla f = L_f \) -Lipschitz in (22). This completes the proof.

### D.2 Proof of Corollary 4.4

In the first case, where \( f \) is convex, we set \( \epsilon_g = \frac{a}{\epsilon_f} \left( \frac{\epsilon_f}{2M} \right)^r \). By Theorem 4.1, we have \( f(x_K) - f^* \leq \epsilon_f \) and \( g(x_K) - g^* \leq \epsilon_g \) when

\[
K \geq \max \left\{ \frac{2L_f D^2}{\epsilon_f} - 1, \frac{4L_g D^2}{\epsilon_g} - 1 \right\} = \max \left\{ \frac{2L_f D^2}{\epsilon_f} - 1, \frac{4rM_r L_g D^2}{\alpha \epsilon_f} - 1 \right\} = \mathcal{O} \left( \frac{1}{\epsilon_f^r} \right).
\]

Moreover, Proposition 4.3 implies that \( f(x_K) - f^* \geq -M \left( \frac{r_e \alpha}{\alpha} \right)^{1/2} \geq -\epsilon_f \). Putting all pieces together, we conclude that \( |f(x_K) - f^*| \leq \epsilon_f \) and \( g(x_K) - g^* \leq \epsilon_g \) after \( K = \mathcal{O}(1/\epsilon_f^r) \) iterations.

In the second case, where \( f \) is non-convex, we set \( \epsilon_g = \min \left\{ \frac{\epsilon_f}{2M}, \frac{\epsilon_f}{2L_f} \right\} \). By Theorem 4.2, we can find \( k^* \in \{0, 1, \ldots, K - 1\} \) such that \( \mathcal{G}(x_{k^*}) \leq \epsilon_f \) and \( g(x_{k^*}) - g^* \leq \epsilon_g \) when

\[
K \geq (f(x_0) - f) \cdot \max \left\{ \frac{2L_f D^2}{\epsilon_f^2}, \frac{2L_g D^2}{\epsilon_f \epsilon_g} \right\} = (f(x_0) - f) \cdot \max \left\{ \frac{2L_f D^2}{\epsilon_f^2}, \frac{2r(2M)^r L_g D^2}{\alpha \epsilon_f^{r+1}}, \frac{2r(2L_f) \frac{r}{2} L_g D^2}{\alpha \epsilon_f^{r+1}} \right\} = \mathcal{O} \left( \epsilon_f^{-1} \right).
\]

Moreover, Proposition 4.3 implies that \( \mathcal{G}(x_{k^*}) \geq -M \left( \frac{r_e \alpha}{\alpha} \right)^{1/2} - L_f \left( \frac{r_e \alpha}{\alpha} \right)^{1/2} \geq -\epsilon_f - \epsilon_f = -\epsilon_f \). Thus, we conclude \( |\mathcal{G}(x_{k^*})| \leq \epsilon_f \) and \( g(x_{k^*}) - g^* \leq \epsilon_g \) after \( K = \mathcal{O}(1/\epsilon_f^{r+1}) \) iterations.

### E Primal-dual method for the bilevel problem

In this section, we discuss the convergence rate of primal-dual type methods for solving the bilevel problem in (1). We consider the setting as in Theorem 4.1, in which both \( f \) and \( g \) are convex and smooth. To simplify the discussion, we further assume \( Z = \{z \in \mathcal{X} \mid Az \leq b\} \) where \( A \in \mathbb{R}^{m \times n}, \ b \in \mathbb{R}^m \), and \( \mathcal{X} \) is a convex and easy-to-project compact set.

To obtain the reformulation in (2), one first needs to estimate the optimal value \( g^* \) of the lower-level problem. Since it is a convex program with linear constraints, we can implement a first-order primal-dual method (e.g., [CP16]) to find \( g_0 \) such that \( |g_0 - g^*| \leq \epsilon_g / 4 \) within at most \( \mathcal{O}(\frac{L_g + \|A\|}{\epsilon_g}) \) iterations\(^1\). Next, Problem (1) can be cast as the following convex optimization problem with linear

\(^1\)Note that this complexity can be improved to the optimal rate of \( \mathcal{O}(\frac{L_g + \|A\|}{\epsilon_g}) \) using an accelerated method.
and nonlinear convex constraints:
\[
\min_{x \in \mathcal{X}} \, f(x) \quad \text{s.t.} \quad Ax \leq b, \; g(x) \leq g_0 + \frac{\epsilon_g}{2},
\]  
(23)
where we add the term \( \frac{\epsilon_g}{2} \) to ensure that the Slater’s condition holds. Now we can apply any classic or accelerated first-order primal-dual methods [HIJN15; Xu21; HA21] to find a solution of Problem (23) that is both \( \epsilon_f \)-suboptimal and \( \frac{\epsilon_g}{2} \)-infeasible. For example, the optimal convergence rates obtained in [Xu21] and [HA21] imply that after \( K \) iterations, the average iterate \( \bar{x}_K \) satisfies
\[
\max \{ \left| f(\bar{x}_K) - f(x^*) \right|, \left| g(\bar{x}_K) - g(x^*) \right| \} \leq \Delta/K,
\]
where \( x^*_k \) denotes an optimal solution of Problem (23), \( \Delta \triangleq O((L_f + L_g + C_g)D^2 + C_g |\lambda_1|^2 + \|A\| \|\lambda_2^*\|^2) \), \( C_g \) is the Lipschitz constant of \( g \), and \( \lambda_1^* \in \mathbb{R} \) and \( \lambda_2^* \in \mathbb{R}^m \) denote an arbitrary dual optimal solution corresponding to the nonlinear and linear constraints in Problem (23), respectively. Using the fact that \( f(x^*_k) \leq f(x^*) \) and \( g(x^*_k) \leq g_0 + \frac{\epsilon_g}{2} \leq g^* + \frac{3}{4} \epsilon_g \), we conclude
\[
f(\bar{x}_K) - f(x^*) \leq \Delta/K \quad \text{and} \quad |g(\bar{x}_K) - g(x^*)| \leq \Delta/K + \frac{3}{4} \epsilon_g.
\]
Therefore, to achieve an \((\epsilon_f, \epsilon_g)\)-optimal solution of Problem (1), a primal-dual method overall requires \( O \left( \frac{L_f + \|A\|}{\epsilon_f} + \frac{\Delta}{\min\{\epsilon_f, \epsilon_g\}} \right) \) primal-dual gradient calls, while our proposed method overall requires \( O \left( \frac{L_g}{\epsilon_g} + \frac{(L_f + L_g)D^2}{\min\{\epsilon_f, \epsilon_g\}} \right) \) linear minimization oracle calls. In particular, we observe that the convergence guarantee of primal-dual methods heavily rely on the norm of the dual optimal variable \( |\lambda_1^*| \), which may tend to infinity as \( \epsilon \) approaches zero and the problem in (23) becomes nearly degenerate.

F Additional motivating examples

In this section, we provide two more examples for the bilevel problem in (1).

**Example 3** (Ill-posed optimization). Consider an empirical risk minimization problem in the form of \( \min_{x \in \mathcal{Z}} \frac{1}{m} \sum_{i=1}^{m} \ell((a_i, x), b_i) \), where \( \mathcal{Z} \subseteq \mathbb{R}^n \) is the constraint set and \( \ell \) is the loss function corresponding to input \( a_i \) and its associated label \( b_i \). Without an explicit regularization, this problem can be ill-posed, i.e., it has multiple optimal solutions or is sensitive to small perturbation in the input data. To tackle this issue, we can consider a secondary objective function \( \mathcal{R}(\cdot) \) as another criterion to select one of the optimal solutions with some desired property. For example, we can find the minimal \( l_2 \)-norm solution by choosing \( \mathcal{R}(x) = \|x\|_2^2 \). Such a problem can be formulated as the following bilevel problem:
\[
\min_{x \in \mathbb{R}^n} \mathcal{R}(x) \quad \text{s.t.} \quad x \in \arg\min_{z \in \mathcal{Z}} \frac{1}{m} \sum_{i=1}^{m} \ell((a_i, z), b_i).
\]

**Example 4** (Representation learning). In meta-learning problems, we aim to pre-train a model that can be easily fine-tuned to new tasks. This can be often achieved by learning a compact representation that is shared among multiple tasks. In particular, consider a multi-task linear representation learning problem with \( T \) tasks at the training time. We assume that the data points for the \( i \)-th task are generated according to \( y_i^j = w_i^* \mathbf{B}_i^* x_i^j + n_i^j \) for \( j = 1, \ldots, m_i \), where \( n_i^j \) is some random noise and \( \mathbf{B}_i^* \in \mathbb{R}^{k \times d} \) is a common representation that maps the input in \( \mathbb{R}^d \) to a
lower dimensional feature vector in $\mathbb{R}^k$. When we have access to a diverse set of tasks such that their heads $\{\mathbf{w}_i^*\}_{i=1}^T$ span $\mathbb{R}^k$, it is shown that one can find the ground truth representation $\mathbf{B}^*$ by solving the following problem:

$$
\min_{\mathbf{B}} \min_{\mathbf{w}_1,\ldots,\mathbf{w}_T} \sum_{i=1}^T \sum_{j=1}^{m_i} \left( y_{ij}^* - \mathbf{w}_i^\top \mathbf{B}^\top \mathbf{x}_{ij} \right)^2 \quad \text{s.t.} \quad \|\mathbf{B}\|_F \leq \Delta, \|\mathbf{w}_i\|_1 \leq \delta, \quad i = 1, \ldots, T,
$$

where we impose the norm constraints on $\mathbf{B}$ and $\{\mathbf{w}_i\}_{i=1}^T$ for some parameters $\Delta, \delta > 0$ to resolve the scale invariance of the problem.

However, if the tasks at the training time are not diverse enough, then we can only learn a partial representation, i.e., a subset of the feature maps in $\mathbf{B}^*$. One way to further improve the learned representation is to leverage the new tasks we observe during the test time. Concretely, let $\hat{\mathbf{w}}_1^*, \ldots, \hat{\mathbf{w}}_T^*$ and $\hat{\mathbf{B}}_{tr}^*$ denote the output of the training procedure. When we are given a new task at the test time, we can improve the representation $\hat{\mathbf{B}}_{tr}^*$ by solving the following bilevel problem:

$$
\min_{\mathbf{B} \in \mathbb{R}^k \times d} \min_{\mathbf{w}_{T+1}} f(\mathbf{B}, \mathbf{w}_{T+1}) \quad \text{s.t.} \quad \mathbf{B} \in \arg\min_{\|\mathbf{B}\|_F \leq \Delta} g(\mathbf{B}), \|\mathbf{w}_{T+1}\|_1 \leq \delta, \quad (24)
$$

where $f(\mathbf{B}, \mathbf{w}_{T+1}) \triangleq \sum_{j=1}^{m_{T+1}} (y^*_{T+1,j} - \mathbf{w}_{T+1}^\top \mathbf{B}^\top \mathbf{x}_{j,T+1})^2$ is the loss over the test set and $g(\mathbf{B}) \triangleq \sum_{i=1}^T \sum_{j=1}^{m_i} (y_{ij}^* - \hat{\mathbf{w}}_i^* \mathbf{B}_i^* \mathbf{x}_{ij})^2$ is the loss over the training set. The rationale is that the solution of Problem (24) can fit to both the old training tasks and the new test task, and hence is a better approximation of $\mathbf{B}^*$ compared to $\hat{\mathbf{B}}_{tr}^*$. This way, we maintain the feature maps learned at the training time and at the same time learn new feature maps from the test task. Note that in Problem (24) the upper-level function is nonconvex, while the lower-level problem is convex with multiple solutions.

\section*{G Experiment details}

In this section, we include more details of the numerical experiments in Section 5. The MATLAB code is also included in the supplemental material.

\subsection*{G.1 Over-parametrized regression}

\textbf{Dataset generation.} The original Wikipedia Math Essential dataset [Roz+21] consists of an 1068×731 matrix. We randomly select one of the columns as the outcome vector $\mathbf{b} \in \mathbb{R}^{1068}$ and the rest as the data matrix $\mathbf{A} \in \mathbb{R}^{1068 \times 730}$.

\textbf{Initialization.} We run the standard FW algorithm with the stepsize chosen as $2/(k + 2)$ on the lower-level problem in (4). We terminate the procedure once the FW gap is no more than $\epsilon_g/2 = 5 \times 10^{-5}$.

\textbf{Implementation details.} For our GBFW algorithm, we choose the stepsizes as $\gamma_k = 2/(k + 12)$ to avoid instability due to large stepsizes. In the setup of Problem (1), the a-IRG algorithm in [KY21a] is given by

$$
\mathbf{x}_{k+1} = \Pi_Z(\mathbf{x}_k - \gamma_k (\nabla g(\mathbf{x}_k) + \eta_k \nabla f(\mathbf{x}_k))),
$$

where $\Pi_Z(\cdot)$ denotes the Euclidean projection onto the set $Z$, $\gamma_k$ is the stepsize, and $\eta_k$ is the regularization parameter. In our experiment, we choose $\gamma_k = 0.01/\sqrt{k + 1}$ and $\eta_k = 1/(k + 1)^{1/4}$. 

Figure 4: The performance of GBFW, a-IRG and the baseline FW algorithms on the bilevel problem in (6) for dictionary learning.

G.2 Dictionary learning

**Dataset generation.** Each of the sparse coefficient vectors \( \{x_i\}_{i=1}^{250} \) and \( \{x'_k\}_{k=1}^{200} \) has 5 nonzero entries, whose locations are randomly chosen. Also, the absolute values of those nonzero weights are drawn uniformly from the interval \([0.2, 1]\). The entries of the random noise vectors \( \{n_i\}_{i=1}^{250} \) and \( \{n'_k\}_{k=1}^{200} \) follow i.i.d. Gaussian distribution with mean 0 and standard deviation 0.01.

**Initialization.** The initialization consists of two phases. In the first phase, we run the standard FW algorithm on both the variables \( D \in \mathbb{R}^{25 \times 40} \) and \( X \in \mathbb{R}^{40 \times 250} \) for \( 10^4 \) iterations with the stepsize chosen as \( 1/\sqrt{k} + 1 \) (\( k \geq 0 \) is the iteration counter). Then in the second phase, we keep the variable \( X \) fixed and update \( D \) using the standard FW algorithm with exact line search. We terminate the procedure and output \( \hat{D} \) and \( \hat{X} \) until the FW gap with respect to \( D \) is no more than \( \epsilon_g = 10^{-6} \).

**Implementation details.** We choose \( \delta = 3 \) in both Problems (5) and (6). All three algorithms start from the same initial point. We initialize \( \tilde{D} \in \mathbb{R}^{25 \times 50} \) as the concatenation of \( \hat{D} \in \mathbb{R}^{25 \times 40} \) and 10 columns of all zeros. Moreover, we initialize the variable \( \tilde{X} \) randomly by drawing its entries from a standard Gaussian distribution and normalizing each column to have a \( \ell_1 \)-norm of \( \delta \). For our GBFW algorithm, we choose the stepsize as \( \gamma_k = 0.3/\sqrt{k+1} \) instead of a constant stepsize as suggested by Theorem 4.2. Empirically, we observe that this leads to faster convergence. The same stepsize rule is also used in the baseline FW algorithm. For the a-IRG algorithm, we use the same stepsize as the one in Section G.1.

We also repeat the experiment for 10 times using different random seeds and the results are reported in Fig. 4. The solid lines represent the average statistics over 10 runs, while the shaded areas around each line indicate the range of all random instances.