# Inexact bilevel stochastic gradient methods for constrained and unconstrained lower-level problems

T. Giovannelli* G. Kent† L. N. Vicente‡

December 7, 2022

**Abstract**

Two-level stochastic optimization formulations have become instrumental in a number of machine learning contexts such as continual learning, neural architecture search, adversarial learning, and hyperparameter tuning. Practical stochastic bilevel optimization problems become challenging in optimization or learning scenarios where the number of variables is high or there are constraints.

In this paper, we introduce a bilevel stochastic gradient method for bilevel problems with lower level constraints. We also present a comprehensive convergence theory that covers all inexact calculations of the adjoint gradient (also called hypergradient) and addresses both the lower-level unconstrained and constrained cases. To promote the use of bilevel optimization in large-scale learning, we introduce a practical bilevel stochastic gradient method (BSG-1) that does not require second-order derivatives and, in the lower-level unconstrained case, dismisses any system solves and matrix-vector products.

## 1 Introduction

Many real-world applications are naturally formulated using hierarchical objectives, which are organized into different nested levels. In the bilevel case, the main goal is placed into an upper optimization level, while the lower optimization level aims to determine the best response to a decision made in the upper level. Bilevel optimization has a rich literature of algorithmic development and theory (see [1, 11, 14, 50, 55] for extensive surveys and books on this topic). The main applications are found in game theory, defense industry, and optimal structural design, and one has recently seen a surge of contributions to machine learning (ML) (see, e.g., [17, 28], and the recent review [29]).

*Department of Industrial and Systems Engineering, Lehigh University, Bethlehem, PA 18015-1582, USA (tog220@lehigh.edu).

†Department of Industrial and Systems Engineering, Lehigh University, Bethlehem, PA 18015-1582, USA (gdk220@lehigh.edu).

‡Department of Industrial and Systems Engineering, Lehigh University, Bethlehem, PA 18015, USA (lnv@lehigh.edu). Support for this author was partially provided by the Centre for Mathematics of the University of Coimbra under grant FCT/MCTES UIDB/MAT/00324/2020.

In this paper, we consider the following nonlinear bilevel optimization problem (BLP) formulation

$$\begin{aligned} \min_{x\in\mathbb{R}^n,\, y\in\mathbb{R}^m} \quad & f_u(x,y) \\ \text{s.t.} \quad & x \in X \\ & y \in \operatorname*{argmin}_{y\in Y(x)} \quad f_\ell(x,y). \end{aligned} \tag{BLP}$$

The goal of the upper level (UL) problem is to determine the optimal value of the UL function $f_u : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$, where the UL variables $x$ are subjected to UL constraints ($x \in X$) and the LL variables $y$ are subjected to being an optimal solution of the lower level (LL) problem. In the LL problem, the LL function $f_\ell : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ is optimized in the LL variables $y$, subject to the LL constraints $y \in Y(x)$. Since the goal of this paper is to propose and analyze a general optimization methodology for a stochastic BLP, the LL problem is assumed to be well-defined, in the sense of having a unique solution $y(x)$ for all $x \in X$. Hence, problem BLP is equivalent to a problem posed solely in the UL variables:

$$\min_{x\in\mathbb{R}^n} \; f(x) = f_u(x, y(x)) \quad \text{s.t.} \quad x \in X. \tag{1.1}$$

Also, note that the UL constraints ($x \in X$) are only posed in the UL variables $x$ as otherwise problem BLP could become intractable due to a disconnected feasible region in the $(x, y)$–space. Whenever applying orthogonal projections within stochastic gradient type methods, the sets $X$ and $Y(x)$ will be assumed closed and convex.

Under appropriate smoothness and non-singularity assumptions, the gradient of $f$ at $x$, when $Y(x) = \mathbb{R}^m$, is given by the well-known so-called adjoint gradient (also called hypergradient in the ML community)

$$\nabla f \;=\; \nabla_x f_u - \nabla^2_{xy} f_\ell \nabla^2_{yy} f_\ell^{-1} \nabla_y f_u, \tag{1.2}$$

where all gradients and Hessians on the right-hand side are evaluated at $(x, y(x))$. We denote the steepest descent direction for $f$ at $x$ as $d(x, y(x)) = -\nabla f(x)$. One arrives at the adjoint formula by first applying the chain rule to $f_u(x, y(x))$

$$\nabla f \;=\; \nabla_x f_u + \nabla y \nabla_y f_u. \tag{1.3}$$

Then, the Jacobian $\nabla y(x)^\top$ of $y(x)$ can be calculated through the (sensitivity) equations $\nabla_y f_\ell(x, y(x)) = 0$. In particular, by taking the derivative of both sides of the equation with respect to $x$, utilizing the chain rule and the implicit differentiation theorem, we obtain $\nabla^2_{yx} f_\ell + \nabla^2_{yy} f_\ell \nabla y^\top = 0$ (all Hessians are evaluated at $(x, y(x))$) which yields

$$\nabla y \;=\; -\nabla^2_{xy} f_\ell \nabla^2_{yy} f_\ell^{-1}. \tag{1.4}$$

Two approaches have been proposed in the literature to deal with $\nabla^2_{yy} f_\ell^{-1}$ in (1.2). One option is to compute the adjoint gradient by first solving the linear system given by the adjoint equation $\nabla^2_{yy} f_\ell \, \lambda = \nabla_y f_u$ for the adjoint variables $\lambda = \lambda(x, y(x))$, and then calculating $\nabla_x f_u - \nabla^2_{xy} f_\ell \, \lambda$. The second option is to truncate the Neumann series given by $\nabla^2_{yy} f_\ell^{-1} = \sum_{i=0}^{\infty} (I - \nabla^2_{yy} f_\ell)^i$.

When $Y(x) \neq \mathbb{R}^m$, it is still possible to use an adjoint formula to compute the gradient of $f$ at $x$ by using sensitivity arguments from nonlinear programming. Such an LL constrained case will be addressed in Section 2.2.

### 1.1 Bilevel machine learning

A variety of problems arising in machine learning can be formulated in terms of bilevel optimization: continual learning, neural architecture search, adversarial training, and hyperparameter tuning are among the most popular examples (see [29] for a review on this topic).

Continual Learning (CL) aims to train ML models when the static task usually considered in learning problems (classification, regression, etc.) is replaced by a sequence of tasks that become available one at a time [31], and for which training and validation datasets are increasingly larger. For each task, a CL instance is formulated as a bilevel problem, where at the UL problem one minimizes the validation error on a subset of model parameters (which includes all hyperparameters), and at the LL problem the training error is minimized on the remaining parameters. Then, a sequence of bilevel problems (one for each task) is solved. In a sense, CL is close to meta-learning [24], where the goal is to determine the best learning process. The increasing interest in CL is motivated by the demand for approaches that help neural networks to learn new tasks without forgetting the previous ones, a phenomenon which is referred to as *catastrophic forgetting* [18, 22, 35, 52]. Another relevant ML area where bilevel optimization is used is Neural Architecture Search (NAS) for Deep Learning. The goal of this problem is to automate the task of designing Deep Neural Networks (DNNs) such that the network's prediction error is minimized. In recent years, NAS has been proposed in a bilevel optimization formulation [28].

Finally, two other popular classes of ML problems that can be formulated by using bilevel optimization are adversarial training and hyperparameter tuning. Adversarial training aims to robustly address adversarial examples [53] which cannot be correctly classified by ML models once a small perturbation is applied. The adversarial training problem is handled by solving a min-max problem [26], which can be reformulated as a bilevel one. The max/LL problem is posed on the variables which perturb the data in a worst-case fashion, where the min/UL problem attempts to minimize the training error on the ML model parameters [21, 34]. Hyperparameter tuning aims to find the best value for the hyperparameters used in a ML model in order to increase its *generalization capability* [3, 6, 15, 17]. In the bilevel formulations proposed in the literature, the UL problem optimizes the validation error over the hyperparameters, while the LL problem has the goal of finding the ML model parameters (e.g., NN weights) that minimize the training error.

### 1.2 Bilevel stochastic descent

In bilevel stochastic optimization, $f_u$ and $f_\ell$ can be interpreted as expected values, namely, $f_u = \mathbb{E}[f_u(x, y, \vartheta_u)]$ and $f_\ell = \mathbb{E}[f_\ell(x, y, \vartheta_\ell)]$, where $\vartheta_u$ and $\vartheta_\ell$ are random variables defined in a probability space (with probability measure independent from $x$ and $y$) such that i.i.d. samples can be observed or generated. The same applies to the functions possibly defining $Y(x)$. (To keep notation simple, we are using the same $f_u$ and $f_\ell$ for deterministic and random variants.)

Having in mind the ML context, the methods that we are considering are Stochastic Approximation (SA) techniques, of the type of the stochastic gradient (SG) method [10, 44, 46] for single-objective optimization. In fact, the bilevel stochastic gradient (BSG) method can be seen as a SG method applied to (1.1), which leads to $x_{k+1} = x_k - \alpha_k g_k^{\mathrm{BSG}}$, where $\alpha_k$ is the stepsize or learning rate and $g_k^{\mathrm{BSG}}$ is a stochastic gradient of $f$. Such a stochastic gradient is obtained by sampling the gradients and Hessians in (1.2) at $(x_k, \tilde{y}_k)$, with $\tilde{y}_k$ denoting an approximation

to the LL optimal solution. The stochastic gradient $g_k^{\text{BSG}}$ is inexact when $\tilde{y}_k \neq y(x_k)$, even in the full-batch (deterministic) case.

In general, BSG methods have only been considered for the LL unconstrained case (i.e., $Y(x) = \mathbb{R}^m$) and are commonly classified according to the approach used to compute the BSG direction $g_k^{\text{BSG}}$ [29]. In particular, a first category, referred to as *implicit differentiation*, is composed of algorithms that compute the BSG direction by applying the implicit function theorem and either solving the adjoint equation [40] or using a truncated Neumann series to approximate the inverse Hessian $\nabla^2_{yy} f_\ell^{-1}$ [32]. A second category, referred to as *iterative differentiation*, includes all the approaches based on automatic differentiation through dynamic systems [16, 17, 33]. Since computing the BSG by using automatic differentiation can be less computationally efficient than using an implicit differentiation method [25], we will focus on the first category, which has been promoted in [9, 12, 13, 20, 23, 51] (see also [7, 29] for recent reviews). These existing approaches either focus on a specific problem structure or require the LL problem to be solved to optimality at each iteration or rely on a truncated Neumann series for the inverse Hessian approximation in the adjoint gradient. Moreover, none of them considers the LL constrained case.

DARTS [28] is an optimization technique related to the BSG method, which has enjoyed great popularity in NAS. It always considers an inexact solution to the LL problem, and it starts an iteration by applying one step of SG to the LL problem, $\tilde{y}_k = y_k - \eta \nabla_y f_\ell(x_k, y_k)$, where $\eta$ is a fixed stepsize. Then, it displaces the UL variables using $x_{k+1} = x_k - \eta\, g_k^{\text{DARTS}}$, where $g_k^{\text{DARTS}}$ is computed by applying the chain rule to $\nabla_x f_u(x, y - \eta \nabla_y f_\ell(x, y))$, leading to (when using full-batch gradients)

$$g_k^{\text{DARTS}} \;=\; \nabla_x f_u(x_k, \tilde{y}_k) - \eta \nabla^2_{xy} f_\ell(x_k, y_k) \nabla_y f_u(x_k, \tilde{y}_k). \tag{1.5}$$

However, in the full-batch (deterministic) case, $g_k^{\text{DARTS}}$ may not be a descent direction. The matrix-vector product in (1.5) is approximated by finite differences, rendering DARTS free of both second-order derivatives and matrix-vector products (see Section 5.2).

### 1.3 Contributions of the paper

A first contribution of this paper is a general framework for the BSG method that applies to both the LL unconstrained ($Y(x) = \mathbb{R}^m$) and constrained ($Y(x) \neq \mathbb{R}^m$) cases. In particular, our paper represents the first work* that proposes a method to address the LL constrained case, which has not been covered elsewhere, neither algorithmically nor theoretically, although important ML problems give rise to bilevel problems with LL constraints.

The convergence theory developed in this paper for the BSG method applies to both the LL unconstrained and constrained cases and is grounded on sensitivity principles of nonlinear optimization. Our theory also comprehensively covers all possible inexact settings such as the inexact solution of the LL problem, inexact computation of the adjoint formula (due to the inexact solution of the adjoint equation or use of a truncated Neumann series), and noisy estimates of the gradients, Jacobians, and Hessians involved. Since the convergence analysis proposed is abstracted from the specifics of the approach used to compute the inverse matrix in the adjoint formula (1.2), our theory unifies two different classes of BSG methods, i.e., the ones based on

*Very recently, we came across an approach [54] where a similar derivation has been proposed for bilevel problems with linear LL constraints.

the adjoint equation and the ones based on the truncated Neumann series, which have been studied separately [20, 40].

To approximate the second-order derivatives in the adjoint formula and avoid explicitly solving the adjoint equation or computing the truncated Neumann series (also dismissing any matrix-vector products), we propose a practical implementation of the BSG method, referred to as BSG-1, which uses rank-1 Hessian approximations. Such approximations are inspired by Gauss-Newton methods for nonlinear least-squares problems (see, e.g,. [39]) and also from the fact that the empirical risk of misclassification in ML is often a sum of non-negative terms matching a function to a scalar which can then be considered in a least-squares fashion [4, 19]. The BSG-1 method can be effectively applied to solve large-scale bilevel ML problems (in both the LL unconstrained and constrained cases), as proved by our numerical results.

We present and discuss a variety of numerical experiments that compare different versions of the BSG methods against DARTS on a synthetic quadratic bilevel problem and a continual learning problem (with and without LL constraints). The numerical results demonstrate that the BSG-1 method performs better than DARTS.

### 1.4 Organization of this paper

This paper is organized as follows. In Section 2, we describe the BSG method for the LL unconstrained case and introduce it for the LL constrained case. The assumptions on the problem functions and inexact calculations required for the convergence analysis of the method are reported in Section 3. Section 4 presents the convergence rates for the nonconvex, strongly convex, and convex cases. Numerical results for a synthetic quadratic bilevel problem and continual learning instances are analyzed in Section 5, which also describes the practical implementations of both the BSG-1 and DARTS methods. Finally, in Section 6 we draw some concluding remarks and we propose ideas for future work. By default, all norms $\|\cdot\|$ used in this paper are the $\ell_2$ ones.

## 2 The bilevel stochastic gradient method

In this section, we introduce the bilevel stochastic gradient (BSG) method for solving stochastic BLPs. Let $\{\vartheta_k^\ell\}_{k\geq 0}$ and $\{\varsigma_k^\ell\}_{k\geq 0}$ be sequences of random variables for the LL gradient and Hessian evaluations, and the Jacobian evaluations, respectively. Similarly, let $\{\vartheta_k^u\}_{k\geq 0}$ be a sequence of random variables for UL gradient evaluations. A realization of $\vartheta_k^\ell$, $\varsigma_k^\ell$, and $\vartheta_k^u$ can be interpreted as a single sample or a batch of samples for mini-batch SG. For compactness of notation, let us set $\xi_k = (\vartheta_k^u, \vartheta_k^\ell)$ in the LL unconstrained case ($Y(x) = \mathbb{R}^m$), and $\xi_k = (\vartheta_k^u, \vartheta_k^\ell, \varsigma_k^\ell)$ in the LL constrained case ($Y(x) \neq \mathbb{R}^m$).

### 2.1 The unconstrained lower level case

Given $(x_k, \tilde{y}_k)$, we denote by $g_x^u(x_k, \tilde{y}_k, \vartheta_k^u)$, $g_y^u(x_k, \tilde{y}_k, \vartheta_k^u)$, and $g_y^\ell(x_k, \tilde{y}_k, \vartheta_k^\ell)$ the stochastic gradient estimates that approximate $\nabla_x f_u(x_k, \tilde{y}_k)$, $\nabla_y f_u(x_k, \tilde{y}_k)$, and $\nabla_y f_\ell(x_k, \tilde{y}_k)$, respectively. The same notation applies to the stochastic Hessian estimates: $H_{xy}^\ell(x_k, \tilde{y}_k, \vartheta_k^\ell)$ and $H_{yy}^\ell(x_k, \tilde{y}_k, \vartheta_k^\ell)$ approximate, respectively, $\nabla_{xy}^2 f_\ell(x_k, \tilde{y}_k)$ and $\nabla_{yy}^2 f_\ell(x_k, \tilde{y}_k)$.

In the LL unconstrained case ($Y(x) = \mathbb{R}^m$), an approximate (negative) BSG (denoted

as $-g_k^{\mathrm{BSG}}$ in Subsection 1.2) can be computed directly from the adjoint formula (1.2), as follows:

$$d(x_k, \tilde{y}_k, \xi_k) \;=\; -\left(g_x^u(x_k, \tilde{y}_k, \vartheta_k^u) - H_{xy}^\ell(x_k, \tilde{y}_k, \vartheta_k^\ell) H_{yy}^\ell(x_k, \tilde{y}_k, \vartheta_k^\ell)^{-1} g_y^u(x_k, \tilde{y}_k, \vartheta_k^u)\right). \tag{2.1}$$

The data in the formula (1.2) is referred to as $D(x, y)$ or $D(x, y(x))$, depending on the point where the gradients and Hessians are evaluated:

$$D(x, y) \;=\; \left(\nabla_x f_u(x, y), \nabla_y f_u(x, y), \nabla_{xy}^2 f_\ell(x, y), \nabla_{yy}^2 f_\ell(x, y)\right). \tag{2.2}$$

The data in the calculation (2.1) is referred to as $D(x_k, \tilde{y}_k, \xi_k)$:

$$D(x_k, \tilde{y}_k, \xi_k) \;=\; \left(g_x^u(x_k, \tilde{y}_k, \vartheta_k^u), g_y^u(x_k, \tilde{y}_k, \vartheta_k^u), H_{xy}^\ell(x_k, \tilde{y}_k, \vartheta_k^\ell), H_{yy}^\ell(x_k, \tilde{y}_k, \vartheta_k^\ell)\right). \tag{2.3}$$

As we have said before, our practical implementation of BSG will use rank-one approximations for $H_{xy}^\ell(x_k, \tilde{y}_k, \vartheta_k^\ell)$ and $H_{yy}^\ell(x_k, \tilde{y}_k, \vartheta_k^\ell)$, and then solve $H_{yy}^\ell(x_k, \tilde{y}_k, \vartheta_k^\ell)\lambda = g_y^u(x_k, \tilde{y}_k, \vartheta_k^u)$ in the least-squares sense.

## 2.2 The constrained lower level case

Let us now handle the LL constrained case, in which we consider

$$Y(x) \;=\; \{y \in \mathbb{R}^m \mid c_i(x, y) \le 0,\ i \in I,\ \text{and } c_i(x, y) = 0,\ i \in E\},$$

where $I$ and $E$ are two finite sets of indices. For the LL problem, assume that at $y(x)$, the gradients of the active constraints are linearly independent (LICQ), the strict complementarity condition holds, and the sufficient second-order optimality conditions are satisfied [39]. Denoting $c_I(x, y) = (c_i(x, y),\ i \in I)$ and $c_E(x, y) = (c_i(x, y),\ i \in E)$, the Lagrangian function of the LL problem is defined as $\mathcal{L}_\ell(x, y, z) = f_\ell(x, y) + c_I(x, y)^\top z_I + c_E(x, y)^\top z_E$, where $z_I$ and $z_E$ are Lagrange multipliers. Assuming that the second-order necessary conditions hold at $y(x)$ with unique Lagrange multipliers $z_I(x)$ and $z_E(x)$ under the LICQ, we can write the following first-order KKT system for the LL problem

$$\begin{cases} \nabla_y f_\ell(x, y(x)) + \nabla_y c_I(x, y(x))\, z_I(x) + \nabla_y c_E(x, y(x))\, z_E(x) = 0, \\ z_I(x) \circ c_I(x, y(x)) = 0, \\ c_E(x, y(x)) = 0, \end{cases}$$

where $\circ$ is the element-wise multiplication operation of two vectors.

Now, given any $x$, define $v(x) = (y(x), z_I(x), z_E(x))^\top$. We can rewrite the KKT system above as $G(x, v(x)) = 0$ by introducing a corresponding vector function $G$. Now, applying the chain rule to $G(x, v(x)) = 0$, we obtain $\nabla_v G^\top \nabla v^\top = -\nabla_x G^\top$, with

$$\nabla_x G^\top = \begin{pmatrix} \nabla_{yx}^2 \mathcal{L}_\ell \\ z_I \circ \nabla_x c_I^\top \\ \nabla_x c_E^\top \end{pmatrix} \quad \text{and} \quad \nabla_v G^\top = \begin{pmatrix} \nabla_{yy}^2 \mathcal{L}_\ell & \nabla_y c_I & \nabla_y c_E \\ z_I \circ \nabla_y c_I^\top & C_I & 0 \\ \nabla_y c_E^\top & 0 & 0 \end{pmatrix}, \tag{2.4}$$

where the Hessian of $\mathcal{L}_\ell$ is evaluated at $(x, y(x), z_I(x), z_E(x))$, the Jacobian matrices $\nabla_x c_I^\top$, $\nabla_x c_E^\top$, $\nabla_y c_I^\top$, $\nabla_y c_E^\top$ are evaluated at $(x, y(x))$, and $C_I$ is a diagonal matrix whose elements are given by $c_I(x, y(x))$.

Since under the assumptions stated at the beginning of this section, the Jacobian $\nabla_v G^\top$ is nonsingular at $(x, v(x))$ [36, 39], we obtain

$$\nabla v = \big(\nabla y, \nabla z_I, \nabla z_E\big) = -\nabla_x G\, \nabla_v G^{-1}.$$

We can now pull out the columns of this system that correspond to the $\nabla y(x)$ term by introducing an appropriate matrix $L = \begin{pmatrix}\mathbf{I}_m & \mathbf{0}\end{pmatrix}^\top$, where $\mathbf{I}_m$ is an identity matrix of size $m$ and $\mathbf{0}$ is a null matrix of size $m \times |I| + |E|$

$$\nabla y(x) \;=\; -\nabla_x G \nabla_v G^{-1} L. \tag{2.5}$$

Substituting (2.5) into (1.3), we obtain the adjoint gradient for the LL constrained case

$$\nabla f = \nabla_x f_u - \nabla_x G \nabla_v G^{-1} L \nabla_y f_u, \tag{2.6}$$

where the gradients of $f_u$ with respect to $x$ and $y$ are evaluated at $(x, y(x))$, while the Jacobians of $G$ with respect to $x$ and $v$ are evaluated at $(x, v(x))$. The negative adjoint gradient provides the steepest descent direction for $f$ at $x$ in the deterministic case.

Similar to the notation used for the LL unconstrained case in Subsection 2.1, given $(x_k, \tilde{v}_k)$, with $\tilde{v}_k = (\tilde{y}_k, (\tilde{z}_I)_k, (\tilde{z}_E)_k)$, the Jacobian estimates $\mathcal{G}_x(x_k, \tilde{v}_k, \varsigma_k^\ell)^\top$ and $\mathcal{G}_v(x_k, \tilde{v}_k, \varsigma_k^\ell)^\top$ approximate $\nabla_x G(x_k, \tilde{v}_k)^\top$ and $\nabla_v G(x_k, \tilde{v}_k)^\top$, respectively. The data in (2.6) is referred to as $D(x, v)$ or $D(x, v(x))$, depending on the point where the gradients, Hessians, and Jacobians are evaluated:

$$D(x, v) \;=\; \left(\nabla_x f_u(x, y),\, L\nabla_y f_u(x, y),\, \nabla_x G(x, v),\, \nabla_v G(x, v)\right). \tag{2.7}$$

An approximate (negative) BSG can be computed directly from the adjoint formula (2.6), as follows:

$$d(x_k, \tilde{v}_k, \xi_k) \;=\; -\Big(g_x^u(x_k, \tilde{y}_k, \vartheta_k^u) - \mathcal{G}_x(x_k, \tilde{v}_k, \varsigma_k^\ell)\, \mathcal{G}_v(x_k, \tilde{v}_k, \varsigma_k^\ell)^{-1} L\, g_y^u(x_k, \tilde{y}_k, \vartheta_k^u)\Big), \tag{2.8}$$

where the data in calculation (2.8) is now referred to as $D(x_k, \tilde{v}_k, \xi_k)$:

$$D(x_k, \tilde{v}_k, \xi_k) \;=\; \Big(g_x^u(x_k, \tilde{y}_k, \vartheta_k^u),\, L\, g_y^u(x_k, \tilde{y}_k, \vartheta_k^u),\, \mathcal{G}_x(x_k, \tilde{v}_k, \varsigma_k^\ell),\, \mathcal{G}_v(x_k, \tilde{v}_k, \varsigma_k^\ell)\Big). \tag{2.9}$$

It is worth mentioning that an alternative approach to obtain a direction colinear with the negative gradient of $f$ in the LL constrained case was proposed in [48]. However, such an approach requires solving an auxiliary linear-quadratic bilevel problem, which is not practical in terms of solving large-scale machine learning problems. Also, the approach in [48] does not yield the exact size of the gradient.

### 2.3 A unified notation

Our goal is to propose a general algorithm that applies to both the LL unconstrained and constrained cases. For each case, one can denote the BSG directions used in the deterministic setting (i.e., (1.2) and (2.6)) and stochastic one (i.e, (2.1) and (2.8)) by using the following unified notation

$$d(D) = -(a - AB^{-1}b), \tag{2.10}$$

where $D = (a, b, A, B)$. In particular, in the LL unconstrained case, when the adjoint formula (1.2) or (2.1) is used, the data $D$ is either the deterministic one (2.2) or the stochastic

one (2.3), respectively. Similarly, in the constrained case, when the adjoint formula (2.6) or (2.8) is used, the data $D$ is again either the deterministic one (2.7) or the stochastic one (2.9), respectively. Moreover, we use the following notation to encapsulate the LL variables in the two cases

$$w \;=\; \begin{cases} y \text{ when } Y(x) = \mathbb{R}^n, \\ v \text{ when } Y(x) \neq \mathbb{R}^n, \end{cases} \tag{2.11}$$

i.e., $w$ is equal to $y$ in the unconstrained case and $v$ in the constrained case.

### 2.4 The BSG method

The schema of the BSG method is presented in Algorithm 1. An initial point $(x_0, \tilde{w}_0)$ and a sequence of positive scalars $\{\alpha_k\}$ are required as input. In Step 1, any arbitrary optimization method can be applied to approximately solve the LL problem, regardless of being unconstrained or constrained. In Step 2, one computes an approximate (negative) BSG, which will be denoted by $d(x_k, \tilde{w}_k, \xi_k)$ and computed through (2.1) or (2.8). Finally, at Step 3, the vector $x$ is updated by using a proper step size taken from the sequence of positive scalars. When $X$ is a closed and convex constrained set different from $\mathbb{R}^n$, we need to compute the orthogonal projection of $x_k + \alpha_k\, d(x_k, \tilde{w}_k, \xi_k)$ onto $X$ (note that such a projection can be computed by solving a convex optimization problem).

**Algorithm 1** Bilevel Stochastic Gradient (BSG) Method

**Input:** $(x_0, \tilde{w}_0)$, $\{\alpha_k\}_{k\geq 0} > 0$.

**For** $k = 0, 1, 2, \ldots$ **do**
  **Step 1.** Obtain an approximation $\tilde{w}_k$ to the LL optimal solution $w(x_k)$.
  **Step 2.** Compute a (negative) stochastic gradient approximation $d(x_k, \tilde{w}_k, \xi_k)$.
  **Step 3.** Compute $x_{k+1} = P_X(x_k + \alpha_k\, d(x_k, \tilde{w}_k, \xi_k))$.
**End do**

We point out that as is usual in the literature related to SG methods, a stopping criterion is not considered due to a lack of reasonable criteria and for the need to study the asymptotic convergence properties.

### 2.5 Computing the BSG direction inexactly

The two approaches proposed in the literature to deal with the inverse matrix $B^{-1}$ in (2.10) consist of either solving the adjoint equation or using a truncated Neumann series. In particular, the first approach requires solving the adjoint equation $B\,\lambda = b$ for the adjoint variables $\lambda$. The BSG direction can thus be calculated by $a - A\,\lambda$. The residual error due to the inexact solution of the adjoint equation is denoted by $\tilde{r}$, i.e., $\tilde{r} = B\,\lambda - b$. Note that the previous expression can be written as $B\,\lambda = b + \tilde{r}$, where the right-hand side can be interpreted as a perturbation of the right-hand side in the adjoint equation. Since $\lambda = B^{-1}(b + \tilde{r})$, the BSG direction becomes

$$-(a - A\,B^{-1}(b + \tilde{r})) \tag{2.12}$$

in the inexact adjoint solve case. Given a positive scalar $q > 0$, the second approach is based on the Neumann series as follows:

$$B^{-1} \;=\; \sum_{i=0}^{\infty}(I-B)^i \;=\; \mathscr{B} + \tilde{R},$$

where $\mathscr{B} = \sum_{i=0}^{q}(I-B)^i$ and $\tilde{R} = \sum_{i=q+1}^{\infty}(I-B)^i$. Note that the accuracy of the approximation is an increasing function of $q$. An approximation to $B^{-1}$ is given by $\mathscr{B}$, i.e., $B^{-1} \simeq \mathscr{B}$. Therefore, the BSG direction becomes

$$-(a - A\,(\mathscr{B} + \tilde{R})\,b) \tag{2.13}$$

in the inexact Neumann series case.

# 3 Assumptions, sensitivity, and smoothness

In this section, we introduce the assumptions used in the convergence analysis of the BSG method, which extends the convergence theory of the SG method to the bilevel case when the stepsize is assumed to be decaying. Using the notation introduced in (2.10)–(2.11), the convergence theory developed in this section covers both the LL unconstrained and constrained cases. The solution of the LL problem is assumed inexact. Our theory also applies when the BSG direction (2.10) is computed inexactly by using the approaches in Subsection 2.5, which comprehensively generalizes all existing approaches in the literature.

Given that we consider the application of the stochastic gradient method (or a similar SA technique) to solve the LL problem (see Step 1 of Algorithm 1), we will denote by $\xi_k^{\mathrm{S1}}$ the set of random variables for all combined iterations of the LL solution process at iteration $k$. Moreover, we will denote by $\xi_k^{\mathrm{all}}$ the set of all random variables for both the LL and UL solution processes at iteration $k$. Therefore, noticing that $\xi_k$ denotes the set of random variables used at Step 2 of Algorithm 1, let us set $\xi_k^{\mathrm{all}} = (\xi_k^{\mathrm{S1}}, \xi_k)$. At each iteration, the iterate $x_k$ is completely determined by the realizations of the independent random variables $\xi_k^{\mathrm{S1}}$ and $\xi_k$.

## 3.1 General assumptions

Let us recall that $f(x) = f_u(x, y(x))$. The true function $f$ will later be assumed sufficiently smooth (Assumption 3.7 below). For the moment we need to impose a certain smoothness on the BLP gradients, Hessians, and Jacobians involved in the computation of the BSG direction (Assumption 3.1 below).

**Assumption 3.1 (Lipschitz continuity of gradients, Hessians, and Jacobians)** *The gradients $\nabla_x f_u$ and $\nabla_y f_u$ are Lipschitz continuous in $(x, y)$. In the LL unconstrained case, the Hessians $\nabla^2_{xy} f_\ell$ and $\nabla^2_{yy} f_\ell$ are Lipschitz continuous in $(x, y)$. In the LL constrained case, the Jacobians $\nabla_x c_I^\top$, $\nabla_x c_E^\top$, $\nabla_y c_I^\top$, and $\nabla_y c_E^\top$ and the Hessians $\nabla^2_{yx} c_i$ and $\nabla^2_{yy} c_i$, for all $i \in I \cup E$, are Lipschitz continuous in $(x, y)$. Further, $D(\cdot, \cdot, \xi)$ is Lipschitz continuous in the second argument.*

We point out that the assumption on $D(\cdot, \cdot, \xi)$ is met in finite-sum ML of the type $\sum_{i=1}^{N} h_i(x)$ if all functions $h_i$ are Lipschitz continuous. In this case, the randomness $\xi$ consists of drawing a batch, which trivially renders the assumption true.

We require the BSG direction $d(x_k, w(x_k), \xi_k)$ to satisfy Assumption 3.2 below, which is a classical assumption [5]. Such an assumption will be used to derive a bound on the second

moment of the approximate BSG direction at the end of Section 3.2. The expected value with respect to the probability distributions of $\xi_k$ and $\xi_k^{\mathrm{S1}}$ are denoted by $\mathbb{E}_{\xi_k}[\cdot]$ and $\mathbb{E}_{\xi_k^{\mathrm{S1}}}[\cdot]$, respectively. The expected value with respect to the joint distribution of $\xi_k$ and $\xi_k^{\mathrm{S1}}$ is denoted by $\mathbb{E}_{\xi_k^{\mathrm{all}}} = \mathbb{E}_{\xi_k}[\mathbb{E}_{\xi_k^{\mathrm{S1}}}[\cdot]]$.

**Assumption 3.2 (Bound on the second moment of the BSG direction)** *There exists a positive scalar $V_d$ such that the vector $d(x_k, w(x_k), \xi_k)$ satisfies the following condition:*

$$\mathbb{E}_{\xi_k}[\|d(x_k, w(x_k), \xi_k)\|^2] \;\leq\; V_d.$$

### 3.2 Sensitivity of the approximated bilevel stochastic gradient direction

To bound the second moment of the approximate BSG direction $d(x_k, \tilde{w}_k, \xi_k)$ and the expectation of the error between the negative gradient $-\nabla f(x_k)$ and $d(x_k, \tilde{w}_k, \xi_k)$, we will need to apply sensitivity analysis arguments from nonlinear optimization. We start by assuming that the calculation process of the BSG direction (2.10) is (approximately) Lipschitz continuous with respect to changes in its data.

**Assumption 3.3 (Sensitivity of the BSG direction)** *Given any pair of data $D_1$ and $D_2$, there exists a constant $L_{BSG} > 0$ such that*

$$\|d(D_1) - d(D_2)\| \;\leq\; L_{BSG}(\|D_1 - D_2\| + \|r_1 - r_2\|), \tag{3.1}$$

*where $r_1$ and $r_2$ are the residual errors in the inexact computations of $d(D_1)$ and $d(D_2)$.*

In Proposition 3.1 below (see Appendix A for the proof), it is shown that the inexact ways (2.12) and (2.13) of calculating adjoint gradients or BSG directions do ensure that Assumption 3.3 is satisfied. Parts of the proof have been published elsewhere [13]. In fact, the arguments used are the known facts that sum is Lipschitz continuous, multiplication is Lipschitz continuous if the factors are bounded, and matrix inversion is Lipschitz continuous if its singular values are bounded away from zero.

**Proposition 3.1** *Given any pair of data $D_1$ and $D_2$, let $r_1$ and $r_2$ be the residual errors incurred when $d(D_1)$ and $d(D_2)$ are computed inexactly by either* (2.12) *(in which case $r_1$, $r_2$ are $\tilde{r}_1$, $\tilde{r}_2$) or* (2.13) *(in which case $r_1$, $r_2$ are $\tilde{R}_1$, $\tilde{R}_2$). Assuming that $b$, $A$, and $B$ are bounded in norm and that the singular values of $B$ are bounded away from zero, there exists a constant $L_{BSG} > 0$ such that inequality* (3.1) *of Assumption 3.3 is satisfied.*

We now introduce Assumption 3.4 on the absolute error of the LL optimal solution, whose validity in practice is discussed in Subsection 4.4.

**Assumption 3.4** *There exists a positive scalar $C_w$ such that*

$$\mathbb{E}_{\xi_k^{\mathrm{S1}}}[\|w(x_k) - \tilde{w}_k\|^2] \;\leq\; (C_w\,\alpha_k)^2.$$

By applying Jensen's inequality, one also has $(\mathbb{E}_{\xi_k^{\mathrm{S1}}}[\|w(x_k) - \tilde{w}_k\|])^2 \;\leq\; \mathbb{E}_{\xi_k^{\mathrm{S1}}}[\|w(x_k) - \tilde{w}_k\|^2]$, thus

$$\mathbb{E}_{\xi_k^{\mathrm{S1}}}[\|w(x_k) - \tilde{w}_k\|] \;\leq\; C_w\,\alpha_k. \tag{3.2}$$

We also need Assumption 3.5 below to hold which essentially amounts to the sampling error in the data. To enforce this assumption in practice, we refer the reader to the discussion reported in Section 4.5. Recall that $D(x_k, \tilde{w}_k)$ represents the deterministic data defining the quantity $d(x_k, \tilde{w}_k)$ (see (2.2) and (2.7)), and $D(x_k, \tilde{w}_k, \xi_k)$ the stochastic data of the calculation of $d(x_k, \tilde{w}_k, \xi_k)$ (see (2.3) and (2.9)).

**Assumption 3.5** *There exists a positive scalar $C_D$ such that*

$$\mathbb{E}_{\xi_k^{\text{all}}}[\|D(x_k, \tilde{w}_k) - D(x_k, \tilde{w}_k, \xi_k)\|] \;\leq\; C_D\, \alpha_k.$$

Finally, we need to bound the residual errors introduced in Assumption 3.3. In practice, when the BSG direction is computed inexactly (see Subsection 2.5), this assumption can be enforced by either increasing the accuracy in the inexact adjoint equation solve or increasing the value of $q$ used to truncate the Neumann series.

**Assumption 3.6** *There exists a positive scalar $C_e$ such that, for all realizations of the algorithm,*

$$\|r_k\| \;\leq\; C_e\, \alpha_k,$$

*where $r_k$ is either $(r_1)_k$ or $(r_2)_k$.*

One is ready to establish the desired bounds (see Lemmas 3.1 and 3.2 below), for which the constants will depend on the above-introduced constants.

**Lemma 3.1** *Under Assumptions 3.1–3.4 and 3.6, and assuming the stepsize sequence $\{\alpha_k\}$ bounded from above by a constant $C_s > 0$, one has*

$$\mathbb{E}_{\xi_k^{\text{all}}}[\|d(x_k, \tilde{w}_k, \xi_k)\|^2] \;\leq\; G_d, \tag{3.3}$$

*where $G_d = 2\,(U_d + V_d)$, $U_d = L_{BSG}^2 C_s^2 (L_{LL}^2 C_w^2 + 2C_e C_w + 4C_e^2)$, and $L_{LL} > 0$ is a constant only dependent on the Lipschitz constants of the gradients and Hessians of Assumption 3.1.*

**Proof.** By considering the data $D_1 = D(x_k, w(x_k), \xi_k)$ and $D_2 = D(x_k, \tilde{w}_k, \xi_k)$ in Assumption 3.3, we obtain

$$\begin{aligned}\|d(x_k, w(x_k), \xi_k) - d(x_k, \tilde{w}_k, \xi_k)\| \;&\leq L_{BSG}\|D(x_k, w(x_k), \xi_k) - D(x_k, \tilde{w}_k, \xi_k)\| \\ &+\; L_{BSG}\|r_1 - r_2\|.\end{aligned} \tag{3.4}$$

Assumption 3.1 implies the existence of a constant $L_{LL} > 0$ such that

$$\|D(x_k, w(x_k), \xi_k) - D(x_k, \tilde{w}_k, \xi_k)\| \;\leq\; L_{LL}\|w(x_k) - \tilde{w}_k\|. \tag{3.5}$$

From (3.4), (3.5), and Assumption 3.6, we obtain

$$\|d(x_k, w(x_k), \xi_k) - d(x_k, \tilde{w}_k, \xi_k)\| \;\leq\; L_{BSG}(L_{LL}\|w(x_k) - \tilde{w}_k\| + 2C_e\alpha_k). \tag{3.6}$$

By raising both sides of (3.6) to the second power, we have

$$\|d(x_k, w(x_k), \xi_k) - d(x_k, \tilde{w}_k, \xi_k)\|^2 \;\leq\; L_{BSG}^2(L_{LL}^2\|w(x_k) - \tilde{w}_k\|^2 + 2C_e\alpha_k\|w(x_k) - \tilde{w}_k\| + 4C_e^2\alpha_k^2).$$

Therefore, by taking expectations with respect to the distribution of $\xi_k^{\text{all}}$, considering Assumption 3.4, (3.2), and the bound on the stepsize sequence, and denoting $U_d = L_{BSG}^2 C_s^2 (L_{LL}^2 C_w^2 + 2C_e C_w + 4C_e^2)$, we obtain

$$\mathbb{E}_{\xi_k^{\text{all}}}[\|d(x_k, w(x_k), \xi_k) - d(x_k, \tilde{w}_k, \xi_k)\|^2] \;\leq\; U_d, \tag{3.7}$$

where we have used that $\mathbb{E}_{\xi_k^{\text{all}}}[\|w(x_k) - \tilde{w}_k\|^i] \;=\; \mathbb{E}_{\xi_k^{\text{S1}}}[\|w(x_k) - \tilde{w}_k\|^i]$, with $i \in \{1, 2\}$.

From Assumption 3.2 and (3.7), we can obtain the desired bound on the second moment of the approximate BSG direction by adding and subtracting $d(x_k, w(x_k), \xi_k)$ and considering that $\mathbb{E}_{\xi_k^{\text{all}}}[\|d(x_k, w(x_k), \xi_k)\|^2] = \mathbb{E}_{\xi_k}[\|d(x_k, w(x_k), \xi_k)\|^2]$. In particular, we have

$$\begin{aligned}\mathbb{E}_{\xi_k^{\text{all}}}[\|d(x_k, \tilde{w}_k, \xi_k)\|^2] \;&\leq\; 2\,\mathbb{E}_{\xi_k^{\text{all}}}[\|d(x_k, \tilde{w}_k, \xi_k) - d(x_k, w(x_k), \xi_k)\|^2] \\ &\quad +2\,\mathbb{E}_{\xi_k}[\|d(x_k, w(x_k), \xi_k)\|^2] \\ &\leq\; 2\,(U_d + V_d).\end{aligned}$$

□

**Lemma 3.2** *Under Assumptions 3.1 and 3.3–3.6*

$$\mathbb{E}_{\xi_k^{\text{all}}}[\| - \nabla f(x_k) - d(x_k, \tilde{w}_k, \xi_k)\|] \;\leq\; C_d\,\alpha_k, \tag{3.8}$$

*where $C_d = L_{BSG}(L_{LL}C_w + C_D + 4C_e)$, and $L_{LL} > 0$ is a constant only dependent on the Lipschitz constants of the gradients and Hessians of Assumption 3.1.*

**Proof.** By adding and subtracting the term $d(x_k, \tilde{w}_k)$ and using the triangle inequality, we have

$$\begin{aligned}\mathbb{E}_{\xi_k^{\text{all}}}[\|d(x_k, w(x_k)) - d(x_k, \tilde{w}_k, \xi_k)\|] \;&\leq\; \mathbb{E}_{\xi_k^{\text{all}}}[\|d(x_k, w(x_k)) - d(x_k, \tilde{w}_k)\|] &(3.9)\\ &\quad + \mathbb{E}_{\xi_k^{\text{all}}}[\|d(x_k, \tilde{w}_k) - d(x_k, \tilde{w}_k, \xi_k)\|]. &(3.10)\end{aligned}$$

Now we derive a bound for the right-hand side in (3.9). By considering the data $D_1 = D(x_k, w(x_k))$ and $D_2 = D(x_k, \tilde{w}_k)$ in Assumption 3.3, and taking the expectation, we obtain

$$\mathbb{E}_{\xi_k^{\text{all}}}[\|d(x_k, w(x_k)) - d(x_k, \tilde{w}_k)\|] \;\leq\; L_{BSG}\,\mathbb{E}_{\xi_k^{\text{all}}}[\|D(x_k, w(x_k)) - D(x_k, \tilde{w}_k)\|] \;+\; 2L_{BSG}C_e\alpha_k, \tag{3.11}$$

where we have applied Assumption 3.6 on $\|(r_1)_k - (r_2)_k\|$.

Note that the right-hand side of (3.11) contains exact BLP gradients and Hessians (or Jacobians in the LL constrained case). Therefore, the Lipschitz continuity of those mappings (Assumption 3.1) implies the existence of a constant $L_{LL} > 0$ such that

$$\|D(x_k, w(x_k)) - D(x_k, \tilde{w}_k)\| \;\leq\; L_{LL}\|w(x_k) - \tilde{w}_k\|. \tag{3.12}$$

Taking expectations with respect to the distribution of $\xi_k^{\text{all}}$ on both sides of (3.12), we can write

$$\mathbb{E}_{\xi_k^{\text{all}}}[\|D(x_k, w(x_k)) - D(x_k, \tilde{w}_k)\|] \;\leq\; L_{LL}\,\mathbb{E}_{\xi_k^{\text{S1}}}[\|w(x_k) - \tilde{w}_k\|], \tag{3.13}$$

where we have used that $\mathbb{E}_{\xi_k^{\text{all}}}[\|w(x_k) - \tilde{w}_k\|] \;=\; \mathbb{E}_{\xi_k^{\text{S1}}}[\|w(x_k) - \tilde{w}_k\|]$. Therefore, from (3.11), (3.13), and (3.2), we obtain

$$\mathbb{E}_{\xi_k^{\text{all}}}[\|d(x_k, w(x_k)) - d(x_k, \tilde{w}_k)\|] \;\leq\; L_{BSG}\,(L_{LL}\,C_w \;+\; 2C_e)\alpha_k. \tag{3.14}$$

Now we derive a bound for (3.10). By considering the data $D_1 = D(x_k, \tilde{w}_k)$ and $D_2 = D(x_k, \tilde{w}_k, \xi_k)$ in Assumption 3.3 and applying Assumption 3.6, we have

$$\|d(x_k, \tilde{w}_k) - d(x_k, \tilde{w}_k, \xi_k)\| \;\leq\; L_{BSG}\|D(x_k, \tilde{w}_k) - D(x_k, \tilde{w}_k, \xi_k)\| \;+\; 2L_{BSG}C_e\alpha_k. \tag{3.15}$$

Taking expectations with respect to the distribution of $\xi_k^{\text{all}}$ on both sides of (3.15), we obtain

$$\mathbb{E}_{\xi_k^{\text{all}}}[\|d(x_k, \tilde{w}_k) - d(x_k, \tilde{w}_k, \xi_k)\|] \;\leq\; L_{BSG}\,\mathbb{E}_{\xi_k^{\text{all}}}[\|D(x_k, \tilde{w}_k) - D(x_k, \tilde{w}_k, \xi_k)\|] \;+\; 2L_{BSG}C_e\alpha_k. \tag{3.16}$$

Hence, from (3.16) and Assumption 3.5, we obtain

$$\mathbb{E}_{\xi_k^{\text{all}}}[\|d(x_k, \tilde{w}_k) - d(x_k, \tilde{w}_k, \xi_k)\|] \;\leq\; L_{BSG}\,(C_D + 2C_e)\alpha_k. \tag{3.17}$$

The proof can be concluded from (3.9)–(3.10), (3.14), and (3.17). □

### 3.3 Smoothness of the true objective function

Our convergence theory requires an explicit assumption on the smoothness of the true function $f$.

**Assumption 3.7 (Smoothness of $f$)** *The gradient $\nabla f$ is Lipschitz continuous in $x$ with constant $L_{\nabla f} > 0$, i.e.,*

$$\|\nabla f(x_1) - \nabla f(x_2)\| \;\leq\; L_{\nabla f}\|x_1 - x_2\| \quad \textit{for all } (x_1, x_2) \in \mathbb{R}^n \times \mathbb{R}^n. \tag{3.18}$$

In both the LL unconstrained and constrained cases, the Lipschitz continuity of $\nabla f$ can be inferred from the Lipschitz continuity of $y(x)$, $w(x)$, and the gradients, Hessians, and Jacobians involved (along with the boundedness away from singularity of Hessian or KKT matrices). Such a result and its proof are given in Appendix B, and again part of the proof has been reported in [13] and relies on elementary arguments.

An important and well-known consequence that follows from Assumption 3.7 is

$$f(x) \;\leq\; f(\bar{x}) + \nabla f(\bar{x})^\top(x - \bar{x}) + \frac{1}{2}L_{\nabla f}\|x - \bar{x}\|^2 \;\text{ for all }\; (x, \bar{x}) \in \mathbb{R}^n \times \mathbb{R}^n. \tag{3.19}$$

## 4 Convergence rate of the BSG method

In this section, we extend the convergence theory of the SG method to the bilevel case when the stepsize is assumed to be decaying. The BLP objective function $f$ is assumed to be nonconvex, strongly convex (leading to a $1/k$ sublinear convergence rate), or simply convex ($1/\sqrt{k}$ rate).

Using the notation introduced in (2.10)–(2.11), the convergence theory developed in this section covers both the LL unconstrained and constrained cases. The BSG method under consideration takes an inexact solution of the LL problem, for which the stochasticity is rigorously included in the analysis for the first time. Moreover, such a theory also applies when the BSG direction (2.10) is computed inexactly regardless of the approach used (see Subsection 2.5), thus leading to an analysis that is considerably more general than the ones proposed in the literature [20, 40].

### 4.1 Rate in the nonconvex case

In this section, the true objective function $f$ is assumed to be possibly nonconvex. We now present two lemmas that will allow us to prove the convergence result. Such lemmas and the resulting theorem are based on the theory provided in [5], where the main differences lie in the use of the projection operator to handle the upper-level constraints $x \in X$ and, importantly, in how inexactly $d(x_k, \tilde{w}_k, \xi_k)$ approximates $-\nabla f(x_k)$ (see Lemmas 3.1 and 3.2). The first lemma is just a Taylor bound derived as a result of Assumption 3.7.

**Lemma 4.1** *Under Assumption 3.7, the iterates of Algorithm 1 satisfy the following inequality for all $k \in \mathbb{N}$*

$$\begin{aligned}\mathbb{E}_{\xi_k^{\text{all}}}\left[f(x_{k+1})\right] - f(x_k) \leq \alpha_k \left(P_X \nabla f(x_k)\right)^\top \mathbb{E}_{\xi_k^{\text{all}}}\left[d(x_k, \tilde{w}_k, \xi_k)\right] \\ + \frac{1}{2}\alpha_k^2 L_{\nabla f} \mathbb{E}_{\xi_k^{\text{all}}}\left[\|d(x_k, \tilde{w}_k, \xi_k)\|^2\right].\end{aligned} \tag{4.1}$$

**Proof.** From equation (3.19), the iterates generated by Algorithm 1 satisfy

$$f(x_{k+1}) - f(x_k) \leq \nabla f(x_k)^\top (x_{k+1} - x_k) + \frac{1}{2} L_{\nabla f} \|x_{k+1} - x_k\|^2.$$

Recalling that Algorithm 1 uses the update $x_{k+1} = P_X\left(x_k + \alpha_k d(x_k, \tilde{w}_k, \xi_k)\right)$, and since $x_k$ is in the feasible region $X$, we know that $x_k = P_X(x_k)$, which yields

$$f(x_{k+1}) - f(x_k) \;\leq\; \alpha_k \nabla f(x_k)^\top P_X d(x_k, \tilde{w}_k, \xi_k) + \frac{1}{2}\alpha_k^2 L_{\nabla f} \|P_X d(x_k, \tilde{w}_k, \xi_k)\|^2.$$

Since $P_X$ is an orthogonal projection, we know $P_X = P_X^\top = P_X^2$ and $\|P_X\| \leq 1$. Using this fact and taking expectations with respect to the distribution of $\xi_k^{\text{all}}$, we obtain (4.1). □

We now need to introduce Assumption 4.1, which states that the size of the gradient is bounded by some constant (which is implied by the Lipschitz continuity of $\nabla f$ and the boundedness of $X$).

**Assumption 4.1** *The gradients generated by the sequence of iterates $\{x_k\}_{k\geq 0}$ are bounded, i.e.,*

$$\|\nabla f(x_k)\| \;\leq\; C_{\nabla f},$$

*for some constant $C_{\nabla f} > 0$.*

The following lemma further extends the result of Lemma 4.1 by using the inexactness of $d(x_k, \tilde{w}_k, \xi_k)$ and the bound on its second-order moment.

**Lemma 4.2** *Under Assumptions 3.1–3.7 and 4.1, the iterates generated by Algorithm 1 satisfy the following inequality for all $k \in \mathbb{N}$:*

$$\mathbb{E}_{\xi_k^{\text{all}}}\left[f(x_{k+1})\right] - f(x_k) \leq -\alpha_k \|P_X \nabla f(x_k)\|^2 + \alpha_k^2 C_{\nabla f} C_d + \frac{1}{2}\alpha_k^2 L_{\nabla f} G_d. \tag{4.2}$$

**Proof.** From inequality (4.1) and Lemma 3.1, we have

$$\mathbb{E}_{\xi_k^{\text{all}}}\left[f(x_{k+1})\right] - f(x_k) \leq \alpha_k \left(P_X \nabla f(x_k)\right)^\top \mathbb{E}_{\xi_k^{\text{all}}}\left[d(x_k, \tilde{w}_k, \xi_k)\right] + \frac{1}{2}\alpha_k^2 L_{\nabla f} G_d.$$

Adding and subtracting $\alpha_k \left(P_X \nabla f(x_k)\right)^\top \mathbb{E}_{\xi_k^{\text{all}}}\left[\nabla f(x_k)\right]$ to the right-hand side and simplifying, we obtain

$$\begin{aligned}\mathbb{E}_{\xi_k^{\text{all}}}\left[f(x_{k+1})\right] - f(x_k) \;\leq\; & \alpha_k \left(P_X \nabla f(x_k)\right)^\top \mathbb{E}_{\xi_k^{\text{all}}}\left[d(x_k, \tilde{w}_k, \xi_k) + \nabla f(x_k)\right] \\ & - \alpha_k \left(P_X \nabla f(x_k)\right)^\top \nabla f(x_k) + \frac{1}{2}\alpha_k^2 L_{\nabla f} G_d.\end{aligned}$$

Applying the properties of orthogonal projections along with the Cauchy-Schwarz and Jensen's inequalities, we obtain

$$\begin{aligned}\mathbb{E}_{\xi_k^{\text{all}}}\left[f(x_{k+1})\right] - f(x_k) \;\leq\; & \alpha_k \|P_X \nabla f(x_k)\| \, \mathbb{E}_{\xi_k^{\text{all}}}\left[\|d(x_k, \tilde{w}_k, \xi_k) + \nabla f(x_k)\|\right] \\ & - \alpha_k \|P_X \nabla f(x_k)\|^2 + \frac{1}{2}\alpha_k^2 L_{\nabla f} G_d.\end{aligned}$$

Finally, from Assumption 4.1 and inequality (3.8) along with the properties of orthogonal projections, we obtain the desired result. □

We will now introduce the final assumptions that are needed for the convergence result of the nonconvex case. The first of these states that for Algorithm 1 to converge, the sequence of function values must be bounded below by some minimum value.

**Assumption 4.2** *The sequence* $\{f(x_k)\}_{k\geq 0}$ *is bounded below by* $f_{inf}$.

Lastly, we require the stepsize to be of decaying type.

**Assumption 4.3** *The sequence of decaying stepsizes* $\{\alpha_k\}_{k\geq 0}$ *satisfies*

$$\sum_{k=0}^{\infty} \alpha_k = \infty \quad \textit{and} \quad \sum_{k=0}^{\infty} \alpha_k^2 < \infty.$$

We can now establish the convergence result for the nonconvex case. We use $\mathbb{E}[\cdot]$ to refer to the *total expectation* of $f$, namely, the expected value with respect to the joint distribution of all the random vectors $\xi_k^{\text{all}}$.

**Theorem 4.3** *Under Assumptions 3.1–3.7, 4.1, and 4.2, suppose that Algorithm 1 is run with a decaying stepsize sequence that satisfies Assumption 4.3. Then, with* $A_K := \sum_{k=0}^{K} \alpha_k$,

$$\lim_{K\to\infty} \mathbb{E}\left[\sum_{k=0}^{K} \alpha_k \|P_X \nabla f(x_k)\|^2\right] < \infty, \tag{4.3}$$

*and therefore*

$$\lim_{K\to\infty} \mathbb{E}\left[\frac{1}{A_K}\sum_{k=0}^{K} \alpha_k \|P_X \nabla f(x_k)\|^2\right] = 0. \tag{4.4}$$

**Proof.** The proof follows [5, Theorem 4.10] closely. Taking the total expectation of (4.2), we have

$$\mathbb{E}\left[f(x_{k+1})\right] - \mathbb{E}\left[f(x_k)\right] \;\leq\; -\alpha_k \mathbb{E}\left[\|P_X \nabla f(x_k)\|^2\right] + \alpha_k^2 C_{\nabla f} C_d + \frac{1}{2}\alpha_k^2 L_{\nabla f} G_d$$

Summing both sides of this inequality for $k \in \{0, 1, ..., K\}$ and by Assumption 4.2, we have

$$\begin{aligned} f_{\inf} - \mathbb{E}\left[f(x_0)\right] \;&\leq\; \mathbb{E}\left[f(x_{K+1})\right] - \mathbb{E}\left[f(x_0)\right] \\ &\leq\; -\sum_{k=0}^{K} \alpha_k \mathbb{E}\left[\|P_X \nabla f(x_k)\|^2\right] + C_{\nabla f} C_d \sum_{k=0}^{K} \alpha_k^2 + \frac{1}{2} L_{\nabla f} G_d \sum_{k=0}^{K} \alpha_k^2. \end{aligned}$$

Rearranging, we obtain

$$\sum_{k=0}^{K} \alpha_k \mathbb{E}\left[\|P_X \nabla f(x_k)\|^2\right] \;\leq\; \mathbb{E}\left[f(x_0)\right] - f_{\inf} + C_{\nabla f} C_d \sum_{k=0}^{K} \alpha_k^2 + \frac{1}{2} L_{\nabla f} G_d \sum_{k=0}^{K} \alpha_k^2.$$

Assumption 4.3 implies that the right-hand side of this inequality converges to a finite limit when $K$ increases, which proves (4.3). To obtain (4.4), we can divide by $A_K$ as follows

$$\frac{1}{A_K} \sum_{k=0}^{K} \alpha_k \mathbb{E}\left[\|P_X \nabla f(x_k)\|^2\right] \;\leq\; \frac{\mathbb{E}\left[f(x_0)\right] - f_{\inf}}{A_K} + \frac{C_{\nabla f} C_d}{A_K} \sum_{k=0}^{K} \alpha_k^2 + \frac{L_{\nabla f} G_d}{2A_K} \sum_{k=0}^{K} \alpha_k^2.$$

Taking the limit as $K \to \infty$, and noting Assumption 4.3, we obtain the desired result. □

### 4.2 Rate in the strongly convex case

In this subsection, we present the convergence rate of the BSG method when $f$ is assumed to be strongly convex. In practice, such a case occurs when the UL objective function $f_u$ is strongly convex and $y(x)$ is an affine function in $x$. Hence, imposing strong convexity of $f$ is a strong assumption in the sense of assuming in practice that the LL problem is a QP problem. Still, we cover this case for completeness of our convergence theory and to address the case when an $\ell_2$ regularization term in $y$ is added to $f_\ell$.

Along with Assumption 3.7, we also need the iterates to lie in a bounded set, which could be ensured by the boundedness of $X$ in the BLP formulation.

**Assumption 4.4 (Boundedness of the iterates)** *The sequence of iterates $\{x_k\}_{k\geq 0}$ yielded by Algorithm 1 is contained in a bounded set.*

Assumption 4.4 implies that there exists a positive constant $\Theta$ such that, for any $(k_1, k_2)$, we have

$$\|x_{k_1} - x_{k_2}\| \;\leq\; \Theta \;<\; \infty.$$

Finally, we assume that the true function $f$ is strongly convex.

**Assumption 4.5 (Strong convexity of $f$)** *The function $f$ is strongly convex, namely, there exists a constant $c > 0$ such that*

$$f(\bar{x}) \;\geq\; f(x) + \nabla f(x)^\top (\bar{x} - x) + \frac{c}{2}\|\bar{x} - x\|^2 \text{ for all } (\bar{x}, x) \in \mathbb{R}^n \times \mathbb{R}^n. \tag{4.5}$$

A well-known equivalent condition to (4.5) (see, e.g., [38]) is given by

$$(\nabla f(x) - \nabla f(\bar{x}))^\top (x - \bar{x}) \geq c\|x - \bar{x}\|^2 \text{ for all } (x, \bar{x}) \in \mathbb{R}^n \times \mathbb{R}^n. \tag{4.6}$$

Let $x_*$ be the unique minimizer of $f$ on $X$, which implies that $\nabla f(x_*)^\top (x - x_*) \geq 0$ for all $x \in X$. Therefore, if in (4.6) we choose $x = x_k$ and $\bar{x} = x_*$, we obtain

$$\nabla f(x_k)^\top (x_k - x_*) \geq c\|x_k - x_*\|^2. \tag{4.7}$$

The next theorem proves that under the assumption of strong convexity and decaying stepsize ($\sum_{k=0}^{\infty} \alpha_k = \infty$ and $\sum_{k=0}^{\infty} \alpha_k^2 < \infty$), the sequence of points yielded by Algorithm 1 generates a sequence of $f$ values that decays sublinearly at the rate of $1/k$. The proof of this theorem is given in Appendix C.

**Theorem 4.4** *Let Assumptions 3.1–3.7 and 4.4–4.5 hold and $x_*$ be the unique minimizer of $f$ on $X$. Consider the schema given by Algorithm 1 and assume a decaying step size sequence of the form $\alpha_k = \gamma/k$, where $\gamma \geq 1/(2c)$ is a positive constant. The sequence of iterates yielded by Algorithm 1 satisfies*

$$\begin{aligned} \mathbb{E}[\|x_k - x_*\|^2] &\leq \frac{\max\{2\,\gamma^2 M (2c\gamma - 1)^{-1}, \|x_0 - x_*\|^2\}}{k} \\ \mathbb{E}[f(x_k)] - f(x_*) &\leq \frac{(L_{\nabla f}/2)\max\{2\,\gamma^2 M (2c\gamma - 1)^{-1}, \|x_0 - x_*\|^2\}}{k}, \end{aligned}$$

*where $M = G_d + 2C_d\Theta$.*

### 4.3 Rate in the convex case

In this subsection, we state the convergence rate of the BSG method assuming that $f$ is convex and attains a minimizer $x_*$. The same comment about the lack of practicality applies to the convex case: $y(x)$ would need to be affine and that restricts considerably the choice of LL problems.

**Assumption 4.6 (Convexity of $f$)** *Given $(\bar{y}, y) \in \mathbb{R}^m \times \mathbb{R}^m$, the (continuously differentiable) function $f$ is convex in $x$, namely,*

$$f(\bar{x}) \;\geq\; f(x) + \nabla f(x)^\top (\bar{x} - x) \text{ for all } (\bar{x}, x) \in \mathbb{R}^n \times \mathbb{R}^n. \tag{4.8}$$

*Moreover, $f$ attains a minimizer.*

The next theorem states that the BSG method exhibits a sublinear convergence rate of $1/\sqrt{k}$, which implies that the convergence is slower than in the strongly convex case (Theorem 4.4). The proof of this theorem is given in Appendix D.

**Theorem 4.5** *Let Assumptions 3.1–3.6, 4.4, and 4.6 hold. Consider the schema given by Algorithm 1 and assume a decaying step size of the form $\alpha_k = \bar{\alpha}/\sqrt{k}$, with $\bar{\alpha} > 0$. Given a minimizer $x_*$ of $f$, the sequence of iterates yielded by Algorithm 1 satisfies*

$$\min_{s=0,\ldots,k} \mathbb{E}[f(x_s)] - f(x_*) \;\leq\; \frac{\frac{\Theta^2}{2\bar{\alpha}} + \bar{\alpha}(G_d M + 2C_d M\Theta)}{\sqrt{k}}.$$

### 4.4 Imposing a bound on the distance from the LL optimal solution

In this subsection, we want to discuss a way to enforce Assumption 3.4 when using the stochastic gradient (SG) method to solve the LL problem at $x_k$. We focus on the LL unconstrained case. Given an initial point $\tilde{y}_k^0$ and a sequence of stepsizes $\{\beta_i\}$, such a SG method can be described as

$$\tilde{y}_k^{i+1} \;=\; \tilde{y}_k^i - \beta_i g_y^\ell(x_k, \tilde{y}_k^i, \xi_{k,i}^{\mathrm{S1}}), \quad i = 0, \ldots, i_k. \tag{4.9}$$

We start by introducing the sampling assumptions that are standard in the literature related to the SG method. First, suppose that the stochastic gradient $g_y^\ell(x_k, \tilde{y}_k, \xi_{k,i}^{\mathrm{S1}})$ is unbiased, i.e., $\mathbb{E}_{\xi_{k,i}^{\mathrm{S1}}}[g_y^\ell(x_k, \tilde{y}_k, \xi_{k,i}^{\mathrm{S1}})] = \nabla_y f_\ell(x_k, \tilde{y}_k)$, and there exists a positive constant $Q > 0$ such that $\mathbb{E}_{\xi_{k,i}^{\mathrm{S1}}}[\|g_y^\ell(x_k, \tilde{y}_k, \xi_{k,i}^{\mathrm{S1}})\|^2] \leq Q^2$. Also, suppose that $f_\ell$ is strongly convex in the $y$ variables with constant $\mu$ with a unique minimizer $y(x_k)$.

Recalling that the convergence rate of the SG method (4.9) with decaying stepsize is $\mathcal{O}(1/\sqrt{i})$, and by choosing $i_k$ equal to $k^2$, one guarantees the existence of a positive constant $C_y$ such that Assumption 3.4 holds. In fact, by choosing a decaying step size sequence $\{\beta_i\}$ given by $\beta_i = \gamma/i$, where $\gamma \geq 1/(2\mu)$ is a positive constant, and under the classical assumptions stated in the previous paragraph, from [37, Equation (2.9)] it follows that the choice $i_k = k^2$ implies (with $\tilde{y}_k = \tilde{y}_k^{i_k+1}$)

$$\mathbb{E}_{\xi_k^{\mathrm{S1}}}[\|\tilde{y}_k - y(x_k)\|^2] \;\leq\; \frac{\max\{\gamma^2 Q(2\mu\gamma - 1)^{-1}, \|\tilde{y}_k^0 - y(x_k)\|^2\}}{k^2}.$$

Such a result also holds in the LL constrained case when the scheme (4.9) incorporates a projection onto $Y(x_k)$, as long as $Y(x_k)$ is a bounded closed convex set. It is also possible to obtain an inequality like (3.4) for the LL constrained case when using a primal-dual stochastic method [56].

### 4.5 Imposing a bound on dynamic sampling

In this subsection, we want to mention a dynamic sampling strategy to enforce the inequality in Assumption 3.5 in both the LL unconstrained and constrained cases. For the sake of simplicity, we will omit the subscript $k$ in this subsection. Such a dynamic sampling strategy allows reducing the level of noise by increasing the size of the batch. Recalling the definition of $D(x, w)$ given in (2.2) and (2.7), and the definition of $D(x, w, \xi)$ given in (2.3) and (2.9), let us assume that $D(x, w, \xi)$ is normally distributed with mean $D(x, w)$ and variance $\sigma^2$ (and that the dimension of the covariance matrix is $n_{cov}$). Such an assumption implies that the stochastic estimates in $D(x, w, \xi)$ are unbiased estimates of the corresponding true gradients and Hessians/Jacobians.

To increase the accuracy of $D(x, w, \xi)$ as an estimator of $D(x, w)$, we can choose a larger batch size, which is denoted by $n_D$. Let $\bar{D}(x, w, \xi) = (1/n_D)\sum_{r=1}^{n_D} D(x, w, \xi^r)$ be the corresponding mini-batch stochastic estimate, where $\{\xi^r\}_{r=1}^{n_D}$ are values sampled from $\xi$. It is known that (for details, see, for instance, [30, Section 5.3])

$$\mathbb{E}_{\xi_k}[\|D(x, w) - \bar{D}(x, w, \xi)\|] \;\leq\; \frac{\sigma\sqrt{n_{cov}}}{\sqrt{n_D}}.$$

To guarantee that Assumption 3.5 holds, we need to choose a mini-batch size $n_D$ and a sample standard deviation $\sigma$ such that $\sigma\sqrt{n_{cov}/n_D} \;\leq\; C_D\,\alpha_k$. Therefore, when $\alpha_k$ decreases, the dynamic sampling strategy would increase $n_D$.

# 5 Numerical experiments

All tests were run using Google Colab (12.6GB of RAM, Intel(R) Xeon(R) processor running at 2.00GHz, and GPU Tesla T4). We averaged all the results over 10 trials by using different random seeds.

## 5.1 Our practical BSG methods (BSG-1)

A major difficulty in the adjoint formulas (1.2) and (2.6) is the use of second-order derivatives of $f_\ell$ and $\mathcal{L}_\ell$, respectively, and the need to solve the adjoint equation $B\lambda = b$ or use a truncated Neumann series, which prevents its application to large-scale ML problems. We can get around this problem by approximating the second-order derivatives with the outer products of the corresponding gradients, i.e.,

$$\nabla^2_{xy} f_\ell \;\simeq\; \nabla_x f_\ell \nabla_y f_\ell^\top \quad \text{and} \quad \nabla^2_{yy} f_\ell \;\simeq\; \nabla_y f_\ell \nabla_y f_\ell^\top, \tag{5.1}$$

$$\nabla^2_{yx} \mathcal{L}_\ell \;\simeq\; \nabla_y \mathcal{L}_\ell \nabla_x \mathcal{L}_\ell^\top \quad \text{and} \quad \nabla^2_{yy} \mathcal{L}_\ell \;\simeq\; \nabla_y \mathcal{L}_\ell \nabla_y \mathcal{L}_\ell^\top. \tag{5.2}$$

As mentioned in Section 1.3, such approximations are inspired by Gauss-Newton methods for nonlinear least-squares problems, where the Hessian matrix of the objective function $\sum_{i=1}^p (r_i - a_i)^2$ (in which each $r_i$ is a scalar function and $a_i$ a scalar) is approximated by $\sum_{i=1}^p \nabla r_i \nabla r_i^\top$, and also from the fact that the empirical risk of misclassification in ML is often a sum of nonnegative terms matching a function to a scalar which can then be considered in a least-squares fashion [4, 19].

In the LL unconstrained case, the resulting approximate adjoint equation $(\nabla_y f_\ell \nabla_y f_\ell^\top)\,\lambda = \nabla_y f_u$ is most likely infeasible, and we suggest solving it in the least-squares sense. One solution is $\lambda = \nabla_y f_u / (\nabla_y f_\ell^\top \nabla_y f_\ell)$. Plugging this and $\nabla^2_{xy} f_\ell \simeq \nabla_x f_\ell \nabla_y f_\ell^\top$ in the adjoint formula (1.2) gives rise to our practical BSG calculation

$$\nabla f \;\simeq\; \nabla_x f_u - \frac{\nabla_y f_\ell^\top \nabla_y f_u}{\nabla_y f_\ell^\top \nabla_y f_\ell} \nabla_x f_\ell. \tag{5.3}$$

This approximate BSG allows us to use the adjoint formula without computing Hessians or even Hessian-vector products, which is prohibitively expensive for the large bilevel problems arising in ML.

In the LL constrained case, we can use the outer products (5.2) to approximate the second-order derivatives of $\mathcal{L}_\ell$ in the Jacobian matrices $\nabla_x G^\top$ and $\nabla_v G^\top$, introduced in (2.4), and obtain approximate Jacobians $\tilde{G}_x^\top$ and $\tilde{G}_v^\top$. The resulting approximate adjoint equation is given by $\tilde{G}_v \tilde{\lambda} = L \nabla_y f_u$, where $L$ is the matrix used in (2.6), and can be solved using an iterative method for non-symmetric linear systems, e.g., GMRES [45]. Plugging a solution $\tilde{\lambda}$ into $\nabla_x f_u - \tilde{G}_x \tilde{\lambda}$, we obtain the practical BSG calculation

$$\nabla f \;\simeq\; \nabla_x f_u - \tilde{G}_x \tilde{\lambda}, \quad \text{where} \quad \tilde{G}_v \tilde{\lambda} = L \nabla_y f_u. \tag{5.4}$$

Both of these rank-1 approaches for the unconstrained and constrained cases will be referred to as BSG-1, the "1" standing for first-order rank-1 approximations of the Hessian and Jacobian matrices. In the numerical experiments considered for both LL unconstrained and constrained BLPs, we are mainly interested in testing a practical BSG-1 method (Algorithm 2 below) as opposed to the standard BSG method (Algorithm 1). In the practical BSG-1 method,

we will use the stochastic data defined by (2.3) and (2.9) (with the rank-1 approximations applied to the Hessians and Jacobians). We do not include the orthogonal projection operator $P_X$ in Step 3 of Algorithm 2 because there are no UL constraints in the problems considered in the numerical experiments (i.e., $X = \mathbb{R}^n$).

**Algorithm 2** BSG-1 Method

**Input:** $(x_0, \tilde{w}_0)$, $\{\alpha_k\}_{k\geq 0} > 0$.

**For** $k = 0, 1, 2, \ldots$ **do**

**Step 1.** Obtain an approximation $\tilde{w}_k$ to the LL optimal solution $w(x_k)$.

**Step 2.** Draw the stochastic gradients and/or Jacobians from (5.3) or (5.4) to compute $d(x_k, \tilde{w}_k, \xi_k)$.

**Step 3.** Compute $x_{k+1} = x_k + \alpha_k\, d(x_k, \tilde{w}_k, \xi_k)$.

**End do**

In the numerical results for the LL unconstrained case, we will also test the BSG method with stochastic Hessians, where the (negative) BSG direction $d(x_k, \tilde{y}_k, \xi_k)$ is calculated from (2.1). This version is referred to as BSG-H. The adjoint system $H^{\ell}_{yy}(x_k, \tilde{y}_k, \vartheta^{\ell}_k)\lambda = g^u_y(x_k, \tilde{y}_k, \vartheta^u_k)$ is solved by the linear conjugate gradient method until non-positive curvature is detected. In both BSG-1 and BSG-H versions of the BSG method for the LL unconstrained case, we will apply the SG method (4.9) in Step 1 of Algorithm 2 for a certain budget $i_k$ of iterations, obtaining an approximation $\tilde{y}_k$ to the LL optimal solution $y(x_k)$.

In the LL constrained case, we will only consider the BSG-1 version of the BSG method. In particular, given $x_k$, to obtain an approximation $\tilde{w}_k$ to $w(x_k)$ in Step 1 of Algorithm 2, we will first determine an approximation $\tilde{y}_k$ to $y(x_k)$ by minimizing the following exact penalty function over $y$

$$\Phi(x_k, y; \mu) = f_\ell(x_k, y) + \frac{1}{\mu}\sum_{i\in I} \max\{0, -c_i(x_k, y)\} + \frac{1}{\mu}\sum_{i\in E} |c_i(x_k, y)|\,, \tag{5.5}$$

where $\mu$ is a penalty parameter and the functions $c_i$, with $i \in I \cup E$, are the LL constraints defined in Subsection 2.2. We recall that for sufficiently small and positive values of $\mu$, the minimization of such an unconstrained problem will yield the optimal solution $y(x_k)$ of the constrained LL problem [39]. To minimize (5.5), which is a nonsmooth function, we will apply a stochastic subgradient method. Then, given $x_k$ and $\tilde{y}_k$, to determine approximations $(\tilde{z}_I, \tilde{z}_E)$ to the optimal multipliers $(z_I(x_k), z_E(x_k))$, we will solve the KKT system $G(x_k, (\tilde{y}_k, z_I, z_E)) = 0$ for the variables $z_I$ and $z_E$, where $G$ is the vector function introduced in Subsection 2.2.

We will consider two inexact schemes for the solution of the LL problem. The first one (1 step) consists of obtaining $\tilde{y}_k$ by taking a single step of the stochastic gradient method applied to $f_\ell$ ($i_k = 1$, $\forall k$, in (4.9) for the LL unconstrained case) or stochastic subgradient method applied to (5.5) (for the LL constrained case). In the second one (inc. acc.), the number of steps of the stochastic gradient/subgradient method increases by 1 every time the difference of the UL objective function between two consecutive iterations is less than a given threshold, thus leading to an increasing accuracy strategy. In both the LL inexact schemes, $\tilde{y}_k$ is determined by using the approximation $\tilde{y}_{k-1}$ obtained at the previous iteration as a starting point. In the LL constrained case, after each step of stochastic subgradient, a KKT system is solved by using a conjugate gradient method to obtain approximate Lagrange multipliers.

### 5.2 DARTS

DARTS was proposed in [28] for the solution of stochastic BLPs arising from NAS, and was briefly mentioned in Section 1.2. Only the LL unconstrained case ($Y(x) = \mathbb{R}^m$) has been considered. To avoid the computation of the second-order derivatives in (1.5), DARTS approximates the matrix-vector product $\nabla^2_{xy} f_\ell(x_k, y_k)\nabla_y f_u(x_k, \tilde{y}_k)$ by a finite-difference scheme [28]:

$$\nabla^2_{xy} f_\ell(x_k, y_k)\nabla_y f_u(x_k, \tilde{y}_k) \;=\; \frac{\nabla_x f_\ell(x_k, y_k^+) - \nabla_x f_\ell(x_k, y_k^-)}{2\varepsilon},$$

where

$$y_k^\pm \;=\; y_k \pm \varepsilon \nabla_y f_u(x_k, \tilde{y}_k) \quad \text{with} \quad \varepsilon \;=\; 0.01/\|\nabla_y f_u(x_k, \tilde{y}_k)\|. \tag{5.6}$$

Algorithm 3 reports the schema of DARTS for the stochastic setting. In Step 1, a single step of SG (with fixed stepsize $\eta$) is applied to the LL problem to obtain an approximation $\tilde{y}_k$ to the LL optimal solution. Then, in Step 2, the UL variables are updated by moving along the "approximated" descent direction using a stepsize $\alpha_k$.

**Algorithm 3** Differentiable Architecture Search (DARTS)

**Input:** $(x_0, y_0) \in \mathbb{R}^n \times \mathbb{R}^m$, $\{\alpha_k\}_{k\geq 0} > 0$, $\eta > 0$.

**For** $k = 0, 1, 2, \ldots$ **do**

**Step 1.** Compute $\tilde{y}_k = y_k - \eta\, g^\ell_y(x_k, y_k, \vartheta^\ell_k)$.

**Step 2.** Compute $x_{k+1} = x_k - \alpha_k \left(g^u_x(x_k, \tilde{y}_k, \vartheta^u_k) - \frac{\eta}{2\varepsilon}(g^\ell_x(x_k, y_k^+, \vartheta^\ell_k) - g^\ell_x(x_k, y_k^-, \vartheta^\ell_k))\right)$, with $y_k^\pm$ and $\varepsilon$ as in (5.6), with $g^u_y(x_k, \tilde{y}_k, \vartheta^u_k)$ instead of $\nabla_y f_u(x_k, \tilde{y}_k)$, and set $y_{k+1} = \tilde{y}_k$.

**End do**

### 5.3 Results for a synthetic quadratic bilevel problem

We first report results for a "synthetic" bilevel problem, where both levels are defined by quadratic objective functions. Given $h_1 \in \mathbb{R}^n$, $h_2 \in \mathbb{R}^m$, symmetric positive definite matrices $H_2 \in \mathbb{R}^{n\times n}$ and $H_3 \in \mathbb{R}^{m\times m}$, and matrices $H_1 \in \mathbb{R}^{n\times m}$ and $H_4 \in \mathbb{R}^{n\times m}$, we consider the following problem

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \; f_u(x, y) \;&=\; h_1^\top x + h_2^\top y + \frac{1}{2} x^\top H_1 y + \frac{1}{2} x^\top H_2 x, \\ y \in \operatorname*{argmin}_{y\in\mathbb{R}^m} \; f_l(x, y) \;&=\; \frac{1}{2} y^\top H_3 y - y^\top H_4 x. \end{aligned} \tag{5.7}$$

In the numerical experiments, we considered a dimension of 50 at both the upper and lower levels (i.e., $n = m = 50$), with $H_2$ and $H_3$ randomly generated and $H_1$ and $H_4$ set equal to the identity matrix. Note that problem (5.7) is deterministic. To investigate the numerical performance of the stochastic methods considered in the experiments, we computed stochastic gradient and Hessian estimates by adding Gaussian noise with mean 0 to each corresponding deterministic gradient (i.e., $\nabla_x f_u$, $\nabla_y f_u$, $\nabla_x f_\ell$, $\nabla_y f_\ell$) and Hessian (i.e., $\nabla^2_{xy} f_\ell$, $\nabla^2_{yy} f_\ell$). The values of the standard deviation were chosen from the set $\{0, 0.05, 4, 20\}$. In the experiments, we compared BSG-1 and BSG-H against DARTS using the best UL and LL fixed stepsizes (i.e.,

$\alpha^u$ and $\alpha^\ell$, respectively) found for each algorithm. Such stepsizes were chosen by performing a grid search over the set $\{i \times 10^{-s_u}\}$ for the UL stepsize and $\{i \times 10^{-s_\ell}\}$ for the LL stepsize, with $i \in \{1, \ldots, 9\}$, where $s_u = 4$ and $s_\ell = 3$ for BSG-1 and DARTS and $s_u = 3$ and $s_\ell = 2$ for BSG-H. Here we are not reporting numerical results for decaying stepsizes because that has led to worse results in terms of function values, but the relative positions of the methods are the same. When using the LL inc. acc. strategy, the threshold for increasing the number of LL steps was set to $10^{-1}$. In all the figures considered in this subsection, we plot the true function $f$ of the BLP ($f(x_k) = f_u(x_k, y(x_k))$). It can be easily shown that $f$ is actually a quadratic function itself. The number of iterations and running time are both used as metrics for the comparison of the algorithms.

From Figure 1, we can see that the BSG-H version of the BSG method performs significantly better than BSG-1 and DARTS when the standard deviation of the stochastic gradient and Hessian estimates is 0 or 0.05. Note that BSG-H yields the smallest objective function value in the least number of iterations. However, this requires a worse performance in terms of time due to the high computational cost of computing Hessian matrices. In this low-noise regime, DARTS achieves a performance comparable to BSG-1. These findings are consistent with the use of second-order derivatives by the three algorithms. When the standard deviation is 4 or 20, we were unable to find stepsizes that ensure a performance of BSG-H comparable to the other algorithms, and we omitted BSG-H from the corresponding plots. Overall, we can see that as the standard deviation increases, the performance of BSG-H and DARTS gets worse, while BSG-1 shows a higher robustness to the noise. The red dotted horizontal lines in all the figures represent the optimal value of the true function $f$.

### 5.4 Continual learning

We are going to use instances of Continual Learning (CL) as practical stochastic bilevel problems to test the performance of BSG and DARTS. CL was briefly described in Section 1, and is now introduced in more detail. Let us denote a whole features/labels dataset by $\mathcal{D} = \{(\mathbf{u}_j, \mathbf{v}_j),\ j \in \{1, \ldots, N\}\}$, consisting of $N$ pairs of a feature vector $\mathbf{u}_j$ and the corresponding true label $\mathbf{v}_j$. For any data point $j$, the classification is deemed correct if the right label is predicted. To evaluate the loss incurred when using the prediction function $\phi(\mathbf{u}; \theta)$, which in this section is supposed to be a DNN, we use a loss function $\ell(\phi(\mathbf{u}; \theta), \mathbf{v})$.

The goal of CL is to minimize the prediction error over a sequence of tasks that become available one at a time. Among the many different formulations proposed for CL, hierarchical objectives have been used in [41, 49]. In order to assess the performance of BSG against DARTS on this class of problems, we adapt the formulation in [41] to an incremental setting where each task is available as a subset of samples. Given $t \in \{1, \ldots, T\}$, let $\mathcal{D}_t$ be the set of samples for a new task $T_t$, which can be split into a training set $D^t_{\text{tr}}$ and a validation set $D^t_{\text{v}}$. Moreover, let us split the parameters $\theta_t$ of the model $\phi(\mathbf{u}; \theta_t)$ into two subvectors, $\lambda_t$ and $\delta_t$, whose roles are to give us flexibility in minimizing the classification error on the training and validation data along the sequence of tasks. According to [17], a reasonable strategy is to choose $\lambda_t$ and $\delta_t$ as the vectors of weights in the hidden and output layers, respectively.

To solve the overall CL problem, one starts from the first task $T_1$ and, after an arbitrary number of iterations or amount of time, we include in the problem the second task $T_2$. One reiterates this procedure until all the tasks have been added to the problem. Let us now suppose that one has already added $t$ tasks. At this stage, the goal of the UL and LL problems is to

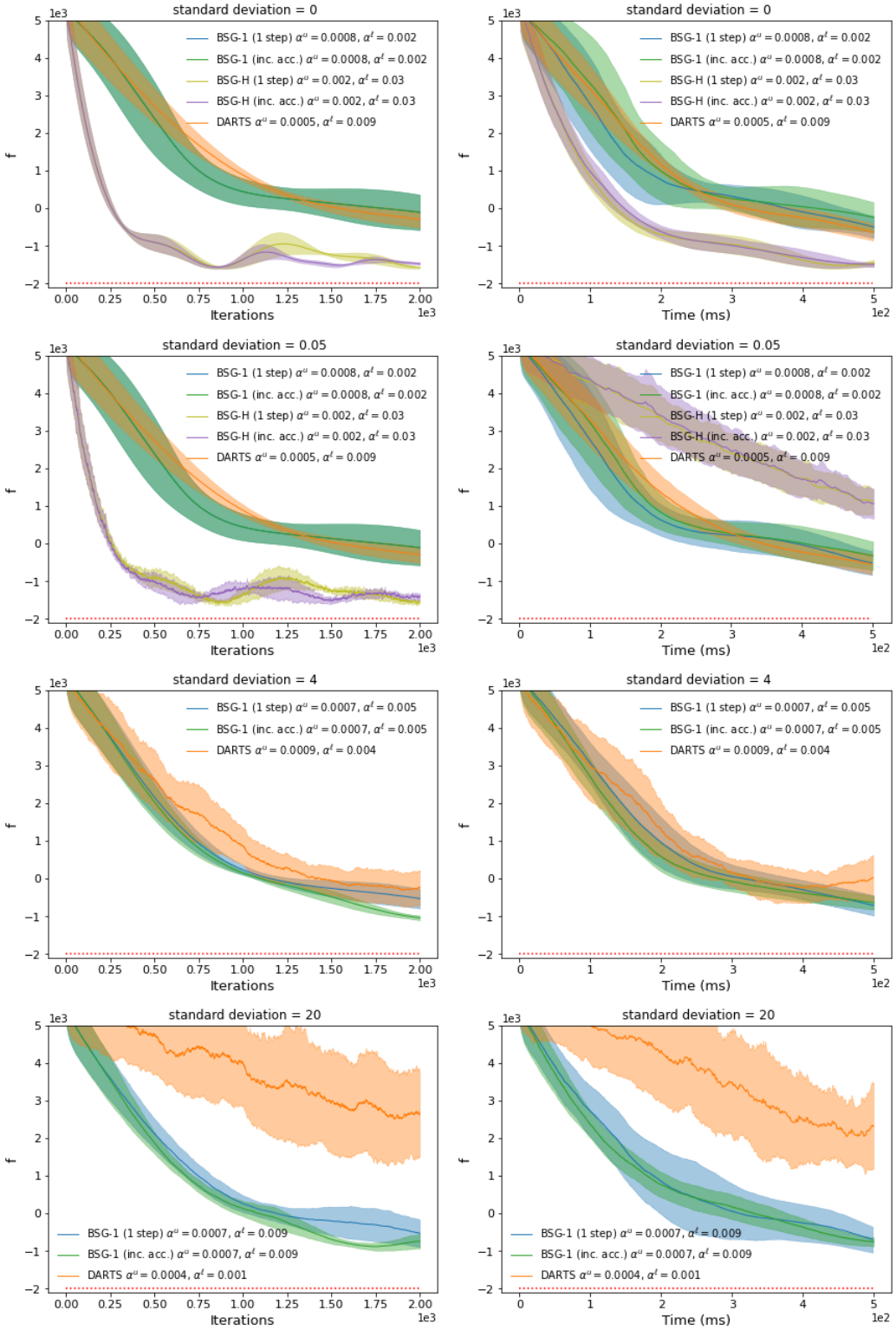


Figure 1: Comparison of different variants of the BSG method and DARTS on a synthetic quadratic bilevel problem for different values of the standard deviation of the stochastic gradient and Hessian estimates. The comparison is based on the number of iterations and time (in milliseconds).

determine the values of $\lambda_t$ and $\delta_t$ that ensure a small classification error on $T_t$ and on all the previous tasks $T_i$, with $i < t$. To this end, the UL problem determines $(\lambda_t, \delta_t)$ by minimizing the prediction error on $D^t_{\text{val}} = \cup_{i \le t} D^i_{\text{v}}$, which is composed of the data sampled from the validation sets associated with the current and previous tasks. Similarly, the LL problem determines $\delta_t$ by minimizing the error on $D^t_{\text{train}} = \cup_{i \le t} D^i_{\text{tr}}$. Note that at each stage one solves a different problem since the objective functions of the UL and LL change as new tasks are included in the problem. The formulation of the problem solved at stage $t$, with $t \in \{1, \ldots, T\}$, can be written as follows

$$\begin{aligned} \min_{(\lambda_t,\, \delta_t)} \quad & f_u(\lambda_t,\, \delta_t) \;=\; \frac{1}{|D^t_{\text{val}}|} \sum_{(\mathbf{u},\mathbf{v}) \in\, D^t_{\text{val}}} \ell(\phi(\mathbf{u}; \lambda_t, \delta_t), \mathbf{v}) \\ \text{s.t.} \quad & \delta_t \in \operatorname*{argmin}_{\delta_t} \quad f_\ell(\lambda_t,\, \delta_t) \;=\; \frac{1}{|D^t_{\text{train}}|} \sum_{(\mathbf{u},\mathbf{v}) \in\, D^t_{\text{train}}} \ell(\phi(\mathbf{u}; \lambda_t, \delta_t), \mathbf{v}). \end{aligned} \tag{5.8}$$

We point out that the large dimension of the datasets usually considered in ML may prevent the use of the whole sets $D^i_{\text{tr}}$ and $D^i_{\text{v}}$ from previous tasks $i$'s, where $i < t$ and $t$ is the current task. In such cases, it may be necessary to resort to subsets $\bar{D}^i_{\text{tr}} \subset D^i_{\text{tr}}$ and $\bar{D}^i_{\text{v}} \subset D^i_{\text{v}}$, which we will not do in this paper given that our interest focuses on the solution of stochastic BLPs.

Once a new task is included in the problem, the classification accuracy of the DNN on the previous tasks tends to deteriorate, thus resulting in the well-studied phenomenon of *catastrophic forgetting* [22], which can be alleviated by adding LL inequality constraints to the lower level of problem (5.8). Such inequality constraints are inspired by [31] and ensure that, at each stage, the current model outperforms the old model on all the previous tasks, thus preventing the deterioration of the classification accuracy when learning new tasks. In particular, for all $i < t$ and $t \ge 2$, we have

$$\sum_{(\mathbf{u},\mathbf{v}) \in\, D^i_{\text{tr}}} \ell(\phi(\mathbf{u}; \lambda_t, \delta_t), \mathbf{v}) - \sum_{(\mathbf{u},\mathbf{v}) \in\, D^i_{\text{tr}}} \ell(\phi(\mathbf{u}; \lambda_{t-1}, \delta_{t-1}), \mathbf{v}) \;\le\; 0. \tag{5.9}$$

We point out that similar constraints were also used in the bilevel formulation proposed in [49], where the violation of the constraints is penalized.

### 5.5 Results for continual learning instances

We now present numerical results comparing BSG-1 and DARTS on the CL problems (5.8) that were posed in Section 5.4. In our implementation, we determine $\lambda_t$ (the UL variables) on the current problem by starting from the parameter values found from the previous problem. However, since each consecutive task increases the output space of the DNN, we entirely re-initialize $\delta_t$ (the LL variables) at the start of each new task (when first applying a LL step) so that the model outputs are not biased from previous tasks.

In order to test our algorithm on a large-scale ML scenario, we chose the well-studied CIFAR-10 dataset [27] that consists of a total of 60,000 colored images ($32 \times 23$) of 10 different classes (i.e., airplanes, automobiles, birds, cats, deer, dogs, frogs, horses, ships, and trucks). The dataset is split into a training set that consists of 50,000 images and a testing set that consists of 10,000 images; however, for our experiments we only used the first set of images. We used a subset of 40,000 images for training and the remaining 9,999 images for validation (since one of the images had an issue, we removed it from the dataset). We solved five problems (5.8) with an

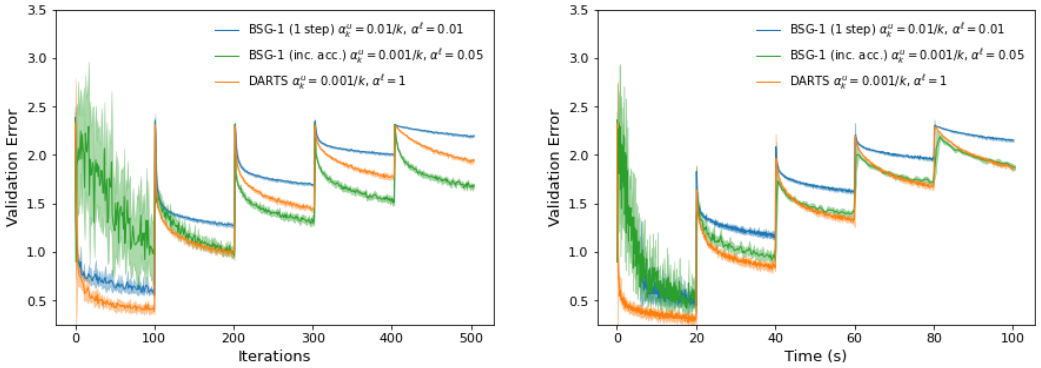


Figure 2: Comparison of two variants of the BSG-1 method (1 step and inc. acc.) and DARTS on a CL problem. The comparison is based on the number of iterations and time (in seconds).

increasing number of tasks from 1 to 5, where the first task datasets ($D^1_{\text{val}}$ and $D^1_{\text{train}}$) consist of only the images with class labels in $\{0, 1\}$ (these correspond to airplanes and automobiles), the second task datasets ($D^2_{\text{val}}$ and $D^2_{\text{train}}$) consist of the images with class labels in $\{0, 1, 2, 3\}$ (airplanes, automobiles, birds, and cats), etc., until the final task datasets ($D^5_{\text{val}}$ and $D^5_{\text{train}}$), which are the original training and validation sets and consist of all the class labels $\{0, 1, ..., 9\}$. Further, we implemented a DNN with two convolutional layers, a max-pooling layer, and one linear fully-connected layer as our model. The network consisted of 19,392 and 163,840 weights in the hidden and output layers, respectively. For the UL and LL problems, we have used batch sizes equal to 0.05% and 0.01% of the sizes of the current task's validation and training datasets, respectively.

The results for the unconstrained LL case are reported in Figure 2, where we plot the approximation $f_u$ of the true function $f$, as it is typically done in bilevel ML [23, 32, 40, 51, 57]. We are not reporting BSG-H because of the extremely high computational cost of dealing with second-order derivatives given the choices of DNN and dataset. Two variants of BSG-1 are compared against DARTS using the best UL decaying stepsize sequence $\{\alpha_k^u\}_{k\in\mathbb{N}}$ and the best LL fixed stepsize $\alpha^\ell$ found for each algorithm. Such stepsizes were chosen by performing a grid search over $\{0.001/k, 0.01/k, 0.1/k\}$ for the UL stepsize and $\{0.01, 0.05, 0.1, 1\}$ for the LL stepsize. For BSG-1, the results were gathered when using in the LL problem either 1 step or an increasing accuracy strategy with threshold for increasing the number of LL iterations equal to $10^{-2}$ (and maximum number of LL iterations equal to 30). The number of iterations and running time (in seconds) were both used as metrics for the comparison.

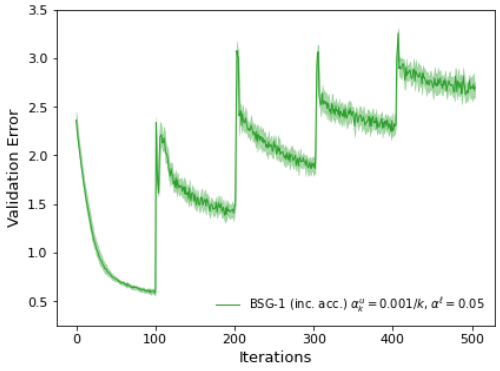


Figure 3: Performance of the BSG-1 method (inc. acc.) on a CL problem with LL constraints.

By the nature of the CL problem, we expect to see five separate "jumps" in the validation

error (the UL objective function) indicating the start of a new task. As the tasks progress, the BSG-1 variant using the increasing accuracy strategy converges faster than DARTS in terms of iterations, despite some initial noise. In terms of running time, both DARTS and BSG-1 with increasing accuracy achieve a comparable validation error as the number of tasks increases. We point out that the observed noise for BSG-1 in the first task could be reduced by using a different choice of the UL and LL stepsizes for that task but that would lead to a higher validation error in the remaining ones. We also note that BSG-1 with increasing accuracy would perform even better in terms of iterations the more accurately the LL problem is solved but it would lead to an increase in time.

Figure 3 shows as a proof of concept the numerical results of the BSG-1 algorithm with increasing accuracy on the CL problems (5.8) when considering LL constraints (5.9). In accordance with the procedure illustrated in Subsection 5.1 for the LL constrained case, approximate Lagrange multipliers are obtained at each iteration by solving the corresponding KKT system with the linear conjugate gradient method (with maximum number of iterations equal to 100 and tolerance equal to $10^{-4}$). The system in (5.4) is solved by using 50 iterations of the GMRES method.

# 6 Concluding remarks and future work

In this paper, we proposed a general framework for bilevel stochastic gradient (BSG) methods that applies to both the LL unconstrained and constrained cases, we provided a corresponding convergence theory that allows for full inexactness in the calculation of adjoint gradients which also rigorously covers the inexact solution of the LL problem, and we introduced a practical BSG method for large-scale bilevel optimization problems (called BSG-1).

The use of rank-1 Hessian approximations in our BSG-1 method allowed us to approximate the second-order derivatives in the adjoint formula and, in the LL unconstrained case, avoid explicitly solving the adjoint equation (also dismissing any matrix-vector products). Although DARTS does not incorporate descent or first-order principles, it is commonly applied to solve NAS problems due to its practical satisfactory performance. Our numerical experiments showed that BSG-1 performs better than DARTS on the synthetic quadratic bilevel problem considered. On the continual learning instances, BSG-1 outperforms DARTS in terms of iterations and achieves a comparable performance in terms of running time. Further exploration of improvements for BSG-1 and DARTS methods on other large-scale learning problems is left for future work. Also left for future work are variance reduction techniques, which can be incorporated into the BSG method to ensure faster convergence, as already proposed in [8, 57].

The promising results on the ML instances considered in this proposal suggest that the BSG-1 method has the potential to perform well on the unconstrained bilevel formulations of NAS, which in the literature are still tackled by using DARTS when a continuous relaxation of the (discrete) search space is used [43]. We point out that using the rank-1 Hessian approximations is crucial to allow the application of the BSG method to NAS, which would not be possible otherwise due to the extreme dimensions of the resulting bilevel problems. Moreover, the fact that BSG can solve bilevel optimization problems with constrained LL problems paves the way for the solution of new NAS formulations. In particular, one could think of including in the LL problem constraints that help the model avoid overfitting [42] or constraints that depend on the specific learning instances considered [47].

## References


[1] J. F. Bard. *Practical Bilevel Optimization: Algorithms and Applications*. Springer Publishing Company, Incorporated, 1st edition, 2010.

[2] A. Beck. *First-Order Methods in Optimization*. SIAM-Society for Industrial and Applied Mathematics, 2017.

[3] K. P. Bennett, G. Kunapuli, J. Hu, and J. S. Pang. *Bilevel Optimization and Machine Learning*, pages 25–47. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[4] A. Botev, H. Ritter, and D. Barber. Practical Gauss-Newton optimisation for deep learning. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 557–565. PMLR, 06–11 Aug 2017.

[5] L. Bottou, F. E. Curtis, and J. Nocedal. Optimization methods for large-scale machine learning. 60:223–311, 2018.

[6] C. Chen, S. Zheng, X. Chen, E. Dong, X. Liu, H. Liu, and D. Dou. Generalized dataweighting via class-level gradient manipulation. In *Advances in Neural Information Processing Systems*, 2021.

[7] C. Chen, X. Chen, C. Ma, Z. Liu, and X. Liu. Gradient-based bi-level optimization for deep learning: A survey. *arXiv preprint arXiv:2207.11719*, 2022.

[8] T. Chen, Y. Sun, and W. Yin. A Single-Timescale Stochastic Bilevel Optimization Method. *arXiv e-prints*, art. arXiv:2102.04671, February 2021.

[9] T. Chen, Y. Sun, and W. Yin. Closing the gap: Tighter analysis of alternating stochastic gradient methods for bilevel problems. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 25294–25307. Curran Associates, Inc., 2021.

[10] K. L. Chung. On a stochastic approximation method. *Annals of Mathematical Statistics*, 25:463 – 483, 1954.

[11] B. Colson, P. Marcotte, and G. Savard. An overview of bilevel optimization. *Annals of Operations Research*, 153:235–256, 2007.

[12] N. Couellan and W. Wang. Bi-level stochastic gradient for large scale support vector machine. *Neurocomputing*, 153:300–308, 2015.

[13] N. Couellan and W. Wang. On the convergence of stochastic bi-level gradient methods. *PREPRINT available at http://www.optimization-online.org/DB_HTML/2016/02/5323.html*, 02 2016.

[14] S. Dempe and A. Zemkoho. *Bilevel Optimization: Advances and Next Challenges*. Springer International Publishing, 2020.

[15] S. Dhar, U. Kurup, and M. Shah. Stabilizing bi-Level hyperparameter optimization using Moreau-Yosida regularization. *arXiv e-prints*, art. arXiv:2007.13322, July 2020.

[16] Weinan E. A proposal on machine learning via dynamical systems. *Communications in Mathematics and Statistics*, 5:1–11, 02 2017.

[17] L. Franceschi, P. Frasconi, S. Salzo, R. Grazzi, and M. Pontil. Bilevel programming for hyperparameter optimization and meta-learning. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1568–1577. PMLR, 2018.

[18] R. M. French. Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3:128–135, 1999.

[19] M. Gargiani, A. Zanelli, M. Diehl, and F. Hutter. On the promise of the stochastic generalized Gauss-Newton method for training DNNs. *arXiv e-prints*, art. arXiv:2006.02409, June 2020.

[20] S. Ghadimi and M. Wang. Approximation methods for bilevel programming, 2018.

[21] I. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

[22] I. J. Goodfellow, M. Mirza, D. Xiao, A. Courville, and Y. Bengio. An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv e-prints*, art. arXiv:1312.6211, December 2013.

[23] M. Hong, H. Wai, Z. Wang, and Z. Yang. A Two-Timescale Framework for Bilevel Optimization: Complexity Analysis and Application to Actor-Critic. *arXiv e-prints*, art. arXiv:2007.05170, July 2020.

[24] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey. Meta-Learning in neural networks: A survey. *arXiv e-prints*, art. arXiv:2004.05439, April 2020.

[25] K. Ji, J. Yang, and Y. Liang. Bilevel Optimization: Convergence Analysis and Enhanced Design. *arXiv e-prints*, art. arXiv:2010.07962, October 2020.

[26] H. Jiang, Z. Chen, Y. Shi, B. Dai, and T. Zhao. Learning to defend by learning to attack. *arXiv e-prints*, art. arXiv:1811.01213, November 2018.

[27] A. Krizhevsky. Learning multiple layers of features from tiny images. pages 32–33, 2009. URL `https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf`.

[28] H. Liu, K. Simonyan, and Y. Yang. DARTS: Differentiable architecture search. *ArXiv*, arXiv:1806.09055, 2019.

[29] R. Liu, J. Gao, J. Zhang, D. Meng, and Z. Lin. Investigating bi-Level optimization for learning and vision from a unified perspective: A survey and beyond. *arXiv e-prints*, art. arXiv:2101.11517, January 2021.

[30] S. Liu and L. N. Vicente. The stochastic multi-gradient algorithm for multi-objective optimization and its application to supervised machine learning. *arXiv e-prints*, art. arXiv:1907.04472, July 2019.

[31] D. Lopez-Paz and M. Ranzato. Gradient episodic memory for continual learning. *arXiv e-prints*, art. arXiv:1706.08840, June 2017.

[32] J. Lorraine, P. Vicol, and D. Duvenaud. Optimizing Millions of Hyperparameters by Implicit Differentiation. *arXiv e-prints*, art. arXiv:1911.02590, November 2019.

[33] Y. Lu, A. Zhong, Q. Li, and B. Dong. Beyond Finite Layer Neural Networks: Bridging Deep Architectures and Numerical Differential Equations. *arXiv e-prints*, art. arXiv:1710.10121, October 2017.

[34] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[35] M. McCloskey and N. J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. volume 24 of *Psychology of Learning and Motivation*, pages 109–165. Academic Press, 1989.

[36] G. P. McCormick. Optimality criteria in nonlinear programming. pages 27–38, Philadelphia, PA, USA, 1976. SIAM.

[37] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on Optimization*, 19:1574–1609, 2009.

[38] Y. Nesterov. *Lectures on Convex Optimization*. Springer Publishing Company, Incorporated, 2nd edition, 2018. ISBN 3319915770.

[39] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer-Verlag, Berlin, second edition, 2006.

[40] F. Pedregosa. Hyperparameter optimization with approximate gradient. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 737–746, New York, New York, USA, 20–22 Jun 2016. PMLR.

[41] Q. Pham, D. Sahoo, C. Liu, and S. C. H. Hoi. Bilevel continual learning, 2020.

[42] S. N. Ravi, T. Dinh, V. S. Lokhande, and V. Singh. Explicitly imposing constraints in deep networks via conditional gradients gives improved generalization and faster convergence. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):4772–4779, 2019.

[43] P. Ren, Y. Xiao, X. Chang, P. Huang, Z. Li, X. Chen, and X. Wang. A comprehensive survey of neural architecture search: Challenges and solutions. *ACM Comput. Surv.*, 54, 2021.

[44] H. Robbins and S. Monro. A stochastic approximation method. *Annals of Mathematical Statistics*, 22:400–407, 1951.

[45] Y. Saad and M. H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on Scientific and Statistical Computing*, 7: 856–869, 1986. doi: 10.1137/0907058. URL `https://doi.org/10.1137/0907058`.

[46] J. Sacks. Asymptotic distribution of stochastic approximation procedures. *Annals of Mathematical Statistics*, 29:373 – 405, 1958.

[47] S. Sangalli, E. Erdil, A. Hoetker, O. Donati, and E. Konukoglu. Constrained optimization to train neural networks on critical and under-represented Classes. *arXiv e-prints*, art. arXiv:2102.12894, February 2021.

[48] G. Savard and J. Gauvin. The steepest descent direction for the nonlinear bilevel programming problem. *Operations Research Letters*, 15:265–272, 1994.

[49] A. Shaker, F. Alesiani, S. Yu, and W. Yin. Bilevel continual learning, 2020.

[50] A. Sinha, P. Malo, and K. Deb. A review on bilevel optimization: From classical to evolutionary approaches and applications. *IEEE Transactions on Evolutionary Computation*, 22:276–295, 2018.

[51] D. Sow, K. Ji, and Y. Liang. On the Convergence Theory for Hessian-Free Bilevel Algorithms. *arXiv e-prints*, 2021.

[52] H. Sun, W. Pu, X. Fu, T.H. Chang, and M. Hong. Learning to continuously optimize wireless resource in a dynamic environment: A bilevel optimization perspective. *IRE Transactions on Audio*, 70:1900–1917, 2022. ISSN 1053-587X.

[53] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *arXiv e-prints*, art. arXiv:1312.6199, December 2013.

[54] I. Tsaknakis, P. Khanduri, and M. Hong. An implicit gradient-type method for linearly constrained bilevel problems. ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings, pages 5438–5442. Institute of Electrical and Electronics Engineers Inc., 2022.

[55] L. N. Vicente and P. H. Calamai. Bilevel and multilevel programming: A bibliography review. *Journal of Global Optimization*, 5:291–306, 1994.

[56] Y. Xu. Primal-dual stochastic gradient method for convex programs with many functional constraints. *SIAM Journal on Optimization*, 30(2):1664–1692, 2020. doi: 10.1137/18M1229869. URL `https://doi.org/10.1137/18M1229869`.

[57] J. Yang, K. Ji, and Y. Liang. Provably Faster Algorithms for Bilevel Optimization. *arXiv e-prints*, art. arXiv:2106.04692, June 2021.

# A Proposition 3.1

**Proof.** There are two cases to consider (inexact adjoint system and truncated Neumann series). In both we will use the fact that when $B_1$ and $B_2$ are nonsingular,

$$\|B_1^{-1} - B_2^{-1}\| \leq \|B_1^{-1}(B_1 - B_2)B_2^{-1}\| \leq \|B_1^{-1}\|\|B_2^{-1}\|\|B_1 - B_2\|. \tag{A.1}$$

*1) Inexact adjoint system.*
The approximate BSG direction is $d(D) = -(a - AB^{-1}\tilde{b})$, where $\tilde{b} = b + \tilde{r}$ and $\tilde{r}$ is the residual error due to the inexact solution of the ajoint equation (see Subsection 2.5). Now, we have

$$\|d(D_1) - d(D_2)\| = \| - a_1 + A_1 B_1^{-1}\tilde{b}_1 + a_2 - A_2 B_2^{-1}\tilde{b}_2\|.$$

Adding and subtracting $A_1 B_1^{-1}\tilde{b}_2$ and using the triangle inequality, we obtain

$$\|d(D_1) - d(D_2)\| \leq \|a_1 - a_2\| + \|A_1 B_1^{-1}\|\|\tilde{b}_1 - \tilde{b}_2\| + \|\tilde{b}_2\|\|A_1 B_1^{-1} - A_2 B_2^{-1}\|.$$

Adding and subtracting $A_2 B_1^{-1}$ in the last norm on the right, we have

$$\begin{aligned}\|d(D_1) - d(D_2)\| \leq\ & \|a_1 - a_2\| + \|A_1\|\|B_1^{-1}\|\|\tilde{b}_1 - \tilde{b}_2\| \\ & + \|\tilde{b}_2\|\|A_2\|\|B_1^{-1} - B_2^{-1}\| + \|\tilde{b}_2\|\|B_1^{-1}\|\|A_1 - A_2\|.\end{aligned}$$

From the assumptions of Proposition 3.1 and (A.1), and setting $r_1 = \tilde{r}_1$ and $r_2 = \tilde{r}_2$, there exist positive constants $C_i$, with $i \in \{1, \ldots, 5\}$, and $L = \max\{1, C_1, C_2 + C_3\|r_2\|, C_4 + C_5\|r_2\|\}$, such that

$$\begin{aligned}\|d(D_1) - d(D_2)\| &\leq \|a_1 - a_2\| + C_1(\|b_1 - b_2\| + \|r_1 - r_2\|) \\ &\quad + (C_2 + C_3\|r_2\|)\,\|B_1 - B_2\| + (C_4 + C_5\|r_2\|)\,\|A_1 - A_2\| \\ &\leq L(\|a_1 - a_2\| + \|b_1 - b_2\| + \|B_1 - B_2\| + \|A_1 - A_2\|) + L\|r_1 - r_2\|. \quad \text{(A.2)}\end{aligned}$$

Thus, the proof of Part 1) is completed, where $L_{BSG}$ can be easily obtained from $L$ and equivalence of norms.

*2) Truncated Neumann series.*
The approximate BSG direction is $d(D) = -(a - A\mathscr{B}b)$, where $\mathscr{B} = B^{-1} - \tilde{R}$ and $\tilde{R}$ is a residual matrix (see Subsection 2.5). By repeating the reasoning used to prove Part 1) until (A.2), we arrive at

$$\begin{aligned}\|d(D_1) - d(D_2)\| \leq\ & \|a_1 - a_2\| + \|A_1\|\,\|\mathscr{B}_1\|\,\|b_1 - b_2\| \\ & + \|b_2\|\,\|A_2\|\,\|\mathscr{B}_1 - \mathscr{B}_2\| + \|b_2\|\,\|\mathscr{B}_1\|\,\|A_1 - A_2\|\,.\end{aligned}$$

Therefore, by using the same arguments as in Part 1), but now with $r_1 = \tilde{R}_1$ and $r_2 = \tilde{R}_2$, there exist positive constants $C_i$, with $i \in \{1, \ldots, 5\}$ and $L = \max\{1, C_1 + C_2\|r_1\|, C_3, C_4 + C_5\|r_1\|\}$, such that

$$\begin{aligned}\|d(D_1) - d(D_2)\| &\leq \|a_1 - a_2\| + (C_1 + C_2\|r_1\|)\,\|b_1 - b_2\| \\ &\quad + C_3(\|B_1 - B_2\| + \|r_1 - r_2\|) + (C_4 + C_5\|r_1\|)\,\|A_1 - A_2\| \\ &\leq L(\|a_1 - a_2\| + \|b_1 - b_2\| + \|B_1 - B_2\| + \|A_1 - A_2\|) + L\|r_1 - r_2\|. \quad \text{(A.3)}\end{aligned}$$

Thus, the proof of Part 2) is completed, where again $L_{BSG}$ can be easily obtained from $L$ and equivalence of norms. □

# B Smoothness of $f$

**Proposition B.1** *Under Assumption 3.1, assuming $\nabla_y f_u$ bounded in norm, and*

- *either $\nabla^2_{xy} f_\ell$ and $\nabla^2_{yy} f_\ell$ (for the unconstrained case) or $\nabla_x c_I$, $\nabla_x c_E$, $\nabla_y c_I$, $\nabla_y c_E$, $\nabla^2_{yx} c_i$, and $\nabla^2_{yy} c_i$ with $i \in I \cup E$ (for the constrained case) bounded in norm,*
- *either $\nabla^2_{yy} f_\ell$ (for the unconstrained case) or $\nabla_v G$ (for the constrained case) bounded away from singularity,*

*there exists a constant $L_{\nabla f} > 0$ such that inequality (3.18) of Assumption 3.7 is satisfied.*

**Proof.** Taking the norm of equations (1.4) and (2.5) and from the boundedness of $\nabla^2_{yy} f_\ell$ and $\nabla_v G$, there exist positive constants $L_y$ and $L_w$ such that $\|\nabla y(x)\| \le L_y$ and $\|\nabla w(x)\| \le L_w$. This implies that $y(x)$ and $w(x)$ are Lipschitz continuous in $x$ with constants $L_y$ and $L_w$, respectively, i.e.,

$$\|y(x_1) - y(x_2)\| \;\le\; L_y \|x_1 - x_2\| \quad \text{and} \quad \|w(x_1) - w(x_2)\| \;\le\; L_w \|x_1 - x_2\|. \tag{B.1}$$

To prove (3.18) for the LL unconstrained case, using the derivation followed for the proof of Proposition 3.1 until (A.2)–(A.3) and considering $r_1 = r_2 = 0$, we have

$$\|\nabla f(x_1) - \nabla f(x_2)\| \;\le\; L\,(\|a_1 - a_2\| + \|b_1 - b_2\| + \|B_1 - B_2\| + \|A_1 - A_2\|), \tag{B.2}$$

with $a_i = \nabla_x f_u(x_i, y(x_i))$, $b_i = \nabla_y f_u(x_i, y(x_i))$, $B_i = \nabla^2_{yy} f_\ell(x_i, y(x_i))$, and $A_i = \nabla^2_{xy} f_\ell(x_i, y(x_i))$, $i = 1, 2$.

By the Lipschitz continuity of $\nabla_x f_u$ in $(x, y)$ due to Assumption 3.1, we have

$$\|a_1 - a_2\| \;=\; \|\nabla_x f_u(x_1, y(x_1)) - \nabla_x f_u(x_2, y(x_2))\| \;\le\; L_1 \|(x_1 - x_2, y(x_1) - y(x_2))^\top\|.$$

Squaring both sides and using (B.1), we obtain

$$\|a_1 - a_2\|^2 \;\le\; L_1^2(\|x_1 - x_2\|^2 + L_y^2 \|x_1 - x_2\|^2) \;=\; L_1^2(1 + L_y^2)\|x_1 - x_2\|^2.$$

Taking the square root of both sides yields $\|a_1 - a_2\| \le L_a \|x_1 - x_2\|$, where $L_a = L_1(1 + L_y^2)^{\frac{1}{2}}$. Since $\nabla_y f_u$ is Lipschitz continuous in $(x, y)$, after performing the same process as above, we will obtain the following bound: $\|\nabla_y f_u(x_1, y(x_1)) - \nabla_y f_u(x_2, y(x_2))\| \;\le\; L_b \|x_1 - x_2\|$. Similarly, since $\nabla^2_{yy} f_\ell$ and $\nabla^2_{xy} f_\ell$ are Lipschitz continuous in $(x, w)$, we have $\|\nabla^2_{yy} f_\ell(x_1, y(x_1)) - \nabla^2_{yy} f_\ell(x_2, y(x_2))\| \;\le\; L_B \|x_1 - x_2\|$ and $\left\|\nabla^2_{xy} f_\ell(x_1, y(x_1)) - \nabla^2_{xy} f_\ell(x_2, y(x_2))\right\| \;\le\; L_A \|x_1 - x_2\|$. Thus, substituting all of these into (B.2), we obtain

$$\|\nabla f(x_1) - \nabla f(x_2)\| \;\le\; L_{\nabla f} \|x_1 - x_2\|,$$

where $L_{\nabla f} = L\,(L_a + L_b + L_A + L_B)$. This concludes the proof for the LL unconstrained case. The proof for the LL constrained case follows very similar steps. □

## C Theorem 4.4

**Proof.** For any $k \in \mathbb{N}$, we can write

$$\begin{aligned}\mathbb{E}_{\xi_k^{\text{all}}}[\|x_{k+1} - x_*\|^2] &= \mathbb{E}_{\xi_k^{\text{all}}}[\|P_X(x_k + \alpha_k\, d(x_k, \tilde{w}_k, \xi_k)) - x_*\|^2] \\ &\leq \mathbb{E}_{\xi_k^{\text{all}}}[\|x_k + \alpha_k\, d(x_k, \tilde{w}_k, \xi_k) - x_*\|^2] \\ &= \|x_k - x_*\|^2 + \alpha_k^2 \mathbb{E}_{\xi_k^{\text{all}}}[\|d(x_k, \tilde{w}_k, \xi_k)\|^2] \\ &\quad + 2\alpha_k \mathbb{E}_{\xi_k^{\text{all}}}[d(x_k, \tilde{w}_k, \xi_k)]^\top (x_k - x_*).\end{aligned}$$

Adding and subtracting the term $2\alpha_k(\mathbb{E}_{\xi_k^{\text{all}}}[d(x_k, w(x_k))])^\top(x_k - x_*)$, noting that $d(x_k, w(x_k)) = -\nabla f(x_k)$, and applying the Cauchy-Schwarz and Jensen's inequalities, we obtain

$$\begin{aligned}\mathbb{E}_{\xi_k^{\text{all}}}[\|x_{k+1} - x_*\|^2] &\leq \|x_k - x_*\|^2 + \alpha_k^2 \mathbb{E}_{\xi_k^{\text{all}}}[\|d(x_k, \tilde{w}_k, \xi_k)\|^2] - 2\alpha_k \nabla f(x_k)^\top (x_k - x_*) \\ &\quad + 2\alpha_k \mathbb{E}_{\xi_k^{\text{all}}}[d(x_k, \tilde{w}_k, \xi_k) - d(x_k, w(x_k))]^\top (x_k - x_*) \\ &\leq \|x_k - x_*\|^2 + \alpha_k^2 \mathbb{E}_{\xi_k^{\text{all}}}[\|d(x_k, \tilde{w}_k, \xi_k)\|^2] - 2\alpha_k \nabla f(x_k)^\top (x_k - x_*) \\ &\quad + 2\alpha_k \mathbb{E}_{\xi_k^{\text{all}}}[\|d(x_k, \tilde{w}_k, \xi_k) - d(x_k, w(x_k))\|]\|x_k - x_*\|.\end{aligned}$$

Then, by using Assumption 4.4 and inequalities (3.3), (3.8), and (4.7),

$$\mathbb{E}_{\xi_k^{\text{all}}}[\|x_{k+1} - x_*\|^2] \;\leq\; (1 - 2c\alpha_k)\|x_k - x_*\|^2 + (G_d + 2C_d\Theta)\,\alpha_k^2.$$

Denoting $M = G_d + 2C_d\Theta$ and taking the total expectation on both sides, one obtains

$$\mathbb{E}[\|x_{k+1} - x_*\|^2] \;\leq\; (1 - 2c\alpha_k)\mathbb{E}[\|x_k - x_*\|^2] + M\alpha_k^2.$$

Using $\alpha_k = \frac{\gamma}{k}$ for some constant $\gamma > \frac{1}{2c}$, it follows by induction [37, Eq. (2.9) and (2.10)] that

$$\mathbb{E}[\|x_k - x_*\|^2] \;\leq\; \frac{\max\{2\,\gamma^2 M(2c\gamma - 1)^{-1}, \|x_0 - x_*\|^2\}}{k}, \tag{C.1}$$

which proves the first result.

From (3.19), one obtains (see, e.g., [2, Lemma 5.7] and $P_X(x_k - x_*) = (x_k - x_*)$)

$$f(x_k) \;\leq\; f(x_*) + (P_X \nabla f(x_*))^\top (x_k - x_*) + \frac{1}{2} L_{\nabla f} \|x_k - x_*\|^2. \tag{C.2}$$

From (C.1) and (C.2), by taking the total expectation and recalling $P_X \nabla f(x_*) = 0$, one can obtain the optimality gap in terms of function values

$$\begin{aligned}\mathbb{E}[f(x_k)] - f(x_*) &\leq \frac{1}{2} L_{\nabla f} \mathbb{E}[\|x_k - x_*\|^2] \\ &\leq \frac{(L_{\nabla f}/2)\max\{2\,\gamma^2 M(2c\gamma - 1)^{-1}, \|x_0 - x_*\|^2\}}{k}.\end{aligned}$$

□

# D Theorem 4.5

**Proof.** Assumption 4.6 implies that

$$\nabla f(x_k)^\top (x_k - x_*) \;\geq\; f(x_k) - f(x_*). \tag{D.1}$$

Repeating the same arguments that in the proof of Theorem 4.4 led to (C.1), but now using (D.1) instead,

$$\mathbb{E}_{\xi_k^{\text{all}}}[\|x_{k+1} - x_*\|^2] \;\leq\; \|x_k - x_*\|^2 + 2\alpha_k(f(x_*) - f(x_k)) + (G_d + 2C_d\Theta)\alpha_k^2.$$

Letting $M = G_d + 2C_d\Theta$, we have

$$\mathbb{E}_{\xi_k^{\text{all}}}\left[\|x_{k+1} - x_*\|^2\right] \;\leq\; \|x_k - x_*\|^2 + 2\alpha_k(f(x_*) - f(x_k)) + M\alpha_k^2.$$

Rearranging, taking total expectations, and dividing by $\alpha_k$, we obtain

$$2(\mathbb{E}[f(x_k)] - f(x_*)) \;\leq\; \frac{\mathbb{E}\left[\|x_k - x_*\|^2\right]}{\alpha_k} - \frac{\mathbb{E}\left[\|x_{k+1} - x_*\|^2\right]}{\alpha_k} + M\alpha_k.$$

If we replace $k$ by $s$ and sum over $s = 0, 1, \ldots, k$, we obtain

$$\begin{aligned}
2\sum_{s=0}^{k}(\mathbb{E}[f(x_s)] - f(x_*)) \;&\leq\; \sum_{s=0}^{k}\left(\frac{\mathbb{E}\left[\|x_s - x_*\|^2\right]}{\alpha_s} - \frac{\mathbb{E}\left[\|x_{s+1} - x_*\|^2\right]}{\alpha_s}\right) + M\sum_{s=0}^{k}\alpha_s \\
&=\; \frac{\mathbb{E}\left[\|x_0 - x_*\|^2\right]}{\alpha_0} + \sum_{s=1}^{k}\left(\frac{1}{\alpha_s} - \frac{1}{\alpha_{s-1}}\right)\mathbb{E}\left[\|x_s - x_*\|^2\right] + M\sum_{s=0}^{k}\alpha_s \\
&=\; \frac{\mathbb{E}\left[\|x_k - x_*\|^2\right]}{\alpha_k} + M\sum_{s=0}^{k}\alpha_s.
\end{aligned}$$

Using Assumption 4.4 and repeating the same steps used in [30, Theorem 5.3], we can obtain the desired result. □