Recent Advances in First-Order Methods for Continuous Bilevel Optimization

Zhaosong Lu

University of Minnesota

International Conference on Bilevel Optimization
Pittsburgh, USA

August 2, 2026

Acknowledgment:
Mingyi Hong (UMN), Stephen Wright (UW-Madison), Xiangyuan Wang (UMN)

Outline

Part I: Introduction to bilevel optimization

Introduction to bilevel optimization (BO)

Mathematical form:

$$\begin{aligned} \min_{x,y} \quad & f(x,y) \\ \text{s.t.} \quad & y \in \operatorname*{argmin}_z \left\{ \tilde{f}(x,z) \mid \tilde{g}(x,z) \leq 0 \right\}. \end{aligned}$$

- f is the upper-level objective function, while $\tilde{f}$ and $\tilde{g}$ are the lower-level objective and constraint functions, respectively.
- x is the upper-level decision variable (leader), and y is the lower-level decision variable (follower).
- The associated ***hyper-objective*** function is

$$\Phi(x) := \min \left\{ f(x,y) \mid y \in \operatorname*{argmin}_z \left\{ \tilde{f}(x,z) \mid \tilde{g}(x,z) \leq 0 \right\} \right\},$$

and the BO problem can be equivalently written as $\min_x \Phi(x)$.

Historical background of BO

In 1934, Heinrich von Stackelberg introduced the Stackelberg game, a strategic game involving two players [Stackelberg 1934].

Leader: moves first and commits to a strategy.

Follower: observes the leader's decision and chooses an optimal response.

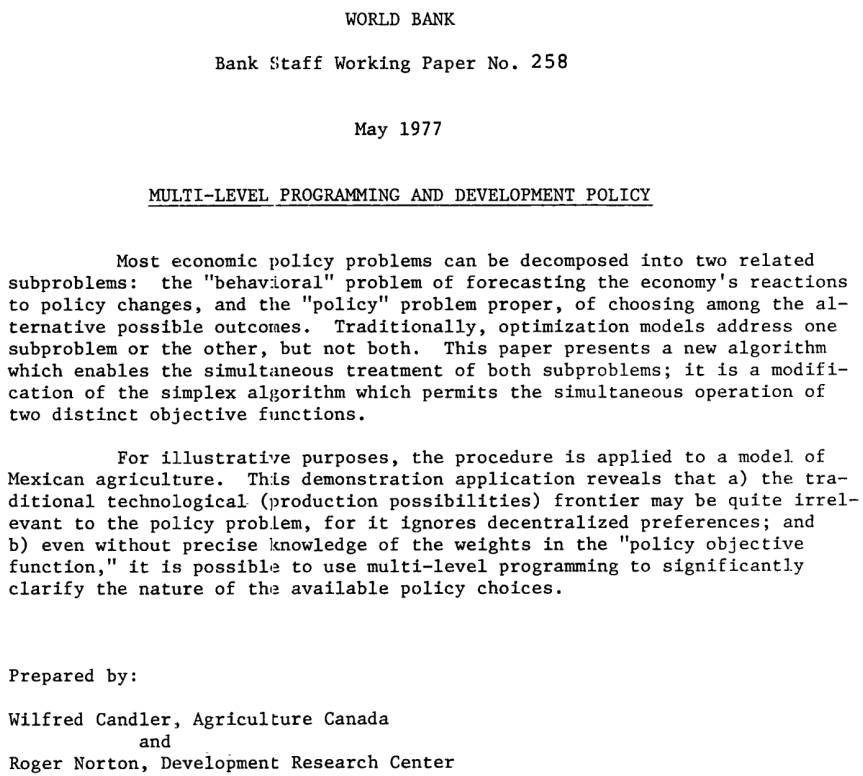
The leader anticipates the reaction of the follower and optimizes its own objective.

This leader-follower paradigm naturally corresponds to the hierarchical structure of BO.

Historical background of BO (cont'd)

The term "bilevel" optimization was formally introduced in the early 1970s, with an application to military resource allocation [Bracken and McGill 1973].

The terms "bilevel" and "multilevel" optimization were subsequently introduced in a World Bank report.



WORLD BANK

Bank Staff Working Paper No. 258

May 1977

MULTI-LEVEL PROGRAMMING AND DEVELOPMENT POLICY

Most economic policy problems can be decomposed into two related subproblems: the "behavioral" problem of forecasting the economy's reactions to policy changes, and the "policy" problem proper, of choosing among the alternative possible outcomes. Traditionally, optimization models address one subproblem or the other, but not both. This paper presents a new algorithm which enables the simultaneous treatment of both subproblems; it is a modification of the simplex algorithm which permits the simultaneous operation of two distinct objective functions.

For illustrative purposes, the procedure is applied to a model of Mexican agriculture. This demonstration application reveals that a) the traditional technological (production possibilities) frontier may be quite irrelevant to the policy problem, for it ignores decentralized preferences; and b) even without precise knowledge of the weights in the "policy objective function," it is possible to use multi-level programming to significantly clarify the nature of the available policy choices.

Prepared by:

Wilfred Candler, Agriculture Canada
and
Roger Norton, Development Research Center

Historical background of BO (cont'd)

Since the 1980s, BO has received increasing attention due to its applications in transportation, economics, power systems, signal processing, and machine learning.

Modern machine learning problems often involve several interacting learning or optimization subproblems. BO provides a natural framework for modeling such problems because it explicitly captures their hierarchical structure.

Application 1: AI learning from experience

The *Era of Experience* envisions an AI learning paradigm based on continued interaction and feedback rather than only static datasets [Silver and Sutton 2025].

This paradigm introduces **nested** learning dynamics:

- Upper level: learn a reward or preference model from human feedback.
- Lower level: adapt the policy through reinforcement learning under the learned reward.

> Furthermore, users could provide feedback during the learning process, such as their satisfaction level, which could be used to fine-tune the reward function. The reward function can then adapt over time, to improve the way in which it selects or combines signals, and to identify and correct any misalignment. This can also be understood as a bi-level optimisation process that optimises user feedback as the top-level goal, and optimises grounded signals from the environment at the low level.[4] In this way, a small amount of human data may facilitate a large amount of autonomous learning.
>
> Silver and Sutton, "Welcome to the Era of Experience," 2025

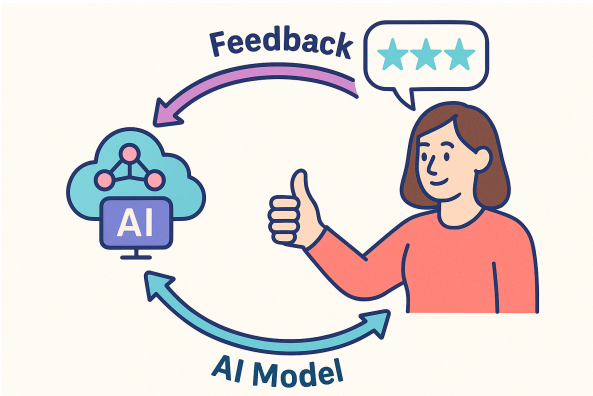
Application 1: AI learning from experience (cont'd)

BO provides a natural algorithmic framework for jointly learning the reward model and the corresponding policy.

A rough BO formulation is

$$\begin{aligned} \min_{\theta_{\text{reward}}} \quad & \mathcal{L}_{\text{user}}\left(\pi^\star(\theta_{\text{reward}})\right) \\ \text{s.t.} \quad & \pi^\star(\theta_{\text{reward}}) \in \operatorname*{argmax}_{\pi} R_{\theta_{\text{reward}}}(\pi). \end{aligned}$$

- Feedback is provided interactively during training through preferences, corrections, or satisfaction scores.
- R_θ is a reward model that is updated using the received feedback.
- $\pi^\star(\theta)$ is the corresponding policy, which is optimized to maximize the learned reward.



Application 1: AI learning from experience (cont'd)

A concrete BO formulation is inverse reinforcement learning, which infers the reward function underlying observed expert policies.

Upper-level problem (leader):
Find reward parameters θ such that the resulting optimal policy $\pi^\star(\theta)$ matches the expert policy π_E:

$$\min_\theta \mathrm{Dist}\big(\pi^\star(\theta), \pi_E\big).$$

The distance can measure, e.g., differences in feature expectations or divergence between the policies.

Lower-level problem (follower):
For a given reward function R_θ, find an optimal policy:

$$\pi^\star(\theta) \in \operatorname*{argmax}_\pi \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0}^{T} R_\theta(s_t, a_t)\right].$$

It is a reinforcement-learning problem under the reward R_θ.

Application 2: hyperparameter tuning

Training a machine-learning model often involves hyperparameters, which determine how the model is trained.

Hyperparameters include regularization parameters, learning rates, architecture parameters, and training-sample weights. They are distinct from **model parameters**, which are learned from the training data.

A common strategy for tuning hyperparameters is to divide the available data into training and validation sets [Franceschi et al. 2018], and then:

- for fixed hyperparameters λ, use the training set to learn the model parameters w;
- use the validation set to evaluate the trained model and select λ.

This naturally gives a BO problem: the upper level selects λ, while the lower level trains w under the selected λ.

Application 2: hyperparameter tuning (cont'd)

Example (regularization parameter tuning): For a given $\lambda = (\lambda_1, \ldots, \lambda_r)$, the model parameters are obtained from

$$w^\star(\lambda) \in \operatorname*{argmin}_w \left\{ \ell_{\mathrm{tr}}(w) + \sum_{i=1}^{r} \lambda_i R_i(w) \right\},$$

where ℓ_{tr} is the training loss and R_i are regularization functions. The hyper-objective is the validation loss of the trained model:

$$\Phi(\lambda) := \ell_{\mathrm{val}}\big(w^\star(\lambda)\big).$$

Therefore, the resulting BO formulation is

$$\begin{aligned} \min_{\lambda \geq 0, w} \quad & \ell_{\mathrm{val}}(w) \\ \text{s.t.} \quad & w \in \operatorname*{argmin}_z \left\{ \ell_{\mathrm{tr}}(z) + \sum_{i=1}^{r} \lambda_i R_i(z) \right\}. \end{aligned}$$

Application 3: neural architecture search

Automate design of the neural network structure (e.g., number of layers, connection types) [Liu et al. 2018, Xue et al. 2021].

Upper-level problem (leader): Find the best architecture a from a search space $\mathcal{A}$ that minimizes validation loss:

$$\min_{a \in \mathcal{A}} \mathcal{L}_{\text{val}}(w^*(a)).$$

Lower-level problem (follower): For a *given* architecture a, train its weights w^* to convergence on the training data:

$$w^*(a) = \arg\min_{w} \mathcal{L}_{\text{train}}(w, a).$$

Application 4: power-grid vulnerability

Consider a power-grid vulnerability problem in which an attacker perturbs transmission lines and a defender responds to restore feasible operation [Kim et al. 2015].

- **Attacker (leader):** identify the most destructive transmission-line attack subject to a limited budget.
- **Defender (follower):** restore feasible grid operation while minimizing the amount of power-adjustment.

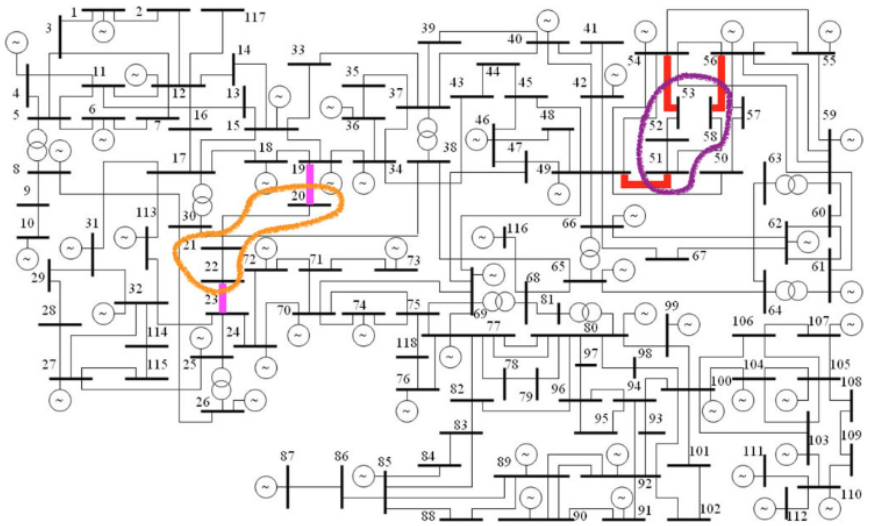


Fig. 5. Voltage disturbance model: attacks on the 118-Bus system. The transmission lines indicated by thicker dark lines are are the optimal attack for $\kappa = 3$. The thick lighter lines are added to the optimal three-line attack when κ is increased to 5. (Image source: [26])

Application 4: power-grid vulnerability (cont'd)

The **attacker** solves

$$\begin{array}{ll} \max\limits_{\gamma} & \mathcal{F}_L(\gamma) \\ \text{s.t.} & \mathbf{1}^\top \gamma \leq \kappa \bar{\gamma}, \quad 0 \leq \gamma \leq \bar{\gamma}\mathbf{1}, \end{array}$$

where γ represents the attack levels on the transmission lines, $\bar{\gamma}$ is the maximum attack level on each line, and κ controls the total attack budget.

For a given attack γ, the **defender** solves

$$\begin{array}{rll} \mathcal{F}_L(\gamma) := \min\limits_{x,y} & p^\top y \\ \text{s.t.} & F_L(x, y; \gamma) = 0, \quad \underline{x} \leq x \leq \bar{x}, \quad 0 \leq y \leq \bar{y}, \end{array}$$

where $F_L(x, y; \gamma) = 0$ represents the power flow feasibility equations, x represents the grid state variables with bounds $\underline{x}$ and $\bar{x}$, y represents the power-adjustment variables with upper bound $\bar{y}$, and p contains their weights.

Application 5: adversarial training

Adversarial training aims to train a model that is robust against worst-case perturbations. It is a minimax problem and can be viewed as a special bilevel problem [Madry et al. 2018].

Upper-level problem (leader-minimizer):
Find model parameters w that minimize the expected loss under the worst-case perturbation:

$$\min_{w} \mathbb{E}_{(u,v)} \left[\mathcal{L}\big(f_w(u + \delta^\star(w,u,v)), v\big) \right],$$

where f_w denotes the model parameterized by w, and (u, v) denotes a data sample with u being the input and v being its corresponding label.

Lower-level problem (follower-maximizer):
For the model w and a data sample (u, v), find a perturbation within the prescribed set Δ that maximizes the loss:

$$\delta^\star(w,u,v) \in \operatorname*{argmax}_{\delta \in \Delta} \mathcal{L}\big(f_w(u+\delta), v\big).$$

Other applications of BO

Other applications of BO include:

- DNN pruning [Sehwag et al. 2020, Zhang et al. 2022]
- Deep reinforcement learning [Gao et al. 2019, Vahdat et al. 2020]
- Wireless telecommunications [Sun et al. 2021, Gao et al. 2020]
- Data reweighting [Pan et al. 2024, Fan et al. 2024]
- LLM unlearning [Reisizadeh et al. 2025]

Several recent surveys provide broader overviews of BO and related topics [Sinha et al. 2017, Zhang et al. 2024].

Part II: Bilevel optimization with strongly convex lower-level objective function

BO with strongly convex lower-level objective function

Consider a stochastic BO without lower-level constraints:

$$\begin{aligned} \min_{x,y} \quad & f(x,y) := \mathbb{E}_\xi F(x,y;\xi) \\ \text{s.t.} \quad & y \in \operatorname*{argmin}_z \left\{ \tilde{f}(x,z) := \mathbb{E}_\zeta \tilde{F}(x,z;\zeta) \right\}, \end{aligned}$$

where f is smooth, $\tilde{f}$ is twice differentiable, and $\tilde{f}(x,\cdot)$ is L-smooth and μ-strongly convex for any x.

Strong convexity of $\tilde{f}(x,\cdot)$ implies that the lower-level minimizer $y^*(x) := \operatorname{argmin}_z \tilde{f}(x,z)$ is unique. Moreover, $y^*(x)$ satisfies

$$\nabla_y \tilde{f}(x, y^*(x)) = 0.$$

This, together with twice differentiability of $\tilde{f}$ and nonsingularity of $\nabla^2_{yy}\tilde{f}$, implies that $y^*(x)$ is continuously differentiable.

Hypergradient formula

Observe that the hyper-objective function $\Phi(x) = f(x, y^*(x))$ is differentiable and

$$\nabla\Phi(x) = \nabla_x f(x, y^*(x)) + \nabla y^*(x)\nabla_y f(x, y^*(x)).$$

In addition, $\nabla_y \tilde{f}(x, y^*(x)) = 0$. Differentiating it yields

$$\nabla^2_{xy} \tilde{f}(x, y^*(x)) + \nabla y^*(x)\nabla^2_{yy} \tilde{f}(x, y^*(x)) = 0,$$

and thus

$$\nabla y^*(x) = -\nabla^2_{xy} \tilde{f}(x, y^*(x)) \left[\nabla^2_{yy} \tilde{f}(x, y^*(x))\right]^{-1}.$$

Therefore,

$$\nabla\Phi(x) = \nabla_x f(x, y^*(x)) - \nabla^2_{xy} \tilde{f}(x, y^*(x)) \left[\nabla^2_{yy} \tilde{f}(x, y^*(x))\right]^{-1} \nabla_y f(x, y^*(x)).$$

Stationarity system

Note that $y = y^*(x)$ if and only if $\nabla_y \tilde{f}(x, y) = 0$. Therefore, $\nabla\Phi(x) = 0$ is equivalent to the coupled system

$$H(x, y) = 0, \qquad G(x, y) = 0,$$

where

$$H(x, y) := \nabla_x f(x, y) - \nabla^2_{xy} \tilde{f}(x, y) \left[\nabla^2_{yy} \tilde{f}(x, y)\right]^{-1} \nabla_y f(x, y),$$
$$G(x, y) := \nabla_y \tilde{f}(x, y).$$

This motivates a stochastic gradient descent scheme for finding an approximate solution of this system.

A Hessian-based stochastic gradient descent (SGD) method

A simple single-loop stochastic approximation scheme is

$$x_{k+1} = x_k - \alpha_k H^k,$$
$$y_{k+1} = y_k - \beta_k G^k,$$

where H^k and G^k are stochastic estimates of $H(x_k, y_k)$ and $G(x_k, y_k)$, respectively.

- Estimating G^k is straightforward since $G(x, y) = \mathbb{E}_\zeta \nabla_y \tilde{F}(x, y; \zeta)$.
- However, $H(x, y)$ involves the inverse Hessian $\left[\nabla^2_{yy} \tilde{f}(x, y)\right]^{-1}$. Since

$$\mathbb{E}_\zeta \left[\nabla^2_{yy} \tilde{F}(x, y; \zeta)\right]^{-1} \neq \left[\mathbb{E}_\zeta \nabla^2_{yy} \tilde{F}(x, y; \zeta)\right]^{-1},$$

it is more challenging to construct an estimator H^k.

A Hessian-based SGD method (cont'd)

Neumann series for matrix inverse estimation: Given a positive definite matrix A, we aim to estimate A^{-1}.

Note that $A = \tau[I - (I - \tau^{-1}A)]$ for any $\tau > 0$, and thus

$$A^{-1} = \tau^{-1}\left[I - \left(I - \tau^{-1}A\right)\right]^{-1}.$$

Suppose $\|A\| \leq L$ and let $\tau > L/2$. Then $\|I - \tau^{-1}A\| < 1$. Since $(I - C)^{-1} = \sum_{i=0}^{\infty} C^i$ for any $\|C\| < 1$, we have

$$A^{-1} = \tau^{-1}\sum_{i=0}^{\infty}\left(I - \tau^{-1}A\right)^i.$$

We can estimate A^{-1} by truncating the series at depth T:

$$\tau^{-1}\sum_{i=0}^{T-1}\left(I - \tau^{-1}A\right)^i.$$

The truncation error is $\mathcal{O}(\epsilon)$ when $T = \mathcal{O}(\log(1/\epsilon))$.

A Hessian-based SGD method (cont'd)

The truncated series can be estimated by sampling p uniformly from $\{0, 1, \ldots, T-1\}$ and using $T\tau^{-1}\left(I - \tau^{-1}A\right)^p$, thanks to the fact

$$\mathbb{E}_p\left[T\tau^{-1}(I - \tau^{-1}A)^p\right] = \tau^{-1}\sum_{i=0}^{T-1}\left(I - \tau^{-1}A\right)^i.$$

Suppose that $A(\zeta)$, depending on some random variable ζ, is such that $A = \mathbb{E}[A(\zeta)]$. Let $\zeta_1, \ldots, \zeta_p$ be independent samples. Then we have

$$\mathbb{E}\Big[\prod_{i=1}^{p}(I - \tau^{-1}A(\zeta_i))\Big] = \prod_{i=1}^{p}\mathbb{E}[(I - \tau^{-1}A(\zeta_i))] = (I - \tau^{-1}A)^p.$$

Let p be uniformly sampled from $\{0, 1, \ldots, T-1\}$. It follows that

$$\mathbb{E}_{p,\zeta_1,\ldots,\zeta_p}\Big[T\tau^{-1}\prod_{i=1}^{p}(I - \tau^{-1}A(\zeta_i))\Big] = \tau^{-1}\sum_{i=0}^{T-1}\left(I - \tau^{-1}A\right)^i.$$

Therefore, $T\tau^{-1}\prod_{i=1}^{p}(I - \tau^{-1}A(\zeta_i))$ provides an unbiased estimate of $\tau^{-1}\sum_{i=0}^{T-1}\left(I - \tau^{-1}A\right)^i$.

A Hessian-based SGD method (cont'd)

Subroutine: estimating $H(x,y)$ with depth t

Sample $p \in \{0, \dots, t-1\}$ uniformly at random, and construct

$$\widehat{H}_t = \nabla_x F(x,y;\xi) - \nabla^2_{xy}\tilde{F}(x,y;\zeta_0)\Big[\frac{tc}{L}\prod_{i=1}^{p}\big(I - \frac{c}{L}\nabla^2_{yy}\tilde{F}(x,y;\zeta_i)\big)\Big]\nabla_y F(x,y;\xi),$$

where $c := \mu/(\mu^2+\sigma^2)$, σ^2 is a variance bound for the stochastic Hessian estimator, and $\xi, \zeta_0, \zeta_1, \dots, \zeta_p$ are independent.

The estimator $\widehat{H}_t$ has exponentially decaying bias and bounded variance:

$$\|H(x,y) - \mathbb{E}[\widehat{H}_t]\| = \mathcal{O}\left((1-c\mu/L)^t\right), \qquad \mathbb{E}\|\widehat{H}_t - \mathbb{E}[\widehat{H}_t]\|^2 \le \widehat{\sigma}^2.$$

Here, $\widehat{\sigma}^2$ is a constant consisting of the variance bounds of stochastic gradients and Hessians.

A Hessian-based SGD method (cont'd)

The two-timescale stochastic approximation (TTSA) algorithm [Hong et al. 2023] uses the above hypergradient estimator with the truncation depth $t = \mathcal{O}(\log(1/\epsilon))$ at each iteration.

TTSA algorithm for BO

Perform the recursion

$$x_{k+1} = x_k - \alpha_k H^k,$$
$$y_{k+1} = y_k - \beta_k \nabla_y \tilde{F}(x_k, y_k; \zeta_{k+1}),$$

where H^k is computed by the previous subroutine.

- Although the algorithm is single-loop, each update of x requires constructing a hypergradient estimator.
- The hypergradient estimator requires stochastic Hessian evaluations of the lower-level objective function.
- It is called *two-timescale* stochastic approximation since α_k and β_k decay at different rates, typically with $\alpha_k/\beta_k \to 0$.

A Hessian-based SGD method (cont'd)

TTSA convergence measures:

Lower-level tracking error

For the lower-level problem, the key quantity is whether y^k tracks $y^*(x^{k-1})$:

$$\Delta_y^k := \mathbb{E}[\|y^k - y^*(x^{k-1})\|^2].$$

Upper-level tracking error

The upper-level error is measured by

$$\Delta_x^k := \begin{cases} \mathbb{E}[\|x^k - x^*\|^2] & \text{if } \Phi \text{ is strongly convex,} \\ \mathbb{E}[\Phi(x^k) - \Phi(x^*)] & \text{if } \Phi \text{ is convex,} \\ \mathbb{E}[\|\nabla \Phi(x^k)\|^2] & \text{if } \Phi \text{ is weakly convex.} \end{cases}$$

A Hessian-based SGD method (cont'd)

TTSA convergence rates:

Hyper-objective Φ	Step size (α_k, β_k)	Outer rate	Inner rate
Strongly convex	$\mathcal{O}(K^{-1})$, $\mathcal{O}(K^{-2/3})$	$\mathcal{O}(K^{-2/3})$	$\mathcal{O}(K^{-2/3})$
Convex	$\mathcal{O}(K^{-3/4})$, $\mathcal{O}(K^{-1/2})$	$\mathcal{O}(K^{-1/4})$	$\mathcal{O}(K^{-1/2})$
Weakly convex	$\mathcal{O}(K^{-3/5})$, $\mathcal{O}(K^{-2/5})$	$\mathcal{O}(K^{-2/5})$	$\mathcal{O}(K^{-2/5})$

- The lower-level objective $\tilde{f}(x, \cdot)$ is strongly convex and K is the total number of iterations.
- The outer rate measures upper-level optimality or stationarity, while the inner rate measures the lower-level tracking error.

A Hessian-based SGD method (cont'd)

TTSA sample complexity:

Table: Total sample complexity of BSA and TTSA

Method	Strongly convex	Convex	Weakly convex
BSA	$\mathcal{O}(\epsilon^{-2})$	$\mathcal{O}(\epsilon^{-4})$	$\mathcal{O}(\epsilon^{-3})$
TTSA	$\widetilde{\mathcal{O}}(\epsilon^{-3/2})$	$\widetilde{\mathcal{O}}(\epsilon^{-4})$	$\widetilde{\mathcal{O}}(\epsilon^{-5/2})$

- The Bilevel Stochastic Approximation (BSA) method [Ghadimi & Wang 2018] is a double-loop method. For each outer iterate x^k, it performs sufficiently many lower-level iterations to obtain $y^k \approx y^*(x^k)$ and estimate the hypergradient. In contrast, TTSA is a single-loop method, which continuously tracks $y^*(x^k)$ using only one lower-level stochastic-gradient update per iteration.
- Compared with BSA, TTSA improves the sample complexity by a factor of $\widetilde{\mathcal{O}}(\epsilon^{-1/2})$ in the strongly convex and weakly convex settings.

A Hessian-based SGD method (cont'd)

Subroutine of TTSA: It approximates the Hessian inverse using a truncated Neumann series and then estimates the resulting products using independent samples.

Alternative subroutine: It directly computes the required inverse-Hessian–vector product by solving a quadratic program problem.

For a given (x, y), consider

$$v^*(x,y) := \operatorname*{argmin}_v \left\{ Q(v,x,y) := \frac{1}{2} v^\top \nabla^2_{yy} \tilde{f}(x,y) v + \nabla_y f(x,y)^\top v \right\}.$$

Observe that $v^*(x,y) = -[\nabla^2_{yy} \tilde{f}(x,y)]^{-1} \nabla_y f(x,y)$, and hence the hypergradient mapping can be written as

$$H(x,y) = \nabla_x f(x,y) + \nabla^2_{xy} \tilde{f}(x,y) v^*(x,y).$$

The above strongly convex quadratic program can be suitably solved by the *conjugate gradient method* (deterministic case) or a *stochastic gradient method* (stochastic case).

A Hessian-based SGD method (cont'd)

Conjugate gradient method for convex quadratic program:

When exact Hessian-vector products are available, the conjugate gradient method successively minimizes Q along directions $d^0, d^1, \ldots$ that are conjugate with respect to $A := \nabla^2_{yy} \tilde{f}(x, y)$:

$$(d^i)^\top A d^j = 0, \qquad i \neq j.$$

- Each iteration only requires a Hessian-vector product Au, without forming the Hessian inverse.
- Compared with ordinary gradient descent, the iteration complexity improves from $\mathcal{O}(\kappa \log(1/\epsilon))$ to $\mathcal{O}(\sqrt{\kappa} \log(1/\epsilon))$, where κ is the condition number of A.

A Hessian-based SGD method (cont'd)

Stochastic gradient method for the above quadratic program:

When only stochastic gradient and Hessian estimators are available, consider the stochastic gradient estimator

$$G(x, y, v; \xi, \zeta) := \nabla^2_{yy} \tilde{F}(x, y; \zeta) v + \nabla_y F(x, y; \xi).$$

It satisfies $\mathbb{E}[G(x, y, v; \xi, \zeta)] = \nabla_v Q(x, y, v)$.

Starting from v^0, perform

$$v^{j+1} = v^j - \eta_j G(x, y, v^j; \xi_j, \zeta_j), \quad j = 0, 1, \ldots.$$

Each iteration also only requires a stochastic Hessian-vector product and a stochastic gradient.

A Hessian-free SGD method

TTSA requires computing the Hessian of $\tilde{f}$. We next introduce a first-order method based on a value-function reformulation [Kwon et al. 2023].

Denote $\tilde{f}^*(x) := \min_y \tilde{f}(x,y)$. Then the BO problem can be equivalently written as

$$\min_{x,y} f(x,y) \qquad \text{s.t.} \qquad \tilde{f}(x,y) - \tilde{f}^*(x) \le 0.$$

A penalty approach considers

$$\mathcal{L}_\lambda(x,y) := f(x,y) + \lambda\big(\tilde{f}(x,y) - \tilde{f}^*(x)\big), \qquad \mathcal{L}_\lambda^*(x) := \min_y \mathcal{L}_\lambda(x,y).$$

If λ is sufficiently large, then $\mathcal{L}_\lambda^*$ serves as an approximation of Φ.

A Hessian-free SGD method (cont'd)

The penalized function $\mathcal{L}_\lambda^*$ is differentiable and

$$\nabla \mathcal{L}_\lambda^*(x) = \nabla_x f(x, y_\lambda^*(x)) + \lambda\big(\nabla_x \tilde{f}(x, y_\lambda^*(x)) - \nabla_x \tilde{f}(x, y^*(x))\big),$$

where $y_\lambda^*(x) := \operatorname{argmin}_y \mathcal{L}_\lambda(x, y)$ and $y^*(x) := \operatorname{argmin}_y \tilde{f}(x, y)$.

For sufficiently large $\lambda > 0$,

$$\|\nabla \Phi(x) - \nabla \mathcal{L}_\lambda^*(x)\| = \mathcal{O}(1/\lambda).$$

Therefore, to satisfy $\|\nabla \Phi(x)\| \le \epsilon$, it suffices to find x such that $\|\nabla \mathcal{L}_\lambda^*(x)\| \le \mathcal{O}(\epsilon)$ with $\lambda = \mathcal{O}(1/\epsilon)$.

A Hessian-free SGD method (cont'd)

Key idea: Perform a single SGD step on $\min_x \mathcal{L}^*_{\lambda_k}(x)$ for some $\{\lambda_k\}$.

Compute two auxiliary vectors

$$y_k \approx y^*_{\lambda_k}(x_k), \qquad z_k \approx y^*(x_k).$$

Then we set

$$d_k := \nabla_x f(x_k, y_k) + \lambda_k \big(\nabla_x \tilde{f}(x_k, y_k) - \nabla_x \tilde{f}(x_k, z_k)\big)$$

as an approximation of $\nabla \mathcal{L}^*_{\lambda_k}(x_k)$.

Finally, $\{x_k\}$ is updated by

$$x_{k+1} = x_k - \xi \alpha_k d_k,$$

while y_k and z_k are updated by first-order steps toward $y^*_{\lambda_k}(x_k)$ and $y^*(x_k)$, respectively.

A Hessian-free SGD method (cont'd)

Fully first-order stochastic approximation (F^2SA) algorithm for BO

Input: step sizes $\{\alpha_k, \gamma_k\}$, multiplier increment $\{\delta_k\}$, inner-loop number T, step-size ratio ξ, and initialization λ_0, x_0, y_0, z_0.

for $k = 0, \ldots, K-1$ **do**
  $z_{k,0} \leftarrow z_k$, $y_{k,0} \leftarrow y_k$
  for $t = 0, \ldots, T-1$ **do**
    $z_{k,t+1} \leftarrow z_{k,t} - \gamma_k h^{k,t}_{\tilde{f},z}$
    $y_{k,t+1} \leftarrow y_{k,t} - \alpha_k (h^{k,t}_{f,y} + \lambda_k h^{k,t}_{\tilde{f},y})$
  end for
  $z_{k+1} \leftarrow z_{k,T}$, $y_{k+1} \leftarrow y_{k,T}$
  $x_{k+1} \leftarrow x_k - \xi\alpha_k \big(h^k_{f,x} + \lambda_k (h^{k,y}_{\tilde{f},x} - h^{k,z}_{\tilde{f},x})\big)$
  $\lambda_{k+1} \leftarrow \lambda_k + \delta_k$
end for

Here the h's are unbiased stochastic gradient estimates of the corresponding gradients of f and $\tilde{f}$.

A Hessian-free SGD method (cont'd)

F^{2}SA convergence analysis:

The analysis is based on showing decrease of the potential function

$$\mathbb{V}_k := \Phi(x_k) - \Phi^* + L_{\tilde{f}}\lambda_k \|y_k - y^*_{\lambda_k}(x_k)\|^2 + \frac{L_{\tilde{f}}\lambda_k}{2}\|z_k - y^*(x_k)\|^2.$$

The form of $\mathbb{V}_k$ ensures to control both tracking errors:

$$\|y_k - y^*_{\lambda_k}(x_k)\|, \qquad \|z_k - y^*(x_k)\|.$$

The parameters are chosen as $\xi = 1$, $T = \mathcal{O}(\kappa)$, and

$$\alpha_k = \mathcal{O}((k+k_0)^{-a}), \quad \gamma_k = \mathcal{O}((k+k_0)^{-c}), \quad \lambda_k = \mathcal{O}((k+k_0)^{a-c}),$$

where $0 \le c \le a \le 1$ and κ is a bound on the condition number of $\nabla^2_{yy}\tilde{f}(x,\cdot)$.

A Hessian-free SGD method (cont'd)

F^{2}SA complexity:

The convergence rates of F^{2}SA for finding x_R with $\mathbb{E}\|\nabla\Phi(x_R)\| \leq \epsilon$ under different settings are given below.

Setting	Parameters (a, c)	Rate	Iterations
$f, \tilde{f}$: stochastic	$(5/7, 4/7)$	$\widetilde{\mathcal{O}}(K^{-1/7})$	$\widetilde{\mathcal{O}}(\epsilon^{-7})$
f: stochastic, $\tilde{f}$: deterministic	$(3/5, 2/5)$	$\widetilde{\mathcal{O}}(K^{-1/5})$	$\widetilde{\mathcal{O}}(\epsilon^{-5})$
$f, \tilde{f}$: deterministic	$(1/3, 0)$	$\widetilde{\mathcal{O}}(K^{-1/3})$	$\widetilde{\mathcal{O}}(\epsilon^{-3})$

A Hessian-free SGD method (cont'd)

Tighter analysis for F^{2}SA:

A key observation in [Chen et al. 2025] is that, for sufficiently large λ,

$$\|\nabla^2 \mathcal{L}_\lambda^*(x)\| = \mathcal{O}(\kappa^3),$$

with no dependence on λ. This follows from the sensitivity estimates

$$\|y^*(x) - y_\lambda^*(x)\| = \mathcal{O}(1/\lambda), \qquad \|\nabla y^*(x) - \nabla y_\lambda^*(x)\| = \mathcal{O}(1/\lambda).$$

Consequently, the x-stepsize can be made significantly larger and the refined algorithm converges faster.

For noisy gradients in both levels, noisy gradients only in f, and deterministic gradients, respectively, the complexities are improved by a factor of ϵ^{-1}:

$$\widetilde{\mathcal{O}}(\epsilon^{-7}),\ \widetilde{\mathcal{O}}(\epsilon^{-5}),\ \widetilde{\mathcal{O}}(\epsilon^{-3}) \quad \Longrightarrow \quad \widetilde{\mathcal{O}}(\epsilon^{-6}),\ \widetilde{\mathcal{O}}(\epsilon^{-4}),\ \widetilde{\mathcal{O}}(\epsilon^{-2}).$$

A Hessian-free SGD method (cont'd)

Complexity lower bound for stochastic f and $\tilde{f}$:

At the exact lower-level solutions, $\nabla \mathcal{L}_\lambda^*$ admits the unbiased estimator

$$\widehat{\nabla} \mathcal{L}_\lambda^*(x) = \nabla_x F(x, y_\lambda^*(x); \xi) + \lambda \big(\nabla_x \tilde{F}(x, y_\lambda^*(x); \zeta_1) - \nabla_x \tilde{F}(x, y^*(x); \zeta_2) \big).$$

Due to $\lambda = \mathcal{O}(1/\epsilon)$ and bounded variance assumption, one has

$$\mathbb{E}\big[\| \widehat{\nabla} \mathcal{L}_\lambda^*(x) - \nabla \mathcal{L}_\lambda^*(x) \|^2 \big] = \mathcal{O}(\lambda^2) = \mathcal{O}(\epsilon^{-2}).$$

Since the oracle complexity lower bound for smooth nonconvex stochastic optimization with gradient variance σ^2 is $\Omega(\sigma^2 \epsilon^{-4})$, the variance amplification suggests that a lower bound of $\Omega(\epsilon^{-6})$ may be unavoidable.

Indeed, [Kwon et al. 2024] constructs a hard instance establishing an $\Omega(\epsilon^{-6})$ lower bound, showing that the $\mathcal{O}(\epsilon^{-6})$ complexity is *optimal*.

A Hessian-free SGD method (cont'd)

Improved complexity under stochastic smoothness condition:

Suppose the following stochastic smoothness condition holds:

$$\mathbb{E}_\zeta \|\nabla_x \tilde{F}(x, y_1; \zeta) - \nabla_x \tilde{F}(x, y_2; \zeta)\|^2 \leq L^2 \|y_1 - y_2\|^2.$$

- Control y_k and z_k such that $\|y_k - y^*_{\lambda_k}(x_k)\|$, $\|z_k - y^*(x_k)\|$, and $\|y^*_{\lambda_k}(x_k) - y^*(x_k)\|$ are all of order $\mathcal{O}(1/\lambda_k)$. Then $\|y_k - z_k\| = \mathcal{O}(1/\lambda_k)$.
- Using the same sample ζ_k for the two lower-level gradients, consider

$$\widehat{\nabla}\mathcal{L}^*_{\lambda_k}(x_k) = \nabla_x F(x_k, y_k; \xi_k) + \lambda_k \big(\nabla_x \tilde{F}(x_k, y_k; \zeta_k) - \nabla_x \tilde{F}(x_k, z_k; \zeta_k)\big).$$

Notice that

$$\mathrm{Var}[\widehat{\nabla}\mathcal{L}^*_{\lambda_k}(x_k)] = \mathrm{Var}[\nabla_x F(x_k, y_k; \xi_k)] + \mathrm{Var}[\lambda_k \big(\nabla_x \tilde{F}(x_k, y_k; \zeta_k) - \nabla_x \tilde{F}(x_k, z_k; \zeta_k)\big)].$$

This and the stochastic smoothness condition imply

$$\mathbb{E}\|\widehat{\nabla}\mathcal{L}^*_{\lambda_k}(x_k) - \mathbb{E}[\widehat{\nabla}\mathcal{L}^*_{\lambda_k}(x_k)]\|^2 \leq \sigma_f^2 + \lambda_k^2 L^2 \|y_k - z_k\|^2 = \mathcal{O}(1).$$

Consequently, the complexity *improves* to $\mathcal{O}(\epsilon^{-4})$ [Kwon et al. 2024].

Faster fully first-order stochastic approximation (F^{3}SA)

The momentum-assisted gradient estimators [Khanduri et al. 2021] can further improve the convergence rates under a stronger assumption (e.g., F and $\tilde{F}$ satisfy smoothness and bounded gradient conditions almost surely on the sample space).

F^{3}SA algorithm for BO

Input: step sizes $\{\alpha_k, \gamma_k\}$, multiplier increment $\{\delta_k\}$, momentum-weight sequence $\{\eta_k\}$, step-size ratio ξ, and initialization λ_0, x_0, y_0, z_0.

1: **for** $k = 0, \ldots, K-1$ **do**
2: $\quad z_{k+1} \leftarrow z_k - \gamma_k \tilde{h}^k_{\tilde{f},z}$
3: $\quad y_{k+1} \leftarrow y_k - \alpha_k(\tilde{h}^k_{f,y} + \lambda_k \tilde{h}^k_{\tilde{f},y})$
4: $\quad x_{k+1} \leftarrow x_k - \xi\alpha_k\big(\tilde{h}^k_{f,x} + \lambda_k(\tilde{h}^{k,y}_{\tilde{f},x} - \tilde{h}^{k,z}_{\tilde{f},x})\big)$
5: $\quad \lambda_{k+1} \leftarrow \lambda_k + \delta_k$
6: **end for**

The momentum-assisted estimator for updating z is

$$\tilde{h}^k_{\tilde{f},z} := \nabla_y \tilde{F}(x_k, z_k; \zeta^k_z) + (1-\eta_k)\big(\tilde{h}^{k-1}_{\tilde{f},z} - \nabla_y \tilde{F}(x_{k-1}, z_{k-1}; \zeta^k_z)\big).$$

The estimators $\tilde{h}^k_{f,y}$, $\tilde{h}^k_{\tilde{f},y}$, $\tilde{h}^k_{f,x}$, $\tilde{h}^{k,y}_{\tilde{f},x}$, and $\tilde{h}^{k,z}_{\tilde{f},x}$ are defined similarly using the same momentum-weight sequence $\{\eta_k\}$.

F^{3}SA parameter choice and complexity

Choose

$$\alpha_k = \mathcal{O}((k+k_0)^{-a}), \qquad \gamma_k = \mathcal{O}((k+k_0)^{-c}), \qquad \eta_k = (k+1)^{-2c},$$

where $0 \le c \le a \le 1$, k_0 is an instance-dependent constant, and $\xi > 0$ is sufficiently small. Moreover,

$$\delta_k = \frac{\gamma_k}{\alpha_k} - \lambda_k, \qquad \lambda_{k+1} = \frac{\gamma_k}{\alpha_k} = \mathcal{O}((k+k_0)^{a-c}).$$

The convergence rates of F^{3}SA for finding x_R with $\mathbb{E}\|\nabla\Phi(x_R)\| \le \epsilon$ under different settings are given below.

Setting	(a, c)	Rate	Iterations
Both f and $\tilde{f}$ stochastic	$(3/5, 2/5)$	$\widetilde{\mathcal{O}}(K^{-1/5})$	$\widetilde{\mathcal{O}}(\epsilon^{-5})$
Only f stochastic	$(1/2, 1/4)$	$\widetilde{\mathcal{O}}(K^{-1/4})$	$\widetilde{\mathcal{O}}(\epsilon^{-4})$
Deterministic	$(1/3, 0)$	$\widetilde{\mathcal{O}}(K^{-1/3})$	$\widetilde{\mathcal{O}}(\epsilon^{-3})$

Part III: Bilevel optimization with convex lower-level objective function

Unconstrained bilevel optimization

Consider **unconstrained** bilevel optimization (UBO)

$$\begin{aligned} \min \quad & f(x,y) \\ \text{s.t.} \quad & y \in \operatorname*{argmin}_z \tilde{f}(x,z). \end{aligned}$$

Assumption:

- $f(x,y) = f_1(x,y) + f_2(x)$ and $\tilde{f}(x,y) = \tilde{f}_1(x,y) + \tilde{f}_2(y)$.
- f_1, $\tilde{f}_1$ are Lipschitz smooth on $\mathcal{X} \times \mathcal{Y}$ with $\mathcal{X} := \operatorname{dom} f_2$ and $\mathcal{Y} := \operatorname{dom} \tilde{f}_2$, and $\tilde{f}_1(x,\cdot)$ is **convex** for any $x \in \mathcal{X}$.
- $f_2, \tilde{f}_2$ are proper closed prox-friendly convex functions.

Challenge: Under mere convexity, the lower-level solution may not be unique, and its solution set may change discontinuously with x. This may lead to an ill-behaved hyper-objective function Φ.

An example on ill-behaved hyper-objective function

Consider

$$\min_{x\in[-1,1]} \{\Phi(x) := \min_{y\in\mathcal{Y}^*(x)} x^2 + y^2\}, \quad \mathcal{Y}^*(x) := \operatorname*{argmin}_{y\in[-1,1]} xy.$$

Then

$$\mathcal{Y}^*(x) = \begin{cases} \{-1\}, & x > 0, \\ [-1,1], & x = 0, \\ \{1\}, & x < 0, \end{cases} \qquad \Phi(x) = \begin{cases} x^2 + 1, & x \neq 0, \\ 0, & x = 0. \end{cases}$$

Observe that $\mathcal{Y}^*(\cdot)$ changes discontinuously at $x = 0$, and Φ is discontinuous at $x = 0$.

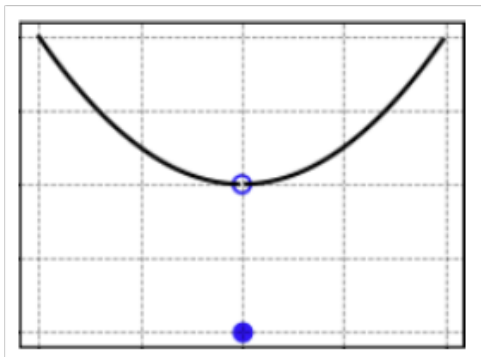


Figure: The hyper-objective function $\Phi(\cdot)$ (left) and the lower-level solution set $\mathcal{Y}^*(\cdot)$ (right)

Related work

Numerous studies on *special* classes of UBO:

- constraint-based methods (Hansen et al. 1992)
- gradient-type methods (Liu et al. 2021; Ji & Liang 2022; Liu et al. 2022; Hu et al. 2023; Kwon et al. 2023)
- interior-point method (Liu et al. 2021), pseudo-second-order method (Sow et al. 2022)
- primal-dual gradient method (Sow et al. 2022)
- perturbed gradient-based method (Huang et al. 2023)
- zeroth-order methods (Chen et al. 2023)
- smoothing method (Alcantara & Takeda 2026)
- many more ...

Our approach: solve UBO via ***minimax optimization*** (Lu & Mei 2024).

Strongly-convex-strongly-concave minimax optimization

Consider a **strongly-convex-strongly-concave** minimax (SCSCM) problem

$$\min_x \max_y \left\{ \bar{H}(x,y) := \bar{h}(x,y) + p(x) - q(y) \right\},$$

where

- $\bar{h}(x,y)$ is σ_x-strongly-convex-σ_y-strongly-concave and $L_{\nabla \bar{h}}$-smooth on dom $p \times$ dom q for some $\sigma_x, \sigma_y > 0$,
- p and q are proper closed prox-friendly convex functions.

Goal: find an $\bar{\epsilon}$-stationary point (x,y) satisfying $\text{dist}(0, \partial_x \bar{H}(x,y)) \leq \bar{\epsilon}$ and $\text{dist}(0, \partial_y \bar{H}(x,y)) \leq \bar{\epsilon}$ for any $\bar{\epsilon} > 0$.

A modified optimal first-order method for SCSCM

Algorithm 1: A modified optimal first-order method for SCSCM [Lu & Mei 2025]

Input: $\bar{\epsilon} > 0$, $\bar{z}^0 = z_f^0 \in -\sigma_x \operatorname{dom} p$, $\bar{y}^0 = y_f^0 \in \operatorname{dom} q$, $(z^0, y^0) = (\bar{z}^0, \bar{y}^0)$,
$\bar{\alpha} = \min\left\{1, \sqrt{8\sigma_y/\sigma_x}\right\}$, $\eta_z = \sigma_x/2$, $\eta_y = \min\{1/(2\sigma_y), 4/(\bar{\alpha}\sigma_x)\}$,
$\beta_t = 2/(t+3)$, $\zeta = \left(2\sqrt{5}(1 + 8L_{\nabla \bar{h}}/\sigma_x)\right)^{-1}$, $\gamma_x = \gamma_y = 8\sigma_x^{-1}$, and
$\bar{\zeta} = \min\{\sigma_x, \sigma_y\}/L_{\nabla \bar{h}}^2$. Set $k = 0$.

1. Set $t = 0$ and compute

$$
\begin{aligned}
(z_g^k, y_g^k) &= \bar{\alpha}(z^k, y^k) + (1 - \bar{\alpha})(z_f^k, y_f^k), \\
(x^{k,-1}, y^{k,-1}) &= (-\sigma_x^{-1} z_g^k, y_g^k), \\
x^{k,0} &= \operatorname{prox}_{\zeta\gamma_x p}(x^{k,-1} - \zeta\gamma_x a_x^k(x^{k,-1}, y^{k,-1})), \\
y^{k,0} &= \operatorname{prox}_{\zeta\gamma_y q}(y^{k,-1} - \zeta\gamma_y a_y^k(x^{k,-1}, y^{k,-1})), \\
b_x^{k,0} &= (x^{k,-1} - \zeta\gamma_x a_x^k(x^{k,-1}, y^{k,-1}) - x^{k,0})/(\zeta\gamma_x), \\
b_y^{k,0} &= (y^{k,-1} - \zeta\gamma_y a_y^k(x^{k,-1}, y^{k,-1}) - y^{k,0})/(\zeta\gamma_y).
\end{aligned}
$$

2. If $\gamma_x \|a_x^k(x^{k,t}, y^{k,t}) + b_x^{k,t}\|^2 + \gamma_y \|a_y^k(x^{k,t}, y^{k,t}) + b_y^{k,t}\|^2 \leq \gamma_x^{-1}\|x^{k,t} - x^{k,-1}\|^2 + \gamma_y^{-1}\|y^{k,t} - y^{k,-1}\|^2$, go to step 5.

A modified optimal first-order method for SCSCM (cont'd)

3. Compute

$$
\begin{aligned}
x^{k,t+1/2} &= x^{k,t} + \beta_t(x^{k,0} - x^{k,t}) - \zeta\gamma_x(a_x^k(x^{k,t}, y^{k,t}) + b_x^{k,t}),\\
y^{k,t+1/2} &= y^{k,t} + \beta_t(y^{k,0} - y^{k,t}) - \zeta\gamma_y(a_y^k(x^{k,t}, y^{k,t}) + b_y^{k,t}),\\
x^{k,t+1} &= \mathrm{prox}_{\zeta\gamma_x p}(x^{k,t} + \beta_t(x^{k,0} - x^{k,t}) - \zeta\gamma_x a_x^k(x^{k,t+1/2}, y^{k,t+1/2})),\\
y^{k,t+1} &= \mathrm{prox}_{\zeta\gamma_y q}(y^{k,t} + \beta_t(y^{k,0} - y^{k,t}) - \zeta\gamma_y a_y^k(x^{k,t+1/2}, y^{k,t+1/2})),\\
b_x^{k,t+1} &= (x^{k,t} + \beta_t(x^{k,0} - x^{k,t}) - \zeta\gamma_x a_x^k(x^{k,t+1/2}, y^{k,t+1/2}) - x^{k,t+1})/(\zeta\gamma_x),\\
b_y^{k,t+1} &= (y^{k,t} + \beta_t(y^{k,0} - y^{k,t}) - \zeta\gamma_y a_y^k(x^{k,t+1/2}, y^{k,t+1/2}) - y^{k,t+1})/(\zeta\gamma_y).
\end{aligned}
$$

4. Set $t \leftarrow t+1$, and go to step 2.

5. Compute

$$
\begin{aligned}
&(x_f^{k+1}, y_f^{k+1}) = (x^{k,t}, y^{k,t}),\\
&(z_f^{k+1}, w_f^{k+1}) = (\nabla_x \hat{h}(x_f^{k+1}, y_f^{k+1}) + b_x^{k,t}, -\nabla_y \hat{h}(x_f^{k+1}, y_f^{k+1}) + b_y^{k,t}),\\
&z^{k+1} = z^k + \eta_z\sigma_x^{-1}(z_f^{k+1} - z^k) - \eta_z(x_f^{k+1} + \sigma_x^{-1} z_f^{k+1}),\\
&y^{k+1} = y^k + \eta_y\sigma_y(y_f^{k+1} - y^k) - \eta_y(w_f^{k+1} + \sigma_y y_f^{k+1}), \quad x^{k+1} = -\sigma_x^{-1} z^{k+1}.
\end{aligned}
$$

A modified optimal first-order method for SCSCM (cont'd)

6. Compute

$$\tilde{x}^{k+1} = \mathrm{prox}_{\bar{\zeta} p}(x^{k+1} - \bar{\zeta}\nabla_x \bar{h}(x^{k+1}, y^{k+1})),$$
$$\tilde{y}^{k+1} = \mathrm{prox}_{\bar{\zeta} q}(y^{k+1} + \bar{\zeta}\nabla_y \bar{h}(x^{k+1}, y^{k+1})).$$

7. Terminate the algorithm and output $(\tilde{x}^{k+1}, \tilde{y}^{k+1})$ if

$$\|\bar{\zeta}^{-1}(x^{k+1} - \tilde{x}^{k+1}, \tilde{y}^{k+1} - y^{k+1}) - (\nabla \bar{h}(x^{k+1}, y^{k+1}) - \nabla \bar{h}(\tilde{x}^{k+1}, \tilde{y}^{k+1}))\| \le \bar{\epsilon}.$$

8. Set $k \leftarrow k + 1$, and go to step 1.

Remark. It is a slight modification of an optimal first-order method for SCSCM (Kovalev & Gasnikov 2022) by incorporating a **verifiable termination criterion**.

Theorem (Complexity of Algorithm 1)

Algorithm 1 outputs an $\bar{\epsilon}$-stationary point (x, y) satisfying $\mathrm{dist}(0, \partial_x \bar{H}(x, y)) \le \bar{\epsilon}$ *and* $\mathrm{dist}(0, \partial_y \bar{H}(x, y)) \le \bar{\epsilon}$ *within* $\mathcal{O}((L_{\nabla \bar{h}}/\sqrt{\sigma_x \sigma_y}) \log \bar{\epsilon}^{-1})$ *evaluations of* $\nabla \bar{h}$ *and the proximal operators of* p *and* q.

Nonconvex-concave minimax optimization

Consider a **nonconvex-concave** minimax (NCCM) problem

$$\min_x \max_y \{H(x,y) := h(x,y) + p(x) - q(y)\},$$

where

- h is $L_{\nabla h}$-smooth on dom $p \times$ dom q and $h(x,\cdot)$ is concave for any $x \in$ dom p,
- p and q are proper closed prox-friendly convex functions, and dom q is compact.

Goal: find an ϵ-stationary point (x,y) satisfying $\text{dist}(0, \partial_x H(x,y)) \leq \epsilon$ and $\text{dist}(0, \partial_y H(x,y)) \leq \epsilon$ for any $\epsilon > 0$.

A first-order method for NCCM

Our approach: apply Algorithm 1 to solve $\min_x \max_y \{h_k(x,y) + p(x) - q(y)\}$, where

$$h_k(x,y) = h(x,y) + L_{\nabla h}\|x - x^k\|^2 - \epsilon\|y - \hat{y}^0\|^2/(4D_{\mathbf{y}}).$$

Algorithm 2: A first-order method for NCCM [Lu & Mei 2025]

Input: $\epsilon > 0$, $\hat{\epsilon}_0 \in (0, \epsilon/2]$, $(\hat{x}^0, \hat{y}^0) \in \operatorname{dom} p \times \operatorname{dom} q$, $(x^0, y^0) = (\hat{x}^0, \hat{y}^0)$, and $\hat{\epsilon}_k = \hat{\epsilon}_0/(k+1)$. Set $k = 0$.

1. Call Algorithm 1 with $\bar{h} \leftarrow h_k$, $\bar{\epsilon} \leftarrow \hat{\epsilon}_k$, $\sigma_x \leftarrow L_{\nabla h}$, $\sigma_y \leftarrow \epsilon/(2D_{\mathbf{y}})$, $L_{\nabla \bar{h}} \leftarrow 3L_{\nabla h} + \epsilon/(2D_{\mathbf{y}})$, $\bar{z}^0 = z_f^0 \leftarrow -\sigma_x x^k$, $\bar{y}^0 = y_f^0 \leftarrow y^k$, and denote its output by (x^{k+1}, y^{k+1}).
2. Terminate the algorithm and output (x^{k+1}, y^{k+1}) if
$$\|x^{k+1} - x^k\| \le \epsilon/(4L_{\nabla h}).$$
3. Set $k \leftarrow k+1$, and go to step 1.

Theorem (Complexity of Algorithm 2)

Algorithm 2 outputs an ϵ-stationary point (x,y) satisfying $\operatorname{dist}(0, \partial_x H(x,y)) \le \epsilon$ *and* $\operatorname{dist}(0, \partial_y H(x,y)) \le \epsilon$ *within* $\mathcal{O}(\epsilon^{-5/2}\log\epsilon^{-1})$ *evaluations of ∇h and the proximal operators of p and q.*

Remark: For nonconvex-**strongly**-concave minimax problem, Algorithm 2 can be modified to achieve an improved complexity of $\mathcal{O}(\epsilon^{-2}\log\epsilon^{-1})$.

Minimax approximation of UBO

Observation: $y \in \operatorname{argmin}_z \tilde{f}(x,z) \;\Leftrightarrow\; \tilde{f}(x,y) \leq \min_z \tilde{f}(x,z)$.

Hence, UBO can be written as

$$\min_{x,y}\{f(x,y) | \tilde{f}(x,y) \leq \min_z \tilde{f}(x,z)\}.$$

It can be approximately solved as

$$\min_{x,y} f(x,y) + \rho(\tilde{f}(x,y) - \min_z \tilde{f}(x,z)),$$

which is equivalent to the *minimax problem*

$$\min_{x,y} \max_z P_\rho(x,y,z), \quad \text{where} \quad P_\rho(x,y,z) := f(x,y) + \rho(\tilde{f}(x,y) - \tilde{f}(x,z)).$$

A conceptual penalty method for UBO

Algorithm 3: A conceptual penalty method for UBO

Input: positive sequences $\{\rho_k\}$ and $\{\epsilon_k\}$ with $\lim_{k\to\infty}(\rho_k, \epsilon_k) = (\infty, 0)$. Set $k = 0$.

1. Find an ϵ_k-**optimal** solution (x^k, y^k, z^k) of problem $\min_{x,y} \max_z P_{\rho_k}(x, y, z)$.[1]
2. Set $k \leftarrow k + 1$, and go to step 1.

Theorem (Convergence of Algorithm 3)

Suppose that $\{(x^k, y^k, z^k)\}$ is generated by Algorithm 3. Then any accumulation point of $\{(x^k, y^k)\}$ is an optimal solution of UBO.

Remark: Algorithm 3 is **not implementable** in general.

[1] (x_ϵ, y_ϵ) is called an ϵ-**optimal** solution of the problem $\Psi^* = \min_x \max_y \Psi(x, y)$ if $\max_y \Psi(x_\epsilon, y) - \Psi(x_\epsilon, y_\epsilon) \le \epsilon$ and $\Psi(x_\epsilon, y_\epsilon) - \Psi^* \le \epsilon$.

ε-KKT solution of UBO

Recall that UBO can be written as

$$\min_{x,y}\{f(x,y)|\tilde{f}(x,y) \leq \min_{z} \tilde{f}(x,z)\}.$$

- A KKT solution (x,y) of it satisfies $\tilde{f}(x,y) \leq \min_z \tilde{f}(x,z)$ and moreover (x,y) is a stationary point of

$$\min_{x',y'} f(x',y') + \rho\big(\tilde{f}(x',y') - \min_{z'} \tilde{f}(x',z')\big) \quad \text{for some } \rho \geq 0.$$

- The latter problem is equivalent to

$$\min_{x',y'} \max_{z'} f(x',y') + \rho(\tilde{f}(x',y') - \tilde{f}(x',z')),$$

whose stationary point satisfies

$$0 \in \partial f(x,y) + \rho\partial\tilde{f}(x,y) - (\rho\nabla_x\tilde{f}(x,z);0), \quad 0 \in \rho\partial_z\tilde{f}(x,z).$$

Definition (ε-KKT solution of UBO)

For any $\varepsilon > 0$, (x,y) is said to be an ε-KKT solution of UBO if there exists $(z,\rho) \in \mathbb{R}^m \times \mathbb{R}_+$ such that

$$\text{dist}\Big(0, \partial f(x,y) + \rho\partial\tilde{f}(x,y) - (\rho\nabla_x\tilde{f}(x,z);0)\Big) \leq \varepsilon, \quad \text{dist}\big(0, \rho\partial_z\tilde{f}(x,z)\big) \leq \varepsilon,$$

$$\tilde{f}(x,y) - \min_{z'} \tilde{f}(x,z') \leq \varepsilon.$$

ε-KKT solution of UBO (cont'd)

Under lower-level strong convexity and suitable smoothness assumptions, an ε-KKT solution of UBO yields an approximate stationary point of the hyper-objective function Φ.

Theorem (ε-KKT $\rightarrow$ stationarity of Φ)

Suppose that $\tilde{f}(x, \cdot)$ is strongly convex and that f and $\tilde{f}$ are smooth. If (x, y) is an ε-KKT solution of UBO, then $\|\nabla \Phi(x)\| = \mathcal{O}(\sqrt{\varepsilon})$. Moreover, if the associated penalty parameter satisfies $\rho = \Omega(\varepsilon^{-1})$, then $\|\nabla \Phi(x)\| = \mathcal{O}(\varepsilon)$.

A practical penalty method for UBO

Goal: find an $\mathcal{O}(\varepsilon)$-KKT solution of UBO.

Algorithm 4: A practical penalty method for UBO [Lu & Mei 2025]

Input: $\varepsilon \in (0, 1/4]$, $\rho = \varepsilon^{-1}$, $(x^0, y^0) \in \mathcal{X} \times \mathcal{Y}$ with $\tilde{f}(x^0, y^0) \leq \min_y \tilde{f}(x^0, y) + \varepsilon$.

1. Call Algorithm 2 with $\epsilon \leftarrow \varepsilon$, $\hat{\epsilon}_0 \leftarrow \varepsilon^{3/2}$, $\hat{x}^0 \leftarrow (x^0, y^0)$, $\hat{y}^0 \leftarrow y^0$, and $L_{\nabla h} \leftarrow L_{\nabla f_1} + 2\varepsilon^{-1} L_{\nabla \tilde{f}_1}$ to find an ϵ-stationary point $(x_\epsilon, y_\epsilon, z_\epsilon)$ of $\min_{x,y} \max_z P_\rho(x, y, z)$.
2. Output (x_ϵ, y_ϵ).

Remark: Algorithm 4 can be modified to solve a sequence of subproblems with dynamic penalty and tolerance parameters.

A practical penalty method for UBO (cont'd)

Theorem (Complexity of Algorithm 4)

Algorithm 4 outputs an $\mathcal{O}(\varepsilon)$-KKT solution of UBO within $\mathcal{O}(\varepsilon^{-4}\log\varepsilon^{-1})$ evaluations of ∇f_1, $\nabla\tilde{f}_1$ and the proximal operators of f_2 and $\tilde{f}_2$.

Remark: For UBO with $\tilde{f}$ being **strongly** convex with respect to y, this complexity can be improved to $\mathcal{O}(\varepsilon^{-3}\log\varepsilon^{-1})$.

Constrained bilevel optimization

Consider **constrained** bilevel optimization (CBO)

$$\begin{aligned} \min \quad & f(x,y) \\ \text{s.t.} \quad & y \in \operatorname*{argmin}_z \{\tilde{f}(x,z) | \tilde{g}(x,z) \leq 0\}, \end{aligned}$$

where f and $\tilde{f}$ satisfy all the above assumptions and also the following additional assumptions:

- $\tilde{g}$ is Lipschitz smooth on $\mathcal{X} \times \mathcal{Y}$,
- $\tilde{g}_i(x,\cdot)$ is convex and there exists $\hat{z}_x \in \mathcal{Y}$ for each $x \in \mathcal{X}$ such that $\tilde{g}_i(x,\hat{z}_x) < 0$ for all i.

Related work

Existing methods for special classes of CBO:

- gradient-based methods (Allende & Still 2013, Luo et al. 1996, Yang et al. 2021) for solving CBO with f and $\tilde{f}$ being smooth and $\tilde{g}$ being convex with respect to y
- second-order method (Ma et al. 2021) for solving CBO with $\tilde{f}$ and $\tilde{g}$ being three times continuously differentiable
- difference-of-convex algorithm (Ye et al. 2022) for solving CBO with f being a DC function and $\tilde{f}, \tilde{g}$ being convex functions
- relaxation method based on lower-level Wolfe duality (Li et al. 2023)
- many more ...

Our approach: solve CBO via ***minimax optimization*** (Lu & Mei 2024).

Minimax approximation of CBO

Introduce a **penalty function** for the lower level problem

$$\widetilde{P}_\mu(x,z) = \tilde{f}(x,z) + \mu \left\| [\tilde{g}(x,z)]_+ \right\|^2 .$$

Observation:

- CBO can be approximately solved as the UBO

$$f_\mu^* = \min_{x,y} \left\{ f(x,y) | y \in \operatorname*{argmin}_z \widetilde{P}_\mu(x,z) \right\}.$$

- This UBO can be approximately solved as the **penalty problem**

$$\min_{x,y} f(x,y) + \rho \Big(\widetilde{P}_\mu(x,y) - \min_z \widetilde{P}_\mu(x,z) \Big)$$

and hence the **minimax problem**

$$\min_{x,y} \max_z \left\{ P_{\rho,\mu}(x,y,z) := f(x,y) + \rho(\widetilde{P}_\mu(x,y) - \widetilde{P}_\mu(x,z)) \right\}.$$

A conceptual penalty method for CBO

Algorithm 5: A conceptual penalty method for CBO

Input: positive sequences $\{\rho_k\}$, $\{\mu_k\}$ and $\{\epsilon_k\}$ with $\lim_{k\to\infty}(\rho_k, \mu_k, \epsilon_k) = (\infty, \infty, 0)$. Set $k = 0$.

1. Find an ϵ_k-**optimal** solution (x^k, y^k, z^k) of problem $\min_{x,y} \max_z P_{\rho_k,\mu_k}(x, y, z)$.
2. Set $k \leftarrow k + 1$, and go to step 1.

Theorem (Convergence of Algorithm 5)

Suppose that $\{(x^k, y^k, z^k)\}$ is generated by Algorithm 5. Then any accumulation point of $\{(x^k, y^k)\}$ is an optimal solution of CBO.

Remark: Algorithm 5 is **not implementable** in general.

ε-KKT solution of CBO

Note that CBO can be viewed as

$$\min_{x,y}\{f(x,y)|\tilde{f}(x,y) \leq \tilde{f}^*(x),\ \tilde{g}(x,y) \leq 0\},$$

where $\tilde{f}^*(x) = \min_z\{\tilde{f}(x,z)|\tilde{g}(x,z) \leq 0\}$.

- Its associated Lagrangian function is given by

$$\mathcal{L}(x,y,\rho,\lambda) = f(x,y) + \rho(\tilde{f}(x,y) - \tilde{f}^*(x)) + \langle\lambda, \tilde{g}(x,y)\rangle.$$

- A KKT solution (x,y) of CBO satisfies

$$\tilde{f}(x,y) \leq \tilde{f}^*(x), \quad \tilde{g}(x,y) \leq 0, \quad \langle\lambda, \tilde{g}(x,y)\rangle = 0,$$

and moreover (x, y) is a stationary point of the problem $\min_{x',y'} \mathcal{L}(x',y',\rho,\lambda)$ for some $\rho, \lambda \geq 0$.

ε-KKT solution of CBO (cont'd)

It can be shown that $\min_{x',y'} \mathcal{L}(x', y', \rho, \lambda)$ is equivalent to

$$\min_{x',y',\tilde{\lambda}'} \max_{z'} \left\{ f(x',y') + \rho\big(\tilde{f}(x',y') - \tilde{f}(x',z') - \langle \tilde{\lambda}', \tilde{g}(x',z')\rangle\big) + \langle \lambda, \tilde{g}(x',y')\rangle + \delta_{\mathbb{R}^l_+}(\tilde{\lambda}') \right\},$$

whose stationary point $(x, y, \tilde{\lambda}, z)$ satisfies

$$0 \in \partial f(x,y) + \rho \partial \tilde{f}(x,y) - \rho(\nabla_x \tilde{f}(x,z) + \nabla_x \tilde{g}(x,z)\tilde{\lambda}; 0) + \nabla \tilde{g}(x,y)\lambda,$$

$$0 \in \rho(\partial_z \tilde{f}(x,z) + \nabla_z \tilde{g}(x,z)\tilde{\lambda}),$$

$$\tilde{\lambda} \in \mathbb{R}^l_+, \quad \tilde{g}(x,z) \le 0, \quad \langle \tilde{\lambda}, \tilde{g}(x,z)\rangle = 0.$$

Definition (ε-KKT solution of CBO)

For any $\varepsilon > 0$, (x, y) is said to be an ε-KKT solution of CBO if there exists $(z, \rho, \lambda, \tilde{\lambda}) \in \mathbb{R}^m \times \mathbb{R}_+ \times \mathbb{R}^l_+ \times \mathbb{R}^l_+$ such that

$$\operatorname{dist}\Big(0, \partial f(x,y) + \rho \partial \tilde{f}(x,y) - \rho(\nabla_x \tilde{f}(x,z) + \nabla_x \tilde{g}(x,z)\tilde{\lambda}; 0) + \nabla \tilde{g}(x,y)\lambda\Big) \le \varepsilon,$$

$$\operatorname{dist}\Big(0, \rho(\partial_z \tilde{f}(x,z) + \nabla_z \tilde{g}(x,z)\tilde{\lambda})\Big) \le \varepsilon,$$

$$\|[\tilde{g}(x,z)]_+\| \le \varepsilon, \quad |\langle \tilde{\lambda}, \tilde{g}(x,z)\rangle| \le \varepsilon,$$

$$|\tilde{f}(x,y) - \tilde{f}^*(x)| \le \varepsilon, \quad \|[\tilde{g}(x,y)]_+\| \le \varepsilon, \quad |\langle \lambda, \tilde{g}(x,y)\rangle| \le \varepsilon,$$

A practical penalty method for CBO

Goal: find an $\mathcal{O}(\varepsilon)$-KKT solution of CBO.

Algorithm 6: A practical penalty method for CBO [Lu & Mei 2024]

Input: $\varepsilon \in (0, 1/4]$, $\rho = \varepsilon^{-1}$, $\mu = \varepsilon^{-2}$, $(x^0, y^0) \in \mathcal{X} \times \mathcal{Y}$ with $\widetilde{P}_\mu(x^0, y^0) \leq \min_y \widetilde{P}_\mu(x^0, y) + \varepsilon$. Set $k = 0$.

1. Call Algorithm 2 with $\epsilon \leftarrow \varepsilon$, $\hat{\epsilon}_0 \leftarrow \varepsilon^{5/2}$, $\hat{x}^0 \leftarrow (x^0, y^0)$, and $\hat{y}^0 \leftarrow y^0$ to find an ϵ-stationary point $(x_\epsilon, y_\epsilon, z_\epsilon)$ of $\min_{x,y} \max_z P_{\rho,\mu}(x, y, z)$.
2. Output (x_ϵ, y_ϵ).

Remark: Algorithm 6 can be modified to solve a sequence of subproblems with dynamic penalty and tolerance parameters.

A practical penalty method for CBO (cont'd)

Theorem (Complexity of Algorithm 6)

Algorithm 6 outputs an $\mathcal{O}(\varepsilon)$-KKT solution of CBO within $\mathcal{O}(\varepsilon^{-7}\log\varepsilon^{-1})$ evaluations of ∇f_1, $\nabla\tilde{f}_1$, $\nabla\tilde{g}$ and the proximal operators of f_2 and $\tilde{f}_2$.

Remark: For CBO with $\tilde{f}$ being **strongly** convex with respect to y, this complexity can be improved to $\mathcal{O}(\varepsilon^{-6}\log\varepsilon^{-1})$.

Numerical results: unconstrained bilevel optimization

$$\begin{aligned} \min \quad & c^T x + d^T y + \delta_{[-1,1]^n}(x) \\ \text{s.t.} \quad & y \in \operatorname*{argmin}_z \left\{ x^T \widetilde{A} z + z^T \widetilde{B} z + \tilde{d}^T z + \delta_{[-1,1]^m}(z) \right\}, \end{aligned}$$

where $\widetilde{A} \in \mathbb{R}^{n \times m}$, $\widetilde{B} \in \mathbb{R}^{m \times m}$, $c \in \mathbb{R}^n$, $d, \tilde{d} \in \mathbb{R}^m$, and $\delta_S(\cdot)$ is the indicator function of S. Apply Algorithm 4 with $\varepsilon = 10^{-4}$.

n	m	Initial objective value	Final objective value
100	100	-0.35	-101.67
200	200	-0.53	-194.91
300	300	-0.48	-307.43
400	400	-0.44	-401.71
500	500	-0.05	-527.45
600	600	0.99	-644.53
700	700	0.49	-759.54
800	800	-1.23	-872.77
900	900	-2.07	-1004.27
1000	1000	-1.06	-1107.61

Numerical results: constrained bilevel optimization

$$\begin{aligned} \min \quad & c^T x + d^T y + \delta_{[-1,1]^n}(x) \\ \text{s.t.} \quad & y \in \operatorname*{argmin}_z \left\{ \tilde{d}^T z + \delta_{[-1,1]^m}(z) \middle| \widetilde{A}x + \widetilde{B}z - \tilde{b} \leq 0 \right\}, \end{aligned}$$

where $c \in \mathbb{R}^n$, $d, \tilde{d} \in \mathbb{R}^m$, $\tilde{b} \in \mathbb{R}^l$, $\widetilde{A} \in \mathbb{R}^{l \times n}$, and $\widetilde{B} \in \mathbb{R}^{l \times m}$. Apply Algorithm 6 with $\varepsilon = 10^{-4}$.

n	m	l	Initial objective value	Final objective value
100	100	5	-0.51	-34.83
200	200	10	-0.15	-121.41
300	300	15	1.56	-208.44
400	400	20	-0.04	-298.25
500	500	25	1.45	-384.77
600	600	30	0.75	-470.31
700	700	35	0.09	-568.26
800	800	40	-0.98	-629.61
900	900	45	1.21	-689.00
1000	1000	50	1.44	-781.79

A sequential minimax optimization method for CBO

Goal: propose a sequential minimax optimization (SMO) method for CBO

$$\begin{aligned} \min \quad & f(x,y) \\ \text{s.t.} \quad & y \in \operatorname*{argmin}_z \{\tilde{f}(x,z) | \tilde{g}(x,z) \leq 0\}. \end{aligned}$$

Observation:

- CBO can be approximately solved as

$$\begin{aligned} \min \quad & f(x,y) \\ \text{s.t.} \quad & y \in \operatorname*{argmin}_z \{\tilde{f}(x,z) + \tfrac{1}{2\rho\mu}\left(\|[\lambda + \mu\tilde{g}(x,z)]_+\|^2 - \|\lambda\|^2\right)\}. \end{aligned}$$

- The latter can be approximately solved as

$$\begin{aligned} \min_{x,y} f(x,y) + \rho\Big(\tilde{f}(x,y) + \frac{1}{2\rho\mu}\left(\|[\lambda + \mu\tilde{g}(x,y)]_+\|^2 - \|\lambda\|^2\right) \\ - \min_z \left\{\tilde{f}(x,z) + \frac{1}{2\rho\mu}\left(\|[\lambda + \mu\tilde{g}(x,z)]_+\|^2 - \|\lambda\|^2\right)\right\}\Big), \end{aligned}$$

which is equivalent to $\min_{x,y} \max_z \mathcal{L}(x,y,z,\lambda;\rho,\mu)$ where

$$\begin{aligned} \mathcal{L}(x,y,z,\lambda;\rho,\mu) = & f(x,y) + \rho\tilde{f}(x,y) + \frac{1}{2\mu}\|[\lambda + \mu\tilde{g}(x,y)]_+\|^2 \\ & - \rho\tilde{f}(x,z) - \frac{1}{2\mu}\|[\lambda + \mu\tilde{g}(x,z)]_+\|^2. \end{aligned}$$

A sequential minimax optimization method for CBO (cont'd)

Define

$$\widetilde{\mathcal{L}}(x, z, \lambda; \rho, \mu) := \tilde{f}(x, z) + \frac{1}{2\rho\mu}\|[\lambda + \mu\tilde{g}(x, z)]_+\|^2.$$

Algorithm 7: A sequential minimax optimization method for CBO [Lu & Mei 2026]

Input: $\varepsilon, \tau \in (0, 1)$, $\epsilon_0 \in (\tau\varepsilon, 1]$, $x^0 \in \mathcal{X}$, $y^0 \in \mathcal{Y}$, $z^0 \in \mathcal{Y}$, $\epsilon_k = \epsilon_0\tau^k$, $\rho_k = \epsilon_k^{-1}$, $\mu_k = \epsilon_k^{-3}$, $\eta_k = \epsilon_k$, and $\lambda^0 \in \mathbb{R}^l_+$. Set $k = 0$.

1. Call Nesterov's accelerated proximal gradient method starting at y^k to find an η_k-optimal solution y^k_{init} of $\min_z \widetilde{\mathcal{L}}(x^k, z, \lambda^k; \rho_k, \mu_k)$.
2. Call Algorithm 2 with $\epsilon \leftarrow \epsilon_k$, $\hat{\epsilon}_0 \leftarrow \epsilon_k/(2\sqrt{\mu_k})$, $\hat{x}^0 \leftarrow (x^k, y^k_{\text{init}})$, and $\hat{y}^0 \leftarrow z^k$ to find an ϵ_k-stationary point $(x^{k+1}, y^{k+1}, z^{k+1})$ of

$$\min_{x,y} \max_z \mathcal{L}(x, y, z, \lambda^k; \rho_k, \mu_k).$$

3. Set $\lambda^{k+1} = [\lambda^k + \mu_k\tilde{g}(x^{k+1}, z^{k+1})]_+$.
4. If $\epsilon_k \le \varepsilon$, terminate the algorithm and output (x^{k+1}, y^{k+1}).
5. Set $k \leftarrow k + 1$, and go to step 1.

A sequential minimax optimization method for CBO (cont'd)

Theorem (Complexity of Algorithm 7)

Algorithm 7 outputs an $\mathcal{O}(\varepsilon)$-KKT solution of CBO within $\mathcal{O}(\varepsilon^{-7}\log\varepsilon^{-1})$ evaluations of ∇f_1, $\nabla\tilde{f}_1$, $\nabla\tilde{g}$ and the proximal operators of f_2 and $\tilde{f}_2$.

Remark: For CBO with $\tilde{f}$ being **strongly** convex with respect to y, the above complexity can be improved to $\mathcal{O}(\varepsilon^{-6}\log\varepsilon^{-1})$.

Numerical result

$$\begin{aligned} \min \quad & c^T x + d^T y + \delta_{[-1,1]^n}(x) \\ \text{s.t.} \quad & y \in \underset{z}{\operatorname{argmin}} \left\{ \tilde{d}^T z + \delta_{[-1,1]^m}(z) \middle| \widetilde{A}x + \widetilde{B}z - \tilde{b} \leq 0 \right\}, \end{aligned}$$

where $c \in \mathbb{R}^n$, $d, \tilde{d} \in \mathbb{R}^m$, $\tilde{b} \in \mathbb{R}^l$, $\widetilde{A} \in \mathbb{R}^{l \times n}$, and $\widetilde{B} \in \mathbb{R}^{l \times m}$. Apply Algorithms 6 and 7 with $\varepsilon = 10^{-4}$.

			Initial obj val	Final obj val		CPU time (seconds)	
n	m	l		SMO	FPM	SMO	FPM
100	100	5	0.22	−75.78	−75.75	7.4	22.1
200	200	10	−0.38	−154.20	−154.18	12.5	107.9
300	300	15	−0.11	−246.70	−246.66	24.1	267.8
400	400	20	0.34	−305.97	−305.89	37.0	561.6
500	500	25	0.96	−394.74	−394.70	63.8	719.8

Improved complexity for finding an ε-KKT solution

[Shen et al. 2026] proposed a saddle-point reformulation of the constrained lower-level problem, avoiding the introduction of an additional penalty parameter for the lower-level constraints.

Fact: For any sufficiently large Λ, $y \in \operatorname{argmin}_z\{\tilde{f}(x,z) \mid \tilde{g}(x,z) \le 0\}$ if and only if there exists $\lambda \in [0,\Lambda]^l$ such that (y,λ) is a saddle point of

$$L(x,y',\lambda';\Lambda) := \tilde{f}(x,y') + \langle \lambda', \tilde{g}(x,y')\rangle - \delta_{[0,\Lambda]^l}(\lambda').$$

- Define the saddle-point gap function by

$$G(x,y,\lambda;\Lambda) := \max_{\tilde{\lambda}} L(x,y,\tilde{\lambda};\Lambda) - \min_{\tilde{y}} L(x,\tilde{y},\lambda;\Lambda).$$

Then (y,λ) is a saddle point if and only if $G(x,y,\lambda;\Lambda) = 0$.

- Penalizing the gap function G yields the minimax approximation:

$$\min_{x,y,\lambda} \max_{\tilde{y},\tilde{\lambda}} \big\{ f(x,y) + \rho\big(L(x,y,\tilde{\lambda};\Lambda) - L(x,\tilde{y},\lambda;\Lambda)\big)\big\}.$$

- Applying an algorithmic framework similar to Algorithm 2 yields an $\widetilde{\mathcal{O}}(\epsilon^{-4})$ oracle complexity for finding an ϵ-KKT solution of CBO.

Summary

- Introduced the historical background of bilevel optimization and its modern applications in machine learning.
- For bilevel optimization with a strongly convex lower-level objective function, presented the two-timescale stochastic approximation (TTSA) algorithm and the fully first-order stochastic approximation (F^2SA) method, together with their iteration complexity guarantees.
- For bilevel optimization with a convex lower-level objective function, discussed the challenges arising from nonunique lower-level solutions, reformulated the bilevel problem as a minimax optimization problem, and presented implementable first-order penalty methods with provable iteration complexity guarantees.

Thank you!

References

- J. H. Alcantara and A. Takeda. Theoretical smoothing frameworks for nonsmooth simple bilevel problems. *Mathematics of Operations Research*, 2026.
- G. B. Allende and G. Still. Solving bilevel programs with the KKT-approach. *Mathematical Programming*, 2013.
- J. Bracken and J. T. McGill. Mathematical programs with optimization problems in the constraints. *Operations Research*, 21(1):37–44, 1973.
- L. Chen, Y. Ma, and J. Zhang. Near-optimal nonconvex-strongly-convex bilevel optimization with fully first-order oracles. *Journal of Machine Learning Research*, 2025.
- L. Chen, J. Xu, and J. Zhang. Bilevel optimization without lower-level strong convexity from the hyper-objective perspective. arXiv:2301.00712, 2023.
- S. Fan, M. Pagliardini, and M. Jaggi. DoGE: Domain reweighting with generalization estimation. ICML, 2024.
- L. Franceschi, P. Frasconi, S. Salzo, R. Grazzi, and M. Pontil. Bilevel programming for hyperparameter optimization and meta-learning. ICML, 2018.
- Y. Gao, H. Yang, P. Zhang, C. Zhou, and Y. Hu. GraphNAS: Graph neural architecture search with reinforcement learning. arXiv:1904.09981, 2019.
- Y. Gao, C. Yong, Z. Xiong, D. Niyato, Y. Xiao, and J. Zhao. A Stackelberg game approach to resource allocation for intelligent reflecting surface aided communications. arXiv:2003.06640, 2020.
- S. Ghadimi and M. Wang. Approximation methods for bilevel programming. arXiv:1802.02246, 2018.

- P. Hansen, B. Jaumard, and G. Savard. New branch-and-bound rules for linear bilevel programming. *SIAM Journal on Scientific and Statistical Computing*, 1992.
- M. Hong. Bilevel Optimization: Recent Algorithmic Theoretical Advances, and Emerging Applications in Training LLMs (talk slides). The 2025 International Conference on Continuous Optimization (ICCOPT 2025), University of Southern California, July 2025.
- M. Hong, H.-T. Wai, Z. Wang, and Z. Yang. A two-timescale stochastic algorithm framework for bilevel optimization: Complexity analysis and application to actor-critic. *SIAM Journal on Optimization*, 2023.
- X. Hu, N. Xiao, X. Liu, and K.-C. Toh. An improved unconstrained approach for bilevel optimization. *SIAM Journal on Optimization*, 2023.
- M. Huang, X. Chen, K. Ji, S. Ma, and L. Lai. Efficiently escaping saddle points in bilevel optimization. *Journal of Machine Learning Research*, 2025.
- K. Ji and Y. Liang. Lower bounds and accelerated algorithms for bilevel optimization. *Journal of Machine Learning Research*, 2023.
- P. Khanduri, S. Zeng, M. Hong, H.-T. Wai, Z. Wang, and Z. Yang. A near-optimal algorithm for stochastic bilevel optimization via double-momentum. NeurIPS, 2021.
- T. Kim, S. J. Wright, D. Bienstock, and S. Harnett. Vulnerability analysis of power systems. arXiv:1503.02360, 2015.
- D. Kovalev and A. Gasnikov. The first optimal algorithm for smooth and strongly-convex-strongly-concave minimax optimization. NeurIPS, 2022.
- J. Kwon, D. Kwon, S. Wright, and R. Nowak. A fully first-order method for stochastic bilevel optimization. ICML, 2023.
- J. Kwon, D. Kwon, S. Wright, and R. Nowak. On penalty methods for nonconvex bilevel optimization and first-order stochastic approximation. ICLR, 2024.

- J. Kwon, D. Kwon, and H. Lyu. On the complexity of first-order methods in stochastic bilevel optimization. ICML, 2024.
- Y. Li, G.-H. Lin, J. Zhang, and X. Zhu. A novel approach for bilevel programs based on Wolfe duality. arXiv:2302.06838, 2023.
- B. Liu, M. Ye, S. Wright, P. Stone, and Q. Liu. BOME! Bilevel optimization made easy: A simple first-order approach. NeurIPS, 2022.
- H. Liu, K. Simonyan, and Y. Yang. DARTS: Differentiable architecture search. ICLR, 2018.
- 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. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.
- R. Liu, X. Liu, X. Yuan, S. Zeng, and J. Zhang. A value-function-based interior-point method for non-convex bi-level optimization. ICML, 2021.
- Z. Lu and S. Mei. First-order penalty methods for bilevel optimization. *SIAM Journal on Optimization*, 2024.
- Z. Lu and S. Mei. A first-order augmented Lagrangian method for constrained minimax optimization. *Mathematical Programming*, 2025.
- Z. Lu and S. Mei. Solving bilevel optimization via sequential minimax optimization. *Mathematics of Operations Research*, 2026.
- Z.-Q. Luo, J.-S. Pang, and D. Ralph. *Mathematical Programs with Equilibrium Constraints*. Cambridge University Press, 1996.

- X. Ma, W. Yao, J. J. Ye, and J. Zhang. Combined approach with second-order optimality conditions for bilevel programming problems. arXiv:2108.00179, 2021.
- A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards deep learning models resistant to adversarial attacks. ICLR, 2018.
- R. Pan, D. Zhang, H. Zhang, X. Pan, M. Xu, J. Zhang, R. Pi, X. Wang, and T. Zhang. ScaleBiO: Scalable bilevel optimization for LLM data reweighting. arXiv:2406.19976, 2024.
- H. Reisizadeh, J. Jia, Z. Bu, B. Vinzamuri, A. Ramakrishna, K.-W. Chang, V. Cevher, S. Liu, and M. Hong. BLUR: A bi-level optimization approach for LLM unlearning. arXiv:2506.08164, 2025.
- V. Sehwag, S. Wang, P. Mittal, and S. Jana. Hydra: Pruning adversarially robust neural networks. NeurIPS, 2020.
- Y. Shen, Y. He, W. Wang, and Q. Lin. Penalty-based first-order methods for bilevel optimization with minimax and constrained lower-level problems. arXiv:2605.08006, 2026.
- D. Silver and R. Sutton. Welcome to the era of experience. Google AI, 2025.
- A. Sinha, P. Malo, and K. Deb. A review on bilevel optimization: From classical to evolutionary approaches and applications. *IEEE Transactions on Evolutionary Computation*, 2017.
- D. Sow, K. Ji, and Y. Liang. On the convergence theory for Hessian-free bilevel algorithms. NeurIPS, 2022.
- D. Sow, K. Ji, Z. Guan, and Y. Liang. A primal-dual approach to bilevel optimization with multiple inner minima. arXiv:2203.01123, 2022.

- H. von Stackelberg. *Marktform und Gleichgewicht*. Springer, 1934.
- H. Sun, W. Pu, M. Zhu, X. Fu, T.-H. Chang, and M. Hong. Learning to continuously optimize wireless resource in episodically dynamic environment. ICASSP, 2021.
- S. Wright. Bilevel Optimization: Background and Applications (talk slides). The Workshop on Optimisation, Metric Bounds, Approximation and Transversality (WOMBAT), University of Queensland, Australia, November 2025.
- S. Wright. Fully First-Order Methods for Bilevel Optimization (talk slides). The Workshop on Optimisation, Metric Bounds, Approximation and Transversality (WOMBAT), University of Queensland, Australia, November 2025.
- A. Vahdat, A. Mallya, M.-Y. Liu, and J. Kautz. UNAS: Differentiable architecture search meets reinforcement learning. CVPR, 2020.
- C. Xue, X. Wang, J. Yan, Y. Hu, X. Yang, and K. Sun. Rethinking bi-level optimization in neural architecture search: A Gibbs sampling perspective. AAAI, 2021.
- J. Yang, K. Ji, and Y. Liang. Provably faster algorithms for bilevel optimization. NeurIPS, 2021.
- J. J. Ye, X. Yuan, S. Zeng, and J. Zhang. Difference of convex algorithms for bilevel programs with applications in hyperparameter selection. *Mathematical Programming*, 2023.
- Y. Zhang, Y. Yao, P. Ram, P. Zhao, T. Chen, M. Hong, Y. Wang, and S. Liu. Advancing model pruning via bi-level optimization. NeurIPS, 2022.
- Y. Zhang, P. Khanduri, I. Tsaknakis, Y. Yao, M. Hong, and S. Liu. An introduction to bilevel optimization: Foundations and applications in signal processing and machine learning. *IEEE Signal Processing Magazine*, 2024.