Benchmarking Stochastic Approximation Algorithms for Fairness-Constrained Training of Deep Neural Networks

Anonymous Author(s)
Affiliation
Address
email

Abstract

The ability to train Deep Neural Networks (DNNs) with constraints is instrumental in improving the fairness of modern machine-learning models. Many algorithms have been analysed in recent years, and yet there is no standard, widely accepted method for the constrained training of DNNs. In this paper, we provide a challenging benchmark of real-world large-scale fairness-constrained learning tasks, built on top of the US Census (Folktables, [22]). We point out the theoretical challenges of such tasks and review the main approaches in stochastic approximation algorithms. Finally, we demonstrate the use of the benchmark by implementing and comparing three recently proposed, but as-of-yet unimplemented, algorithms both in terms of optimization performance, and fairness improvement. We release the code of the benchmark as a Python package at https://github.com/humancompatible/train.

1 Introduction

There has been a considerable interest in detecting and mitigating bias in artificial intelligence (AI) systems, recently. Multiple legislative frameworks, including the AI Act in the European Union, require the bias to be removed, but there is no agreement on what the correct definition of bias is or how to remove it. A natural translation of the requirement of removing bias into the formulation of training of deep neural network (DNN) utilizes constraints bounding the difference in empirical risk across multiple subgroups [13, 42, 48]. Over the past five years, there have been numerous algorithms ([29, 5, 17, 43, 6, 40, 41, 10, 18, 49, 27, 32, 33]) proposed to solve convex and non-convex empirical-risk minimization (ERM) problems subject to constraints bounding the absolute value of empirical risk. Numerous other algorithms of this kind could be construed, based on a number of design choices, including:

- sampling techniques for the ERM objective and the constraints, either the same or different;
- use of first-order or higher-order derivatives, possibly in quasi-Newton methods;
- use of globalization strategies such as filters or line search;
- use of "true" globalization strategies including random initial points and random restarts in order to reach global minimizers.

Nevertheless, there is no single toolkit implementing the algorithms, which would allow for their easy comparison, and there is no benchmark to test the combinations of design choices on.

In this paper, we consider the constrained ERM problem:

$$\min_{x\in\mathbb{R}^n} \mathbb{E}[f(x,\xi)] \quad \text{s.t.} \quad \mathbb{E}[c(x,\zeta)] \leq 0, \tag{1}$$

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

Table 1: Particular formulations of the constraint function c to enforce fairness.

Model	Our formulation
Demographic Parity [24]	$\lvert\mathbb{E}_{\mathcal{D}[\text{group } A]}[\ell(f_\theta(X), Y)] - \mathbb{E}_{\mathcal{D}[\text{group } B]}[\ell(f_\theta(X), Y)]\rvert \le \delta$
Equal opportunity [31]	$\lvert\mathbb{E}_{\mathcal{D}[\text{group } A, Y=+]}[\ell(f_\theta(X), Y)] - \mathbb{E}_{\mathcal{D}[\text{group } B, Y=+]}[\ell(f_\theta(X), Y)]\rvert \le \delta$
Equalized odds [31]	$\sum_{v\in\{+,-\}} \lvert\mathbb{E}_{\mathcal{D}[\text{group } A, Y=v]}[\ell(f_\theta(X), Y)] - \mathbb{E}_{\mathcal{D}[\text{group } B, Y=v]}[\ell(f_\theta(X), Y)]\rvert \le \delta$

where ξ and ζ are random variables. Further, we provide an automated way of constructing the ERM formulations out of a computation graph of a neural network defined by PyTorch or TensorFlow, the choice of the constraints (see Table 1), and a definition of the protected subgroups to apply the constraints to. Specifically, we provide means of utilizing the US Census data via the Python package Folktables, together with definitions of up to 5.7 billion protected subgroups. This presents a challenging benchmark in stochastic approximation for the constrained training of deep neural networks.

Our contributions. The contributions of this paper are:

- a literature review of algorithms subject to handling (1);
- a toolbox that *(i)* implements four algorithms applicable in real-world situations, and *(ii)* provides an easy-to-use benchmark on real-world fairness problems;
- numerical experiments that compare these algorithms on a real-world dataset, and a comparison with alternative approaches to fairness.

Paper structure. The rest of the paper is organized as follows. Section 2 reviews related works and presents the relevant notions of fairness. Section 3 introduces the algorithms. Section 4 reports on our experiments. Section 5 concludes.

2 Related work, and background in fairness

In the literature on fairness, one distinguishes among pre-processing, in-processing, and post-processing. Pre-processing methods focus on modifying the training data to mitigate biases [50, 23]. In-processing methods enforce fairness during the training process by modifying the learning algorithm itself [53]. Post-processing methods adjust the model's predictions after training [35]. The constrained ERM approach (1) belongs to the class of in-processing methods.

In-processing methods include several approaches. One trend consists in jointly learning a predictor function and an adversarial agent that aims to reconstitute the subgroups from the predictor [1, 38, 39, 25]. Another approach consists in adding "penalization" terms to the empirical risk term. These additional penalization terms, commonly referred to as regularizers, promote models that are a compromise between fitting the training data, and optimizing a fairness metric. Differentiable regularizers include, among others, HSIC [37], Fairret [11], or Prejudice Remover [34].

Closer to our setting, [16] considers minimizing the empirical risk subject to the so-called rate constraints based on the model's prediction rates on different datasets. These rates, derived from a dataset, give rise to non-convex, non-smooth, and large-scale inequality constraints akin to (1). The authors of [16] argue that hard constraints, although leading to a more difficult optimization problem, offer advantages over using a weighted sum of multiple penalization terms. Indeed, while the choice of weights for the penalization terms may depend on the dataset, specifying one constraint for each goal is easier for practitioners. In addition, a penalization-based model provides a predictor that balances minimizing the data-fit term and penalties in an opaque way, whereas a constraint-based model allows for a clearer understanding of the model design: minimizing the data-fit term subject to "hard" fairness constraints. Rate constraints differ from those in (1) in that they are piecewise-constant, rendering first-order methods unsuitable for solving them.

Major toolboxes for evaluating the fairness of models or for training models with fairness guarantees include AIF360 [4] and FairLearn [8]. Other libraries include [21], which computes the Pareto front of accuracy and fairness metrics for high-capacity models, and [11], which provides differentiable fairness-inducing penalization terms.

A detailed survey of fairness-oriented datasets is provided in [36], and new datasets are derived in [22]. The benchmark [30] provides a review of the existence of biases in prominent datasets, finding that "not all widely used fairness datasets stably exhibit fairness issues", and assesses the performance of a wide range of in-processing fairness methods in addressing biases, focusing on differentiable minimization only. Other benchmarks of fairness methods include [20, 26, 46, 15]. The statistical aspects of the fairness-constrained Empirical Risk Minimization have only been considered recently; see e.g. [12].

The template problem (1) encompasses fairness-enforcing approaches that find applications in high-risk domains, such as credit scoring, hiring processes, medicine and healthcare [14], ranking and recommendation [47], but also in forecasting the observations of linear dynamical systems [55], or in two-sided economic markets [54]. In addition, solving (1) is of interest in other fields, such as compression of neural networks [13], improving statistical performance of neural networks [42, 48], or the training of neural networks with constraints on the Lipschitz bound [45]. We note that the presence of large-scale constraints is a common feature to all the aforementioned methodologies.

Deep neural networks (DNNs). Consider a dataset of N observations $\mathcal{D} = \{(X_i, Y_i), i = 1, ..., N\}$. We seek some function f_θ such that $f_\theta(X_i) \approx Y_i$. A typical formulation of this task is the following regression problem:

$$\min_{\theta \in \mathbb{R}^n} \frac{1}{N} \sum_{i=1}^{N} \ell(f_\theta(X_i), Y_i) + \mathcal{R}(\theta). \tag{2}$$

Here, $\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ is a loss function, such as the logistic loss $\ell(y; z) = \log(1 + e^{-yz})$, the hinge loss $\ell(y; z) = \max\{0, 1 - yz\}$, the absolute deviation loss $\ell(y; z) = |y - z|$, or the square loss $\ell(y; z) = \frac{1}{2}(y - z)^2$. The term $\mathcal{R}$ is a regularizer, and f_θ is a deep neural network (DNN) of depth L with parameters θ. The DNN f_θ is defined recursively, for some input X, as

$$a_0 = X, \qquad a_i = \rho_i(V_i(\theta) a_{i-1}), \ \text{ for every } i = 1, \ldots, L, \qquad f_\theta(X) = a_L, \tag{3}$$

where $V_i(\cdot)$ are linear maps into the space of matrices, and ρ_i are activation functions applied coordinate-wise, such as ReLU $\max(0, t)$, quadratics t^2, hinge losses $\max\{0, t\}$, and SoftPlus $\log(1 + e^t)$. A dataset $\mathcal{D}$ is described by attributes (or features), such as age, income, gender, etc. The attribute which the DNN is trained to predict is called the class attribute. We denote the class attribute by Y, whereas the predicted value given by the DNN is denoted by $\hat{Y}$. Both Y and $\hat{Y}$ are binary and take values in $\{+, -\}$.

Fairness-aware learning applied to DNNs. The goal of this approach is to reduce discriminatory behavior in the predictions of a DNN across different demographic groups (e.g., male vs. female). The demographic groups are also reffered to as subgroups. The attributes such as race or gender which must be handled cautiously are called protected. We denote by S the protected attribute $S \in \{s, \bar{s}\}$ where s denotes the protected group and $\bar{s}$ denotes the non-protected group. Denote by $\mathcal{D}[s]$ and $\mathcal{D}[\bar{s}]$ the observations in $\mathcal{D}$ such that $S = s$ and $S = \bar{s}$, respectively. A way to impose fairness on the learned predictor is to equip (2) with suitable constraints. Some possible constraint choices are shown in Table 1. Choosing loss difference bound as the constraint and setting $\delta > 0$ yields formulation:

$$\begin{aligned} \min_{\theta \in \mathbb{R}^n} \quad & \frac{1}{N} \sum_{i=1}^{N} \ell(f_\theta(X_i), Y_i) + \mathcal{R}(\theta) \\ \text{s.t.} \quad & -\delta \le \frac{1}{|\mathcal{D}[s]|} \sum_{X_i, Y_i \in \mathcal{D}[s]} \ell(f_\theta(X_i), Y_i) - \frac{1}{|\mathcal{D}[\bar{s}]|} \sum_{X_i, Y_i \in \mathcal{D}[\bar{s}]} \ell(f_\theta(X_i), Y_i) \le \delta. \end{aligned} \tag{4}$$

Fairness metrics. There exist tens of fairness metrics [51], however, it was pointed out in [3, Ch. 3] that most fairness metrics may be seen as combinations of independence, separation, and sufficiency. These baseline fairness criteria cannot be attained simultaneously. Moreover, there is a trade-off between attaining the baseline fairness metrics and the prediction accuracy, i.e., the probability that the predicted value is equal to the actual value. As a result, we seek an optimal trade-off between attaining the fairness metrics and minimizing the prediction inaccuracy. We follow the definitions in [3] of the baseline fairness metrics applied to a binary classification task.

Independence (Ind) This fairness criterion requires the prediction $\hat{Y}$ to be statistically independent of the protected attribute S. Equivalent definitions of independence for a binary classifier $\hat{Y}$ are referred to as statistical parity (SP), demographic parity, and group fairness. Independence is the simplest criterion to work with, both mathematically and algorithmically. In a binary classification task, independence implies the equality of $P(\hat{Y} = + \mid S = s)$ and $P(\hat{Y} = + \mid S = \bar{s})$ and the fairness gap is computed as

$$|P(\hat{Y} = + \mid S = s) - P(\hat{Y} = + \mid S = \bar{s})|.$$

Separation (Sp) Unlike independence, the separation criterion requires the prediction $\hat{Y}$ to be statistically independent of the protected attribute S, given the true label Y. The separation criterion also appears under the name Equalized odds (EO). In a binary classification task, the separation criterion requires that all groups experience the same true negative rate and the same true positive rate. Formally, we require the equality of $P(\hat{Y} = + \mid S = s, Y = v)$ and $P(\hat{Y} = + \mid S = \bar{s}, Y = v)$, for every $v \in \{+, -\}$. The fairness gap may be computed as

$$\sum_{v \in \{+,-\}} |P(\hat{Y} = + \mid S = s, Y = v) - P(\hat{Y} = + \mid S = \bar{s}, Y = v)|.$$

Sufficiency (Sf) The sufficiency criterion is satisfied if the true label Y is statistically independent of the protected attribute S, given the prediction $\hat{Y}$. In a binary classification task, the sufficiency criterion requires a parity of positive and negative predictive values across the groups. Formally, we require the equality of $P(Y = + \mid \hat{Y} = v, S = s)$ and $P(Y = + \mid \hat{Y} = v, S = \bar{s})$, for every $v \in \{+, -\}$, and the fairness gap may be computed as

$$\sum_{v \in \{+,-\}} |P(Y = + \mid S = s, \hat{Y} = v) - P(Y = + \mid S = \bar{s}, \hat{Y} = v)|.$$

3 Algorithms

We recall that we consider the optimization problem

$$\min_{x \in \mathbb{R}^n} F(x) \quad \text{s.t.} \quad C(x) \leq 0, \tag{5}$$

where the functions $F : \mathbb{R}^n \to \mathbb{R}$ and $C : \mathbb{R}^n \to \mathbb{R}^m$ are defined as expectations of functions f and c, which depend on random variables ξ and ζ, respectively. Solving (5) has the following challenges:

- large-scale objective and constraint functions, which require sampling schemes,
- the necessity of incorporating inequality constraints, not merely equality constraints (see fairness formulations in Table 1),
- the necessity to cope with the nonconvexity and nonsmoothness of F and C, due to the presence of neural networks.

In this section, we identify the algorithms that address these challenges most precisely. However, we note that there exists currently no algorithm with guarantees for such a general setting.

Recalls and notation. We denote the projection of a point x onto a set $\mathcal{X}$ by $\mathrm{proj}_{\mathcal{X}}(x) = \arg\min_{v \in \mathcal{X}} \|x - v\|^2$. We denote by $N \sim \mathcal{G}(p_0)$ sampling a random variable from the geometric distribution with a parameter p_0, i.e., the probability that $N = n$ equals $(1 - p_0)^n p_0$ for $n \geq 0$. We distinguish between the random variable ξ associated with the objective function and the random variable ζ associated with the constraint function. Their probability distributions are denoted by $\mathcal{P}_\xi$ and $\mathcal{P}_\zeta$. For an integer $J \in \mathbb{N}$, a set $\{\xi_j\}_{j=1}^J$ of independent and identically distributed random variables $\xi_1, \ldots, \xi_J \overset{iid}{\sim} \mathcal{P}_\xi$ is called a mini-batch. Inspired by [40], we use the following notation for the stochastic estimates computed from a mini-batch of size J:

$$\overline{\nabla}^J f(x) = \tfrac{1}{J} \sum_{j=1}^J \nabla f(x, \xi_j), \quad \overline{c}^J(x) = \tfrac{1}{J} \sum_{j=1}^J c(x, \zeta_j), \quad \overline{\nabla}^J c(x) = \tfrac{1}{J} \sum_{j=1}^J \nabla c(x, \zeta_j). \tag{6}$$

Table 2: Assumptions on objective and constraint functions, F and C, which allow for theoretical convergence proofs.

	Objective function F				Constraint function C						
Algorithm	stochastic	weakly convex	$\mathcal{C}^1$ with Lipschitz ∇F	tame loc. Lipschitz	stochastic	$C(x)=0$	$C(x)=0$ and $C(x)\leq 0$	linear	weakly convex	$\mathcal{C}^1$ with Lipschitz ∇C	tame loc. Lipschitz
SGD	✓	(✓)	(✓)	✓							
[6] [29] [17]	✓	–	✓	–	–	✓	–	–	–	✓	–
[40]	✓	–	✓($\mathcal{C}^3$)	–	–	✓	–	–	–	✓($\mathcal{C}^3$)	–
[49] [18]	✓	–	✓	–	–	(✓)	✓	–	–	✓	–
[41]	✓	–	✓($\mathcal{C}^2$)	–	–	(✓)	✓	–	–	✓($\mathcal{C}^2$)	–
[10]	✓	–	✓(+ cvx)	–	–	✓	–	✓	–	–	–
[43]	✓	–	✓	–	✓	✓	–	–	–	✓	–
SSL-ALM [32]	✓	–	✓	–	✓	(✓)	✓	✓	–	–	–
Stoch. Ghost [27]	✓	–	✓	–	✓	(✓)	✓	–	–	✓	–
Stoch. Switch. Subg. [33]	✓	✓	–	–	✓	(✓)	✓	–	✓	–	–

3.1 Review of methods for constrained ERM

We compare recent constrained optimization algorithms considering a stochastic objective function in Table 2. We note that most of them do not consider the case of stochastic constraints. Among those which do consider stochastic constraints, only three admit inequality constraints. Moreover, with the exception of [33], all the algorithms in Table 2 assume F to be at least $\mathcal{C}^1$, which makes addressing the challenge of nonsmoothness of F infeasible. The recent paper [19] leads us to the conclusion that assuming the objective and constraint functions to be tame and locally Lipschitz is a suitable requirement for solving (5) with theoretical guarantees of convergence. At this point, however, no such algorithm exists, to the best of our knowledge.

Consequently, we consider the practical performance of the algorithms that address the challenges of solving (5) most closely: Stoch. Ghost [27], SSL-ALM [32], and Stoch. Switching Subgradient [33].

3.2 Stochastic Ghost Method (StGh)

The Stochastic Ghost method was described in [27] where a method for solving (1) in the non-stochastic setting [28] was combined with the stochastic sampling inspired by an unbiased Monte Carlo method [9]. The method [28] for the non-stochastic setting is based on solving subproblem (7) to obtain a direction d to preform the classical line search. Here, $e \in \mathbb{R}^m$ is a vector with all elements equal to one, τ and $\beta > 0$ are user-prescribed constants and κ_k is defined as a certain convex combination of optimization subproblems related to C and ∇C. The definition of κ_k enables to expand the feasibility region so that (7) is always feasible. As the problem (1) is stochastic, the subproblem (7) is modified to a stochastic version (8), using the notation in (6):

$$
\begin{aligned}
\min_d \quad & \nabla F(x_k)^\top d + \tfrac{\tau}{2}\|d\|^2, \\
\text{s.t.} \quad & C(x_k) + \nabla C(x_k)^\top d \leq \kappa_k e, \\
& \|d\|_\infty \leq \beta,
\end{aligned}
\qquad (7)
\qquad\qquad
\begin{aligned}
\min_d \quad & \overline{\nabla}^J f(x_k)^\top d + \tfrac{\tau}{2}\|d\|^2, \\
\text{s.t.} \quad & \overline{c}^J(x_k) + \overline{\nabla}^J c(x_k)^\top d \leq \overline{\kappa_k}^J e, \\
& \|d\|_\infty \leq \beta.
\end{aligned}
\qquad (8)
$$

In the stochastic setting, an unbiased estimate $d(x_k)$ of the line search direction d is needed and it is computed using four particular mini-batches as follows. To facilitate comprehension, we denote $X_k^J = \{X_{k,j}\}_{j=1}^J$ a mini-batch of size J with the j-th element $X_{k,j} = (\nabla f(x_k, \xi_{k,j}), c(x_k, \zeta_{k,j}), \nabla c(x_k, \zeta_{k,j}))$. First, we sample a random variable $N \sim \mathcal{G}(p_0)$ from the geometric distribution. Then we sample the mini-batches X_k^1 and $X_k^{2^{N+1}}$ and we partition the mini-batch $X_k^{2^{N+1}}$ of size 2^{N+1} into two mini-batches $\mathrm{odd}(X_k^{2^{N+1}})$ and $\mathrm{even}(X_k^{2^{N+1}})$ of size 2^N. Finally, we solve (8) for each of the four mini-batches, denoting by $d(x_k; X_k^J)$ the solution of (8) for the corresponding mini-batch X_k^J. We obtain

$$d(x_k) = \frac{d(x_k; X_k^{2^{N+1}}) - \frac{1}{2}\left(d(x_k; \mathrm{odd}(X_k^{2^{N+1}})) + d(x_k; \mathrm{even}(X_k^{2^{N+1}}))\right)}{(1-p_0)^N p_0} + d(x_k; X_k^1). \quad (9)$$

An update between the iterations x_k and x_{k+1} is computed as

$$x_{k+1} = x_k + \alpha_k d(x_k),$$

where the deterministic stepsize α_k fulfills the classical requirement to be square-summable $\sum_{k=1}^{\infty} (\alpha_k)^2 < \infty$ but not summable $\sum_{k=1}^{\infty} \alpha_k = \infty$. For more details, see Algorithm 1.

3.3 Stochastic Smoothed and Linearized AL Method (SSL-ALM)

The Stochastic Smoothed and Linearized AL Method (SSL-ALM) was described in [32] for optimization problems with stochastic linear constraints. Although problem (1) has non-linear inequality constraints, we use the SSL-ALM due to the lack of algorithms in the literature dealing with stochastic non-linear constraints; see Table 2. The transition between equality and inequality constraints is handled with slack variables. Following the structure of [32], we minimize over the set $\mathcal{X} = \mathbb{R}^n \times \mathbb{R}^m_{\geq 0}$. The method is based on the augmented Lagrangian (AL) function $L_\rho(x, y) = F(x) + y^\top C(x) + \frac{\rho}{2}\|C(x)\|^2$, which is a result of merging the Lagrange function with the penalty methods [7]. Adding a smoothing term yields the proximal AL function

$$K_{\rho,\mu}(x, y, z) = L_\rho(x, y) + \frac{\mu}{2}\|x - z\|^2.$$

The SSL-ALM method was originally proposed in [32] where it is interpreted as an inexact gradient descent step on the Moreau envelope. An important property of the Moreau envelope is that its stationary points coincide with those of the original function.

The strength of this method is that, as opposed to the Stochastic Ghost method, it does not use large mini-batch sizes. In each iteration, we sample $\xi \overset{iid}{\sim} \mathcal{P}_\xi$ to evaluate the objective and $\zeta_1, \zeta_2 \overset{iid}{\sim} \mathcal{P}_\zeta$ to evaluate the constraint function and its Jacobian matrix, respectively. The function

$$G(x, y, z; \xi, \zeta_1, \zeta_2) = \nabla f(x, \xi) + \nabla c(x, \zeta_1)^\top y + \rho \nabla c(x, \zeta_1)^\top c(x, \zeta_2) + \mu(x - z) \quad (10)$$

is defined so that, in iteration k, $\mathbb{E}_{\xi,\zeta_1,\zeta_2}[G(x_k, y_{k+1}, z_k; \xi, \zeta_1, \zeta_2)] = \nabla K_{\rho,\mu}(x_k, y_{k+1}, z_k)$. Omitting some details, the updates are performed using some parameters η, τ, and β as follows:

$$\begin{aligned} y_{k+1} &= y_k + \eta c(x, \zeta_1), \\ x_{k+1} &= \mathrm{proj}_{\mathcal{X}}(x_k - \tau G(x_k, y_{k+1}, z_k; \xi, \zeta_1, \zeta_2)), \\ z_{k+1} &= z_k + \beta(x_k - z_k). \end{aligned} \quad (11)$$

For more details, see Algorithm 2.

3.4 Stochastic Switching Subgradient Method (SSw)

The Stochastic Switching Subgradient method was described in [33] for optimization problems over a closed convex set $\mathcal{X} \subset \mathbb{R}^d$ which is easy to project on and for weakly convex objective and constraint functions F and C which may be non-smooth. This is why the notion of gradient of F and C is replaced by a more general notion of subgradient, which is an element of a subdifferential.

The algorithm requires as input a prescribed sequence of infeasibility tolerances ϵ_k and sequences of stepsizes η_k^f and η_k^c. In iteration k, we sample $\zeta_1, \ldots, \zeta_J \overset{iid}{\sim} \mathcal{P}_\zeta$ to compute an estimate $\overline{c}^J(x_k)$. If $\overline{c}^J(x_k)$ is smaller than ϵ_k, we sample $\xi \overset{iid}{\sim} \mathcal{P}_\xi$ and an update between x_k and x_{k+1} is computed using a stochastic estimate $S^f(x_k, \xi)$ of an element of the subdifferential $\partial F(x_k)$ of the objective function:

$$x_{k+1} = \mathrm{proj}_{\mathcal{X}}(x_k - \eta_k^f S^f(x_k, \xi)).$$

Otherwise, we sample $\zeta \overset{iid}{\sim} \mathcal{P}_\zeta$ and the update is computed using a stochastic estimate $S^c(x_k, \zeta)$ of an element of the subdifferential $\partial C(x_k)$ of the constraint function:

$$x_{k+1} = \mathrm{proj}_{\mathcal{X}}(x_k - \eta_k^c S^c(x_k, \zeta)).$$

In either case, the updates are only saved starting from a prescribed index k_0 and the final output is sampled randomly from the saved updates. For more details, see Algorithm 3. The algorithm presented here is slightly more general than the one presented in [33]: we allow for the possibility of different stepsizes for the objective update, η_k^f, and the constraint update η_k^c, while the original method employs equal stepsizes $\eta_k^f = \eta_k^c$.

4 Experimental evaluation

In this section, we illustrate the presented algorithms on a real-world instance of the ACS dataset, comparing how they fare with optimization and fairness metrics.

4.1 Dataset for fair ML

[22] proposed a large-scale dataset for fair Machine Learning, based on the ACS PUMS data sample (American Community Survey Public Use Microdata Sample). The ACS survey is sent annually to approximately 3.5 million US households in order to gather information on features such as ancestry, citizenship, education, employment, or income. Therefore, it has the potential to give rise to large-scale learning and optimization problems.

We use the ACSIncome dataset over the state of Oklahoma, and choose the binary classification task of predicting whether an individual's income is over \$50,000. The dataset contains 9 features and 17,917 data points, and may be accessed via the Python package Folktables. We choose race (**RAC1P**) as the protected attribute. In the original dataset, it is a categorical variable with 9 values. For the purposes of this experiment, we binarize it to obtain the non-protected group of "white" people and the protected group of "non-white" people. The dataset is split randomly into train (80%, 14,333 points) and test (20%, 3,584 points) subsets and it is stratified with respect to the protected attribute, i.e., the proportion of "white" and "non-white" samples in the training and test sets is equivalent to that in the full dataset (30.8% of positive labels in group "white", 20.7% in the group "non-white"). The protected attributes are then removed from the data so that the model cannot learn from them directly. The data is normalized using Scikit-Learn StandardScaler.

Note that ACSIncome is a real-world dataset for which ERM-based predictors without fairness safeguards are known to learn biases [30]. Accordingly, Table 3 (line 1) shows that an ERM predictor without fairness safeguards has poor fairness metrics; see also Figure 4.

4.2 Experiments

Numerical setup. Experiments are conducted on an Asus Zenbook UX535 laptop with AMD Ryzen 7 5800H CPU, and 16GB RAM. The code is written in Python with the PyTorch package [44].

Problems. We consider the constrained ERM problem (4) – $\mathcal{R} = 0$, and, as baselines, the ERM problem (2) without any regularization, $\mathcal{R} = 0$, and with a fairness inducing regularizer $\mathcal{R}$ that promotes small difference in accuracy between groups, provided by the Fairret library [11]. In all problems, we take as loss function the Binary Cross Entropy with Logits Loss

$$\ell(f_\theta(X_i), Y_i) = -Y_i \cdot \log \sigma(f_\theta(X_i)) - (1 - Y_i) \cdot \log(1 - \sigma(f_\theta(X_i))), \tag{12}$$

where $\sigma(z) = \frac{1}{1+e^{-z}}$ is the sigmoid function, and the prediction function f_θ is a neural network with 2 interconnected hidden layers of sizes 64 and 32 and ReLU activation, with a total of 194 parameters.

Algorithms and parameters. We assess the performance of four algorithms for solving the constrained problem (4): *(1)* Stochastic Ghost (StGh) (Sec. 3.2 - parameters $p_0 = 0.4$, $\alpha_0 = 0.05$, $\rho = 0.8$, $\tau = 1$, $\beta = 10$, $\lambda = 0.5$, $\hat{\alpha} = 0.05$), *(2)* SSL-ALM (Sec. 3.3 - parameters $\mu = 2.0$, $\rho = 1.0$, $\tau = 0.01$, $\eta = 0.05$, $\beta = 0.5$, $M_y = 10$), *(3)* plain Augmented Lagrangian Method ALM (Sec. 3.3, smoothing term removed $\mu = 0$, otherwise the same setting as SSL-ALM), and *(4)* Stochastic Switching Subgradient (SSw) (Sec. 3.4 - $\eta_k^f = 0.5$, $\eta_k^c = 0.05$, $\epsilon_k = 10^{-4}$ if $k < 500$, $\epsilon_k = 0.97\epsilon_{k-1}$ for every $k \geq 500$ at each epoch). We also provide the behavior of SGD for solving the ERM problem, both with no fairness safeguards (SGD), and with fairness regularization provided by the Fairret library [11] (SGD-Fairret). These methods serve as baselines. When estimating the constraints, we sample an equal number of data points for every subgroup.

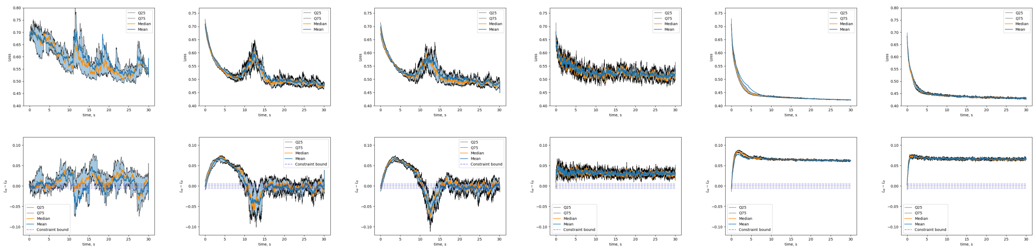


Figure 1: Train loss and constraint values (first and second row) over time (s) on the ACS Income dataset for each algorithm. From left to right: StGh, SSL-ALM, ALM, SSw, SGD, SGD-Fairret.

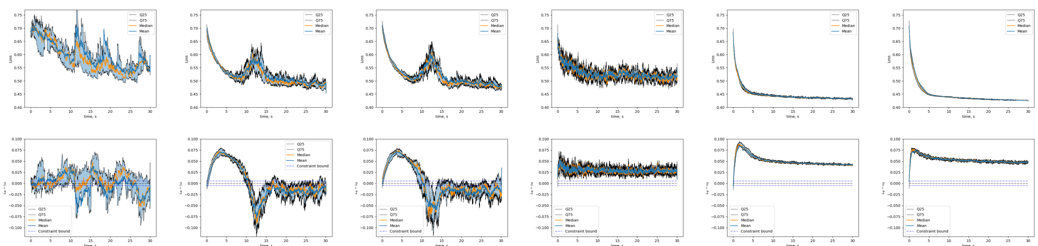


Figure 2: Test loss and constraint (first and second row) values over time (s) on the ACS Income dataset for each algorithm. From left to right: StGh, SSL-ALM, ALM, SSw, SGD, SGD-Fairret.

Optimization performance. Figures 1 and 2 present the evolution of loss and constraint values over the train and test datasets for the four algorithms addressing the constrained problem (columns 1–4), as well as for the two baselines: SGD without fairness (col. 5), and SGD with fairness regularization (col. 6). Each algorithm is run 10 times, and the plots display the mean, median, and quartiles values.

To a certain extent, the four algorithms (col. 1–4) succeed in minimizing the loss and satisfying the constraints on the train set. The AL-based methods (col. 2 and 3) demonstrate a better behavior compared to StGh and SSw; indeed, StGh exhibits higher variability in both loss and constraint values (col. 1), while SSw fails to satisfy the constraints within the required bounds (col. 4). We were unable to identify parameter settings for SSw that simultaneously satisfy the constraints and minimize the objective function. Appendix B provides the behavior of SSw with equal objective and constraint and stepsizes; the constraints are satisfied well, but the objective function is barely minimized. The ERM baselines (col. 5 and 6) exhibit lower variability in the trajectories, and minimize the loss in less time, but as expected, they do not satisfy the constraints.

The ALM and SSL-ALM schemes are the closest to satisfying the constraints on the train set. On the test set, however, they are slightly biased towards negative values. Such bias is expected on unseen data and reflects the generalization behavior of fairness-constrained estimators. This is beyond the scope of the current work; see e.g. [12].

Fairness performance. Figure 3 presents the distribution of predictions over both groups. The distribution of prediction without fairness guarantees (col. 5) clearly does not meet the group fairness standard. Indeed, the "non-white" group has a significantly higher likelihood than the "white" group of receiving small predicted values, and the converse holds for large predicted values. The SGD-Fairret model (col. 6) lies between the four constrained models and SGD. Among the fairness-constrained models, the ALM and SSL-ALM distributions are the closest to the distributions of SGD without fairness, which is consistent with retaining good prediction information. The four models that approximately solve the fairness formulation (col. 1–4) all have closer distributions across groups. Numerically, this is expressed in Table 3 (col. Wd), which reports the value of the Wasserstein distance between group distributions for each model.

Table 3 displays the fairness metrics presented in Section 2: independence (Ind), separation (Sp), and sufficiency (Sf), along with inaccuracy (Ina). The mean value and standard deviation over 10 runs are presented for the four fairness-constrained models and the two baselines, both on train and test sets. Figure 4 presents the mean values as spider plots. For all metrics, smaller is better.

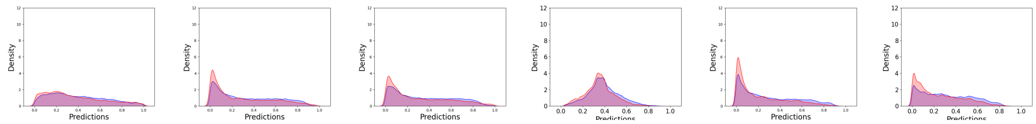
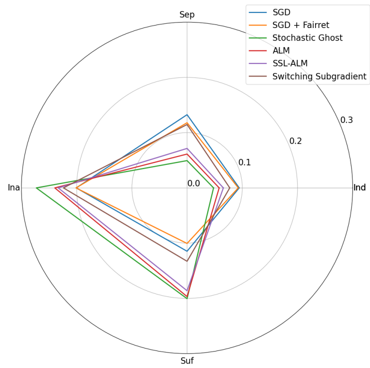
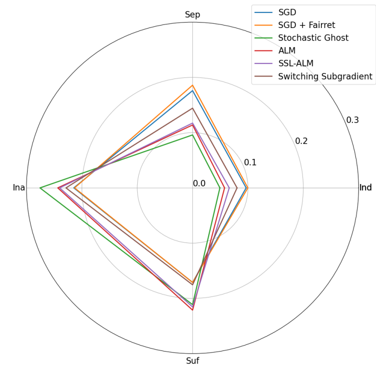


Figure 3: Distribution of predictions for each algorithm. Left to right: StGh, SSL-ALM, ALM, SSw, SGD, SGD-Fairret. Blue and red denote “white” and “non-white” groups.



(a) Train set



(b) Test set

Figure 4: Average value of the three fairness metrics (independence (Ind), separation (Sp), and sufficiency (Sf)), along with inaccuracy (Ina). For all metrics, smaller values are better.

Table 3: Fairness metrics (independence, separation, sufficiency), inaccuracy, and Wasserstein distances between groups (Wd) for the four constrained estimators and the two baselines.

	Train					Test				
Algname	Ind	Sp	Ina	Sf	Wd	Ind	Sp	Ina	Sf	Wd
SGD	0,094±0,004	0,132±0,007	**0,201±0,001**	0,115±0,006	0,008±0,000	0,097±0,006	0,176±0,016	0,215±0,002	**0,171±0,009**	0,008±0,000
StGh	**0,048±0,026**	**0,049±0,028**	0,273±0,024	0,200±0,038	0,002±0,001	**0,049±0,029**	**0,096±0,039**	0,276±0,022	0,211±0,033	0,003±0,002
ALM	0,058±0,007	0,061±0,016	0,240±0,012	0,197±0,011	0,003±0,000	0,058±0,012	0,114±0,014	0,244±0,007	0,221±0,017	0,003±0,001
SSL-ALM	0,066±0,009	0,071±0,015	0,233±0,017	0,186±0,013	0,003±0,001	0,066±0,011	0,117±0,023	0,240±0,012	0,215±0,022	0,004±0,001
SSw	0,077±0,029	0,115±0,029	0,224±0,017	0,133±0,015	**0,001±0,001**	0,080±0,029	0,144±0,050	0,229±0,013	0,175±0,031	**0,002±0,001**
SGD-Fairret	0,091±0,012	0,121±0,017	**0,201±0,002**	**0,106±0,010**	0,005±0,001	0,094±0,010	0,174±0,019	**0,213±0,002**	0,180±0,022	0,006±0,001

Among the four fairness-constrained models, StGh performs best in terms of independence and separation, but worst in terms of accuracy. SSw achieves fairness and accuracy metrics that have intermediate values relative to those of the unconstrained SGD model, and those of the other constrained models. This is consistent with the observation that the optimization method, with our choice of parameters, favored minimizing the objective over satisfying the constraints. The ALM and SSL-ALM methods provide the best compromise: they improve independence and separation relative to the SGD model, while moderately degrading accuracy. SGD-Fairret slightly improves sufficiency relative to the SGD model. The four models constrained in the difference of loss between subgroups have higher values of sufficiency. Similar observations hold for metrics on the test set.

5 Conclusion

To the best of our knowledge, this paper provides the first benchmark for assessing the performance of optimization methods on real-world instances of fairness constrained training of models. We highlight the challenges of this approach, namely that objective and constraints are non-convex, non-smooth, and large-scale, and review the performance of four practical algorithms.

Limitations Our work identifies that there is currently no algorithm with guarantees for solving the fairness constrained problem. Above all, we hope that this work, along with the Python toolbox for easy benchmarking of new optimization methods, will stimulate further interest in this topic. Also, we caution readers that the method present here is not a silver-bullet that handles all biases and ethical issues of training ML models. In particular, care must be taken that fair ML is part of a interdisciplinary pipeline that integrates the specifics of the use-case, and that it does not serve as an excuse for pursuing Business-As-Usual policies that fail to tackle ethical issues [2, 52].

References

[1] Tameem Adel, Isabel Valera, Zoubin Ghahramani, and Adrian Weller. One-network adversarial fairness. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 2412–2420, 2019.

[2] Agathe Balayn, Mireia Yurrita, Jie Yang, and Ujwal Gadiraju. "✓ Fairness Toolkits, A Checkbox Culture?" On the Factors that Fragment Developer Practices in Handling Algorithmic Harms. In *Proceedings of the 2023 AAAI/ACM Conference on AI, Ethics, and Society*, AIES '23, pages 482–495, New York, NY, USA, August 2023. Association for Computing Machinery.

[3] Solon Barocas, Moritz Hardt, and Arvind Narayanan. *Fairness and Machine Learning: Limitations and Opportunities*. MIT Press, 2023.

[4] Rachel K. E. Bellamy, Kuntal Dey, Michael Hind, Samuel C. Hoffman, Stephanie Houde, Kalapriya Kannan, Pranay Lohia, Jacquelyn Martino, Sameep Mehta, Aleksandra Mojsilovic, Seema Nagar, Karthikeyan Natesan Ramamurthy, John Richards, Diptikalyan Saha, Prasanna Sattigeri, Moninder Singh, Kush R. Varshney, and Yunfeng Zhang. AI Fairness 360: An Extensible Toolkit for Detecting, Understanding, and Mitigating Unwanted Algorithmic Bias, October 2018.

[5] Albert Berahas, Frank E. Curtis, Daniel Robinson, and Baoyu Zhou. Sequential quadratic optimization for nonlinear equality constrained stochastic optimization. *SIAM Journal on Optimization*, 31:1352–1379, 05 2021.

[6] Albert S. Berahas, Frank E. Curtis, Michael J. O'Neill, and Daniel P. Robinson. A stochastic sequential quadratic optimization algorithm for nonlinear equality constrained optimization with rank-deficient jacobians, 2023.

[7] D.P. Bertsekas and W. Rheinboldt. *Constrained Optimization and Lagrange Multiplier Methods*. Computer science and applied mathematics. Academic Press, 2014.

[8] Sarah Bird, Miro Dudík, Richard Edgar, Brandon Horn, Roman Lutz, Vanessa Milan, Mehrnoosh Sameki, Hanna Wallach, and Kathleen Walker. Fairlearn: A toolkit for assessing and improving fairness in AI. Technical Report MSR-TR-2020-32, Microsoft, May 2020.

[9] José H. Blanchet, Peter W. Glynn, and Yanan Pei. Unbiased multilevel monte carlo: Stochastic optimization, steady-state simulation, quantiles, and other applications. *arXiv: Statistics Theory*, 2019.

[10] Raghu Bollapragada, Cem Karamanli, Brendan Keith, Boyan Lazarov, Socratis Petrides, and Jingyi Wang. An adaptive sampling augmented lagrangian method for stochastic optimization with deterministic constraints. *Computers and Mathematics with Applications*, 149:239–258, 2023.

[11] Maarten Buyl, Marybeth Defrance, and Tijl De Bie. fairret: a framework for differentiable fairness regularization terms. In *International Conference on Learning Representations*, 2024.

[12] Luiz FO Chamon, Santiago Paternain, Miguel Calvo-Fullana, and Alejandro Ribeiro. Constrained learning with non-convex losses. *IEEE Transactions on Information Theory*, 69(3):1739–1760, 2022.

[13] Changan Chen, Frederick Tung, Naveen Vedula, and Greg Mori. Constraint-aware deep neural network compression. In *Computer Vision – ECCV 2018: 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part VIII*, page 409–424, Berlin, Heidelberg, 2018. Springer-Verlag.

[14] Richard J. Chen, Judy J. Wang, Drew F. K. Williamson, Tiffany Y. Chen, Jana Lipkova, Ming Y. Lu, Sharifa Sahai, and Faisal Mahmood. Algorithmic fairness in artificial intelligence for medicine and healthcare. *Nature Biomedical Engineering*, 7(6):719–742, Jun 2023.

[15] Zhenpeng Chen, Jie M. Zhang, Max Hort, Mark Harman, and Federica Sarro. Fairness testing: A comprehensive survey and analysis of trends. *ACM Trans. Softw. Eng. Methodol.*, 33(5), June 2024.

[16] Andrew Cotter, Heinrich Jiang, Serena Wang, Taman Narayan, Seungil You, Karthik Sridharan, and Maya R. Gupta. Optimization with non-differentiable constraints with applications to fairness, recall, churn, and other goals. *Journal of Machine Learning Research*, 20(172):1–59, 2019.

[17] Frank E. Curtis, Michael J. O'Neill, and Daniel P. Robinson. Worst-case complexity of an sqp method for nonlinear equality constrained stochastic optimization. *Mathematical Programming*, 205(1):431–483, May 2024.

[18] Frank E. Curtis, Daniel P. Robinson, and Baoyu Zhou. Sequential quadratic optimization for stochastic optimization with deterministic nonlinear inequality and equality constraints. *SIAM Journal on Optimization*, 34(4):3592–3622, 2024.

[19] Damek Davis, Dmitriy Drusvyatskiy, Sham Kakade, and Jason D. Lee. Stochastic subgradient method converges on tame functions, 2018.

[20] MaryBeth Defrance, Maarten Buyl, and Tijl De Bie. Abcfair: an adaptable benchmark approach for comparing fairness methods, 2024.

[21] Eoin Delaney, Zihao Fu, Sandra Wachter, Brent Mittelstadt, and Chris Russell. OxonFair: A Flexible Toolkit for Algorithmic Fairness, November 2024.

[22] Frances Ding, Moritz Hardt, John Miller, and Ludwig Schmidt. Retiring adult: New datasets for fair machine learning. *Advances in Neural Information Processing Systems*, 34, 2021.

[23] Mengnan Du, Subhabrata Mukherjee, Guanchu Wang, Ruixiang Tang, Ahmed Hassan Awadallah, and Xia Hu. Fairness via representation neutralization. In *Neurips*, 2021.

[24] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, ITCS '12, page 214–226, New York, NY, USA, 2012. Association for Computing Machinery.

[25] Harrison Edwards and Amos Storkey. Censoring representations with an adversary, 2016.

[26] Alessandro Fabris, Stefano Messina, Gianmaria Silvello, and Gian Antonio Susto. Algorithmic fairness datasets: the story so far. *Data Mining and Knowledge Discovery*, 36(6):2074–2152, Nov 2022.

[27] Francisco Facchinei and Vyacheslav Kungurtsev. Stochastic approximation for expectation objective and expectation inequality-constrained nonconvex optimization, 2023.

[28] Francisco Facchinei, Vyacheslav Kungurtsev, Lorenzo Lampariello, and Gesualdo Scutari. Ghost penalties in nonconvex constrained optimization: Diminishing stepsizes and iteration complexity. *Mathematics of Operations Research*, 46(2):595–627, 2021.

[29] Yuchen Fang, Sen Na, Michael W. Mahoney, and Mladen Kolar. Fully stochastic trust-region sequential quadratic programming for equality-constrained optimization problems. *SIAM Journal on Optimization*, 34(2):2007–2037, 2024.

[30] Xiaotian Han, Jianfeng Chi, Yu Chen, Qifan Wang, Han Zhao, Na Zou, and Xia Hu. FFB: A Fair Fairness Benchmark for In-Processing Group Fairness Methods. In *The Twelfth International Conference on Learning Representations*, October 2023.

[31] Moritz Hardt, Eric Price, and Nathan Srebro. Equality of opportunity in supervised learning. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 3323–3331, Red Hook, NY, USA, 2016. Curran Associates Inc.

[32] Ruichuan Huang, Jiawei Zhang, and Ahmet Alacaoglu. Stochastic smoothed primal-dual algorithms for nonconvex optimization with linear inequality constraints, 2025.

[33] Yankun Huang and Qihang Lin. Oracle complexity of single-loop switching subgradient methods for non-smooth weakly convex functional constrained optimization, 2023.

[34] Toshihiro Kamishima, Shotaro Akaho, Hideki Asoh, and Jun Sakuma. Fairness-Aware Classifier with Prejudice Remover Regularizer. In Peter A. Flach, Tijl De Bie, and Nello Cristianini, editors, *Machine Learning and Knowledge Discovery in Databases*, pages 35–50, Berlin, Heidelberg, 2012. Springer.

[35] Michael P. Kim, Amirata Ghorbani, and James Zou. Multiaccuracy: Black-box post-processing for fairness in classification. In *Proceedings of the 2019 AAAI/ACM Conference on AI, Ethics, and Society*, AIES '19, page 247–254, New York, NY, USA, 2019. Association for Computing Machinery.

[36] Tai Le Quy, Arjun Roy, Vasileios Iosifidis, Wenbin Zhang, and Eirini Ntoutsi. A survey on datasets for fairness-aware machine learning. *WIREs Data Mining and Knowledge Discovery*, 12(3), 03 2022.

[37] Zhu Li, Adrián Pérez-Suay, Gustau Camps-Valls, and Dino Sejdinovic. Kernel dependence regularizers and Gaussian processes with applications to algorithmic fairness. *Pattern Recognition*, 132:108922, December 2022.

[38] Gilles Louppe, Michael Kagan, and Kyle Cranmer. Learning to pivot with adversarial networks. *Advances in neural information processing systems*, 30, 2017.

[39] David Madras, Elliot Creager, Toniann Pitassi, and Richard Zemel. Learning adversarially fair and transferable representations. In *International Conference on Machine Learning*, pages 3384–3393. PMLR, 2018.

[40] Sen Na, Mihai Anitescu, and Mladen Kolar. An adaptive stochastic sequential quadratic programming with differentiable exact augmented lagrangians. *Mathematical Programming*, 199(1):721–791, May 2023.

[41] Sen Na, Mihai Anitescu, and Mladen Kolar. Inequality constrained stochastic nonlinear optimization via active-set sequential quadratic programming, 2023.

[42] Yatin Nandwani, Abhishek Pathak, Mausam, and Parag Singla. A primal dual formulation for deep learning with constraints. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[43] Figen Oztoprak, Richard Byrd, and Jorge Nocedal. Constrained optimization in the presence of noise. *SIAM Journal on Optimization*, 33(3):2118–2136, 2023.

[44] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[45] Patricia Pauli, Anne Koch, Julian Berberich, Paul Kohler, and Frank Allgöwer. Training robust neural networks using lipschitz bounds. *IEEE Control Systems Letters*, 6:121–126, 2021.

[46] Dana Pessach and Erez Shmueli. A review on fairness in machine learning. *ACM Comput. Surv.*, 55(3), February 2022.

[47] Evaggelia Pitoura, Kostas Stefanidis, and Georgia Koutrika. Fairness in rankings and recommendations: an overview. *The VLDB Journal*, 31(3):431–458, May 2022.

[48] Sathya N. Ravi, Tuan Dinh, Vishnu Suresh Lokhande, and Vikas 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, Jul. 2019.

[49] Qiankun Shi, Xiao Wang, and Hao Wang. A momentum-based linearized augmented lagrangian method for nonconvex constrained stochastic optimization. *Optimization Online*, 2022.

[50] Amal Tawakuli and Thomas Engel. Make your data fair: A survey of data preprocessing techniques that address biases in data towards fair ai. *Journal of Engineering Research*, 2024.

[51] Sahil Verma and Julia Rubin. Fairness definitions explained. In *Proceedings of the International Workshop on Software Fairness*, FairWare '18, page 1–7, New York, NY, USA, 2018. Association for Computing Machinery.

[52] Sandra Wachter, Brent Mittelstadt, and Chris Russell. Bias Preservation in Machine Learning: The Legality of Fairness Metrics Under EU Non-Discrimination Law, January 2021.

[53] Mingyang Wan, Daochen Zha, Ninghao Liu, and Na Zou. In-processing modeling techniques for machine learning fairness: A survey. *ACM Trans. Knowl. Discov. Data*, 17(3), March 2023.

[54] Quan Zhou, Jakub Mareček, and Robert Shorten. Subgroup fairness in two-sided markets. *Plos one*, 18(2):e0281443, 2023.

[55] Quan Zhou, Jakub Mareček, and Robert Shorten. Fairness in forecasting of observations of linear dynamical systems. *Journal of Artificial Intelligence Research*, 76:1247–1280, April 2023.

A Algorithms in more detail

In this section, we provide the pseudocodes of algorithms presented in Section 3 as Algorithms 1 to 3. Recall that we denote by $X_k^J = \{X_{k,j}\}_{j=1}^J$ a mini-batch of size J with the j-th element

$$X_{k,j} = (\nabla f(x_k, \xi_{k,j}), c(x_k, \zeta_{k,j}), \nabla c(x_k, \zeta_{k,j})). \tag{13}$$

B Additional experiments on SSw

This Section provides additional information on the behavior of SSw. Figure 5 shows the evolution of the objective value and constraints for 10 runs of the SSw algorithm, over train and test, with equal objective and constraint stepsizes $\eta_k^f = \eta_k^c = 0.02$. In that case, the constraints are satisfied well, but the objective function is barely minimized.

Algorithm 1 Stochastic Ghost algorithm

Require: Training dataset $\mathcal{D}$, constraint dataset $\mathcal{C}$, initial neural network weights x_0
Require: Parameters $p_0 \in (0, 1)$, α_0, $\hat{\alpha}$, ρ, τ, β

for Iteration $k = 0$ **to** $K - 1$ **do**
  Sample $\xi \overset{iid}{\sim} \mathcal{P}_\xi$ and $\zeta \overset{iid}{\sim} \mathcal{P}_\zeta$
  Sample $N \sim \mathcal{G}(p_0)$
  Set $J = 2^{N+1}$
  Sample a mini-batch $\{\zeta_j\}_{j=1}^J$ so that $\zeta_1, \ldots, \zeta_J \overset{iid}{\sim} \mathcal{P}_\zeta$
  Sample a mini-batch $\{\xi_j\}_{j=1}^J$ so that $\xi_1, \ldots, \xi_J \overset{iid}{\sim} \mathcal{P}_\xi$
  Set X_k^1 and $X_k^{2^{N+1}}$ using (13)
  Compute $d(x_k)$ from (9)
  Set $\alpha_k = \alpha_{k-1}(1 - \hat{\alpha}\alpha_{k-1})$
  Update $x_{k+1} = x_k + \alpha_k d(x_k)$
end for

Algorithm 2 Stochastic Smoothed and Linearized AL Method for solving (1)

Require: Training dataset $\mathcal{D}$, constraint dataset $\mathcal{C}$, initial neural network weights x_0
Require: Parameters μ, η, $M_y > 0$, τ, β, $\rho \geq 0$

for Iteration $k = 0$ **to** $K - 1$ **do**
  Sample $\xi \overset{iid}{\sim} \mathcal{P}_\xi$ and $\zeta_1, \zeta_2 \overset{iid}{\sim} \mathcal{P}_\zeta$
  $y_{k+1} = y_k + \eta c(x, \zeta_1)$
  if $||y_{k+1}|| \geq M_y$ **then**
    $y_{k+1} = 0$
  end if
  $x_{k+1} = \text{proj}_{\mathcal{X}}(x_k - \tau G(x_k, y_{k+1}, z_k; \xi, \zeta_1, \zeta_2))$, where G is defined in (10)
  $z_{k+1} = z_k + \beta(x_k - z_k)$
end for

Algorithm 3 Stochastic Switching Subgradient Method

Require: Training dataset $\mathcal{D}$, constraint dataset $\mathcal{C}$, initial neural network weights $x_0 \in \mathcal{X}$
Require: Total number of iterations K, sequence of tolerances of infeasibility $\epsilon_k \geq 0$, sequences of stepsizes η_k^f and η_k^c, mini-batch size J, starting index k_0 for recording outputs, $I = \emptyset$

for Iteration $k = 0$ **to** $K - 1$ **do**
  Sample a mini-batch $\{\zeta_j\}_{j=1}^J$ so that $\zeta_1, \ldots, \zeta_J \overset{iid}{\sim} \mathcal{P}_\zeta$
  Set $\overline{c}^J(x_k) = \frac{1}{J}\sum_{j=1}^J c(x_k, \zeta_j)$
  if $\overline{c}^J(x_k) \leq \epsilon_k$ **then**
    Sample $\xi \overset{iid}{\sim} \mathcal{P}_\xi$ and generate $S^f(x_k, \xi)$
    Set $x_{k+1} = \text{proj}_{\mathcal{X}}(x_k - \eta_k^f S^f(x_k, \xi))$ and, if $k \geq k_0$, $I = I \cup \{k\}$
  else
    Sample $\zeta \overset{iid}{\sim} \mathcal{P}_\zeta$ and generate $S^c(x_k, \zeta)$
    Set $x_{k+1} = \text{proj}_{\mathcal{X}}(x_k - \eta_k^c S^c(x_k, \zeta))$ and, if $k \geq k_0$, $I = I \cup \{k\}$
  end if
end for
Output: x_τ with τ randomly sampled from I using $P(\tau = k) = \frac{\eta_k}{\sum_{s \in I} \eta_s}$.

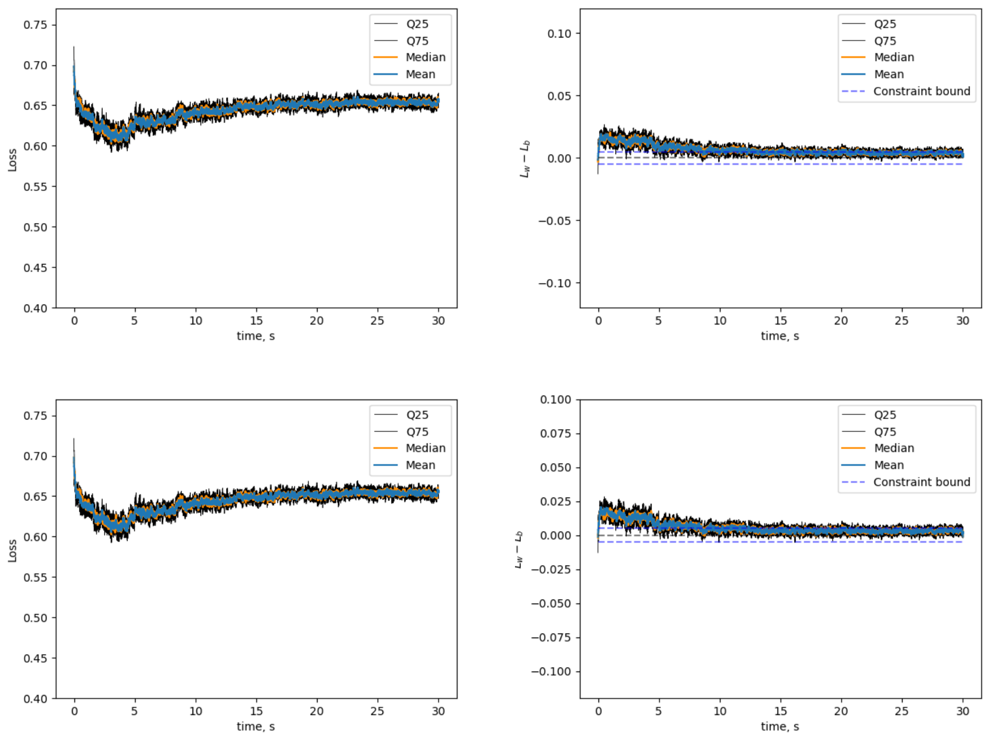


Figure 5: Loss and constraint values over time (s) on the train and test set (first and second row) on the ACS Income dataset for the SSw algorithm.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims of the paper are summarized in the "contributions" section of the introduction; the Section "paper organization" points to which Section supports which claim in the paper.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss limitations of this work in a dedicated paragraph of the Conclusion section.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: The paper contains no theoretical results.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Code to reproduce the experiments is available at `github.com/humancompatible/train`. This repository includes a readme file with instructions to reproduce experiments in the exact same software environment. In addition, the Experimental Section 4 details hardware specifications, explains the details of the implementation, and the instructions to obtain and run the code.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: As mentioned in the abstract and the Numerical Section 4, the code and instructions to reproduce the experiments are available on Github at `https://github.com/humancompatible/train` We use the publicly available dataset Folktables; the dataset is fetched automatically as part of our code.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.

- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: A description of the setup used to produce experiments is provided at the top of Section 4.2. The same applies for the figures of the Appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: All the plots, except the spider plots (Fig. 4), show mean, median, first and third quartiles. All the tables report mean and standard deviation. The spider plots (Fig. 4) provide a visual representation of the mean data in Table 3. Standard deviation are not displayed in Fig. 4, but they are reported in Table 3.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer ”Yes” if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: Again, we detail the hardware specifications (laptop model, CPU and RAM details) at the top of Section 4.2. Time to run the experiments is reported directly in the Figures.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics https://neurips.cc/public/EthicsGuidelines?

Answer: [Yes]

Justification: We reviewed the NeurIPS Code of Ethics, and found that none of the problematic cases in it conform with the numerical experiments.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: We mention broader impacts in the limitations paragraph of the Conclusion.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: The paper does not release any data or models.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: The paper uses the ACS dataset, and the torch library; both are cited in the paper. The README of the code also provides a link to the license of the dataset.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: The paper introduces code. Details on how to use and extend the code are provided in the readme.md file, and each code file is commented.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.

- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: The paper does not conduct crowdsourcing and research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: Again, the paper does not conduct research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: The core method development in this research does not involve LLMs, nor does the writing of the paper.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.