Optimization for Data Science

Zeroth Order optimization for (Black box) Adversarial Attacks

Giacomo Virginio, Mikhail Kolobov

Department of Mathematics, Università degli Studi di Padova

September 18, 2023

Abstract

In this project, our primary objective is to analyze three gradient-free modifications of the original Frank-Wolfe algorithm, namely FZCGS [1], ZO-SCGS [2] and SGFFW [3]. These algorithms are tailored for constrained stochastic optimization problems and aim to improve iteration complexity, particularly in terms of oracle queries, while also striving to rival their first-order counterparts.

Our investigation encompasses a thorough theoretical examination of these algorithms, delving into their underlying principles and theoretical performance guarantees. Subsequently, we subject these methods to practical tests, evaluating their efficacy in a black-box attack scenario as reported in Section 4.3 [1] of the literature. Through this comprehensive analysis, we aim to gain insights into the advancements and practical applicability of these gradient-free Frank-Wolfe variants in the realm of adversarial attacks and optimization.

1 Introduction

The Frank-Wolfe algorithm, also known as the Conditional Gradient method, finds widespread application in machine learning and deep learning. Its fundamental principle revolves around approximating the objective function by a first-order Taylor approximation. It excels in constrained optimization problems within a closed convex set C:

$$\min_{x \in C} f(x) \tag{1}$$

This formulation encompasses a range of problems, including non-convex and stochastic scenarios. One variant employs a stochastic zeroth-order oracle (SZO) to estimate the loss function $f(x)$:

$$\min_{x \in C} f(x) = \min_{x \in C} \mathbb{E}_y[F(x, y)] \tag{2}$$

Another variant is tailored for finite-sum minimization problems, where component functions $f_i(x)$ are summed:

$$\min_{x \in \Omega} F(x) = \frac{1}{n} \sum_{i=1}^{n} f_i(x) \tag{3}$$

In the context of deep learning and optimization, the Frank-Wolfe algorithm plays a crucial role in solving constrained optimization problems efficiently. One of its key applications is in training deep neural networks with constraints, such as in adversarial attacks, where the objective is to find perturbations that fool a neural network while adhering to certain constraints.

1.1 Deterministic Frank-Wolfe Algorithm

This method is particularly valuable in scenarios where exact first-order information is available, making it a powerful tool in optimization tasks.

The core steps of the deterministic Frank-Wolfe method, when exact first-order information is accessible through an incremental first-order oracle (IFO), can be summarized by the following formulas:

$$v_t = \arg\min_{v \in \mathcal{C}} \langle h, \nabla f(x_t) \rangle \tag{4}$$

$$x_{t+1} = (1 - \gamma_{t+1}) x_t + \gamma_{t+1} v_t, \tag{5}$$

where

x_t - Current iterate

v_t - Direction to minimize the linear approximation

γ_{t+1} - Step size or learning rate $\gamma_{t+1} = \dfrac{2}{t+2}$

$\mathcal{C}$ - Convex set

$f(x_t)$ - Objective function to be minimized

h - Vector in the same space as x_t

$\langle \cdot, \cdot \rangle$ - Inner product

In practice, the exact minimization in the first formula can be replaced by an inexact minimization, where a vector v is chosen to satisfy certain conditions. This relaxation maintains the same convergence rate and can be advantageous in scenarios where exact minimization is computationally expensive.

1.2 Stochastic Frank-Wolfe Algorithm

The Stochastic Frank-Wolfe algorithm is a valuable tool when dealing with scenarios where only stochastic zeroth-order information is available. This variant of the Frank-Wolfe method excels in optimization tasks where exact first-order information is unattainable.

The fundamental steps of the Stochastic Frank-Wolfe algorithm, equipped with a stochastic zeroth-order oracle (SZO), can be expressed in the following way:

$$v_t = \arg\min_{v \in \mathcal{C}} \langle h, \nabla f(x_t, y_t) \rangle \tag{6}$$

$$x_{t+1} = (1 - \gamma_{t+1}) x_t + \gamma_{t+1} v_t, \tag{7}$$

where $f(x_t, y_t)$ is a stochastic objective function depending on x_t and a random variable y_t.

1.3 Zeroth-Order Optimization

Zeroth-order optimization is a powerful approach used when the gradient of a function is not directly available. In scenarios like these, where exact first-order information is elusive, zeroth-order optimization techniques come into play to efficiently explore the function space while minimizing reliance on specific assumptions about the function's mathematical properties. This methodology relies on a combination of sampling, interpolation, and search strategies to guide the optimization process effectively.

One key concept in zeroth-order optimization is the estimation of the gradient or gradient-like information using limited function evaluations. Instead of having access to the gradient itself, these methods estimate it based on function values at selected points.

One well-known method for such gradient estimation is the coordinate-wise gradient estimator, which can be represented as:

$$\hat{\nabla} f(\mathbf{x}) = \sum_{j=1}^{d} \frac{f\left(\mathbf{x} + \mu_j \mathbf{e}_j\right) - f\left(\mathbf{x} - \mu_j \mathbf{e}_j\right)}{2\mu_j} \mathbf{e}_j, \qquad (8)$$

where

$\hat{\nabla} f(\mathbf{x})$ - Estimated gradient of the function f at the point $\mathbf{x}$

d - Dimensionality of the optimization space

$\mu_j > 0$ - Smoothing parameter

$\mathbf{e}_j \in \mathbb{R}^d$ - Basis vector

The algorithms under investigation in this research incorporate various strategies for approximating the gradient using techniques like the coordinate-wise gradient estimator (8). These approaches enable efficient optimization in cases where direct access to gradient information is challenging or impractical.

2 Algorithms

2.1 Zeroth Order Frank-Wolfe Stochastic (SGFFW) Algorithm

This paper presents a stochastic zeroth order Frank-Wolfe algorithm designed to solve the following stochastic optimization problem:

$$\min_{x \in C} f(x) = \min_{x \in C} \mathbb{E}_y[F(x; y)] \qquad (1)$$

where $C \in \mathbb{R}^d$ represents a closed convex set. This problem is of significant interest in both convex and non-convex contexts, particularly in deep learning. Two classes of algorithms, those requiring projection and projection-free methods, are typically employed to solve it. This algorithm focuses on a stochastic version of the projection-free Frank-Wolfe algorithm with access to a zeroth order oracle. This is the first application of stochastic gradient descent in zeroth-order context.

The proposed zeroth order stochastic Frank-Wolfe algorithm uses gradient approximation schemes to address non-smoothness and variance challenges. It employs an averaging trick to stabilize the algorithm, leading to the following updates:

$$\begin{aligned} d_t &= (1 - \rho_t) d_{t-1} + \rho_t g(x_t, y_t) \\ v_t &= \arg\min_{s \in C} \langle s, d_t \rangle \\ x_{t+1} &= (1 - \gamma_{t+1}) x_t + \gamma_{t+1} v_t \end{aligned}$$

Three gradient approximation schemes are utilized, including Kiefer Wolfowitz stochastic approximation (KWSA), random directions (RDSA), and an improvised random directions scheme (I-RDSA):

1. *KWSA*

$$\mathbf{g}\left(\mathbf{x}_t; \mathbf{y}\right) = \sum_{i=1}^{d} \frac{F\left(\mathbf{x}_t + c_t \mathbf{e}_i; \mathbf{y}\right) - F\left(\mathbf{x}_t; \mathbf{y}\right)}{c_t} \mathbf{e}_i$$

2. *RDSA*

Sample $\mathbf{z}_t \sim \mathcal{N}\left(0, \mathbf{I}_d\right)$,

$$\mathbf{g}\left(\mathbf{x}_t; \mathbf{y}, \mathbf{z}_t\right) = \frac{F\left(\mathbf{x}_t + c_t \mathbf{z}_t; \mathbf{y}\right) - F\left(\mathbf{x}_t; \mathbf{y}\right)}{c_t} \mathbf{z}_t$$

3. *I-RDSA*

Sample $\{\mathbf{z}_{i,t}\}_{i=1}^{m} \sim \mathcal{N}\left(0, \mathbf{I}_d\right)$,

$$\mathbf{g}\left(\mathbf{x}_t; \mathbf{y}, \mathbf{z}_t\right) = \frac{1}{m} \sum_{i=1}^{m} \frac{F\left(\mathbf{x}_t + c_t \mathbf{z}_{i,t}; \mathbf{y}\right) - F\left(\mathbf{x}_t; \mathbf{y}\right)}{c_t} \mathbf{z}_{i,t}$$

The choice of gradient approximation affects the algorithm's dimension dependence and computational cost.

The pseudocode of the algorithm is presented below:

Stochastic Gradient Free Frank Wolfe

Require: Input, Loss Function $F(x)$, Convex Set C,
number of directions m, sequences $\gamma_t = 2t + 8$,

$$(\rho_t, c_t)_{RDSA} = \left(\frac{4}{d^{1/3}(t+8)^{2/3}}, \frac{2}{d^{3/2}(t+8)^{1/3}}\right),$$

$$(\rho_t, c_t)_{I-RDSA} = \left(\frac{4}{(1+d/m)^{1/3}(t+8)^{2/3}}, \frac{2\sqrt{m}}{d^{3/2}(t+8)^{1/3}}\right),$$

$$(\rho_t, c_t)_{KWSA} = \left(\frac{4}{(t+8)^{2/3}}, \frac{2}{d^{1/2}(t+8)^{1/3}}\right).$$

Output: x_T or $\frac{1}{T}\sum_{t=0}^{T-1} x_t$.

1 : Initialize $x_0 \in C$

2 : for $t = 0, 2, \ldots, T-1$ do

3 : Compute

$$KWSA : g(x_t; y) = \sum_{i=1}^{d} \frac{F(x_t + c_t e_i; y) - F(x_t; y)}{c_t} e_i$$

$RDSA$: Sample $z_t \sim N(0, I_d)$,

$$g(x_t; y, z_t) = \frac{F(x_t + c_t z_t; y) - F(x_t; y)}{c_t} z_t$$

$I - RDSA$: Sample $\{z_i, t\}_{i=1}^{m} \sim N(0, I_d)$,

$$g(x_t; y, z_t) = \frac{1}{m}\sum_{i=1}^{m} \frac{F(x_t + c_t z_i, t; y) - F(x_t; y)}{c_t} z_{i,t}$$

4 : Compute $d_t = (1 - \rho_t)d_{t-1} + \rho_t g(x_t, y_t)$

5 : Compute $v_t = \arg\min_{s \in C} \langle s, d_t \rangle$,

6 : Compute $x_{t+1} = (1 - \gamma_t)x_t + \gamma_t v_t$.

7 : end for

Primal Gap:

Theorem 3.4 from the paper quantifies the primal suboptimality gap for the algorithm in 2 with different gradient approximation schemes:

1. For the RDSA gradient approximation scheme:

$$E[f(x_t) - f(x^*)] = O\left(d^{1/3}(t+9)^{1/3}\right)$$

2. For the I-RDSA gradient approximation scheme:

$$E[f(x_t) - f(x^*)] = O\left(\frac{d}{m^{1/3}}(t+9)^{1/3}\right)$$

3. For the KWSA gradient approximation scheme:

$$E[f(x_t) - f(x^*)] = O\left(\frac{1}{(t+9)^{1/3}}\right)$$

The dimension dependence of the primal gap is characterized as $d^{1/3}$, and the dependence on iterations follows $O(T^{-1/3})$, matching that of the stochastic Frank-Wolfe with first-order information. The number of queries to the SZO to obtain a primal gap of ϵ is given by $O(\frac{d}{\epsilon^3})$.

Dual Gap:

Theorem 3.5 from the paper quantifies the dual gap for the algorithm in 2 with different gradient approximation schemes:

1. For the RDSA gradient approximation scheme:

$$E\left[\min_{t=0,\ldots,T-1} G(x_t)\right] \leq \frac{7(F(x_0) - F(x^*))}{2T} + \frac{LR^2 \ln(T+7)}{T} + \frac{Q_0 + R\sqrt{2Q}}{2T^{2/3}} \quad (9)$$

2. For the I-RDSA gradient approximation scheme:

$$E\left[\min_{t=0,\ldots,T-1} G(x_t)\right] \leq \frac{7(F(x_0) - F(x^*))}{2T} + \frac{LR^2 \ln(T+7)}{T} + \frac{Q_{ir} + R\sqrt{2Q_{ir}}}{2T^{2/3}} \quad (10)$$

3. For the KWSA gradient approximation scheme:

$$E\left[\min_{t=0,\ldots,T-1} G(x_t)\right] \leq \frac{7(F(x_0) - F(x^*))}{2T} + \frac{LR^2 \ln(T+7)}{T} + \frac{Q_{kw} + R\sqrt{2Q_{kw}}}{2T^{2/3}} \quad (11)$$

The dimension dependence of the Frank-Wolfe duality gap is characterized as $d^{1/3}$, and the dependence on iterations follows $O(T^{-1/3})$. The number of queries to the SZO to obtain a Frank-Wolfe duality gap of ϵ is given by $O(\frac{d}{\epsilon^3})$. Additionally, the theorem asserts that the initial conditions are forgotten as $O(1/T)$.

The algorithm's effectiveness is demonstrated in black-box optimization scenarios, specifically in optimizing complex forward models from physics without analytical expressions. The proposed algorithm converges to a first-order stationary point, showcasing its applicability in challenging black-box optimization problems.

The paper introduces a stochastic zeroth order Frank-Wolfe algorithm suitable for non-smooth optimization problems. The algorithm's reliance on hard-to-estimate parameters is minimal, making it practical for real-world applications. Convergence rates for the primal gap and Frank-Wolfe duality gap are quantified, highlighting its efficiency. Experimental results on various datasets further support its efficacy.

2.2 Faster Zeroth-Order Conditional Gradient (FZCGS) Method

This paper addresses a constrained finite-sum minimization problem of the form:

$$\min_{x\in\Omega} F(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x) \tag{1}$$

$\Omega \subset \mathbb{R}^d$ - a closed convex feasible set; $f_i(x)$ - smooth and non-convex component functions; n - the number of component functions. This problem has broad applications in machine learning, including tasks like robust low-rank matrix completion.

The key contributions of this paper include:

- Introduction of a faster stochastic zeroth-order Frank-Wolfe method with a Function Query Oracle (FQO) achieving an oracle complexity of $O(\frac{n^{1/2}d}{\epsilon^2})$.
- Introduction of a faster stochastic zeroth-order conditional gradient sliding method with an improved function queries oracle, achieving an oracle complexity of $O(\frac{n^{1/2}d}{\epsilon})$.
- Introduction of a new stochastic first-order conditional gradient sliding method with an Incremental First-Order Oracle (IFO) achieving an oracle complexity of $O(\frac{n^{1/2}}{\epsilon})$.

The paper utilizes different oracle models to compare the iteration complexity of various algorithms:

- Function Query Oracle (FQO): Samples a component function and returns its function value $f_i(x)$.
- Incremental First-Order Oracle (IFO): Samples a component function and returns its gradient $\nabla f_i(x)$.
- Linear Oracle (LO): Solves a linear programming problem and returns $\text{argmax}_{u\in\Omega}\langle u, v\rangle$.

The algorithm introduced in the paper, termed Faster Zeroth-Order Conditional Gradient Sliding (FZCGS), leverages acceleration techniques inspired by non-convex Frank-Wolfe methods. It employs gradient estimation methods and utilizes conditional gradient sliding for updates. The convergence rate is determined by the choice of parameters.

The pseudocode is presented below:

Conditional Gradient Update

1 : $u_1 = u, \quad t = 1$

2 : v_t be an optimal solution for

$$V_{g,u,\gamma}(u_t) = \max_{x\in\Omega}\left(g + \frac{1}{\gamma}(u_t - u), u_t - x_i\right)$$

3 : If $V_{g,u,\gamma}(u_t) \le \eta$, return $u^+ = u_t$.

4 : Set $u_{t+1} = (1-\alpha_t)u_t + \alpha_t v_t$, where

$$\alpha_t = \min\left\{1, \frac{\left(\frac{1}{\gamma}(u-u_t) - g, v_t - u_t\right)}{\left(\frac{1}{\gamma}\|v_t - u_t\|^2\right)}\right\}.$$

5 : Set $t \leftarrow t+1$ and go to step 2.

FZCGS

Input: $x_0, q > 0, \mu > 0, K > 0, \eta > 0, \gamma > 0, n$

1 : for $k = 0, \ldots, K-1$ do

2 : if $\text{mod}(k, q) = 0$ then

3 : Sample S_1 without replacement to compute $v_k = \nabla f_{S_1}(x_k)$

4 : else

5 : Sample S_2 with replacement to compute

$$v_k = \frac{1}{|S_2|}\sum_{i\in S_2}[\nabla f_i(x_k) - \nabla f_i(x_{k-1}) + v_{k-1}]$$

6 : end if

7 : $x_{k+1} = \text{condg}(v_k, x_k, k, \eta_k)$

8 : end for

Output: Randomly choose x_α from $\{x_k\}$ and return it.

The main results can be summarized as follows:

- For FZCGS with specific parameter settings, the expected squared norm of the gradient estimation error converges as per the given rate.
- The amortized function queries oracle complexity for FZCGS is $O(\frac{n^{1/2}d}{\epsilon})$.
- The linear oracle complexity is $O(\frac{1}{\epsilon^2})$.

Experimental results are presented for both zeroth-order and first-order methods, focusing on the non-convex maximum correntropy criterion-induced regression (MCCR) model. The experiments demonstrate the superiority of the proposed methods over baseline methods in terms of convergence speed and performance. FZCSG incorporates the acceleration technique, which gives this method a significant advantage compared to the analogues.

The paper presents innovative stochastic zeroth and first-order optimization algorithms with accelerated convergence and reduced oracle complexities. Experimental results validate the effectiveness of these algorithms.

2.3 Zero-Order Stochastic Conditional Gradient Sliding (ZOSCGS) Algorithm

The ZOSCGS algorithm is a groundbreaking gradient-free approach designed for non-smooth convex stochastic optimization problems with constraints. It introduces a novel strategy to tackle optimization challenges in environments where exact gradient information is unavailable.

- **Robust for Black-Box Problems**: Theoretical analysis showcases ZOSCGS's robustness not only in non-smooth scenarios but also in smooth settings. It outperforms state-of-the-art (SOTA) algorithms in terms of oracle calls, with an estimation of ϵ^{-2} compared to ZSCG's ϵ^{-3}.
- **Empirical Validation**: The algorithm's theoretical advantages are empirically validated through comparisons with ZSCG in a model case within a smooth setting, shedding light on the reasons behind its advantage.

The ZOSCGS algorithm addresses the following non-smooth convex stochastic optimization problem with constraints:

$$f^* := \min_{x \in Q} [f(x) := \mathbb{E}_\xi [f(x, \xi)]] \tag{12}$$

In this problem:

- $Q \subset \mathbb{R}^d$ represents a convex compact set.
- $f : Q \to \mathbb{R}$ is a convex function.

It is commonly referred to as a black-box problem, where a zero-order (gradient-free) oracle provides function values $f(x, \xi)$ at a requested point x, possibly with adversarial noise $\delta(x)$.

The ZOSCGS algorithm operates as follows:

- It employs the Stochastic Conditional Gradient Sliding Method as an accelerated batched algorithm.
- Instead of relying on the exact gradient, ZOSCGS approximates the gradient using l_2 randomization, denoted as $\nabla f_\gamma(x, \xi, e)$.
- The algorithm's parameters include the number of iterations N, batch size B, stepsize ζ, learning rate η, and accuracies β.
- The algorithm is equipped to achieve an ϵ-accuracy with a specific number of iterations and gradient-free oracle calls.

The pseudocode is presented below:

ZOSCGS

Input: Start point $x_0 \in \mathbb{Q}$,
maximum number of iterations $N \in \mathbb{Z}^+$.
Let stepsize $\zeta_k \in [0, 1]$, learning rate $\eta_k > 0$,
accuracies β_k, batch size $B_k \in \mathbb{Z}^+$,
smoothing parameter $\gamma > 0$.

Initial: Generate independently vectors $e_1, e_2, \ldots$
uniformly distributed on the unit ℓ_2-sphere,
and set $y_0 \leftarrow x_0$.

1: for k = 1, . . . , N do
2: $z_k \leftarrow (1 - \zeta_k)x_{k-1} + \zeta_k y_{k-1}$
3 : Sample $\{e_1, \ldots, e_{B_k}\}$ and $\{\xi_1, \ldots, \xi_{B_k}\}$ independently
4 : $g_k \leftarrow \frac{1}{B_k} \sum_{i=1}^{B_k} \left(\frac{d}{2\gamma}\right) (f(z_k + \gamma e_i, \xi_i) - f(z_k - \gamma e_i, \xi_i))\, e_i$
5 : $y_k \leftarrow \mathrm{CG}(g_k, y_{k-1}, \eta_k, \beta_k)$ ▷ See CG in Algorithm 2
6 : $x_k \leftarrow (1 - \zeta_k)x_{k-1} + \zeta_k y_k$
7 : end for

Output: x_N.

Conditional Gradient Procedure

Input: $g_k, y_{k-1}, \eta_k, \beta_k$, where k is the iteration

1: **for** $t = 0, \ldots, T$
2: $v_t \leftarrow \arg\min_{v \in \mathbb{Q}} \langle g_t, v \rangle$
3: **if** $\langle g_t, u_t - v_t \rangle \leq \beta$
4: **return** u_t
5: **end if**
6: $\alpha_t \leftarrow \min\left\{\frac{\langle g_t, u_t - v_t \rangle}{\eta \cdot \|u_t - v_t\|^2}, 1\right\}$
7: $u_{t+1} \leftarrow u_t + \alpha_t \cdot (v_t - u_t)$
8: $g_{t+1} \leftarrow g_0 + \eta \cdot (u_{t+1} - u_0)$

Output: u.

ZOSCGS represents a pioneering gradient-free conditional gradient-type algorithm tailored for non-smooth convex stochastic optimization problems. The algorithm's robustness extends to both non-smooth and smooth black-box problems, as highlighted by its superior performance in terms of oracle complexity when compared to existing SOTA algorithms.

Several factors contribute to the algorithm's advantage:

- ZOSCGS utilizes Central Finite Difference (CFD) for gradient estimation, as opposed to Direct Finite Difference (DFD).
- Gaussian smoothing is chosen over l_2 randomization for smoothing.

- The algorithm is based on an accelerated batched first-order algorithm, distinguishing it from the unaccelerated first-order Stochastic Frank–Wolfe (SFW) method used by ZSCG.

Theoretical results demonstrate ZOSCGS's superiority in solving non-smooth black-box problems with robustness and efficiency, making it a promising choice for optimization challenges.

3 Experiments

In our experiments, we focus on the performance of the models on the task of adversarial attack on black-box DNN. The task is to find the adversarial perturbation such that the DNN model makes the incorrect prediction.

We follow the setup (Liu et al., 2019), with the nn-carlini pretrained DNN for the MNIST dataset.

3.1 Changes from theoretical papers

We performed the following changes to the parameters and algorithms presented theoretically in the papers:

- In ZOSCGS we fixed the parameter B[k] to 50, which identifies the number of directions in which the algorithm calculates the gradient in each epoch; this is done because the theoretical parameter grows exponentially, requiring more queries to the oracle and more time, furthermore the paper shows that the algorithm with fixed parameters has better performances.
- In FZCGS the gradient is calculated over a set of randomized directions, same as ZOSCGS, instead of over all the directions built by basis vectors; this is done because while less "precise" over the same amount of epochs, calculating the gradient over all basis vector directions requires too many queries to the oracle and time per epoch.
- In FZCGS we substituted the function used for the Conditional Gradient with the same function used in ZOSCGS because, while the results should be the same between the two methods, the one used in FZCGS created problems for the convergence to a result and we weren't able to find a set of parameters to make it work properly.
- In the CG step for both ZOSCGS and FZCGS we added a stopping condition on the size of alpha, as with the parameter getting too small the function took a lot of time to reach the stopping condition while effectively making negligible changes.

3.2 Feasible Set

We tested our algorithms on two different types of feasible sets:

- $\|\delta\|_\infty \leq s$ where s is set to 8 for the KW version of SGFFW and to 4 for the other versions of SGFFW and FZCGS;
- The set containing all possible directions in which the sum of elements is equal to 0, all elements except one are equal to 2000/d, and their opposite, which is built with the following commands:
  num=2000,
  Q=np.eye(d)*num,
  Qrm=np.full(Q.shape,num/d),
  Q=Q-Qrm,
  Q=np.concatenate((Q,-Q), axis=1)

The first type of feasible set is used in SGFFW methods as it effectively changes by the same quantity all elements in the direction, based on their sign, while the second type of feasible set is not usable effectively since SGFFW doesn't use the conditional gradient.

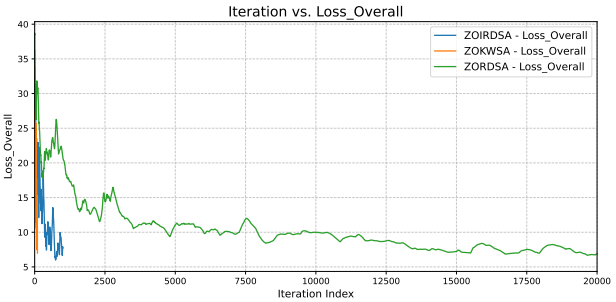
The second type of feasible set is instead more efficient in learning with FZCGS and ZOSCGS compared to the first, as it prioritized the element with the biggest absolute value, hence at each GC step it improves towards the single component direction with the largest gradient.

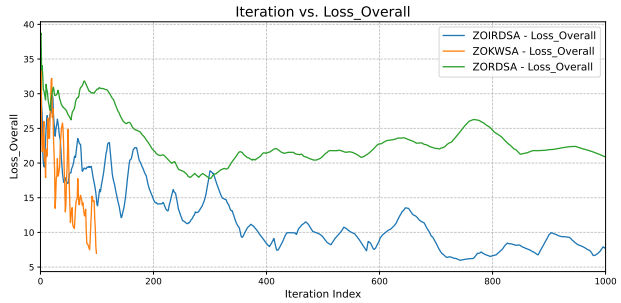
4 Results

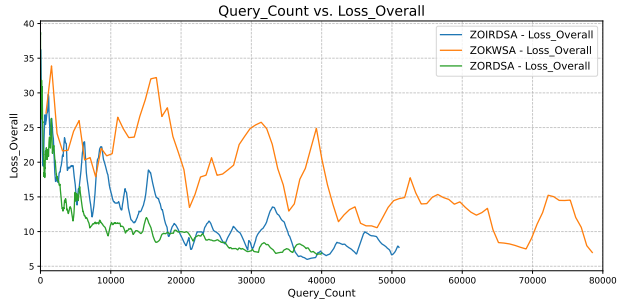
We will compare the results of our adversarial attacks in terms of overall loss over iterations (epochs), number of queries to the oracle and time.

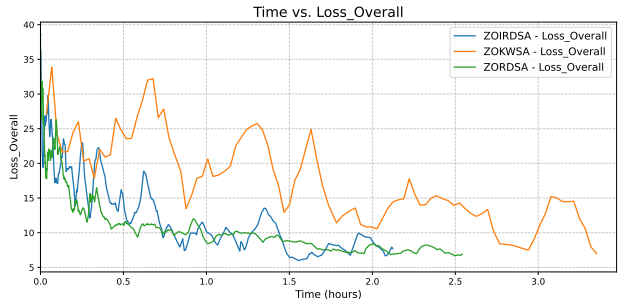
4.1 SGFFW comparison

First, we compare the results obtained between the different SGFFW: KWSA, I-RDSA and RDSA.





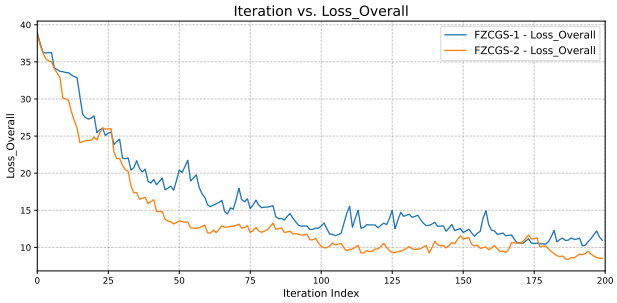




The graphs show respectively the total loss over the number of iterations, the total loss over the number of iterations limited to 200 iterations on the x-axis, the total loss over the number of calls to the oracle and the total loss over the elapsed time in hours. As expected the iteration graph shows FWSA as the best performing, I-RDSA as the second best and RDSA as the worst; however due to the size of the direction vector (784), compared to the parameter m used in I-RDSA (50), KWSA performance is worse per query and time. I-RDSA and RDSA have comparable results both regarding elapsed time and oracle queries. Since this paper's focus is on improving performance over queries, but the other papers focus more on improving performance over iterations we deem I-RDSA the best-performing method for SGFFW, so we will later compare this to the other algorithms.

4.2 FZCGS comparison with different feasible sets

We briefly compare the results obtained by the FZGCS method using the two different feasible sets, in this case, the time it takes for the algorithm is the same between the two sets and the iteration and query graphs are perfectly correlated, so we will only show the iteration graph.

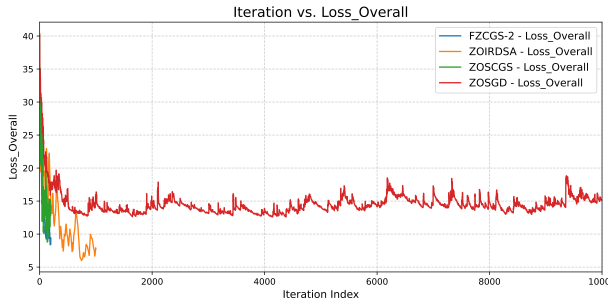


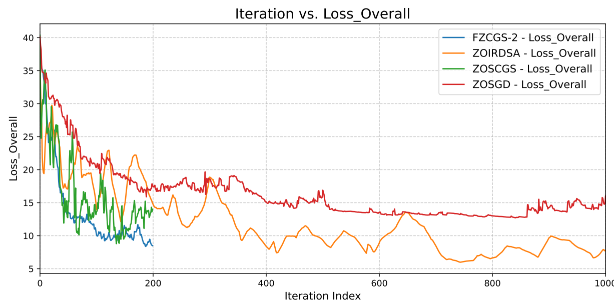
The graph shows the performance over iterations; we can notice that using the second feasible set clearly improves performance, however, the performance is comparable, showing only a small difference.

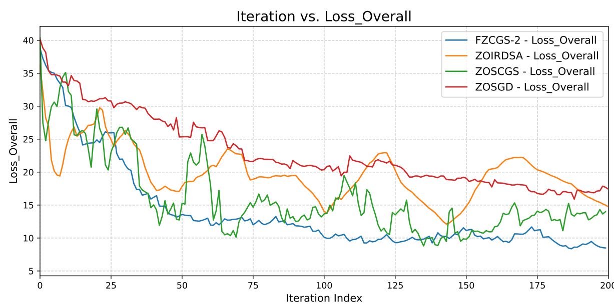
ZOSCGS showed the same results with the two feasible sets applied.

4.3 Comparison between methods

Here we will compare the results of the best methods observed in the subsections above with ZOSCGS, we have also added the well-known ZOSGD methods to offer a comparison to "classical" methodologies.



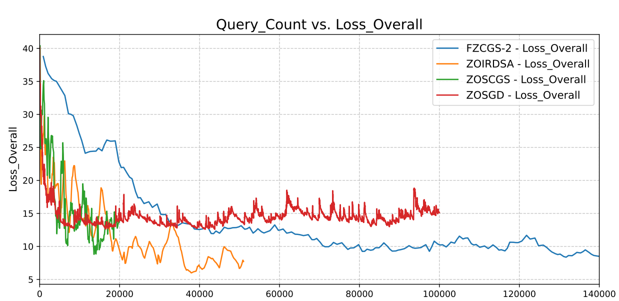


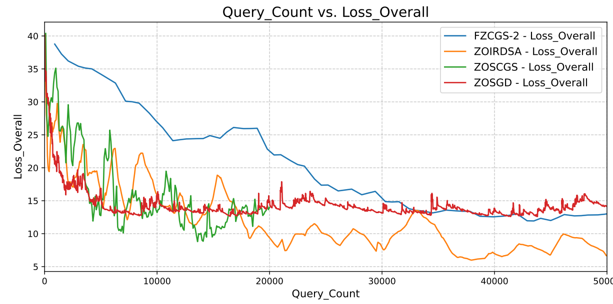


The graphs above show total loss over iteration, with the second and third graphs limited respectively to 1000 and 200 iterations.

The first thing we can notice is the fact that ZOSGD runs by far the highest number of iterations and yet it has stagnating performance which is at all comparable with the other algorithms.

The newer methods, ZOSCGS and FZCGS have comparable results (with the latter showing slightly better results) and are more efficient than I-RDSA, which isn't surprising as they focus on performance over iterations with the use of conditional gradient.



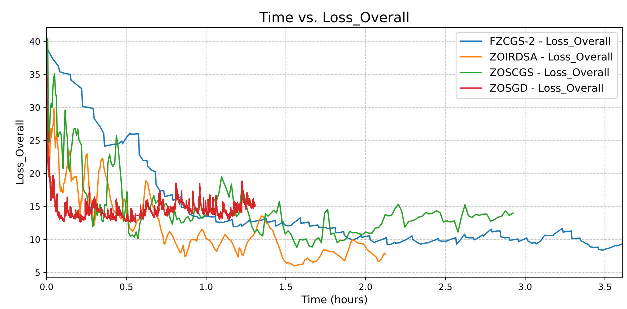


The graphs above show the total loss over queries to oracle, with the second being limited to 50000 queries.

ZOSGD shows a good performance with a small number of queries, however as we already noted it's performance doesn't improve much with many more queries.

FZCGS has really bad performance in this case, this is due to the fact that it uses many queries to improve performance over iterations, compared to the other methods (as FZCGS uses multiple stochastic examples at each epoch for each direction analyzed).

I-RDSA and ZOSCGS have similar performance.



The last graph shows total loss over time, which isn't one of the focuses of the papers, but shows a real-world "limitation" we encountered.

ZOSGD again shows very good performance but only in a small amount of time, while FZCGS shows the opposite results.

ZOSCGS have a similar performance initially, however, I-RDSA shows a better improvement after around 40 minutes.

5 Conclusions

The final consideration that can be made is that there is no overall best method, however, each method has its strengths and weaknesses:

- FZCGS has the best performance when consider-

ing iterations, but is the worst among the three methods when considering queries and time;

- I-RDSA isn't as good as the others when considering iterations, however, it has very good performance considering query and time, reaching the lowest overall loss in our tests;
- ZOSCGS isn't the single best in any category, but its performance is the most "balanced" having very good results in all three categories.

References

[1] Hongchang Gao and Heng Huang. "Can Stochastic Zeroth-Order Frank-Wolfe Method Converge Faster for Non-Convex Problems?" In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, 13–18 Jul 2020, pp. 3377–3386. URL: https://proceedings.mlr.press/v119/gao20b.html.

[2] Aleksandr Lobanov et al. *Zero-Order Stochastic Conditional Gradient Sliding Method for Non-smooth Convex Optimization*. 2023. arXiv: 2303.02778 [math.OC].

[3] Anit Kumar Sahu, Manzil Zaheer, and Soummya Kar. "Towards Gradient Free and Projection Free Stochastic Optimization". In: *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*. Ed. by Kamalika Chaudhuri and Masashi Sugiyama. Vol. 89. Proceedings of Machine Learning Research. PMLR, 16–18 Apr 2019, pp. 3468–3477. URL: https://proceedings.mlr.press/v89/sahu19a.html.