ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 978-1-7281-7605-5/20/$31.00 ©2021 IEEE | DOI: 10.1109/ICASSP39728.2021.9414485

ACCELERATING FRANK-WOLFE WITH WEIGHTED AVERAGE GRADIENTS

Yilang Zhang *Bingcong Li* *Georgios B. Giannakis*

Dept. of ECE and DTC, University of Minnesota, Minneapolis, MN, USA

ABSTRACT

Relying on a conditional gradient based iteration, the Frank-Wolfe (FW) algorithm has been a popular solver of constrained convex optimization problems in signal processing and machine learning, thanks to its low complexity. The present contribution broadens its scope by replacing the gradient per FW subproblem with a weighted average of gradients. This generalization speeds up the convergence of FW by alleviating its zigzag behavior. A geometric interpretation for the averaged gradients is provided, and convergence guarantees are established for three different weight combinations. Numerical comparison shows the effectiveness of the proposed methods.

***Index Terms*—** Frank-Wolfe method, conditional gradient approach, convex optimization

1. INTRODUCTION

Consider the following constrained optimization problem

$$\min_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}) \tag{1}$$

where f is a convex function with Lipschitz continuous gradient, and the constraint set $\mathcal{X} \subset \mathbb{R}^d$ is assumed convex and compact. Throughout, let $\mathbf{x}^* \in \mathcal{X}$ denote a minimizer of (1). For a wide range of signal processing and machine learning problems, the set $\mathcal{X}$ has structure, but it can be difficult or expensive to project onto. Examples include matrix completion in recommender systems [1] and image reconstruction [2], where $\mathcal{X}$ represents the nuclear- and the total-variation-norm ball, respectively. The applicability of projected gradient descent (GD) in these cases is challenged by the computational burden of projection, especially when d is large [3].

An alternative to GD for solving (1) is the Frank-Wolfe (FW) method [4, 5, 6], also known as conditional gradient iteration. FW circumvents the projection in GD by first solving a subproblem with a linear loss function, namely $\min_{\mathbf{v}\in\mathcal{X}}\langle\nabla f(\mathbf{x}_k),\mathbf{v}\rangle$ to obtain $\mathbf{v}_{k+1}$, and then updating $\mathbf{x}_{k+1}$ via a convex combination of $\mathbf{x}_k$ and $\mathbf{v}_{k+1}$. When dealing with a structured $\mathcal{X}$, an efficient or even closed-form solution for finding $\mathbf{v}_{k+1}$ can be available [5, 7], and can afford a cheaper implementation than projection. However, when using e.g. an n-support norm ball, that is, $\mathcal{X} := \text{conv}\{\mathbf{x} \mid \|\mathbf{x}\|_0 \le n,\ \|\mathbf{x}\|_2 \le R\}$ to promote sparse solutions, the complexity per FW iteration is $\mathcal{O}(d\log n)$ [8], while no efficient projection over such a constraint set is available. Providing easy implementation to effect structured solutions, justifies the popularity of FW in several applications. Besides those already mentioned, additional ones include structured SVM [9], video colocation [10], particle filtering [11], traffic assignment [12], optimal transport [13], and electronic vehicle charging [14]. FW also works for nonconvex tasks such as neural network training [15], and submodular optimization [16].

Research in this paper was supported in part by the NSF grant 1901134.
Emails:{zhan7453,lixx5599, georgios}@umn.edu

While FW avoids the costly projection per iteration, it gives rise to a zigzag trajectory of the solution over certain constraint sets [6]. To alleviate zig-zagging, which could significantly slow down convergence, a prudent remedy is to adjust the update direction. In this work, we will introduce three novel options to decide the direction per iteration by replacing the gradient in FW with the weighted average of gradients in previous iterations. The price paid for the performance improvement gained by mitigating the zigzag behavior of FW is minimal — just maintaining the weighted average gradient that can readily be updated online. Different from [17, 18] however, where the averaged gradients serve the necessary steps for momentum, here we deal with a generic setup without momentum. Due to space limitation, we delegate a more detailed case study to Appendix, where our averaged gradients are shown to provably improve the convergence of vanilla FW. All in all, our contribution is threefold.

i) FW is generalized by replacing the ordinary gradient in the subproblem per iteration with a weighted average gradient – a generalization that can afford a neat geometric interpretation.

ii) Three types of weighted averaging for gradients are proposed with guaranteed convergence of the resultant iterative solvers.

iii) Numerical tests confirm the effectiveness of the proposed methods on benchmark datasets.

Notation. Bold lowercase letters denote column vectors; $\mathbb{E}$ represents mathematical expectation; $\|\mathbf{x}\|$ stands for the ℓ_2-norm of $\mathbf{x}$; and $\langle\mathbf{x},\mathbf{y}\rangle$ for the inner product of vectors $\mathbf{x}$ and $\mathbf{y}$. A complete version of this work with proofs can be found online [1].

2. PRELIMINARIES

To better illustrate the intuition behind FW with averaged gradients, we first outline the vanilla FW on the class of problems we will deal subsequently.

Assumption 1. *(Lipschitz continuous gradient.) The function $f : \mathcal{X} \to \mathbb{R}$ has L-Lipchitz continuous gradients; that is, $\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\| \le L\|\mathbf{x}-\mathbf{y}\|,\ \forall\,\mathbf{x},\mathbf{y}\in\mathcal{X}$.*

Assumption 2. *(Convex objective function.) The function $f : \mathcal{X} \to \mathbb{R}$ is convex; that is, $f(\mathbf{y}) - f(\mathbf{x}) \ge \langle\nabla f(\mathbf{x}),\mathbf{y}-\mathbf{x}\rangle,\ \forall\,\mathbf{x},\mathbf{y}\in\mathcal{X}$.*

Assumption 3. *(Convex and compact constraint set.) The constraint set $\mathcal{X}$ is convex and compact with diameter D; that is, $\|\mathbf{x}-\mathbf{y}\| \le D,\ \forall\mathbf{x},\mathbf{y}\in\mathcal{X}$.*

Assumptions 1 – 3 are standard for FW and its variants, and they are assumed to hold true throughout.

[1] https://www.dropbox.com/s/7d9lf5ry3a0o5ka/AvgFW_icassp_with_proofs.pdf?dl=0

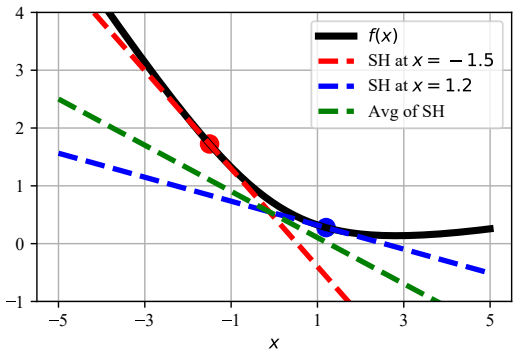

Fig. 1. Supporting hyperplanes (SH) of $f(\mathbf{x})$, and their average for the logistic loss function (black curve). The red (blue) line is the tangent SH of $f(\mathbf{x})$ at the red (blue) point. The green tangent is the average of the red and blue SHs.

FW for solving problems satisfying Assumptions 1 – 3 is summarized in Alg. 1. A subproblem with linear loss function having coefficient $\nabla f(\mathbf{x}_k)$ is solved per iteration to obtain the auxiliary variable $\mathbf{v}_{k+1}$ (cf. line 3). With reference to Fig. 1, the aforementioned linear function $f(\mathbf{x}_k) + \langle \nabla f(\mathbf{x}_k), \mathbf{x} - \mathbf{x}_k \rangle$ is the supporting hyperplane (SH) of $f(\mathbf{x})$ at $\mathbf{x}_k$; and thus it lower bounds $f(\mathbf{x})$ due to convexity; see e.g., the red and blue lines in Fig. 1. Variable $\mathbf{v}_{k+1}$ minimizes this lower bound over $\mathcal{X}$. Devoid of projection, the FW iterate is then updated as the convex combination of $\mathbf{x}_k$ and $\mathbf{v}_{k+1}$; thus, the update direction in the kth iteration is $\mathbf{v}_{k+1} - \mathbf{x}_k$. The most commonly used step size for FW is $\eta_k = \frac{2}{k+2}$, which ensures convergence of the error $f(\mathbf{x}_k) - f(\mathbf{x}^*) = \mathcal{O}(\frac{LD^2}{k})$ [5].

It turns out however, that the update direction in Alg. 1 does not always lead to the best empirical performance. Indeed, it has been observed that improved numerical results are obtained when the direction is found using matching pursuit [19]. In addition, if the loss function is strongly convex and the constraint set is a polytope, linear convergence is ensured by relying on "away steps" that move iterates away from the original FW updated direction [6]. These considerations motivate well the investigation of alternative update directions per FW iteration. This is the subject of the ensuing section.

3. FW WITH WEIGHTED AVERAGE GRADIENTS

This section introduces our FW approach based on averaged gradients, along with its geometric interpretation, and relevant convergence claims.

3.1. Averaged gradients

To have options for the update direction while maintaining FW's geometric interpretation, we will rely on the average of gradients from previous iterations. A generic form of FW with averaged gradients is summarized in Alg. 2, where the vector $\mathbf{g}_{k+1}$ is the key difference with vanilla FW. Clearly, with $\delta_k \equiv 1$, Alg. 2 boils down to FW. As we will see shortly, varying δ_k yields different FW iterations. But before that we will gain insights through a detailed interpretation per subproblem of the proposed algorithm.

Geometric interpretation. Consider the weighted average gradient $\mathbf{g}_{k+1} = \sum_{\tau=0}^{k} w_k^\tau \nabla f(\mathbf{x}_\tau)$, where $w_k^\tau = \delta_\tau \prod_{j=\tau+1}^{k}(1-\delta_j) > 0$, $\forall \tau \geq 1$, and $w_k^0 = \prod_{j=1}^{k}(1-\delta_j) > 0$. It can be easily verified

Algorithm 1 FW [4]

1: **Initialize:** $\mathbf{x}_0 \in \mathcal{X}$
2: **for** $k = 0, 1, \ldots, K-1$ **do**
3: $\quad \mathbf{v}_{k+1} = \arg\min_{\mathbf{x}\in\mathcal{X}} \langle \nabla f(\mathbf{x}_k), \mathbf{x} \rangle$
4: $\quad \mathbf{x}_{k+1} = (1-\eta_k)\mathbf{x}_k + \eta_k \mathbf{v}_{k+1}$
5: **end for**
6: **Return:** $\mathbf{x}_K$

Algorithm 2 A generic form of FW with averaged gradients

1: **Initialize:** $\mathbf{x}_0 \in \mathcal{X}, \mathbf{g}_0 = \nabla f(\mathbf{x}_0)$
2: **for** $k = 0, 1, \ldots, K-1$ **do**
3: $\quad \mathbf{g}_{k+1} = (1-\delta_k)\mathbf{g}_k + \delta_k \nabla f(\mathbf{x}_k)$
4: $\quad \mathbf{v}_{k+1} = \arg\min_{\mathbf{x}\in\mathcal{X}} \langle \mathbf{g}_{k+1}, \mathbf{x} \rangle$
5: $\quad \mathbf{x}_{k+1} = (1-\eta_k)\mathbf{x}_k + \eta_k \mathbf{v}_{k+1}$
6: **end for**
7: **Return:** $\mathbf{x}_K$

that $\sum_{\tau=0}^{k} w_k^\tau = 1$, $\forall k \geq 0$. Then, define a sequence of linear functions $\{\Phi_{k+1}(\mathbf{x})\}$ as

$$\Phi_{k+1}(\mathbf{x}) := \sum_{\tau=0}^{k} w_k^\tau \big[f(\mathbf{x}_\tau) + \langle \nabla f(\mathbf{x}_\tau), \mathbf{x} - \mathbf{x}_\tau \rangle \big], \ \forall k \geq 0 \quad (2)$$

which represents a weighted average of the supporting hyperplanes of $f(\mathbf{x})$ at $\{\mathbf{x}_\tau\}_{\tau=0}^{k}$; see the green line in Fig. 1. Properties of $\Phi_{k+1}(\mathbf{x})$, and its relevance with the proposed algorithm are summarized in the next lemma.

Lemma 1. *For the linear function $\Phi_{k+1}(\mathbf{x})$ in (2) it holds that: i) $\mathbf{v}_{k+1}$ minimizes $\Phi_{k+1}(\mathbf{x})$ over $\mathcal{X}$; and ii) $\Phi_{k+1}(\mathbf{x})$ is a global lower bound for $f(\mathbf{x})$, that is, $\Phi_{k+1}(\mathbf{x}) \leq f(\mathbf{x})$.*

Similar to the vanilla FW, Lemma 1 establishes that $\mathbf{v}_{k+1}$ minimizes a global lower bound of $f(\mathbf{x})$ even for general weighted average gradients.

3.2. Parameter choices

Next, we will provide three different choices for the parameters to ensure convergence. The first one we shall consider is

$$\text{FW-SA:} \quad \delta_k = \frac{4k}{4k+1}, \ \eta_k = \frac{2}{k+2}, \ \forall k \geq 0. \quad (3)$$

As k grows, a larger weight is applied to the current gradient $\nabla f(\mathbf{x}_k)$ when computing $\mathbf{g}_{k+1}$, while the weights for previous gradients decay rapidly. We thus term Alg. 2 with parameters in (3) as *FW with short-term averaged gradients (FW-SA)*. Clearly, when k is sufficiently large, the averaged gradients $\mathbf{g}_{k+1}$ in FW-SA will approach the gradient $\nabla f(\mathbf{x}_k)$ in the vanilla FW, thereby yielding similar update directions. In contrast, the following two parameter choices will lead to update directions that are markedly different from those of the vanilla FW.

Our second choice of parameters is (cf. (3))

$$\text{FW-EA:} \quad \delta_k = \delta, \ \eta_k = \frac{c}{k+k_0}, \ \forall k \geq 0 \quad (4)$$

where $\delta \in (0,1)$, and c and k_0 are constants to be specified later. With $\delta_k = \delta$, the average gradient $\mathbf{g}_{k+1}$ amounts to an *exponentially moving average of previous gradients*, thus the abbreviation *FW-EA* when Alg. 2 utilizes the parameters in (4). The moving average

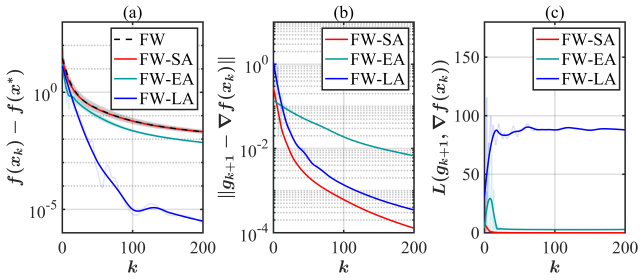

Fig. 2. A glance of FW with averaged gradients using logistic regression with an ℓ_2-norm ball constraint on dataset *mushroom*. (a) The optimality error; (b) $\|\mathbf{g}_{k+1} - \nabla f(\mathbf{x}_k)\|$; (c) the angle between $\mathbf{g}_{k+1}$ and $\nabla f(\mathbf{x}_k)$.

gradient was also adopted in [16] for stochastic optimization in order to reduce the variance of the noisy gradient. Clearly, our use here is for a different purpose.

Our third choice of parameters is

$$\text{FW-LA:} \quad \delta_k = \eta_k = \frac{2}{k+2}, \ \forall k \geq 0. \tag{5}$$

With δ_k as in (5), it can be verified that $w_k^\tau = \mathcal{O}(\frac{\tau}{k^2})$. Hence, recent gradients in $\mathbf{g}_{k+1}$ are weighted less compared to (3) and (4). We term Alg. 2 with the parameters chosen by (5) as *FW with long-term averaged gradients (FW-LA)*.

Before moving on to convergence analysis, it is instructive to check how the averaged gradient differs from the one in the vanilla FW. To this end, we carried numerical tests using logistic regression over an ℓ_2-norm ball constraint. The convergence results are depicted in Fig. 2(a). As expected, FW-SA shows performance comparable to the vanilla FW, while FW-LA outperforms the vanilla one by a significant margin. Fig. 2(b) shows how the amplitude difference between $\mathbf{g}_{k+1}$ and $\nabla f(\mathbf{x}_k)$ decreases with k. This is intuitively reasonable since more weight is put on recent gradients. Moreover, we can deduce from Fig. 2(c) that the direction of $\mathbf{g}_{k+1}$ is considerably different from that of $\nabla f(\mathbf{x}_k)$ in FW-LA, and sometimes the pertinent angle becomes obtuse. Such differences eventually lead to different update directions that speed up convergence.

3.3. Convergence of FW-SA

Next, we analyze the convergence of the FW-SA, FW-EA, and FW-LA algorithms. The following lemma is essential for analyzing FW-SA, because it upper bounds the distance between the weighted average gradient $\mathbf{g}_k$ and the gradient $\nabla f(\mathbf{x})$ for any $\mathbf{x} \in \mathcal{X}$.

Lemma 2. *If $f(\mathbf{x})$ has L-Lipschitz gradient, it holds for $\forall \mathbf{x} \in \mathcal{X}$ and $\forall k \geq 1$, that*

$$\|\mathbf{g}_k - \nabla f(\mathbf{x})\| \leq LD. \tag{6}$$

Building upon Lemma 2, the convergence rate of FW-SA is established in the following theorem.

Theorem 1. *With $\mathbf{g}_0 = \nabla f(\mathbf{x}_0)$, $\delta_k = \frac{4k}{4k+1}$, and $\eta_k = \frac{2}{k+2}$, the optimality error of FW-SA satisfies for $\forall k \geq 1$ that*

$$f(\mathbf{x}_k) - f(\mathbf{x}^*) = \mathcal{O}\Big(\frac{LD^2}{k}\Big). \tag{7}$$

Theorem 1 asserts that the convergence rate of FW-SA coincides with that of FW, which also agrees with our observation in Fig. 2(a).

3.4. Convergence of FW-EA

The next lemma demonstrates that $\|\mathbf{g}_{k+1} - \nabla f(\mathbf{x}_k)\|^2$ in the FW-EA iteration converges faster than that in FW-SA of Lemma 2.

Lemma 3. *Let $\mathbf{g}_0 = \nabla f(\mathbf{x}_0)$, and $\eta_k = \frac{c_1}{k+k_0+1}$ in the FW-EA iteration. If there is a constant c_0 satisfying*

$$c_1^2 \leq \left[1 - (1-\delta)\frac{(k_0+1)^2}{k_0^2}\right]\delta c_0^2 \tag{8}$$

the weighted average gradient in FW-EA then obeys

$$\|\mathbf{g}_{k+1} - \nabla f(\mathbf{x}_k)\|^2 \leq \frac{c_0^2 L^2 D^2}{(k+k_0)^2}.$$

To avoid the burden of choosing constants, consider the instance where $k_0 = 2$, $\delta = 0.8$, $c_1 = 2$, and $c_0 \approx 3.05$, which can be readily verified that they satisfy (8). Lemma 3 then ensures that the distance between $\mathbf{g}_{k+1}$ and $\nabla f(\mathbf{x}_k)$ will converge with rate $\mathcal{O}(\frac{1}{k})$; see also Fig. 2(b). Thus, FW-EA can be proved convergent by leveraging Lemma 3.

Theorem 2. *With $\mathbf{g}_0 = \nabla f(\mathbf{x}_0)$, $\eta_k = \frac{2}{k+3}$, and $\delta = 0.8$, the optimality error (convergence rate) of FW-EA satisfies $\forall k \geq 1$ that*

$$f(\mathbf{x}_k) - f(\mathbf{x}^*) = \mathcal{O}\Big(\frac{LD^2}{k}\Big).$$

Clearly, Theorem 2 establishes that the FW-SA iteration converges with rate $\mathcal{O}(\frac{LD^2}{k})$, which is the same as that of FW-SA.

3.5. Convergence of FW-LA

Recall that $\Phi_{k+1}(\mathbf{x})$ in (2) lower bounds $f(\mathbf{x})$ over $\mathcal{X}$. We rely on this fact to prove convergence of the FW-LA iteration next.

Theorem 3. *Under Assumptions 1-3, and upon choosing $\delta_k = \eta_k = \frac{2}{k+2}$, the so-termed generalized optimality gap of the FW-LA iteration satisfies for $\forall k \geq 1$ that*

$$f(\mathbf{x}_k) - \Phi_k(\mathbf{v}_k) = \mathcal{O}\Big(\frac{LD^2}{k}\Big).$$

This generalized FW gap broadens the standard definition of FW gap in e.g., [5]. Note that Theorem 3 is even stronger than that of [5, Theorem 2], because our result here holds for every $k \geq 1$.

Relative to the optimality gap, the generalized one satisfies

$$f(\mathbf{x}_k) - \Phi_k(\mathbf{v}_k) \geq f(\mathbf{x}_k) - \Phi_k(\mathbf{x}^*) \geq f(\mathbf{x}_k) - f(\mathbf{x}^*) \tag{9}$$

where the inequalities follow because $\Phi_k(\mathbf{v}_k) \leq \Phi_k(\mathbf{x}^*) \leq f(\mathbf{x}^*)$ (cf. Lemma 1). As a result, convergence of the generalized FW gap directly implies convergence of the optimality gap.

Corollary 1. *With parameters as those in Theorem 3, FW-LA guarantees for $\forall k \geq 1$ that*

$$f(\mathbf{x}_k) - f(\mathbf{x}^*) = \mathcal{O}\Big(\frac{LD^2}{k}\Big).$$

Proof. The proof relies on Theorem 3 and (9). □

As with the standard FW gap, the generalized one can also be used as a stopping criterion. Specifically, if $f(\mathbf{x}_k) - \Phi_k(\mathbf{v}_k) \leq \epsilon$ for some prescribed accuracy ϵ, (9) directly implies that $f(\mathbf{x}_k) - f(\mathbf{x}^*) \leq \epsilon$.

Other averaging schemes. One can also let $\delta_k = \frac{1}{k}$, $\forall k \geq 1$, which boils down to FW with uniformly averaged gradients (FW-UA) [20]. Such a choice with $\mathbf{g}_{k+1}$ formed by the average of previous gradients, slows down convergence to a rate $\mathcal{O}(\frac{LD^2 \ln k}{k})$. This explains why FW-SA, FW-EA, and FW-LA, all outperform FW-UA.

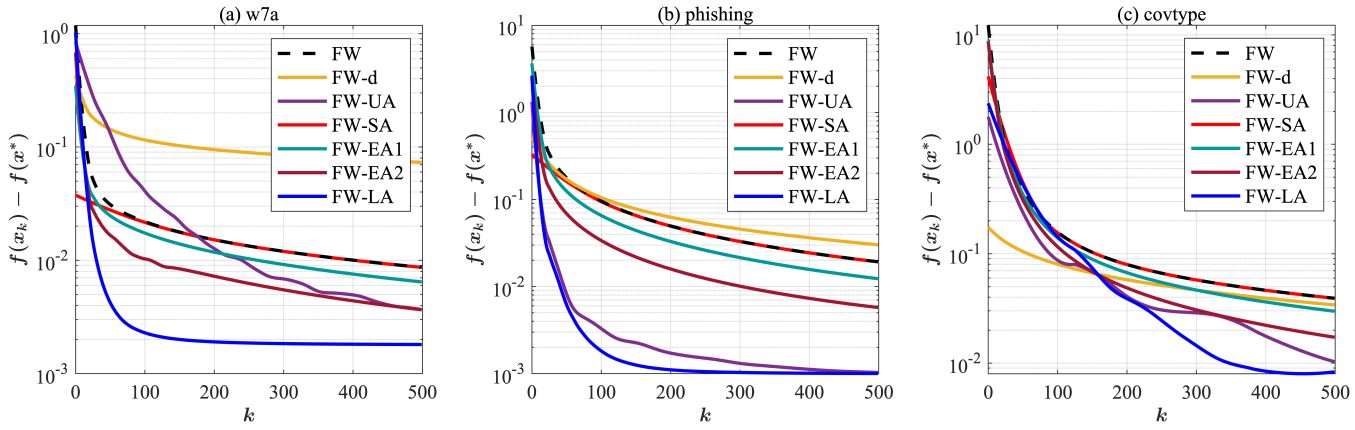

Fig. 3. Performance of FW with averaged gradients on different datasets.

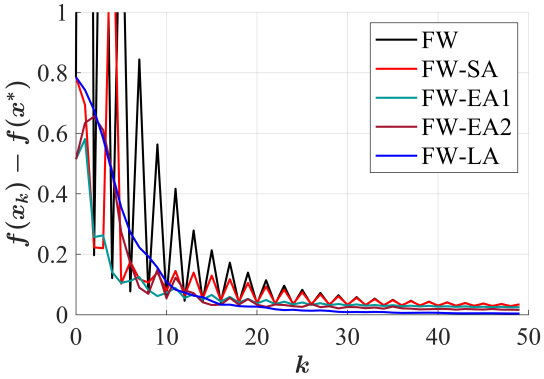

Fig. 4. Zigzag phenomenon on dataset $w7a$.

4. NUMERICAL TESTS

Having established the convergence rates of FW-SA, FW-EA and FW-LA, we proceed here to test their performance numerically. Logistic regression for binary classification is employed for the tests with an ℓ_2-norm ball constraint (that is, $\mathcal{X} = \{\mathbf{x} \mid \|\mathbf{x}\| \leq R\}$) to enhance generalization.

With $\mathbf{x}$ collecting the classifier weights, the loss function is

$$f(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N} \ln\left(1 + \exp(-b_i\langle \mathbf{a}_i, \mathbf{x}\rangle)\right) \tag{10}$$

where $(\mathbf{a}_i, b_i)$ is the (feature, label) pair of datum i, and N is the number of data. It can be verified that Assumptions 1 – 3 are satisfied. Benchmark datasets from LIBSVM[2] are used in the numerical tests. Details regarding the datasets are summarized in Table 1.

As for the baselines, we use vanilla FW with standard step size $\eta_k = \frac{2}{k+2}$ (labeled FW), and a step size variant from [1] to guarantee per iteration descent (FW-d). The performance of FW with uniformly averaged gradients (FW-UA) [20] is also plotted for comparison. For the proposed methods, the parameter choices for FW-SA and FW-LA are the same as those in Theorems 1 and 3. Whereas, FW-EA uses the parameters $\delta = 0.8$ and $\delta = 0.5$, which are denoted as FW-EA1 and FW-EA2, respectively.

[2] https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html.

Table 1. A summary of datasets used in numerical tests

Dataset	d	N (train)	nonzeros
w7a	300	24,692	3.89%
covtype	54	406,709	22.12%
phishing	68	11,055	44.77%

Fig. 4 illustrates how averaged gradients reduce zigzagging in the oscillating optimality error. The experiment is conducted on dataset $w7a$ for the first 50 iterations, and the results are displayed on a linear scale. It can be deduced that the proposed algorithms exhibit milder oscillation compared with vanilla FW, and they are able to converge faster to a lower stability level. These results further validate our assertion that averaged gradients can alleviate zigzagging, hence improving empirical performance over vanilla FW.

Performance of the proposed algorithms relative to baselines are shown in Fig. 3, where all curves are smoothed for display. It can be seen that in all tests, most of the proposed approaches converge faster than vanilla FW and FW-d. This observation also corroborates that the gradient is not always the 'best direction' to employ in the FW subproblem. Additional tests on ℓ_1-norm ball and ℓ_∞-norm ball constraints can be found in the Appendix, where the proposed algorithms again outperform vanilla FW and FW-d.

5. CONCLUSIONS AND FUTURE WORK

Frank-Wolfe solver with improved performance for constrained optimization problems was pursued here using weighted averages of gradients to update the per-iteration direction. Three types of weights offering complementary characteristics were introduced with the corresponding algorithms abbreviated as FW-SA, FW-EA, and FW-LA. Convergence guarantees and their speeds were established for all three options, along with insightful geometric interpretations. Numerical tests confirmed the efficiency of the proposed approaches.

Our future research agenda includes investigating convergence guarantees for the generalized FW gap of FW-EA, and also crafting per-step descent step sizes for FW with averaged gradients.

6. REFERENCES

[1] R. M. Freund, P. Grigas, and R. Mazumder, "An extended Frank–Wolfe method with "in-face" directions, and its application to low-rank matrix completion," *SIAM Journal on Optimization*, vol. 27, no. 1, pp. 319–346, 2017.

[2] Z. Harchaoui, A. Juditsky, and A. Nemirovski, "Conditional gradient algorithms for norm-regularized smooth convex optimization," *Mathematical Programming*, vol. 152, no. 1-2, pp. 75–112, 2015.

[3] Y. Nesterov, *Introductory Lectures on Convex Optimization: A Basic Course*. Springer Science & Business Media, 2004, vol. 87.

[4] M. Frank and P. Wolfe, "An algorithm for quadratic programming," *Naval Research Logistics Quarterly*, vol. 3, no. 1-2, pp. 95–110, 1956.

[5] M. Jaggi, "Revisiting Frank-Wolfe: Projection-free sparse convex optimization." in *Proc. Intl. Conf. on Machine Learning*, 2013, pp. 427–435.

[6] S. Lacoste-Julien and M. Jaggi, "On the global linear convergence of Frank-Wolfe optimization variants," in *Proc. Advances in Neural Info. Process. Syst.*, 2015, pp. 496–504.

[7] D. Garber and E. Hazan, "Faster rates for the Frank-Wolfe method over strongly-convex sets," in *Proc. Intl. Conf. on Machine Learning*, 2015.

[8] A. Argyriou, R. Foygel, and N. Srebro, "Sparse prediction with the k-support norm," in *Proc. Advances in Neural Info. Process. Syst.*, 2012, pp. 1457–1465.

[9] S. Lacoste-Julien, M. Jaggi, M. W. Schmidt, and P. Pletscher, "Block-coordinate Frank-Wolfe optimization for structural svms," in *Proc. Intl. Conf. on Machine Learning*, no. 1, 2013, pp. 53–61.

[10] A. Joulin, K. Tang, and L. Fei-Fei, "Efficient image and video co-localization with Frank-Wolfe algorithm," in *Proc. European Conf. on Computer Vision*. Springer, 2014, pp. 253–268.

[11] S. Lacoste-Julien, F. Lindsten, and F. Bach, "Sequential kernel herding: Frank-Wolfe optimization for particle filtering," in *Proc. Intl. Conf. on Artificial Intelligence and Statistics*, 2015, pp. 544–552.

[12] M. Fukushima, "A modified Frank-Wolfe algorithm for solving the traffic assignment problem," *Transportation Research Part B: Methodological*, vol. 18, no. 2, pp. 169–177, 1984.

[13] G. Luise, S. Salzo, M. Pontil, and C. Ciliberto, "Sinkhorn barycenters with free support via Frank-Wolfe algorithm," in *Proc. Advances in Neural Info. Process. Syst.*, 2019, pp. 9318–9329.

[14] L. Zhang, V. Kekatos, and G. B. Giannakis, "Scalable electric vehicle charging protocols," *IEEE Transactions on Power Systems*, vol. 32, no. 2, pp. 1451–1462, 2016.

[15] L. Berrada, A. Zisserman, and M. P. Kumar, "Deep Frank-Wolfe for neural network optimization," in *Proc. of Intl. Conf. on Learning Representations*, 2018.

[16] A. Mokhtari, H. Hassani, and A. Karbasi, "Stochastic conditional gradient methods: From convex minimization to submodular maximization," *arXiv preprint arXiv:1804.09554*, 2018.

[17] B. Li, M. Coutino, G. B. Giannakis, and G. Leus, "How does momentum help Frank Wolfe?" *arXiv preprint arXiv:2006.11116*, 2020.

[18] B. Li, L. Wang, G. B. Giannakis, and Z. Zhao, "Enhancing Frank Wolfe with an extra subproblem," in *Proc. of 35th AAAI Conf. on Artificial Intelligence*, 2021.

[19] C. W. Combettes and S. Pokutta, "Boosting Frank-Wolfe by chasing gradients," *arXiv preprint arXiv:2003.06369*, 2020.

[20] J. D. Abernethy and J.-K. Wang, "On Frank-Wolfe and equilibrium computation," in *Proc. Advances in Neural Info. Process. Syst.*, 2017, pp. 6584–6593.