Revisiting Projection-Free Optimization for Strongly Convex Constraint Sets

Jarrid Rector-Brooks
2260 Hayward St
Ann Arbor, MI, 48104
University of Michigan, Ann Arbor
jrectorb@umich.edu

Jun-Kun Wang
226 Ferst Drive NW
Atlanta, GA, 30332
Georgia Institute of Technology
jimwang@gatech.edu

Barzan Mozafari
226 Hayward St
Ann Arbor, MI, 48104
University of Michigan, Ann Arbor
mozafari@umich.edu

Abstract
We revisit the Frank-Wolfe (FW) optimization under strongly convex constraint sets. We provide a faster convergence rate for FW without line search, showing that a previously overlooked variant of FW is indeed faster than the standard variant. With line search, we show that FW can converge to the global optimum, even for smooth functions that are not convex, but are quasi-convex and locally-Lipschitz. We also show that, for the general case of (smooth) non-convex functions, FW with line search converges with high probability to a stationary point at a rate of $O(\frac{1}{t^2})$, as long as the constraint set is strongly convex—one of the fastest convergence rates in non-convex optimization.

1 Introduction
A popular family of optimization algorithms are so-called gradient descent algorithms: iterative algorithms that are comprised of a gradient descent step at each iteration, followed by a projection step when there is a feasibility constraint. The purpose of the projection is to ensure that the update vector remains within the feasible set.

In many cases, however, the projection step may have no closed-form and thus requires solving another optimization problem itself (e.g., for $l_1$ norm balls or matroid polytopes (Hazar and others 2016; Hazan and Kale 2012), the closed-form may exist but involve an expensive computation (e.g., the SVD of the model matrix for Schatten-1, Schatten-2, and Schatten-$\infty$ norm balls (Hazar and others 2016), or there may simply be no method available for computing the projection in general (e.g., the convex hull of rotation matrices (Hazar, Kale, and Warmuth 2010), which arises as a constraint set in online learning settings (Hazar, Kale, and Warmuth 2010)). In these scenarios, each iteration of the gradient descent may require many “inner” iterations to compute the projection (Jaggi, Sulovsk, and others 2010; Lacoste-Julien and Jaggi 2015; Hazan and Kale 2012). This makes the projection step quite costly, and can account for much of the execution time of each iteration (e.g., see Appendix B).

Frank-Wolfe (FW) optimization — In this paper, we focus on (FW) approaches, also known as projection-free or conditional gradient algorithms (Frank and Wolfe 1956). Unlike gradient descent, these algorithms avoid the projection step altogether by ensuring that the update vector always lies within the feasible set. At each iteration, FW solves a linear program over a constraint set. Since linear programs have closed-form solutions for most constraint sets, each iteration of FW is, in many cases, more cost effective than conducting a gradient descent step and then projecting it back to the constraint set (Jaggi 2013; Hazan and Kale 2012; Hazan and others 2016).

Another main advantage of FW is the sparsity of its solution. Since the solution of a linear program is always a vertex (i.e., extreme point) of the feasible set (when the set itself is convex), each iteration of FW can add, at most, one new vertex to the solution vector. Thus, at iteration $t$, the solution is a combination of, at most, $t + 1$ vertices of the feasible set, thereby guaranteeing the sparsity of the eventual solution (Clarkson 2010; Jaggi 2013; Jaggi 2011). For these reasons, FW optimization has drawn growing interest in recent years, especially in matrix completion, structural SVM, computer vision, sparse PCA, metric learning, and many other settings (Jaggi, Sulovsk, and others 2010; Lacoste-Julien et al. 2013; Osokin et al. 2016; Wang et al. 2016; Chari et al. 2015; Harchaoui et al. 2012; Hazan and Kale 2012; Shalev-Shwartz, Gonen, and Shamir 2011). Unfortunately, while faster in each iteration, standard FW requires many more iterations to converge than gradient descent, and therefore is slower overall. This is because FW’s convergence rate is typically $O\left(\frac{1}{t}\right)$ while that of (accelerated) gradient descent is $O\left(\frac{1}{t^2}\right)$, where $t$ is the number of iterations (Jaggi 2013).

We make several contributions (summarized in Table I):

1. We revisit a non-conventional variant of FW optimization, called Primal Averaging (PA) (Lan 2013), which has been largely neglected in the past, as it was believed to have the same convergence rate as FW without line search, yet incurring extra computations (i.e., matrix averaging step) at each iteration. However, we discover that, when the constraint set is strongly convex, this non-conventional variant enjoys a much faster convergence rate with high probability, $O\left(\frac{1}{t^2}\right)$ versus $O\left(\frac{1}{t}\right)$, which more than com-
pensates for its slightly more expensive iterations. This surprising result has important ramifications in practice, as many classification, regression, multitask learning, and collaborative filtering tasks rely on norm constraints that are strongly convex, e.g., generalized linear models with $l_p$ norm, squared loss regression with $l_p$ norm, multitask learning with Group Matrix norm, and matrix completion with Schatten norm (Kim and Xing 2010; Lacoste-Julien and Jaggi 2015; Demnyanov and Rubinov 1970; Dunn 1979)

2. While previous work on FW optimization has generally focused on convex functions, we show that FW with line search can converge to the global optimum, even for smooth functions that are not convex, but are quasi-convex and locally-Lipschitz.

3. We also study the general case of (smooth) non-convex functions, showing that FW with line search can converge to a stationary point at a rate of $O\left(\frac{1}{t}\right)$ with high probability, as long as the constraint set is strongly convex. To the best of our knowledge, we are not aware of such a fast convergence rate in the non-convex optimization literature.

4. Finally, we conduct extensive experiments on various benchmark datasets, empirically validating our theoretical results, and comparing the actual performance of various FW variants in practice.

---

1Without any assumptions, converging to local optima for continuous non-convex functions is NP-hard (Carmon et al. 2017; Agarwal et al. 2017).

2 Related Work

Table 1 compares the state-of-the-art on projection-free optimization to our contributions.

| Convex Loss Function | Additional Assumptions about the Loss Function | Constraint Set Assumption | Convergence Rate | Requires Line Search (In Each Iteration) |
|----------------------|-----------------------------------------------|---------------------------|-----------------|----------------------------------------|
| This Paper           | None                                          | Strongly convex           | $O\left(\frac{1}{t}\right)$ with high probability | No                                      |
| State-of-the-Art Result(s) | Jaggi 2013                                    | Convex                    | $O\left(\frac{1}{t}\right)$ | No                                      |
| Garber and Hazan 2015 | Strongly convex                               | Strongly convex           | $O\left(\frac{1}{\sqrt{t}}\right)$ | Yes                                     |
| Lacoste-Julien and Jaggi 2015 | Strongly convex     | Polytope                  | $O\left(\exp(-t)\right)$ | Yes                                     |
| Levitin and Polyak 1966 | Norm of the gradient lower bounded | Strongly convex           | $O\left(\exp(-t)\right)$ | No                                      |
| Demyanov and Rubinov 1970 | Strongly convex | Polytope                  | $O\left(\exp(-t)\right)$ | No                                      |
| Dunn 1979            | Strongly convex                               | Convex                    | $O\left(\exp(-t)\right)$ | No                                      |

| Quasi-Convex Loss Function | Additional Assumptions about the Loss Function | Constraint Set Assumption | Convergence Rate | Requires Line Search (In Each Iteration) |
|---------------------------|-----------------------------------------------|---------------------------|-----------------|----------------------------------------|
| This Paper                | Locally-Lipschitz, Norm of the gradient lower bounded | Strongly convex           | $O\left(\frac{1}{\sqrt{t}}\right)$ | Yes                                     |
| State-of-the-Art Result(s) | Does not exist                                | Does not exist            | Does not exist  | Does not exist                           |

| Non-Convex Loss Function | Additional Assumptions about the Loss Function | Constraint Set Assumption | Convergence Rate | Requires Line Search (In Each Iteration) |
|--------------------------|-----------------------------------------------|---------------------------|-----------------|----------------------------------------|
| This Paper               | None                                          | Strongly convex           | $O\left(\frac{1}{\sqrt{t}}\right)$ with high probability | Yes                                     |
| State-of-the-Art Result(s) | Lacoste-Julien 2016 | Convex                    | $O\left(\frac{1}{\sqrt{t}}\right)$ | No                                      |

Table 1: Our contributions compared to the state-of-the-art results for projection-free optimization. Here, $t$ is the number of iterations. For non-convex functions, convergence is defined in terms of a stationary point instead of a global minimum. Note that although our bound is probabilistic for convex loss functions, we use no additional assumptions on the loss function and do not require line search, which can be a costly operation for big data (see Section 2).
line search (see Section 7.2).

Prior work ([Levitin and Polyak 1966][Demyanov and Rubinov 1970][Dunn 1979]) shows that standard FW without line search for smooth functions can achieve an exponential convergence rate, by making a strict assumption that the gradient is lower-bounded everywhere in the feasible set. In our analysis of PA, however, we do not assume the gradient is lower-bounded everywhere, allowing our result to be more widely applicable.

Quasi-convex optimization — Hazan et al. study quasi-convex and locally-Lipschitz loss functions that admit some saddle points ([Hazan, Levy, and Shalev-Shwartz 2015]). One of the optimization algorithms for this class of functions is the so-called normalized gradient descent, which converges to an $\epsilon$-neighborhood of the global minimum. The analysis in ([Hazan, Levy, and Shalev-Shwartz 2015]) is for unconstrained optimization. In this paper, we analyze FW for the same class of functions, but with strongly convex constraint sets. Interestingly, when the constraint set is an $l_2$ ball, FW becomes equivalent to normalized gradient descent. In this paper, we both 1) show that FW can converge to a neighborhood of a global minimum, and 2) derive a convergence rate for non-convex optimization using FW, which is $O\left(\frac{1}{t}\right)$, where $t$ is the standard inner product.

Strongly convex constraint sets are quite common in machine learning. In contrast, we study a much more general class of quasi-convex functions, including several popular models (e.g., generalized linear models with a sigmoid loss).

3.1 Preliminaries

Strongly convex constraint sets are quite common in machine learning. For example, when $p \in (1, 2]$, $l_p$ balls $\{u \in \mathbb{R}^n : \|u\|_p \leq r\}$ and Schatten-$p$ balls $\{X \in \mathbb{R}^{m \times n} : \|X\|_{sp} \leq r\}$ are all strongly convex ([Garber and Hazan 2015]), where $\|X\|_{sp} = \left(\sum_{i=1}^{\min(m,n)} \sigma(X)_{i}^{p}\right)^{1/p}$ is the Schatten-$p$ norm and $\sigma(X)_{i}$ is the $i^{th}$ largest singular value of $X$. Group $l_{p,q}$ balls, used in multitask learning ([Garber and Hazan 2015][Kim and Xing 2010]), are also strongly convex when $p, q \in (1, 2]$. In this paper, we use the following definitions.

Definition 1 (Strongly convex set). A convex set $\Omega \subseteq \mathbb{R}^d$ is an $\alpha$-strongly convex set with respect to a norm $\|\cdot\|$ if for any $u, v \in \Omega$ and any $\theta \in [0, 1]$, the ball induced by $\|\cdot\|$ which is centered at $\theta u + (1 - \theta) v$ with radius $\theta(1 - \theta) F_2 \|u - v\|_2$ is also included in $\Omega$.

Definition 2 (Quasi-convex functions). A function $f : \mathbb{R}^d \rightarrow \mathbb{R}$ is quasi-convex if for all $u, v \in \mathbb{R}^d$ such that $f(u) \leq f(v)$, it follows that $\langle \nabla f(v), u - v \rangle \leq 0$, where $\langle \cdot, \cdot \rangle$ is the standard inner product.

Definition 3 (Strictly-quasi-convex functions). A function $f : \mathbb{R}^d \rightarrow \mathbb{R}$ is strictly-quasi-convex if it is quasi-convex and its gradients only vanish at the global minimum. That is, for all $u \in \mathbb{R}^d$, it follows that $f(u) > f(u^*) \Rightarrow \|\nabla f(u)\| \neq 0$ where $u^*$ is the global minimum.

Definition 4 (Strictly-locally-quasi-convex functions). Let $u, v \in \mathbb{R}^d, \kappa, \varepsilon > 0$. Further, write $B_r(x)$ as the Euclidean norm ball centered at $x$ of radius $r$ where $x \in \mathbb{R}^d$ and $r \in \mathbb{R}$. We say $f : \mathbb{R}^d \rightarrow \mathbb{R}$ is $(\kappa, \varepsilon, v)$-strictly-locally-quasi-convex in $u$ if at least one of the following applies:

1. $f(u) - f(v) \leq \varepsilon$
2. $\|\nabla f(v)\| > 0$ and for every $y \in B_{2\varepsilon}(v)$ it holds that $\langle \nabla f(u), y - u \rangle \leq 0$

3.2 A Brief Overview of Frank-Wolfe (FW)

The Frank-Wolfe (FW) algorithm (Algorithm 1) attempts to solve the constrained optimization problem $\min_{x \in \Omega} f(x)$ for some convex constraint set $\Omega$ (a.k.a. feasible set) and some function $f : \Omega \rightarrow \mathbb{R}$. FW begins with an initial solution $w_0 \in \Omega$. Then, at each iteration, it computes a search direction $v_t$ by minimizing the linear approximation of $f$ at $w_t$, $v_t = \min_{v \in \Omega} \langle \nabla f(w_t), v \rangle$, where $\nabla f(w_t)$ is the gradient of $f$ at $w_t$. Next, FW produces a convex combination of the current iterate $w_t$ and the search direction $v_t$ to find the next iterate $w_{t+1} = (1 - \gamma_t)w_t + \gamma_tv_t$ where $\gamma_t \in [0, 1]$ is the learning rate for the current iteration. There are a number of ways to choose the learning rate $\gamma_t$. Chief among these are setting $\gamma_t = \frac{2}{t+1}$ (Algorithm 1 option A) or finding $\gamma_t$ via line search (Algorithm 1 option B).

4 Faster Convergence Rate for Smooth Convex Functions

4.1 Primal Averaging (PA)

PA ([Lan 2013]) is a variant of FW that operates in a style similar to Nesterov’s acceleration method. PA maintains three sequences, $(z_{t-1})_{t=1,2,...}$, $(v_{t-1})_{t=1,2,...}$, and $(w_{t-1})_{t=1,2,...}$. The first is the accelerating sequence (as in Nesterov acceleration), the second is the sequence of search directions, and the third is the sequence of solution vectors. At each iteration, PA updates its sequences by computing two convex combinations and consulting the linear oracle, such that

$z_{t-1} = (1 - \gamma_{t-1})w_{t-1} + \gamma_tv_{t-1}$
Algorithm 1 Standard Frank-Wolfe algorithm

1: Input: loss $f : \Omega \to \mathbb{R}$.
2: Input: linear opt. oracle $\mathcal{O}(\cdot)$ for $\Omega$.
3: Initialize: any $w_1 \in \Omega$.
4: for $t = 1, 2, 3, \ldots$ do
5:  $v_t = \mathcal{O}(\nabla f(w_t)) = \arg\min_{v \in \Omega} \langle v, \nabla f(w_t) \rangle$.
6:  Option (A): Predefined decay learning rate $\{\gamma_t \in [0, 1]\}_{t=1, 2, \ldots}$.
7:  Option (B): $\gamma_t = \arg\min_{\gamma \in [0, 1]} \gamma \langle v_t - w_t, \nabla f(w_t) \rangle + \gamma^2 \frac{L^2}{2} \|v_t - w_t\|^2$.
8:  $w_{t+1} \leftarrow (1 - \gamma_t)w_t + \gamma_tw_t$.
9: end for

Algorithm 2 Primal Averaging

1: Initialize any $v_0 \in \Omega \subset \mathbb{R}^d$. Set $w_0 = v_0$.
2: for $t = 1, 2, 3, \ldots$ do
3:  $\gamma_t = \frac{1}{t+1}$.
4:  $z_{t-1} = (1 - \gamma_t)w_{t-1} + \gamma_tw_t$.
5:  Option (A): $p_t = \frac{\sum_{i=1}^t \theta_i}{\gamma_t}$, where $\Theta_t = \sum_{i=1}^t \theta_i$, $\theta_t = t$, and $\frac{\theta_t}{\theta_t} = \gamma_t$.
6:  Option (B): $p_t = \nabla f(z_{t-1})$.
7:  $v_t = \arg\min_{v \in \Omega} \langle v, p_t \rangle$.
8:  $w_t = (1 - \gamma_t)w_{t-1} + \gamma_tw_t$.
9: end for

\[ v_t = \arg\min_{v \in \Omega} \langle \Theta_t^{-1} \sum_{i=1}^t \theta_i \nabla f(z_{t-1}), v \rangle \]
\[ w_t = (1 - \gamma_t)w_{t-1} + \gamma_tw_t \]
where $\Theta_t = \sum_{i=1}^t \theta_i$ and the $\theta_i$ are chosen, such that $\gamma_t = \frac{\theta_t}{\theta_t}$ for all $t$. Note that choosing $\theta_t$ does not require significant computation as setting $\theta_t = t$ satisfies the requirement $\gamma_t = \frac{\theta_t}{\theta_t}$.

Since $z_{t-1}$ and $w_t$ are convex combinations of elements of the constraint set $\Omega$, $z_{t-1}$ and $w_t$ are themselves in $\Omega$. While the input to the linear oracle is a single gradient vector in standard FW, PA uses an average of the gradients seen in iterations $1, 2, \ldots , t$ as the input to the linear oracle.

In standard FW, the sequence $(w_t)_{t=1, 2, \ldots}$ has the following property (Jaggi 2013; Lan 2013; Hazan and others 2016):

\[ f(w_t) - f(w^*) \leq \frac{2L}{t(t+1)} \sum_{i=1}^t \| v_i - w_{i-1} \|^2 \]  

where $w^*$ is an optimal point and $L$ is the smoothness parameter of $f$. We observe that the $\frac{1}{t} \sum_{i=1}^t \| v_i - w_{i-1} \|$ factor of the average distance between the search direction and solution vector pairs. Denote the diameter of $\Omega$ as $D = \sup_{u, v \in \Omega} \| u - v \|$. Then, since $w_1$ and $w_t$ are both in $\Omega$, we find that $\frac{1}{t} \sum_{i=1}^t \| v_i - w_{i-1} \| \leq D$. That is, the average distance of $v_i$ and $w_{i-1}$ is upper bounded by diameter $D$ of $\Omega$. Combining this with (1) yields standard FW’s convergence rate:

\[ f(w_t) - f(w^*) \leq \frac{2L}{t(t+1)} \sum_{i=1}^t \| v_i - w_{i-1} \|^2 \]

PA has a similar guarantee for the sequence $(w_t)_{t=1, 2, \ldots}$ (Lan 2013). Namely

\[ f(w_t) - f(w^*) \leq \frac{2L}{t(t+1)} \sum_{i=1}^t \| v_i - w_{i-1} \|^2 \]

While the inability to guarantee an arbitrarily small distance between $v_i$ and $w_t$ in Equation (1) caused standard FW to converge as $O\left(\frac{1}{t}\right)$, this is not the case for the distance between $v_t$ and $w_t$ in Equation (2). Should we be able to bound the distance $\| v_t - w_t \|$ to be arbitrarily small, we can show that PA converges as $O\left(\frac{1}{t}\right)$ with high probability. We observe that the sequence $(v_t)_{t=1, 2, \ldots}$ expresses this behavior when the constraint set is strongly convex. We have the following theorem:

**Theorem 1.** Assume the convex function $f$ is smooth with parameter $L$. Further, define the function $h$ as $h(w) = f(w) + \theta \xi^Tw$ where $\theta \in [0, \frac{1}{2L}]$, $\xi \in \mathbb{R}^d$, $w \in \Omega$, $\Omega$ is an $\alpha$-strongly convex set, $D$ is the diameter of $\Omega$, and $\xi$ is uniform on the unit sphere. Applying PA to $h$ yields the following convergence rate for $f$ with probability $1 - \delta$,

\[ f(w_t) - f(w^*) = O\left(\frac{dL}{\alpha^2\delta^2D^2}\right) \]

**Theorem 1** states that applying PA to a perturbed function $h$ over an $\alpha$-strongly convex constraint set allows any smooth,
convex function \( f \) to converge as \( O \left( \frac{1}{\delta} \right) \) with probability \( 1 - \delta \), albeit depending on \( \delta \) and \( d \). However, as \( t \) grows, the \( t^2 \) term in the convergence rate’s denominator quickly dominates the rate’s \( \delta \) and \( d \) terms. This, combined with PA’s non-reliance on line search, allows it to outperform the method proposed in (Garber and Hazan 2015). We note that, although Theorem 1 requires us to run PA on the perturbed function \( h, f \) itself still converges as \( O \left( \frac{1}{\delta} \right) \) with high probability. That is, the iterates \( w_t \) produced by running PA on \( h \) themselves have the guarantee of \( f(w_t) - f(w^*) = O \left( \frac{dL}{\sqrt{n\delta}} \right) \) for \( w^* = \arg\min f(w) \) with probability \( 1 - \delta \). We also empirically investigate this result in Section 4.2.

4.2 Stochastic Primal Averaging (SPA)

Here we provide a stochastic version of Primal Averaging. While in the previous section we studied PA with Option (A) of Algorithm 2, we now consider PA with Option (B) of Algorithm 2 providing an analysis of its stochastic version. That is, \( p_t = \nabla f(z_{t-1}) \), where \( \nabla f \) represents the aggregated stochastic gradient constructed as \( \nabla f(z_{t-1}) = \sum_{i \in S_t} \nabla f_i(z_{t-1}) \). Further, \( \nabla f_i(\cdot) \) is the stochastic gradient computed with the \( i \)th item of a dataset of size \( N \), while \( S_t \) is the set of indices sampled without replacement from \( \{1, 2, \ldots, N\} \) at iteration \( t \). We note that \( |S_t| = \min(t^4, N) \).

**Theorem 2.** Assume the convex function \( f \) is smooth with parameter \( L \). Denote \( \alpha \) as the variance of a stochastic gradient. Suppose \( p_t = \nabla f(z_{t-1}) \) and the number of samples used to obtain \( p_t \) is \( n_t = O(t^4) \). Further, define the function \( h \) as \( h(w) = f(w) + \theta^T w \) where \( \theta \in (0, \frac{\pi}{4\sqrt{D}}) \), \( \xi \in \mathbb{R}^d \), \( w \in \Omega \), \( \Omega \) is an \( \alpha \)-strongly-convex set, \( D \) is the diameter of \( \Omega \), and \( \xi \) is uniform on the unit sphere. Then applying PA to \( h \) yields the following convergence rate for \( f \) with probability \( 1 - \delta \):

\[
E[f(w_t)] - f(w^*) = O \left( \frac{dL^2(\alpha^2 + \sigma^2) \log t}{\alpha^2 \sigma^2 t^2} \right)
\]

Theorem 2 states that the stochastic version of PA maintains an \( O \left( \frac{\log t}{t^2} \right) \) convergence rate with high probability, using \( h \) in a manner similar to Theorem 1. Note that \( n_t \) grows as \( O(t^4) \) until it begins to use all the data points to compute the gradient. Thus, for earlier iterations of SPA, the algorithm requires far less computation than its deterministic counterpart. However, the samples required in each iteration grows quickly, causing later iterations of SPA to share the same computational cost as deterministic Primal Averaging.

5 Strictly-Locally-Quasi-Convex Functions

In this section we show that FW with line search can converge within an \( \epsilon \)-neighborhood of the global minimum for strictly-locally-quasi-convex functions. Furthermore, if it is assumed that the norm of the gradient is lower bounded, then FW with line search can converge within an \( \epsilon \)-neighborhood of the global minimum in \( O \left( \max(\frac{1}{\epsilon^2}, \frac{1}{\epsilon^2}) \right) \) iterations.

**Theorem 3.** Assume that the function \( f \) is smooth with parameter \( L \), and that \( f \) is \((\epsilon, \kappa, w^*)\)-strictly-locally-quasi-convex, where \( w^* \) is a global minimum. Then, the standard FW algorithm with line search (Algorithm 1 option (B)) can converge within an \( \epsilon \)-neighborhood of the global minimum when the constraint set is strongly convex. Furthermore, if one assumes that \( f(w) - f(w^*) \geq \epsilon \) implies that the norm of the gradient is lower bounded as \( \|\nabla f(w)\| \geq \theta_\epsilon \) for some \( \theta \in \mathbb{R} \), then the algorithm needs \( t = O(\max(\frac{\pi}{4\sqrt{D}}, \frac{8L\kappa}{\theta_\epsilon^2})) \) iterations to produce an iterate that is within an \( \epsilon \)-neighborhood of the global minimum.

Hazan et al. (2015) provide several examples of strictly-locally-quasi-convex functions. First, if \( \epsilon \in (0, 1) \) and \( x \in \{-x_1, x_2\} \in [-10, 10]^2 \), then the function

\[
g(x) = (1 + e^{-x_1})^{-1} + (1 + e^{-x_2})^{-1}
\]

is \( (\epsilon, 1, x^*) \)-strictly-locally-quasi-convex in \( x \). Second, if \( \epsilon \in (0, 1) \) and \( w \in \mathbb{R}^d \), then the function

\[
h(w) = \frac{1}{m} \sum_{i=1}^{m} (y_i - \phi(\langle w, x_i \rangle))^2
\]

is \( (\epsilon, \frac{\sigma}{\sqrt{d}}, w^*) \)-strictly-locally-quasi-convex in \( w \). Here, \( \phi(z) = \mathbb{1}_{z \geq 0}, \gamma \in \mathbb{R} \) is the margin of a perceptron, and we have \( m \) samples \( \{x_i, y_i\}_{i=1}^{m} \in B_1(0) \times \{0, 1\} \) where \( B_1(0) \subset \mathbb{R}^d \).

6 Smooth Non-Convex Functions

In this section, we show that, with high probability, FW with line search converges as \( O \left( \frac{1}{t^2} \right) \) to a stationary point when the loss function is non-convex and the constraint set is strongly convex. To our knowledge, a rate this rapid does not exist in the non-convex optimization literature.

To help demonstrate our theoretical guarantee, we introduce a measure called the FW gap. The FW gap of \( f \) at a point \( w_t \in \Omega \) is defined as \( k_t := \max_{m \in \mathbb{N}} \langle v - w_t, -\nabla f(w_t) \rangle \). This measure is adopted in (Lacoste-Julien 2016), which is the first work to show that, for smooth non-convex functions, FW has an \( O \left( \frac{1}{\sqrt{t}} \right) \) convergence rate to a stationary point over arbitrary convex sets. The \( O \left( \frac{1}{\sqrt{t}} \right) \) rate matches the rate of projected gradient descent when the loss function is smooth and non-convex. It has been shown (Lacoste-Julien 2016) that a point \( w_t \) is a stationary point for the constrained optimization problem if and only if \( k_t = 0 \).

**Theorem 4.** Assume that the non-convex function \( f \) is smooth with parameter \( L \) and the constraint set \( \Omega \) is \( \alpha \)-strongly-convex and has dimensionality \( d \). Further, define the function \( h \) as \( h(w) = f(w) + \theta^T w \) where \( \theta \in (0, \frac{\pi}{4\sqrt{D}}), \xi \in \mathbb{R}^d \), \( w \in \Omega \), \( D \) is the diameter of the \( \Omega \), and \( \xi \) is uniform on the unit sphere. Let \( \ell_1 = f(w_1) - f(w^*) + C' = \frac{m\min \{\epsilon, C\}}{8L\sqrt{d}} \). Then applying FW with line search to \( h \) yields the following guarantee for the FW gap of \( f \) with probability \( 1 - \delta \):

\[
\min_{1 \leq s \leq t} k_s \leq \frac{\ell_1}{t \min \{\frac{1}{2}, C'\}} = O \left( \frac{1}{t} \right)
\]

We would further discuss the result stated in the theorem. In non-convex optimization literature, Nesterov and Polyak
| Convexity of Loss Function | Loss Function | Constraint | Task   |
|---------------------------|--------------|------------|--------|
| Convex                    | Quadratic Loss | $l_2$ norm | Regression |
| Strictly-Locally-Quasi-Convex | Observed Quadratic Loss | Schatten-$p$ norm | Matrix Completion |
| Non-Convex                | Squared Sigmoid | $l_p$ norm | Classification |
|                           | Bi-Weight Loss   | $l_p$ norm | Robust Regression |

Table 2: Various loss functions and constraint sets used in our experiments.

We refer the interested reader to Appendix B for additional details.

We have conducted extensive experiments on different combinations of loss functions, constraint sets, and real-life datasets (Table 2). Here, we only report two main sets of experiments: the empirical validation of our theoretical results in terms of convergence rates (Section 7.1) and the comparison of various optimizations in terms of actual run times (Section 7.2). We refer the interested reader to Appendix B for additional experiments.

For classification and regression, we used the logistic and quadratic loss functions. For matrix completion, we used the observed quadratic loss (Freund, Grigas, and Mazumder 2017), defined as $f(X) = \sum_{(i,j) \in P(M)} (X_{i,j} - M_{i,j})^2$, where $X$ is the estimated matrix, $M$ is the observed matrix, and $P(M) = \{(i,j) : M_{i,j}$ is observed$\}$. As a non-convex, but strictly-locally-quasi-convex loss, we also used the squared sigmoid loss $\varphi(z) = (1 + \exp(-z))^{-1}$ (Hazar, Levy, and Shalev-Shwartz 2015) for classification. For robust regression, we used the bi-weight loss (Belagiannis et al. 2015), as a non-convex (but smooth) loss $\psi(f(x_i), y_i) = (f(x_i) - y_i)^2 / (1 + (f(x_i) - y_i)^2)$.

For regression, we used the YearPredictionMSD dataset (500K observations, 90 features) (Lichman 2013). For classification, we used the Adult dataset (49K observations, 14 features) (Lichman 2013). For matrix completion, we used the MovieLens dataset (1M movie ratings from 6,040 users on 3,900 movies) (Harper and Konstan 2016).

### 7.1 Empirical Validation of Convergence Rates

We ran several experiments to empirically validate our convergence results. In particular, we studied the performance of Primal Averaging (PA) and standard FW with Line Search (FWLS) with both $l_2$ and Schatten-2 norm balls as our strongly convex constraint sets.

Theorem 1 guarantees a convergence rate of $O(\frac{1}{t^2})$ for PA when the constraint set is strongly convex and the loss function is convex. We experimented with both $l_2$ (logistic classifier) and Schatten-2 norm (matrix completion) balls, measuring the loss value at each iteration. As shown in Figure 1a, a slope of $-2.41$ confirms Theorem 1’s guarantee, which predicts a slope of at least $-2$.

Theorem 3 shows that FWLS converges to the global minimum at the rate of $O\left(\min\left(\frac{1}{T^2}, \frac{1}{T^4}\right)\right)$ when the constraint set is strongly convex and the loss function is strictly-locally-quasi-convex. We investigated this result with the squared sigmoid loss and an $l_2$ norm constraint. Figure 1b exhibits...
our results, showing a slope of $-2.12$, a finding better than the worst-case bounds given by Theorem 5, i.e., a slope of $-0.5$ (see appendix for a detailed discussion).

From Theorem 3, we expect FWLS to converge to a stationary point of a (smooth) non-convex function at a rate of $O(\frac{1}{t})$ when constrained to a strongly convex set. Using the bi-weight loss and an $l_2$ norm constraint, we measured the loss value at each iteration. As shown in Figure 1c, the results confirmed our theoretical results, showing an even steeper slope ($-1.46$ instead of $-1$, since Theorem 4 only provides a worst-case upper bound).

### 7.2 Comparison of Different Optimization Algorithms

To compare the actual performance of various optimization algorithms, we measure the run times, instead of the number of iterations to convergence, in order to account for the time spent in each iteration. In Figure 2, dotted vertical lines mark the convergence points of various algorithms.

First, we compared all three variants of FW: PA, standard FW With Predefined Learning Rate (FWPLR) defined in Algorithm 1 with option A, and standard FW With Line Search (FWLS) defined in Algorithm 1 with option B. All methods were tested on a regression task (quadratic loss) with an $l_2$ norm ball constraint.

As shown in Figure 2a, PA converged $3.7 \times$ and $15.6 \times$ faster than FWPLR and FWLS, respectively. This considerable speedup has significant ramifications in practice. Traditionally, PA has been shied away from, due to its slower iterations, while its convergence rate was believed to be the same as the more efficient variants (Lan 2013). However, as proven in Section 7, PA does converge in fewer iterations.

We also compared the run time of PA versus projected gradient descent (regression task with a quadratic loss). We compared their deterministic versions in Figure 2b, where PA converged significantly faster ($7.7 \times$), as expected. For a fair comparison of their stochastic versions, Stochastic Primal Averaging (SPA) and Stochastic Gradient Descent (SGD), we considered two cases: an $l_2$ constraint (which has an efficient projection) and $l_{2,1}$ constraint (which has a costly projection). As expected, for an efficient projection, SGD converged $4.6 \times$ faster than SPA (Figure 2c), and when the projection was costly, SPA converged $25.1 \times$ faster (see Appendix B for detailed plots).

### 8 Conclusion

In this paper, we revisited an important class of optimization techniques, FW methods, and offered new insight into their convergence properties for strongly convex constraint sets, which are quite common in machine learning. Specifically, we discovered that, for convex functions, a non-conventional variant of FW (i.e., Primal Averaging) converges significantly faster than the commonly used variants of FW with high probability. We also showed that PA’s $O(\frac{1}{t})$ convergence rate more than compensates for its slightly more expensive computational cost at each iteration. We also proved that for strictly-locally-quasi-convex functions, FW can converge to within an $\epsilon$-neighborhood of the global minimum in $O\left(\max\left(\frac{1}{\epsilon}, \frac{1}{\sqrt{t}}\right)\right)$ iterations. Even for non-convex functions, we proved that FW’s convergence rate is better than the previously known results in the literature with high probability. These new convergence rates have significant ramifications for practitioners, due to the widespread applications of strongly convex norm constraints in classification, regression, matrix completion, and collaborative filtering. Finally, we conducted extensive experiments on real-world datasets to validate our theoretical results and investigate our improvements over existing methods. In summary, we showed that PA reduces optimization time by $2.8$–$15.6 \times$ compared to standard FW variants, and by $7.7$–$25.1 \times$ compared to projected gradient descent. Our plan is to integrate PA in machine learning libraries libraries, including our BlinkML project (Park et al. 2018).

### 9 Acknowledgments

This work is in part supported by the National Science Foundation (grants 1629397 and 1553169).
References

[Agarwal et al. 2017] Agarwal, N.; Allen-Zhu, Z.; Bullins, B.; Hazan, E.; and Ma, T. 2017. Finding approximate local minima for nonconvex optimization in linear time. STOC.

[Agarwal, Negahban, and Wainwright 2010] Agarwal, A.; Negahban, S.; and Wainwright, M. J. 2010. Fast global convergence rates of gradient methods for high-dimensional statistical recovery. In Advances in Neural Information Processing Systems, 37–45.

[Beck and Teboulle 2004] Beck, A., and Teboulle, M. 2004. A conditional gradient method with linear rate of convergence for solving convex linear systems. Mathematical Methods of Operations Research 59(2):235–247.

[Belagiannis et al. 2015] Belagiannis, V.; Rupprecht, C.; Carneiro, G.; and Navab, N. 2015. Robust optimization for deep regression. In Proceedings of the IEEE International Conference on Computer Vision, 2830–2838.

[Carmon et al. 2017] Carmon, Y.; Duchi, J.; Hinder, O.; and Yuan, Y. 2015. A conditional gradient method with linear rate of convergence for solving convex linear systems. Mathematical Methods of Operations Research 59(2):235–247.

[Candès and Recht 2009] Candès, E. J., and Recht, B. 2009. Exact matrix completion via convex optimization. Foundations of Computational mathematics 9(6):717.

[Carmon et al. 2017] Carmon, Y.; Duchi, J.; Hinder, O.; and Sidford, A. 2017. Accelerated methods for non-convex optimization. https://arxiv.org/pdf/1611.00756.pdf.

[Chari et al. 2015] Chari, V.; Lacoste-Julien, S.; Laptev, I.; and Sivic, J. 2015. On pairwise costs for network flow multi-object tracking. CVPR.

[Clarkson 2010] Clarkson, K. L. 2010. Coresets, sparse greedy approximation, and the frank-wolfe algorithm. ACM Transactions on Algorithms (TALG) 6(4):63.

[Demyanov and Rubinov 1970] Demyanov, V. F., and Rubinov, A. M. 1970. Approximate methods in optimization problems. Elsevier Publishing Company.

[Dunn 1979] Dunn, J. C. 1979. Rates of convergence for conditional gradient algorithms near singular and nonsingular extremals. SIAM Journal on Control and Optimization.

[Frank and Wolfe 1956] Frank, M., and Wolfe, P. 1956. An algorithm for quadratic programming. Naval research logistics quarterly 3(1-2):95–110.

[Freund, Grigas, and Mazumder 2017] Freund, R. M.; Grigas, P.; and Mazumder, R. 2017. An extended frank–wolfe method with “in-face” directions, and its application to low-rank matrix completion. SIAM Journal on Optimization 27(1):319–346.

[Garber and Hazan 2015] Garber, D., and Hazan, E. 2015. Faster rates for the frank–wolfe method over strongly-convex sets. In International Conference on Machine Learning, 541–549.

[Ge et al. 2015] Ge, R.; Huang, F.; Jin, C.; and Yuan, Y. 2015. Escaping from saddle points – online stochastic gradient for tensor decomposition. COLT.

[Ghadimi and Lan 2016] Ghadimi, S., and Lan, G. 2016. Accelerated gradient methods for nonconvex nonlinear and stochastic programming. Mathematical Programming 156(1-2):59–99.

[Grant and Boyd ] Grant, M., and Boyd, S. Cvx: Matlab software for disciplined convex programming.

[Grant and Boyd 2008] Grant, M. C., and Boyd, S. P. 2008. Graph implementations for nonsmooth convex programs. In Recent advances in learning and control. Springer. 95–110.

[Harchaoui et al. 2012] Harchaoui, Z.; Douze, M.; Paulin, M.; Dudik, M.; and Malick, J. 2012. Large-scale image classification with trace-norm regularization. In Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on, 3386–3393. IEEE.

[Harper and Konstan 2016] Harper, F. M., and Konstan, J. A. 2016. The movielens datasets: History and context. ACM Transactions on Interactive Intelligent Systems (TiiS) 5(4):19.

[Hazan and others 2016] Hazan, E., et al. 2016. Introduction to online convex optimization. Foundations and Trends® in Optimization 2(3-4):157–325.

[Hazan and Kale 2012] Hazan, E., and Kale, S. 2012. Projection-free online learning. arXiv preprint arXiv:1206.4657.

[Hazan and others 2016] Hazan, E., et al. 2016. Introduction to online convex optimization. Foundations and Trends® in Optimization 2(3-4):157–325.

[Hazan, Kale, and Warmuth 2010] Hazan, E.; Kale, S.; and Warmuth, M. K. 2010. Learning rotations with little regret. In COLT, 144–154.

[Hazan, Levy, and Shalev-Shwartz 2015] Hazan, E.; Levy, K. Y.; and Shalev-Shwartz, S. 2015. Beyond convexity: Stochastic quasi-convex optimization. Neural Information Processing Systems.

[Huang and Chen 2011] Huang, W., and Chen, D.-R. 2011. Gradient iteration with lp-norm constraints. Journal of Mathematical Analysis and Applications.

[Huang et al. 2016] Huang, R.; Lattimore, T.; György, A.; and Szepesvari, C. 2016. Following the leader and fast rates in linear prediction: Curved constraint sets and other regularities.

[Jaggi, Sulovsk, and others 2010] Jaggi, M.; Sulovsk, M.; et al. 2010. A simple algorithm for nuclear norm regularized problems. In Proceedings of the 27th international conference on machine learning (ICML-10), 471–478.

[Jaggi 2011] Jaggi, M. 2011. Sparse convex optimization methods for machine learning. Technical report, ETH Zürich.

[Jaggi 2013] Jaggi, M. 2013. Revisiting frank-wolfe: Projection-free sparse convex optimization. In ICML (1), 427–435.

[Kalai and Sastry 2009] Kalai, A. T., and Sastry, R. 2009. The isotron algorithm: High-dimensional isotonic regression. In COLT.

[Kim and Xing 2010] Kim, S., and Xing, E. P. 2010. Tree-guided group lasso for multi-task regression with structured sparsity.

[Koltchinskii et al. 2011] Koltchinskii, V.; Lounici, K.; Tsybakov, A. B.; et al. 2011. Nuclear-norm penalization and
optimal rates for noisy low-rank matrix completion. The Annals of Statistics 39(5):2302–2329.

[Lacoste-Julien and Jaggi 2015] Lacoste-Julien, S., and Jaggi, M. 2015. On the global linear convergence of frank-wolfe optimization variants. In Advances in Neural Information Processing Systems, 496–504.

[Lacoste-Julien et al. 2013] Lacoste-Julien, S.; Jaggi, M.; Schmidt, M.; and Pletscher, P. 2013. Block-coordinate frank-wolfe optimization for structural svms. ICML.

[Lacoste-Julien 2016] Lacoste-Julien, S. 2016. Convergence rate of frank-wolfe for non-convex objectives., arXiv:1607.00345.

[Lan 2013] Lan, G. 2013. The complexity of large-scale convex programming under a linear optimization oracle. https://arxiv.org/abs/1309.5550.

[Lee et al. 2016] Lee, J. D.; Simchowitz, M.; Jordan, M. I.; and Recht, B. 2016. Gradient descent converges to minimizers. COLT.

[Lewis 1966] Levitin, E. S., and Polyak, B. T. 1966. Constrained minimization methods. USSR Computational mathematics and mathematical physics.

[Lichman 2013] Lichman, M. 2013. UCI machine learning repository.

[Liu et al. 2015] Liu, J.; Wright, S. J.; Ré, C.; Bittorf, V.; and Sridhar, S. 2015. An asynchronous parallel stochastic coordinate descent algorithm. The Journal of Machine Learning Research 16(1):285–322.

[Nesterov and Polyak 2006] Nesterov, Y., and Polyak, B. T. 2006. Cubic regularization of newton method and its global performance. Mathematical Programming 108(1):177–205.

[Neter et al. 1996] Neter, J.; Kutner, M. H.; Nachtsheim, C. J.; and Wasserman, W. 1996. Applied linear statistical models, volume 4. Irwin Chicago.

[Osokin et al. 2016] Osokin, A.; Alayrac, J.-B.; Lukasewitz, I.; Dokania, P.; and Lacoste-Julien, S. 2016. Minding the gaps for block frank-wolfe optimization of structured svms. In International Conference on Machine Learning, 593–602.

[Park et al. 2018] Park, Y.; Qing, J.; Shen, X.; and Moazafari, B. 2018. BlinkML: Approximate machine learning with probabilistic guarantees y. Technical Report http://web.eecs.umich.edu/~mozafari/php/data/uploads/blinkml_report.pdf.

[Recht and Ré 2013] Recht, B., and Ré, C. 2013. Parallel stochastic gradient algorithms for large-scale matrix completion. Mathematical Programming Computation 5(2):201–226.

[Recht et al. 2011] Recht, B.; Re, C.; Wright, S.; and Niu, F. 2011. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In Advances in neural information processing systems, 693–701.

[Reddi et al. 2016] Reddi, S. J.; Sra, S.; Poczos, B.; and Smola, A. 2016. Stochastic frank-wolfe methods for non-convex optimization. Allerton.

[Shalev-Shwartz, Gonen, and Shamir 2011] Shalev-Shwartz, S.; Gonen, A.; and Shamir, O. 2011. Large-scale convex minimization with a low-rank constraint. arXiv preprint arXiv:1106.1622.

[Sun and Yuan 2006] Sun, W., and Yuan, Y.-X. 2006. Optimization theory and methods: nonlinear programming, volume 1. Springer Science & Business Media.

[Wang et al. 2016] Wang, Y.-X.; Sadhanala, V.; Dai, W.; Neiswanger, W.; Sra, S.; and Xing, E. 2016. Parallel and distributed block-coordinate frank-wolfe algorithms. In International Conference on Machine Learning, 1548–1557.

[Yu, Zhang, and Schuurmans 2014] Yu, Y.; Zhang, X.; and Schuurmans, D. 2014. Generalized conditional gradient for structured estimation. arXiv:1410.4828.
\section*{A Proofs}

\subsection*{A.1 Proof of Theorem\cite{huang2016stochastic}}

We begin by providing two lemmas which aid in our proof of Theorem\cite{huang2016stochastic}. In particular, Lemma\cite{huang2016stochastic} allows us to upper-bound the distance between two outputs of the linear oracle by a scaled distance of the oracle’s inputs. Lemma\cite{huang2016stochastic} shows that if running PA on an L-smooth function \( f \) allows \( f \) to converge as

\[
\frac{\alpha^2}{\alpha^2 + \delta^2 + \epsilon^2} L \|
\frac{\alpha^2}{\alpha^2 + \delta^2 + \epsilon^2} L \|
\]

then running PA on a perturbed function \( h \) allows \( f \) to converge as

\[
\frac{\alpha^2}{\alpha^2 + \delta^2 + \epsilon^2} L \|
\frac{\alpha^2}{\alpha^2 + \delta^2 + \epsilon^2} L \|
\]

with probability \( 1 - \delta \). Here, \( g \) is the smallest value of the norm of averaged gradients and \( f^* = \min_{w \in \Omega} f(w) \).

Given this, our proof of Theorem\cite{huang2016stochastic} proceeds by first showing that running PA on an L-smooth function \( f \) over an \( \alpha \)-strongly convex constraint set \( \Omega \) causes \( f \) to converge as

\[
\frac{\alpha^2}{\alpha^2 + \delta^2 + \epsilon^2} L \|
\frac{\alpha^2}{\alpha^2 + \delta^2 + \epsilon^2} L \|
\]

We then apply Lemma\cite{huang2016stochastic} thereby showing that running PA on a perturbed function \( h \) allows \( f \) to converge as

\[
\frac{\alpha^2}{\alpha^2 + \delta^2 + \epsilon^2} L \|
\frac{\alpha^2}{\alpha^2 + \delta^2 + \epsilon^2} L \|
\]

with probability \( 1 - \delta \).

We now state the Lemmas and provide their proofs.

**Lemma 1.** Denote

\[
\begin{align*}
x_p &= \arg \max_{x \in \Omega} \langle p, x \rangle \\
x_q &= \arg \max_{x \in \Omega} \langle q, x \rangle
\end{align*}
\]

where \( p, q \in \mathbb{R}^d \) are any non-zero vectors. If a compact set \( \Omega \) is an \( \alpha \)-strongly convex set, then

\[
\| x_p - x_q \| \leq \frac{\| p - q \|}{\alpha (\| p \| + \| q \|)}
\]

**Proof.** It is shown in Proposition A.1 of \cite{huang2016stochastic} that an \( \alpha \)-strongly convex set can be expressed as the intersection of infinitely many Euclidean balls. Denote the \( d \) dimensional unit sphere as \( U = \{ u \in \mathbb{R}^d : \| u \|^2 = 1 \} \). Furthermore, write \( x_u = \arg \max_{x \in \Omega} \langle x, u \rangle \) for some \( u \in U \). Then the \( \alpha \)-strongly convex set \( \Omega \) can be written

\[
\Omega = \bigcap_{u \in U} \mathbb{B}_{\frac{\alpha}{\sqrt{2d}}} (x_u - \frac{u}{\alpha})
\]

Now, let

\[
\begin{align*}
x_p &= \arg \max_{x \in \Omega} \langle \frac{p}{\| p \|}, x \rangle \\
x_q &= \arg \max_{x \in \Omega} \langle \frac{q}{\| q \|}, x \rangle
\end{align*}
\]

and

Based on the above interpretation of strongly convex sets, we see that

\[
x_q \in \mathbb{B}_{\frac{\alpha}{\sqrt{2d}}} (x_p - \frac{p}{\| p \|})
\]

and

\[
x_p \in \mathbb{B}_{\frac{\alpha}{\sqrt{2d}}} (x_q - \frac{q}{\| q \|})
\]

Therefore,

\[
\| x_q - x_p \| \leq \frac{2}{\alpha} \langle x_p - x_q, \frac{p}{\| p \|} \rangle
\]

which leads to

\[
\| x_p - x_q \| \leq \frac{2}{\alpha} \langle x_p - x_q, \frac{p}{\| p \|} \rangle
\]

which results in

\[
\| x_p - x_q \|^2 \leq \frac{2}{\alpha} \langle x_p - x_q, \frac{q}{\| q \|} \rangle
\]

Summing (5) and (6) then applying the Cauchy-Schwarz inequality completes the proof.

We note that Lemma\cite{huang2016stochastic} also implies that the sub-gradient allows

\[
\delta \sqrt{\frac{\pi}{2d}} \leq \| \nabla h(w) \|
\]

Further, if \( h \) is smooth with parameter \( L \) and applying PA to \( h \) over an \( \alpha \)-strongly convex constraint set \( \Omega \) allows \( h \) to converge as

\[
h(w_t) - h^* = O \left( \frac{L}{\alpha^2 \| p \|^2} \right)
\]

then it also holds that

\[
f(w_t) - f^* = O \left( \frac{Ld}{\alpha^2 \| p \|^2} \right)
\]

with probability \( 1 - \delta \).

**Proof.** Note that \( \| \nabla^2 h(w) \| \geq \| \nabla h(w)\| \) where \( y[i] \) is the \( i \)th component of \( y \in \mathbb{R}^d \). Since \( \nabla h(w)[1] = \nabla f(w)[1] + \theta \xi_i \), \( \xi_i \) is uniform on the unit sphere and \( \theta > 0 \). Then with probability \( 1 - \delta \),

\[
Pr \left( \| \nabla h(w) \| \leq \frac{\theta \delta \sqrt{\pi}}{\sqrt{2d}} \right) \leq Pr \left( \| \xi_i \| \leq \delta \frac{\sqrt{\pi}}{\sqrt{2d}} \right) \leq \delta
\]

as \( \xi_i \sim Beta(\frac{\delta}{\sqrt{2d}}, \frac{1}{\sqrt{2d}}) \). Thus, we can lower bound the gradient norm by \( \frac{\delta \sqrt{\pi}}{\sqrt{2d}} \) with probability \( 1 - \delta \).

Now, denote

\[
h^* = \min_{w \in \Omega} h(w)
\]

\[
w^* = \arg \min_{w \in \Omega} f(w)
\]
Recall our assumption that $h$ is smooth with parameter $L$ and that applying PA to $h$ causes $h$ to converge as

$$h(w_t) - h^* = O\left(\frac{L}{\alpha^2 g^2 t^2}\right)$$

Note that $g \geq \frac{\delta}{\sqrt{D}} \geq 0$, so

$$h(w_t) - h^* = O\left(\frac{Ld}{\alpha^2 g^2 t^2}\right)$$

Further, observe that by the construction of $h$ we have

$$f(w) - h(w) \leq \theta \|w\|$$

for any $w$. Now, whenever $h(w_t) - h^* = O\left(\frac{Ld}{\alpha^2 g^2 t^2}\right) = \epsilon$ we see that

$$f(w_t) \leq f(w_t) + \theta \|w_t\| \leq h^* + \epsilon + \theta \|w_t\| \leq h(w^*) + \epsilon + \theta \|w_t\| \leq f(w^*) + \epsilon + \alpha \|w_t\|$$

where we used the fact that $\sum_{i=1}^{\infty} \frac{1}{t} = \frac{\pi}{2}$ in the final equality.

The first and third lines follow from (7), the second from $h^*$ being the minimum value of $h$, the fourth from $D$ being the diameter of the constraint set, and fifth from the choice of $\theta = \frac{\epsilon}{\pi}$. Thus we obtain the convergence rate of $O\left(\frac{Ld}{\alpha^2 g^2 t^2}\right)$ for $f$ with probability $1 - \delta$.

We now proceed with our proof of Theorem 1.

**Proof.** According to Theorem 8 in [Lan 2013], we already have

$$f(w_t) - f(w^*) \leq \frac{2L}{t(t+1)} \sum_{\tau=1}^{t} \|v_{\tau} - v_{\tau-1}\|^2$$

Fix a $t$. Denote

$$p_t = \frac{1}{\Theta_t} \sum_{i=1}^{t} \theta_i \nabla f(z_{i-1})$$

and

$$p_{t-1} = \frac{1}{\Theta_{t-1}} \sum_{i=1}^{t-1} \theta_i \nabla f(z_{i-1})$$

By using Lemma 1 we have

$$\|v_t - v_{t-1}\| \leq \frac{\|p_t - p_{t-1}\|}{\alpha(\|p_t\| + \|p_{t-1}\|)}$$

Based on the update rule

$$p_t = \Theta_t^{-1}(p_{t-1} \Theta_{t-1} + \theta_t \nabla f(z_{t-1}))$$

$$= \Theta_t - \frac{\theta_t}{\Theta_t} p_{t-1} + \frac{\theta_t}{\Theta_t} \nabla f(z_{t-1})$$

So

$$p_t - p_{t-1} = \gamma_t (\nabla f(z_{t-1}) - p_{t-1})$$

given that $\gamma_t = \frac{\theta_t}{\Theta_t}$. By substituting the result back into (9) and noting that $\gamma_t = O\left(\frac{1}{t}\right)$, we find that

$$\|v_t - v_{t-1}\| \leq \frac{\gamma_t(\|p_{t-1}\| + \|\nabla f(z_{t-1})\|)}{\alpha(\|p_t\| + \|p_{t-1}\|)}$$

$$= O\left(\frac{1}{\alpha t^2}\right), \forall t$$

By combining (8) and (11), we get

$$f(w_t) - f(w^*) \leq \frac{2L}{t(t+1)} \sum_{\tau=1}^{t} \|v_{\tau} - v_{\tau-1}\|^2$$

$$\leq \frac{2L}{t(t+1)} \sum_{\tau=1}^{t} O\left(\frac{1}{\alpha^2 g^2 t^2}\right)$$

$$= O\left(\frac{L}{\alpha^2 g^2 t^2}\right)$$

Finally, applying the result of Lemma 2 yields with probability $1 - \delta$ the convergence rate $f(w_t) - f(w^*) = O\left(\frac{Ld}{\alpha^2 g^2 t^2}\right)$ as claimed. □
A.2 Proof of Theorem 2

Proof. We note that
\[
E[f(w_t)] \leq f(z_{t-1}) + \langle \nabla f(z_{t-1}), w_t - z_{t-1} \rangle \\
+ \frac{L}{2} \|w_t - z_{t-1}\|^2
\]

(0) follows from the fact that, as \(w_t = (1 - \gamma_t)w_{t-1} + \gamma_t v_t\)
and \(z_{t-1} = (1 - \gamma_t)^2w_{t-1} + \gamma_t v_{t-1},\)
\(w_t - z_{t-1} = \gamma_t(v_t - v_{t-1})\)

Furthermore, (1) is implied by the convexity of \(f,\) (2) follows from the application of the linear oracle, and (3) follows again from the convexity of \(f.\) Moreover, by taking the expectation over the randomness, we find that
\[
E[f(w_t)] \leq (1 - \gamma_t)f(w_{t-1}) + \gamma_t f(w^*) \\
+ \frac{L\gamma_t^2}{2} \|v_t - v_{t-1}\|^2 + \gamma_t \sigma_D \sqrt{\frac{\gamma_t}{n_t}}
\]
since
\[
E[\|\nabla f(z_{t-1}) - \tilde{\nabla} f(z_{t-1})\|]\n\leq \sqrt{E[\|\nabla f(z_{t-1}) - \tilde{\nabla} f(z_{t-1})\|^2]}
\leq \sigma/\sqrt{n_t}
\]

To maintain an \(O(1/t)\) convergence rate, \(\frac{\sigma D n_t}{\sqrt{n_t}}\) must decay as \(\frac{L\gamma_t^2}{2} \|v_t - v_{t-1}\|^2.\) Recall that the latter term is \(O(1/t).\)

This implies that \(n_t\) must be \(O(t^6)\) so that \(\gamma_t \frac{\sigma D}{\sqrt{n_t}}\) can decay as \(\frac{1}{t^3}\). However, if \(n_t = O(t^4),\) stochastic Primal Averaging yields the slightly worse \(O\left(\frac{\log t}{t^7}\right)\) convergence rate.

Finally, we can remove the reliance on the \(\min_{t \leq s \leq t} \|\nabla f(z_s)\| + \|\nabla f(z_{s-1})\|\) term in the convergence rate by repeating the analysis given in Lemma 2. As this analysis is very similar to the previously provided analysis of Lemma 2, it is omitted. Then when \(n_t = O(t^4)\) we have with probability \(1 - \delta\)

\[
E[f(w_t)] - f^* = O\left(\frac{dL^2(D^2 + \sigma) \log t}{\alpha^2 \delta^2 t^2}\right)
\]

\(\square\)

A.3 Proof of Theorem 3

We state the following lemma from [Garber and Hazan 2015].

Lemma 3. Write the dual norm as \(\|\cdot\|_{\infty}.\) For iteration \(t\) of FW with line search, if \(L < \frac{\alpha}{4} \|\nabla f(w)\|\) set \(\gamma_t = 1;\) otherwise, set \(\gamma_t = \alpha \|\nabla f(w)\| / (4L).\) Then, under the conditions of Theorem 3, Algortihm 1 option (B) has the following guarantee:

\[
f(w_{t+1}) \leq f(w_t) + \frac{\gamma_t}{2} (w^* - w_t, \nabla f(w_t))
\]

We use Lemma 3 to prove Theorem 3.

Proof. Assume that,

\[
f(w_t) - f(w^*) > \epsilon
\]

Otherwise, the algorithm has reached the \(\epsilon\)-neighborhood of \(w^*.\) By the strictly-locally-quasi-convexity of \(f,\) we must have,

\[
\|\nabla f(w)\| > 0
\]

and for every \(x \in B_{\epsilon/\kappa}(w)\) it holds that,

\[
\langle \nabla f(w), x - w \rangle \leq 0
\]

Now choose a point \(y\) such that,

\[
y = w^* + \frac{\epsilon \nabla f(w_t)}{\kappa \|\nabla f(w_t)\|}
\]

and,

\[
y \in B_{\epsilon/\kappa}(w^*)
\]

Then we have the following,

\[
\langle \nabla f(w_t), y - w_t \rangle \leq 0
\]

\[
\equiv \langle \nabla f(w_t), y - w_t \rangle \leq \frac{\kappa \|\nabla f(w_t)\|}{\epsilon} \langle w - w^* \rangle \
\leq \|\nabla f(w_t)\| \langle w_t - w^* \rangle \geq \frac{\epsilon}{\kappa} \|\nabla f(w_t)\|
\]

(17)
Case 1: \( L < \frac{\alpha \| \nabla f(w_t) \|_*}{4} \). Set \( \gamma_t = 1 \):

\[
f(w_{t+1}) \leq f(w_t) + \gamma_t \langle \nabla f(w_t), w^* - w_t \rangle \]

\[
\leq f(w_t) - \frac{\gamma_t \epsilon}{2 \kappa} \| \nabla f(w_t) \| \]

\[
= f(w_t) - \frac{\epsilon}{2 \kappa} \| \nabla f(w_t) \| \tag{18}
\]

Case 2: \( L \geq \frac{\alpha \| \nabla f(w_t) \|_*}{4} \). Set \( \gamma_t = \frac{\alpha \| \nabla f(w_t) \|_*}{4L} \):

\[
f(w_{t+1}) \leq f(w_t) + \frac{\gamma_t}{2} \| \nabla f(w_t) \|_* \langle \nabla f(w_t), w^* - w_t \rangle \]

\[
= f(w_t) + \frac{\alpha \| \nabla f(w_t) \|_* \| \nabla f(w_t) \|}{16L} \]

\[
f(w_t) - \frac{\alpha \epsilon \| \nabla f(w_t) \|_* \| \nabla f(w_t) \|}{8 \kappa L} \]

\[
\leq f(w_t) - \frac{\alpha \epsilon \| \nabla f(w_t) \|_* ^2}{8 \kappa L} \tag{19}
\]

By \(18\) and \(19\), we observe that the loss function monotonically decreases until it enters an \( \epsilon \)-neighborhood of the global minimum, thereby proving that FW with line search can converge within an \( \epsilon \)-neighborhood of the global minimum. To prove that the algorithm requires \( t = O(max(\frac{1}{L}, \frac{1}{\kappa}) \), we use the additional assumption,

\[
f(w) - f(w^*) \geq \epsilon \rightarrow \| \nabla f(w) \| \geq \theta \epsilon
\]

Now, assume that after iteration \( t \) the algorithm reaches the target \( \epsilon \)-neighborhood. Denote the solution vector at iteration \( t \) as \( w_t \). Then, in case (1), we have,

\[
f(w_t) \leq f(w_1) - \frac{te^2 \theta}{2 \kappa} \tag{20}
\]

or

\[
t \leq \frac{2 \kappa (f(w_1) - f(w_t))}{e^2 \theta} \]

\[
\leq \frac{2 \kappa (f(w_1) - f(w^*))}{e^2 \theta} \tag{21}
\]

while for case (2), we have,

\[
f(w_t) \leq f(w_1) - \frac{te^3 \theta}{8 \kappa L} \tag{22}
\]

or

\[
t \leq \frac{8 L \kappa (f(w_1) - f(w_t))}{e^2 \theta} \]

\[
\leq \frac{8 L \kappa (f(w_1) - f(w^*))}{e^3 \theta} \tag{23}
\]

This shows that it requires \( t = O(max(\frac{1}{L}, \frac{1}{\kappa}) \) iterations for the algorithm to produce an iterate that is within the \( \epsilon \)-neighborhood of the global minimum.

\[
A.4 \quad Proof \ of \ Theorem 4
\]

**Proof.** Let \( w_{\gamma} = w_t + \gamma (p_t - w_t) \) for some \( \gamma \in [0, 1] \). Then, \( f(w_{t+1}) \leq f(w_{\gamma}) \) as \( w_{t+1} \) is obtained by line search and thus uses an optimal step size.

\[
f(w_{t+1}) \leq f(w_t) + \gamma \langle \nabla f(w_t), v_t - w_t \rangle
\]

\[
\leq f(w_t) + \frac{\gamma^2 L \| v_t - w_t \|^2}{2} \tag{24}
\]

where (0) follows from \( v_t = \arg \min_{v \in X} \langle \nabla f(w_t), v \rangle \). Let \( c_t \) above be,

\[
c_t = \frac{u_t + v_t + \alpha \| w_t - v_t \|^2}{8} \tag{25}
\]

where \( c_t \in \Omega \) by the definition of a strongly convex set. Let us write,

\[
u_t = \arg \min_{u \in X} \| \nabla f(w_t) \|_* = -\| \nabla f(w_t) \|_*
\]

where the last equality is obtained by the definition of the dual norm. Then,

\[
(c_t - w_t, \nabla f(w_t)) \leq \frac{1}{2} \| v_t - w_t, \nabla f(w_t) \|
\]

\[
+ \frac{\alpha}{8} \| v_t - w_t \|^2 \| \nabla f(w_t) \|_*
\]

\[
\leq \frac{1}{2} \| v_t - w_t, \nabla f(w_t) \|
\]

\[
- \frac{\alpha}{8} \| v_t - w_t \|^2 \| \nabla f(w_t) \|_*
\]

\[
= -k_t \frac{1}{2} - \frac{\alpha}{8} \| v_t - w_t \|^2 \| \nabla f(w_t) \|_*
\]

where the last line is due to the definition of the FW gap. Combining (24) and (25) gives,

\[
f(w_{t+1}) \leq f(w_t) - \frac{\gamma k_t}{2} + \frac{\| v_t - w_t \|^2}{2} \left( \gamma^2 L - \frac{\alpha \| \nabla f(w_t) \|_*}{4} \right)
\]

Case 1: \( L \leq \frac{\alpha \| \nabla f(w_t) \|_*}{4} \), set \( \gamma = 1 \), we get,

\[
f(w_{t+1}) \leq f(w_t) - \frac{k_t}{2}
\]

Case 2: \( L \geq \frac{\alpha \| \nabla f(w_t) \|_*}{4} \), set \( \gamma = \frac{\alpha \| \nabla f(w_t) \|_*}{4L} \), we get,

\[
f(w_{t+1}) \leq f(w_t) - \frac{\alpha k_t \| \nabla f(w_t) \|_*}{8L}
\]

By recursively applying the above inequality, we get,

\[
f(w_{t+1}) \leq f(w_1) - \sum_{s=1}^{t} \min \left( \frac{k_s}{2} - \frac{\alpha \| \nabla f(w_s) \|_*}{8L} \right)
\]

Denote \( \tilde{k}_t = \min_{1 \leq s \leq t} k_s \). We have,

\[
f(w_{t+1}) \leq f(w_1) - \tilde{k}_t \sum_{s=1}^{t} \min \left( \frac{1}{2} - \frac{\alpha \| \nabla f(w_s) \|_*}{8L} \right)
\]

(26)
Furthermore, if we assume that,

$$\| \nabla f(w) \|_s \geq c > 0, \forall s$$  \hspace{1cm} (27)

then,

$$\tilde{k}_t \leq \frac{(f(w_1) - f(w_t))}{t \min \{\frac{1}{2}, \frac{\alpha c}{\delta L} \}}$$

Since $f(w_1) - f(w_t) \leq f(w_1) - f(w^*) = \ell_1$, we get,

$$\tilde{k}_t \leq \frac{\ell_1}{t \min \{\frac{1}{2}, C'\}}$$

where $C' = \frac{\alpha c}{8L\sqrt{\delta}}$.

Finally, by repeating the analysis of Lemma 2, we find with probability $1 - \delta$ that $c \leq \frac{8\sqrt{\pi}}{\sqrt{2d}}$ and thus

$$\tilde{k}_t \leq \frac{\ell_1}{t \min \{\frac{1}{2}, C'\}}$$

for $C' = \frac{\alpha \delta \sqrt{\pi}}{8\sqrt{2d}}$.

\[ \mathbb{Q} \]

### B Additional Experiments

In this appendix, we provide a more detailed version of our experimental results. Our experiments aim to answer the following questions:

1. In what situations do the projections become a performance bottleneck for gradient descent algorithms? (Section B.2)
2. When optimizing convex functions over strongly convex sets, does Primal Averaging (PA) outperform standard FW in practice (as our theory from Section 4 suggests)? If so, by how much? (Sections B.3 and B.4)
3. Does PA also outperform projected gradient descent when optimizing convex functions over strongly convex sets? If so, by how much? (Section B.5)
4. For strictly-locally-quasi-convex loss functions (Hazan, Levy, and Shalev-Shwartz 2015), does FW’s convergence rate in practice match our theoretical results from Section 5? (Section B.6)
5. When optimizing non-convex loss functions, how fast does FW converge in practice? Does it match our results from Section 6? (Section B.7)

In summary, our empirical results show the following:

1. Projections are costly and responsible for a considerable portion of the overall runtime of gradient descent, whenever the projection step has no closed-form solution (e.g., when the constraint set is an $l_1, l_5$ ball), or when the closed-form solution itself is expensive (e.g., projecting a matrix onto a nuclear norm ball, which requires computing the SVD (Agarwal, Negahban, and Wainwright 2010)).
2. In practice, the convergence rate of primal averaging matches our theoretical result of $O(\frac{1}{t})$ for smooth, convex functions with a strongly convex constraint set. Furthermore, under these conditions, primal averaging outperforms FW both with and without line search by 3.7–15.6× for a regression task and 2.8–11.7× for a matrix completion task in terms of the overall optimization time. It also outperforms projected gradient descent by 7.7× and can outperform stochastic gradient descent by up to 25.1×.

3. When optimizing strictly-locally-quasi-convex functions over strongly convex constraint sets, FW with line search converges to an $\epsilon$-neighborhood of the global minimum within $O(\max(\frac{1}{\ell}, \frac{1}{\epsilon^2}))$ iterations, as predicted by Theorem 3.

4. When the loss function is non-convex but the constraint set is strongly convex, FW with line search converges to a stationary point at a convergence rate of $O(\frac{1}{t})$, as predicted by Theorem 4.

#### B.1 Experiment Setup

**Hardware and Software** — Unless stated otherwise, all experiments were conducted on a Red Hat Enterprise Linux 7.1 server with 112 Intel(R) Xeon(R) CPU E7-4850 v3 processors and 2.20GHz cores and 1T DDR4 memory. All algorithms were implemented in Matlab R2015a. For projections that required solving a convex optimization problem, we used the CVX package (Grant and Boyd). Grant and Boyd (2008).

**Loss Functions** — In our experiments, we used a variety of popular loss functions to cover various types of convexity and different types of machine learning tasks used in practice. These functions, summarized in Table 5, are as follows:

- **Logistic Loss.** Logistic regression uses a convex loss function, which is also commonly used in classification tasks (Buja, Stuetzle, and Shen 2005) and is defined as:

  $$\ell(f(x_i), y_i) = \log(1 + e^{-y_if(x_i)})$$  \hspace{1cm} (28)

  where $f$ is a hypothesis function for the learning task and $y_i$ is the target value corresponding to $x_i$. Logistic loss is often used with an $l_p$ norm constraint to avoid overfitting (Huang and Chen 2011). The optimization problem is thus stated as follows:

  $$\min_{w \in \mathbb{R}^d \ b \in \mathbb{R}} \sum_{i=1}^{N} \ell(w^T x_i + b, y_i)$$  \hspace{1cm} (29)

  $$\text{s.t. } \|w\|_p \leq r.$$

  where $w$ is the coefficient vector, $b$ is the linear offset, $N$ is the number of data points, and $r$ is the radius of the $l_p$ norm ball.

- **Quadratic Loss.** The quadratic loss is a convex loss function and is commonly used in regression tasks (a.k.a. least squares loss) (Neter et al. 1996):

  $$\psi(f(x_i), y_i) = (f(x_i) - y_i)^2$$  \hspace{1cm} (30)

  Similar to logistic regression, a typical choice of constraint here is the $l_p$ norm. The optimization is stated as follows:

  $$\min_{w \in \mathbb{R}^d \ b \in \mathbb{R}} \sum_{i} \psi(w^T x_i + b, y_i)$$  \hspace{1cm} (31)

  $$\text{s.t. } \|w\|_p \leq r.$$
### Loss Function

| Convexity of Loss Function | Loss Function | Constraint | Task                  |
|----------------------------|---------------|------------|-----------------------|
| Convex                     | Logistic Loss | $l_p$ norm | Classification        |
| Convex                     | Quadratic Loss | $l_p$ norm | Regression            |
| Strictly-Locally-Quasi-Convex | Squared Sigmoid | $l_p$ norm | Classification        |
| Non-Convex                 | Bi-Weight Loss | $l_p$ norm | Robust Regression     |

### Observed Quadratic Loss

This loss function is also convex, but is typically used in matrix completion tasks (Freund and Grigas, Mazumder 2017), and is defined as:

$$\|X - M\|_{OB}^2 = \sum_{(i,j) \in P(M)} (X_{i,j} - M_{i,j})^2, \quad (32)$$

where $X, M \in \mathbb{R}^{m \times n}$, $X$ is the estimated matrix, $M$ is the observed matrix, and $P(M) = \{(i, j) : M_{i,j} \text{ is observed}\}$. In matrix completion, the loss function is often constrained within a Schatten-$p$ norm ball (Candès and Recht 2009, Koltchinskii et al. 2011, Recht and Ré 2013), which is a convex constraint set. Here, the optimization problem is stated as follows:

$$\min_{X \in \mathbb{R}^{m \times n}} \|X - M\|_{OB}^2 \quad (33)$$

s.t. $\|X\|_{Sp} \leq r$

where $\| \cdot \|_{Sp}$ is the Schatten-$p$ norm.

### Squared Sigmoid Loss

This function is non-convex, but it is strictly-locally-quasi-convex (see Section 3.1.1 of Hazan, Levy, Shalev-Shwartz 2015), and is defined as:

$$\varphi(z) = (1 + \exp(-z))^{-1} \quad (34)$$

where $z \in \mathbb{R}^n$. We can state the optimization problem as follows:

$$\min_{w \in \mathbb{R}^d, b \in \mathbb{R}} \frac{1}{n} \sum_{i=1}^{n} (y_i - \varphi(w^T x_i + b))^2 \quad (35)$$

s.t. $\|w\|_p \leq r$, where $n$ is the number of data points.

### Bi-Weight Loss

This loss function is non-convex, and is defined as follows:

$$\phi(f(x_i), y_i) = \frac{(f(x_i) - y_i)^2}{1 + (f(x_i) - y_i)^2} \quad (36)$$

The bi-weight loss is typically used for robust regression tasks (Belagiannis et al. 2015). Using the $l_p$ norm as a constraint, the optimization problem here is stated as follows:

$$\min_{w \in \mathbb{R}^d, b \in \mathbb{R}} \sum_{i=1}^{n} \phi(w^T x_i + b, y_i) \quad (37)$$

s.t. $\|w\|_p \leq r$.

### Datasets

We ran our experiments using several datasets of different sizes and dimensionalities:

- Robust regression is less sensitive to outliers in the dataset.

### Compared Methods

- **Standard Gradient Descent (GD).** In the $k^{th}$ iteration, GD moves the opposite direction of the gradient:

  $$w^{(k+1)} = w^{(k)} - \eta \sum_{x_i} \nabla f(x_i) \quad (38)$$

  where $f(x_i)$ is the loss on data point $x_i$.

- **Stochastic Gradient Descent (SGD).** Unlike GD, SGD uses only one data point in each iteration:

  $$w^{(k+1)} = w^{(k)} - \eta \nabla f(x_{k+1}) \quad (39)$$

- **Standard Frank-Wolfe with Predefined Learning Rate (FWPLR).** This variant of Frank-Wolfe corresponds to Algorithm 1 with option (A).

- **Standard Frank-Wolfe with Line Search (FWLS).** This corresponds to Algorithm 1 with option (B).

- **Primal Averaging (PA).** Primal averaging is the variant of FW algorithm, which we advocate in this paper. This algorithm was presented in Algorithm 2.

- **Stochastic Primal Averaging (SPA).** This is the stochastic version of PA, as described in Section 4.2.

### B.2 Projection Overhead in Gradient Descent

To understand the overhead of the projection step in gradient descent algorithm, we experimented with various machine learning tasks and constraint sets. Specifically, we studied $l_1$, $l_{1.5}$, $l_2$, $l_\infty$, Schatten-1, Schatten-2, and Schatten-$\infty$ norms as our constraint sets. We used the $l_p$ norm balls in a logistic loss classifier (Adult dataset) and used the Schatten-$p$ norms.
in a matrix completion task (MovieLens dataset) with observed quadratic loss (see Section B.1 and Table 3). To study the effect of data size, we also ran each experiment using different portions of its respective dataset: 1%, 10%, and 100%.

These constraint sets can be divided into three categories (Hazan and others 2016): (i) projection onto the $l_1$, $l_2$, and $l_\infty$ balls have a closed-form and thus can be computed efficiently, (ii) projection onto Schatten-1 (a.k.a. nuclear or trace norm), Schatten-2, and Schatten-$\infty$ norms has a closed-form but the closed-form requires the SVD of the model matrix, and is thus costly, and (iii) projection onto $l_{1.5}$ balls does not have any closed-form and requires solving another optimization problem.

Figure 3 shows the average portion of the total time spent in each iteration of the gradient descent in performing the projection step. As expected, the projection step did not account for much of the overall runtime when there was an efficient closed-form, i.e., less than 7%, 0.03%, and 3% for the $l_1$, $l_2$, and $l_\infty$ norms, respectively. In contrast, projections that involved a costly closed-form or required solving a separate optimization problem introduced a significant overhead. Specifically, the projection time was responsible for 69–99% of the overall runtime for $l_{1.5}$, 95–99% for Schatten-1, 70–99% for Schatten-2, and 71–99% for Schatten-$\infty$.

Another observation is that this overhead decreased with the data size. This is expected, as the cost of computing the gradient itself in GD grows with the data size and becomes the dominating factor, hence reducing the relative ratio of the projection time to the overall runtime of each iteration. This is why, for massive datasets, stochastic gradient descent (SGD) is much more popular than standard GD (Recht et al. 2011; Liu et al. 2015; Recht and Ré 2013). Therefore, we measured the projection overhead for SGD as well. However, since SGD’s runtime does not depend on the overall data size, we did not vary the dataset size.

The results for SGD are shown in Figure 4. The trend here is similar to GD, but the overheads are more pronounced. Constraint sets without a computationally efficient projection caused a significant overhead in SGD. However, for SGD, even the projections with efficient closed-form solutions introduced a noticeable overhead: 5–20% for $l_1$, 5–11% for $l_2$, and 50–65% for $l_\infty$. While SGD takes significantly less time than GD to compute its descent direction for large datasets, the time to compute the projection remains constant. Hence, the fraction of the overall computation time spent on projection is larger in SGD than in GD.

The reason for the particularly higher overhead in case of $l_\infty$ is that projecting onto an $l_\infty$ ball cannot be vectorized. In other words, projection onto $l_1$ and $l_2$ balls can take better advantage of the underlying hardware than projection onto an $l_\infty$ ball, causing the observed disparity in runtimes. In summary, the projection overhead is a major concern for both GD and SGD, whenever there is no efficient closed-form. Furthermore, this problem is still important for SGD, even when there is an efficient procedure for projection.

B.3 Primal Averaging’s Convergence Rate

In this section, we report experiments on various machine learning tasks and datasets to compare PA’s performance against other variants of FW when solving (smooth) convex functions with strongly convex constraint sets. In particular, we studied the performance of PA, FWLS, and FWPLR for both $l_2$ and Schatten-2 balls as our strongly convex constraint sets (see Section 3.1 for a discussion of why these constraints are strongly convex). We used the $l_2$ norm ball for a logistic classifier on the Adult dataset, as well as a linear regression task on the YearPredictionMSD dataset. We used the Schatten-2 norm ball for a matrix completion task on the MovieLens dataset.

First, we measured the $\epsilon$-neighborhood of the global minimum reached by PA at each iteration. Our theoretical results (Theorem 1) predict a convergence rate of $O\left(\frac{1}{t^\alpha}\right)$ in this case. To confirm this empirically, we plotted the logarithm of the $\epsilon$-neighborhood against the logarithm of the iteration number. If the convergence rate of $O\left(\frac{1}{t^\alpha}\right)$ were to hold, we would
expect a straight line with a slope of $-2$ after taking the logarithms.

The plots are shown in Figures 5a and 5b for the classification and matrix completion tasks, respectively. The results confirm our theoretical results, as the plots exhibit a slope of -2.34 and -2.41, respectively. Note that a slightly steeper slope is expected in practice, since our theoretical results only provide a worst-case upper bound on the convergence rate.

**B.4 Primal Averaging’s Performance versus Other FW Variants**

To compare the actual performance of various FW variants on convex functions, we used the same settings as Section B.3. However, instead of the number of iterations to convergence, this time we measured the actual runtimes.

Here, we compared all three variants: PA, FWLS, and FWPLR. Figures 6a and 6b report the time taken to achieve each value of the loss function for the regression and matrix completion tasks, respectively. To compare the performance of these algorithms, we measured the difference between the time it took for each of them to converge. To determine convergence, here picked the first iteration at which the loss value was within $\pm 2\%$ of the previous loss value (practical convergence), and was also within $\pm 2\%$ of the global minimum (actual convergence). The first time at which these iterations were reached for each algorithm are marked by vertical, striped lines in Figures 6a and 6b.

For the regression task, PA converged 3.7× and 15.6× faster than FWPLR and FWLS, respectively. For the matrix completion task, PA converged 2.8× and 11.7× faster than FWPLR and FWLS, respectively. These considerable speedups have significant ramifications in practice. Traditionally, PA has been shied away from, simply because it is slower in each iteration (due to PA’s use of auxiliary sequences and its extra summation step), while its convergence rate was believed to be the same as the more efficient variants [Lan2013]. However, as we formally proved in Section 4, PA does indeed converge within much fewer iterations. Thus, the results shown in Figures 6a and 6b validate our hypothesis that PA’s faster convergence rate more than compensates for its additional computation at each iteration. Figure 7 reports the per-iteration cost of these FW variants on average, showing that PA is only 1.2–1.3× slower than than FWPLR in each iteration. This is why PA’s much faster convergence rate leads to much better performance in practice, compared to FWPLR. On the other hand, although FWLS offers the same convergence rate as PA, PA’s cost per iteration is 3.2–7.1× faster than FWLS, which also explains PA’s superior performance over FWLS.

Finally, we note that PA’s improvements were much more drastic for the regression task than the matrix completion task (3.7–15.6× versus 2.8–11.7×). This is due to of the following reason. We recall that the Schatten-$p$ norm ball with radius $r$ is $\alpha$-strongly convex for $p \in (1, 2]$ and with $\alpha = \frac{p}{p-1}$. The matrix completion task on the MovieLens dataset requires us to predict the values of a $6,040 \times 3,900$ matrix (6,040 users and 3,900 movies). Thus, to be able to maintain a reasonable number of potential matrices within our constraint set, we had to set $r = 12000$, namely $\alpha = \frac{p}{p-1}$. According to Theorem 1, the convergence rate is $\mathcal{O}(\frac{L}{\alpha^2 g^2 t^2})$, which is why a small value of $\alpha$ slows down PA’s convergence.

**B.5 Primal Averaging’s Performance versus Projected Gradient Descent**

In this section, we compare the performance of PA and projected gradient descent. We evaluated deterministic and
stochastic versions of both algorithms on the regression task with the same settings as in Section B.4. We used the same methodology to determine convergence as in Section B.4.

The results are shown in Figure 8. As expected, PA significantly outperformed projected GD, converging $7.7 \times$ faster (Figure 8a). To better compare their stochastic versions (SPA and SGD), however, we used two different settings. The first used the $l_2$ ball as the constraint set, as an example of a case with an efficient projection, and the second used the $l_1$ ball as an example of a case with a costly projection.

The results conformed with our expectation again. When the projection onto the constraint set was efficient, SGD converged $4.6 \times$ faster than SPA (Figure 8b). On the other hand, when the projection was costly, SPA far outperformed SGD, converging $25.1 \times$ faster (Figure 8c).

B.6 Frank-Wolfe for (Smooth) Strictly-Locally-Quasi-Convex Functions

According to Theorem 3, even when the loss function is not convex, FWLS still converges (to an $\epsilon$-neighborhood of the global minimum) within $O\left(\max\left(\frac{1}{\epsilon^2}, \frac{1}{\epsilon^3}\right)\right)$ iterations, as long as the loss function is strictly-locally-quasi-convex. To verify this empirically, we used the squared Sigmoid loss function for a classification task (Adult dataset) with an $l_2$ ball as our constraint set.

Note that, to conform with our theoretical result, FWLS must exhibit an $O\left(\frac{1}{\epsilon^2}\right)$ convergence rate when $\epsilon > 1$ and an $O\left(\frac{1}{\epsilon^3}\right)$ convergence rate when $\epsilon < 1$. To better illustrate this difference, we examine two plots: Figure 9a displays the iterations where $\epsilon > 1$, while Figure 9b displays the iteration where $\epsilon < 1$. Both plots show the logarithm of the $\epsilon$-neighborhood against the logarithm of the iteration number. This means we should expect to see the loss values decreasing at a slope steeper than or equal to $-\frac{1}{2}$ and $-\frac{1}{3}$ in Figure 9a and Figure 9b, respectively.

The plots confirm our theoretical results, exhibiting a slope of $-2.12$ when $\epsilon > 1$ and $-0.377$ when $\epsilon < 1$. Note that the steeper slopes here are expected, as our theoretical results only provide a worst-case upper bound on the convergence rate. Notably, FWLS showed a significantly steeper slope when $\epsilon > 1$. We observe that the convergence rate bound for $\epsilon > 1$ is missing the smoothness parameter $L$ of the $\epsilon < 1$ bound. It is noted in (Hazan, Levy, and Shalev-Shwartz 2015) that using the squared Sigmoid loss is equivalent to the perceptron problem with a $\gamma$-margin, and Kalai and Sastry (Kalai and Sastry 2009) show that the smoothness parameter $L$ of the latter is $\frac{1}{\gamma^2}$. Thus, when the margin is large, the $\epsilon < 1$ case is able to converge at a rate closer to $O\left(\frac{1}{\epsilon^3}\right)$ than the $\epsilon > 1$ case to its rate of $O\left(\frac{1}{\epsilon^2}\right)$. Thus, when the margin is large, the $\epsilon < 1$ case converges at a slower rate.
Figure 9: Iterations for FWLS to converge to an $\epsilon$-neighborhood when optimizing a strictly-locally-quasi-convex function.

B.7 Frank-Wolfe for (Smooth) Non-Convex Functions

In Theorem 4, we proved that FWLS converges to a stationary point of (smooth) non-convex functions at a rate of $O(\frac{1}{t})$, as long as it is constrained to a strongly convex set. To empirically verify whether this upper bound is tight, we use the bi-weight loss (see Section B.1 and Table 3) in a classification task (Adult dataset) with an $l_2$ ball constraint.

In Figure 10, we measured the $\epsilon$-neighborhood reached by FWLS at each iteration, plotting the logarithm of the $\epsilon$-neighborhood against the logarithm of the iteration number. To confirm the $O(\frac{1}{t})$ convergence rate found in Theorem 4, we expect to see a straight line of slope $-1$ in Figure 10.

The empirical results confirm our theoretical results, showing a slope of $-1.46$. Again, we note that a steeper slope is expected in practice as Theorem 4 only provides a worst-case upper bound on the convergence rate.