Solving Empirical Risk Minimization in the Current Matrix Multiplication Time

Yin Tat Lee YINTAT@UW.EDU
University of Washington & Microsoft Research Redmond.
Zhao Song ZHAOSONG@UW.EDU
University of Washington & UT-Austin.
Qiuyi Zhang QIUYI@MATH.BERKELEY.EDU
University of California, Berkeley.

Editors: Alina Beygelzimer and Daniel Hsu

[1] **Abstract**

Many convex problems in machine learning and computer science share the same form:

$$\min_x \sum_i f_i(A_i x + b_i),$$

where f_i are convex functions on $\mathbb{R}^{n_i}$ with constant n_i, $A_i \in \mathbb{R}^{n_i \times d}$, $b_i \in \mathbb{R}^{n_i}$ and $\sum_i n_i = n$. This problem generalizes linear programming and includes many problems in empirical risk minimization.

In this paper, we give an algorithm that runs in time

$$O^*((n^\omega + n^{2.5-\alpha/2} + n^{2+1/6}) \log(n/\delta))$$

where ω is the exponent of matrix multiplication, α is the dual exponent of matrix multiplication, and δ is the relative accuracy. Note that the runtime has only a log dependence on the condition numbers or other data dependent parameters and these are captured in δ. For the current bound $\omega \sim 2.38$ [Vassilevska Williams'12, Le Gall'14] and $\alpha \sim 0.31$ [Le Gall, Urrutia'18], our runtime $O^*(n^\omega \log(n/\delta))$ matches the current best for solving a dense least squares regression problem, a special case of the problem we consider. Very recently, [Alman'18] proved that all the current known techniques can not give a better ω below 2.168 which is larger than our $2 + 1/6$.

Our result generalizes the very recent result of solving linear programs in the current matrix multiplication time [Cohen, Lee, Song'19] to a more broad class of problems. Our algorithm proposes two concepts which are different from [Cohen, Lee, Song'19] :

• We give a robust deterministic central path method, whereas the previous one is a stochastic central path which updates weights by a random sparse vector.

• We propose an efficient data-structure to maintain the central path of interior point methods even when the weights update vector is dense.

1. Introduction

Empirical Risk Minimization (ERM) problem is a fundamental question in statistical machine learning. There are a huge number of papers that have considered this topic Nesterov (1983); Vapnik

1. Extended abstract. Full version is available on https://arxiv.org/pdf/1905.04447.

(1992); Polyak and Juditsky (1992); Nesterov (2004); Bartlett et al. (2005); Bottou and Bousquet (2008); Nemirovski et al. (2009); Moulines and Bach (2011); Feldman et al. (2012); Le Roux et al. (2012); Johnson and Zhang (2013); Vapnik (2013); Shalev-Shwartz and Zhang (2013); Défossez and Bach (2014); Defazio et al. (2014); Frostig et al. (2015); Dieuleveut and Bach (2016); Shang et al. (2017); Zhang et al. (2017); Zhang and Xiao (2017); Zheng et al. (2017); Gonen and Shalev-Shwartz (2017); Murata and Suzuki (2017); Nesterov and Stich (2017); Agarwal et al. (2017); Csiba (2018); Jin et al. (2018) as almost all convex optimization machine learning can be phrased in the ERM framework Shalev-Shwartz and Ben-David (2014); Vapnik (1992). While the statistical convergence properties and generalization bounds for ERM are well-understood, a general runtime bound for general ERM is not known although fast runtime bounds do exist for specific instances Adil et al. (2019).

Examples of applications of ERM include linear regression, LASSO Tibshirani (1996), elastic net Zou and Hastie (2005), logistic regression Cox (1958); Hosmer Jr et al. (2013), support vector machines Cortes and Vapnik (1995), ℓ_p regression Clarkson (2005); Dasgupta et al. (2009); Bubeck et al. (2018); Adil et al. (2019), quantile regression Koenker (2000); Koenker and Hallock (2001); Koenker (2005), AdaBoost Freund and Schapire (1997), kernel regression Nadaraya (1964); Watson (1964), and mean-field variational inference Xing et al. (2002).

The classical Empirical Risk Minimization problem is defined as

$$\min_x \sum_{i=1}^m f_i(a_i^\top x + b_i)$$

where $f_i : \mathbb{R} \to \mathbb{R}$ is a convex function, $a_i \in \mathbb{R}^d$, and $b_i \in \mathbb{R}$, $\forall i \in [m]$. Note that this formulation also captures most standard forms of regularization as well.

Letting $y_i = a_i^\top x + b_i$, and $z_i = f_i(a_i^\top x + b_i)$ allows us to rewrite the original problem in the following sense,

$$\begin{aligned} \min_{x,y,z} \quad & \sum_{i=1}^m z_i \\ \text{s.t.} \quad & Ax + b = y \\ & (y_i, z_i) \in K_i = \{(y_i, z_i) : f_i(y_i) \leq z_i\}, \forall i \in [m] \end{aligned} \tag{1}$$

We can consider a more general version where dimension of K_i can be arbitrary, e.g. n_i. Therefore, we come to study the general n-variable form

$$\min_{x \in \prod_{i=1}^m K_i, Ax=b} c^\top x$$

where $\sum_{i=1}^m n_i = n$. We state our main result for solving the general model.

Theorem 1 (Main result, informal version of Theorem 38) *Given a matrix $A \in \mathbb{R}^{d \times n}$, two vectors $b \in \mathbb{R}^d$, $c \in \mathbb{R}^n$, and m compact convex sets $K_1, K_2, \cdots, K_m$. Assume that there is no redundant constraints and $n_i = O(1)$, $\forall i \in [m]$. There is an algorithm (procedure* MAIN *in Algorithm 6) that solves*

$$\min_{x \in \prod_{i=1}^m K_i, Ax=b} c^\top x$$

up to δ precision and runs in expected time

$$\widetilde{O}\left((n^{\omega+o(1)} + n^{2.5-\alpha/2+o(1)} + n^{2+1/6+o(1)}) \cdot \log(\frac{n}{\delta})\right)$$

where ω is the exponent of matrix multiplication, α is the dual exponent of matrix multiplication.

For the current value of $\omega \sim 2.38$ Williams (2012); Le Gall (2014) and $\alpha \sim 0.31$ Le Gall and Urrutia (2018), the expected time is simply $n^{\omega+o(1)}\widetilde{O}(\log(\frac{n}{\delta}))$.

Remark 2 *More precisely, when n_i is super constant, our running time depends polynomially on $\max_{i\in[m]} n_i$ (but not exponential dependence).*

Also note that our runtime depends on diameter, but logarithmically to the diameter. So, it can be applied to linear program by imposing an artificial bound on the solution.

1.1. Related Work

First-order algorithms for ERM are well-studied and a long series of accelerated stochastic gradient descent algorithms have been developed and optimized Nesterov (1998); Johnson and Zhang (2013); Xiao and Zhang (2014); Shalev-Shwartz and Zhang (2014); Frostig et al. (2015); Lin et al. (2015); Ma et al. (2015); Allen-Zhu and Yuan (2016); Reddi et al. (2016); Shalev-Shwartz (2016); Allen-Zhu and Hazan (2016); Schmidt et al. (2017); Murata and Suzuki (2017); Lin et al. (2017); Lei et al. (2017); Allen-Zhu (2017a,b, 2018a,b). However, these rates depend polynomially on the Lipschitz constant of ∇f_i and in order to achieve a $\log(1/\epsilon)$ dependence, the runtime will also have to depend on the strong convexity of the $\sum_i f_i$. In this paper, we want to focus on algorithms that depend logarithmically on diameter/smoothness/strong convexity constants, as well as the error parameter ϵ. Note that gradient descent and a direct application of Newton's method do not belong to these class of algorithms, but for example, interior point method and ellipsoid method does.

Therefore, in order to achieve high-accuracy solutions for non-smooth and non strongly convex case, most convex optimization problems will rely on second-order methods, often under the general interior point method (IPM) or some sort of iterative refinement framework. So, we note that our algorithm is thus optimal in this general setting since second-order methods require at least n^ω runtime for general matrix inversion.

Our algorithm applies the interior point method framework to solve ERM. The most general interior point methods require $O(\sqrt{n})$-iterations of linear system solves Nesterov (1998), requiring a naive runtime bound of $O(n^{\omega+1/2})$. Using the inverse maintenance technique Vaidya (1989); Cohen et al. (2019), one can improve the running time for LP to $O(n^\omega)$. This essentially implies that almost all convex optimization problems can be solved, up to subpolynomial factors, as fast as linear regression or matrix inversion!

The specific case of ℓ_2 regression can be solved in $O(n^\omega)$ time since the solution is explicitly given by solving a linear system. In the more general case of ℓ_p regression, Bubeck et al. (2018) proposed a $\widetilde{O}_p(n^{|1/2-1/p|})$-iteration iterative solver with a naive $O(n^\omega)$ system solve at each step. Recently, Adil et al. (2019) improved the runtime to $\widetilde{O}_p(n^{\max(\omega,7/3)})$, which is current matrix multiplication time as $\omega > 7/3$. However, both these results depend exponentially on p and fail to be impressive for large p. Otherwise, we are unaware of other ERM formulations that have have general runtime bounds for obtaining high-accuracy solutions.

Recently several works Alman and Williams (2018a,b); Alman (2018) try to show the limitation of current known techniques for improving matrix multiplication time. Alman and Vassilevska Williams Alman and Williams (2018b) proved limitations of using the Galactic method applied to many tensors of interest (including Coppersmith-Winograd tensors Coppersmith and Winograd (1987)). More recently, Alman Alman (2018) proved that by applying the Universal method on those tensors, we cannot hope to achieve any running time better than $n^{2.168}$ which is already above our $n^{2+1/6}$.

2. Overview of Techniques

In this section, we discuss the key ideas in this paper. Generalizing the stochastic sparse update approach of Cohen et al. (2019) to our setting is a natural first step to speeding up the matrix-vector multiplication that is needed in each iteration of the interior point method. In linear programs, maintaining approximate complementary slackness means that we maintain x, s to be close multiplicatively to the central path under some notion of distance. However, the generalized notion of complementary slackness requires a barrier-dependent notion of distance. Specifically, if $\phi(x)$ is a barrier function, then our distance is now defined as our function gradient being small in a norm depending on $\nabla^2\phi(x)$. One key fact of the stochastic sparse update is that the variance introduced does not perturb the approximation too much, which requires understanding the second derivative of the distance function. For our setting, this would require bounding the 4th derivative of $\phi(x)$, which may not exist for self-concordant functions. So, the stochastic approach may not work algorithmically (not just in the analysis) if $\phi(x)$ is assumed to be simply self-concordant. Even when assumptions on the 4th derivative of $\phi(x)$ are made, the analysis will become significantly more complicated due to the 4th derivative terms. To avoid these problems, the main contributions of this paper is to 1) introduce a robust version of the central path and 2) exploit the robustness via sketching to apply the desired matrix-vector multiplication fast.

More generally, our main observation is that one can generally speed up an iterative method using sketching if the method is robust in a certain sense. To speed up interior point methods, in Section 4 and A, we give a robust version of the interior point method; and in Section B, we give a data structure to maintain the sketch; and in Section C, we show how to combine them together. We provide several basic notations and definitions for numerical linear algebra in Section 3. In Section D, we provide some classical lemmas from the literature of interior point methods. In Section E, we prove some basic properties of the sketching matrix. Now, we first begin with an overview of our robust central path and then proceed with an overview of sketching iterative methods.

2.1. Central Path Method

We consider the following optimization problem

$$\min_{x \in \prod_{i=1}^m K_i, Ax=b} c^\top x \tag{2}$$

where $\prod_{i=1}^m K_i$ is the direct product of m low-dimensional convex sets K_i. We let x_i be the i-th block of x corresponding to K_i. Interior point methods consider the path of solutions to the following optimization problem:

$$x(t) = \arg\min_{Ax=b} c^\top x + t \sum_{i=1}^m \phi_i(x_i) \tag{3}$$

where $\phi_i : K_i \to \mathbb{R}$ are self-concordant barrier functions. This parameterized path is commonly known as the *central path*. Many algorithms solve the original problem (2) by following the central path as the path parameter is decreased $t \to 0$. The rate at which we decrease t and subsequently the runtimes of these path-following algorithms are usually governed by the self-concordance properties of the barrier functions we use.

Definition 3 *We call a function ϕ a ν self-concordant barrier for K if $\mathrm{dom}\phi = K$ and for any $x \in \mathrm{dom}\phi$ and for any $u \in \mathbb{R}^n$*

$$|D^3\phi(x)[u,u,u]| \leq 2\|u\|_x^{3/2} \quad \text{and} \quad \|\nabla\phi(x)\|_x^* \leq \sqrt{\nu}$$

where $\|v\|_x := \|v\|_{\nabla^2\phi(x)}$ and $\|v\|_x^* := \|v\|_{\nabla^2\phi(x)^{-1}}$, for any vector v.

Remark 4 *It is known that $\nu \geq 1$ for any self-concordant barrier function.*

Nesterov and Nemirovsky showed that for any open convex set $K \subset \mathbb{R}^n$, there is a $O(n)$ self-concordant barrier function Nesterov (1998). In this paper, the convex set K_i we considered has $O(1)$ dimension. While Nesterov and Nemirovsky gave formulas for the universal barrier; in practice, most ERM problems lend themselves to explicit $O(1)$ self-concordant barriers for majority of the convex functions people use. For example, for the set $\{x : \|x\| < 1\}$, we use $-\log(1-\|x\|^2)$; for the set $\{x : x > 0\}$, we use $-\log(x)$, and so on. That is the reason why we assume the gradient and hessian can be computed in $O(1)$ time. Therefore, in this paper, we assume a ν_i self-concordant barrier ϕ_i is provided and that we can compute $\nabla\phi_i$ and $\nabla^2\phi_i$ in $O(1)$ time. The main result we will use about self-concordance is that the norm $\|\cdot\|_x$ is stable when we change x.

Theorem 5 (Theorem 4.1.6 in Nesterov (1998)) *If ϕ is a self-concordant barrier and if $\|y - x\|_x < 1$, then we have :*

$$(1 - \|y-x\|_x)^2\nabla^2\phi(x) \preceq \nabla^2\phi(y) \preceq \frac{1}{(1-\|y-x\|_x)^2}\nabla^2\phi(x).$$

In general, we can simply think of ϕ_i as a function penalizing any point $x_i \notin K_i$. It is known how to transform the original problem (2) by adding $O(n)$ many variables and constraints so that

- The minimizer $x(t)$ at $t = 1$ is explicitly given.
- One can obtain an approximate solution of the original problem using the minimizer at small t in linear time.

For completeness, we show how to do it in Lemma 41. Therefore, it suffices to study how we can move efficiently from $x(1)$ to $x(\epsilon)$ for some tiny ϵ where $x(t)$ is again the minimizer of the problem (3).

2.2. Robust Central Path

In the standard interior point method, we use a tight ℓ_2-bound to control how far we can deviate from $x(t)$ during the entirety of the algorithm. Specifically, if we denote $\gamma_i^t(x_i)$ as the appropriate measure of error (this will be specified later and is often called the Newton Decrement) in each block coordinate x_i at path parameter t, then as we let $t \to 0$, the old invariant that we are maintaining is,

$$\Phi_{\mathrm{old}}^t(x) = \sum_{i=1}^m \gamma_i^t(x_i)^2 \leq O(1).$$

It can be shown that a Newton step in the standard direction will allow for us to maintain Φ^t_{old} to be small even as we decrease t by a multiplicative factor of $O(m^{-1/2})$ in each iteration, thereby giving a standard $O(\sqrt{m})$ iteration analysis. Therefore, the standard approach can be seen as trying to remain within a small ℓ_2 neighborhood of the central path by centering with Newton steps after making small decreases in the path parameter t. Note however that if each γ_i can be perturbed by an error that is $\Omega(m^{-1/2})$, $\Phi^t_{\text{old}}(x)$ can easily become too large for the potential argument to work.

To make our analysis more robust, we introduce a robust version that maintains the soft-max potential:

$$\Phi^t_{\text{new}}(x) = \sum_{i=1}^{m} \exp(\lambda \gamma^t_i(x_i)) \leq O(m)$$

for some $\lambda = \Theta(\log m)$. The robust central path is simply the region of all x that satisfies our potential inequality. We will specify the right constants later but we always make λ large enough to ensure that $\gamma_i \leq 1$ for all x in the robust central path. Now note that a ℓ_∞ perturbation of γ translates into a small multiplicative change in Φ^t, tolerating errors on each γ_i of up to $O(1/\text{poly}\log(n))$.

However, maintaining $\Phi^t_{\text{new}}(x) \leq O(m)$ is not obvious because the robust central path is a much wider region of x than the typical ℓ_2-neighborhood around the central path. We will show later how to modify the standard Newton direction to maintain $\Phi^t_{\text{new}}(x) \leq O(m)$ as we decrease t. Specifically, we will show that a variant of gradient descent of Φ^t_{new} in the Hessian norm suffices to provide the correct guarantees.

2.3. Speeding up via Sketching

To motivate our sketching algorithm, we consider an imaginary iterative method

$$z^{(k+1)} \leftarrow z^{(k)} + P \cdot F(z^{(k)})$$

where P is some dense matrix and $F(z)$ is some simple formula that can be computed efficiently in linear time. Note that the cost per iteration is dominated by multiplying P with a vector, which takes $O(n^2)$ time. To avoid the cost of multiplication, instead of storing the solution explicitly, we store it implicitly by $z^{(k)} = P \cdot u^{(k)}$. Now, the algorithm becomes

$$u^{(k+1)} \leftarrow u^{(k)} + F(P \cdot u^{(k)}).$$

This algorithm is as expensive as the previous one except that we switch the location of P. However, if we know the algorithm is robust under perturbation of the $z^{(k)}$ term in $F(z^{(k)})$, we can instead do

$$u^{(k+1)} \leftarrow u^{(k)} + F(R^\top RP \cdot u^{(k)})$$

for some random Gaussian matrix $R : \mathbb{R}^{b \times n}$. Note that the matrix RP is fixed throughout the whole algorithm and can be precomputed. Therefore, the cost of per iteration decreases from $O(n^2)$ to $O(nb)$.

For our problem, we need to make two adjustments. First, we need to sketch the change of z, that is $F(P \cdot u^{(k)})$, instead of $z^{(k)}$ directly because the change of z is smaller and this creates a

smaller error. Second, we need to use a fresh random R every iteration to avoid the randomness dependence issue in the proof. For the imaginary iterative process, it becomes

$$\overline{z}^{(k+1)} \leftarrow \overline{z}^{(k)} + R^{(k)\top} R^{(k)} P \cdot F(\overline{z}^{(k)}),$$
$$u^{(k+1)} \leftarrow u^{(k)} + F(\overline{z}^{(k)}).$$

After some iterations, $\overline{z}^{(k)}$ becomes too far from $z^{(k)}$ and hence we need to correct the error by setting $z^{(k)} = P \cdot u^{(k)}$, which zeros the error.

Note that the algorithm explicitly maintains the approximate vector $\overline{z}$ while implicitly maintaining the exact vector z by $Pu^{(k)}$. This is different from the classical way to sketch Newton method Pilanci and Wainwright (2016, 2017), which is to simply run $z^{(k+1)} \leftarrow z^{(k)} + R^\top RP \cdot F(z^{(k)})$ or use another way to subsample and approximate P. Such a scheme relies on the iteration method to fix the error accumulated in the sketch, while we are actively fixing the error by having both the approximate explicit vector $\overline{z}$ and the exact implicit vector z.

Without precomputation, the cost of computing $R^{(k)}P$ is in fact higher than that of $P \cdot F(z^{(k)})$. The first one involves multiplying multiple vectors with P and the second one involves multiplying 1 vector with P. However, we can precompute $[R^{(1)\top}; R^{(2)\top}; \cdots; R^{(T)\top}]^\top \cdot P$ by fast matrix multiplication. This decreases the cost of multiplying 1 vector with P to $n^{\omega-1}$ per vector. This is a huge saving from n^2. In our algorithm, we end up using only $\widetilde{O}(n)$ random vectors in total and hence the total cost is still roughly n^ω.

2.4. Maintaining the Sketch

The matrix P we use in interior point methods is of the form

$$P = \sqrt{W} A^\top (AWA^\top)^{-1} A \sqrt{W}$$

where W is some block diagonal matrix. Cohen et al. (2019) showed one can approximately maintain the matrix P with total cost $\widetilde{O}(n^\omega)$ across all iterations of interior point method. However, the cost of applying the dense matrix P with a vector z is roughly $O(n\|z\|_0)$ which is $O(n^2)$ for dense vectors. Since interior point methods takes at least $\sqrt{n}$ iterations in general, this gives a total runtime of $O(n^{2.5})$. The key idea in Cohen et al. (2019) is that one can design a stochastic interior point method such that each step only need to multiply P with a vector of density $\widetilde{O}(\sqrt{n})$. This bypasses the $n^{2.5}$ bottleneck.

In this paper, we do not have this issue because we only need to compute RPz which is much cheaper than Pz. We summarize why it suffices to maintain RP throughout the algorithm. In general, for interior point method, the vector z is roughly an unit vector and since P is an orthogonal projection, we have $\|Pz\|_2 = O(1)$. One simple insight we have is that if we multiply a random $\sqrt{n} \times n$ matrix R with values $\pm\frac{1}{\sqrt{n}}$ by Pz, we have $\|RPz\|_\infty = \widetilde{O}(\frac{1}{\sqrt{n}})$ (Lemma 47). Since there are $\widetilde{O}(\sqrt{n})$ iterations in interior point method, the total error is roughly $\widetilde{O}(1)$ in a correctly reweighed ℓ_∞ norm. In Section A, we showed that this is exactly what interior point method needs for convergence. Furthermore, we note that though each step needs to use a fresh random matrix R_l of size $\sqrt{n} \times n$, the random matrices $[R_1^\top; R_2^\top; \cdots; R_T^\top]^\top$ we need can all fit into $\widetilde{O}(n) \times n$ budget. Therefore, throughout the algorithm, we simply need to maintain the matrix $[R_1^\top; R_2^\top; \cdots; R_T^\top]^\top P$ which can be done with total cost $\widetilde{O}(n^\omega)$ across all iterations using idea similar to Cohen et al. (2019).

The only reason the data structure looks complicated is that when the block matrix W changes in different location in $\sqrt{W}A^\top(AWA^\top)^{-1}A\sqrt{W}$, we need to update the matrix $[R_1; R_2; \cdots; R_T]P$ appropriately. This gives us few simple cases to handle in the algorithm and in the proof. For the intuition on how to maintain P under W change, see (Cohen et al., 2019, Section 2.2 and 5.1).

2.5. Fast rectangular matrix multiplication

Given two size $n \times n$ matrices, the time of multiplying them is $n^{2.81} < n^3$ by applying Strassen's original algorithm Strassen (1969). The current best running time takes n^ω time where $\omega < 2.373$ Williams (2012); Le Gall (2014). One natural extension of multiplying two square matrices is multiplying two rectangular matrices. What is the running time of multiplying one $n \times n^a$ matrix with another $n^a \times n$ matrix? Let α denote the largest upper bound of a such that multiplying two rectangular matrices takes $n^{2+o(1)}$ time. The α is called the dual exponent of matrix multiplication, and the state-of-the-art result is $\alpha = 0.31$ Le Gall and Urrutia (2018). We use the similar idea as Cohen et al. (2019) to delay the low-rank update when the rank is small.

3. Preliminaries

Given a vector $x \in \mathbb{R}^n$ and m compact convex sets $K_1 \subset \mathbb{R}^{n_1}, K_2 \subset \mathbb{R}^{n_2}, \cdots, K_m \subset \mathbb{R}^{n_m}$ with $\sum_{i=1}^m n_i = n$. We use x_i to denote the i-th block of x, then $x \in \prod_{i=1}^m K_i$ if $x_i \in K_i, \forall i \in [m]$.

We say a block diagonal matrix $A \in \oplus_{i=1}^m \mathbb{R}^{n_i \times n_i}$ if A can be written as

$$A = \begin{bmatrix} A_1 & & & \\ & A_2 & & \\ & & \ddots & \\ & & & A_m \end{bmatrix}$$

where $A_1 \in \mathbb{R}^{n_1 \times n_1}$, $A_2 \in \mathbb{R}^{n_2 \times n_2}$, and $A_m \in \mathbb{R}^{n_m \times n_m}$. For a matrix A, we use $\|A\|_F$ to denote its Frobenius norm and use $\|A\|$ to denote its operator norm. There are some trivial facts $\|AB\|_2 \leq \|A\|_2 \cdot \|B\|_2$ and $\|AB\|_F \leq \|A\|_F \cdot \|B\|_2$.

For notation convenience, we assume the number of variables $n \geq 10$ and there are no redundant constraints. In particular, this implies that the constraint matrix A is full rank.

For a positive integer n, let $[n]$ denote the set $\{1, 2, \cdots, n\}$.

For any function f, we define $\widetilde{O}(f)$ to be $f \cdot \log^{O(1)}(f)$. In addition to $O(\cdot)$ notation, for two functions f, g, we use the shorthand $f \lesssim g$ (resp. $\gtrsim$) to indicate that $f \leq Cg$ (resp. $\geq$) for some absolute constant C. For any function f, we use $\mathrm{dom} f$ to denote the domain of function f.

For a vector v, We denote $\|v\|$ as the standard Euclidean norm of v and for a symmetric PSD matrix A, we let $\|v\|_A = (v^\top Av)^{1/2}$. For a convex function $f(x)$ that is clear from context, we denote $\|v\|_x = \|v\|_{\nabla^2 f(x)}$ and $\|v\|_x^* = \|v\|_{\nabla^2 f(x)^{-1}}$.

4. Robust Central Path

In this section we show how to move move efficiently from $x(1)$ to $x(\epsilon)$ for some tiny ϵ by staying on a robust version of the central path. Because we are maintaining values that are slightly off-center, we show that our analysis still goes through despite ℓ_∞ perturbations on the order of $O(1/\mathrm{poly}\log(n))$.

4.1. Newton Step

To follow the path $x(t)$, we consider the optimality condition of (3):

$$\begin{aligned} s/t + \nabla\phi(x) &= 0, \\ Ax &= b, \\ A^\top y + s &= c \end{aligned}$$

where $\nabla\phi(x) = (\nabla\phi_1(x_1), \nabla\phi_2(x_2), \cdots, \nabla\phi_m(x_m))$. To handle the error incurred in the progress, we consider the perturbed central path

$$\begin{aligned} s/t + \nabla\phi(x) &= \mu, \\ Ax &= b, \\ A^\top y + s &= c \end{aligned}$$

where μ represent the error between the original central path and our central path. Each iteration, we decrease t by a certain factor. It may increase the error term μ. Therefore, we need a step to decrease the norm of μ. The Newton method to move μ to $\mu + h$ is given by

$$\begin{aligned} \frac{1}{t} \cdot \delta_s^{\mathrm{ideal}} + \nabla^2\phi(x) \cdot \delta_x^{\mathrm{ideal}} &= h, \\ A\delta_x^{\mathrm{ideal}} &= 0, \\ A^\top \delta_y^{\mathrm{ideal}} + \delta_s^{\mathrm{ideal}} &= 0 \end{aligned}$$

where $\nabla^2\phi(x)$ is a block diagonal matrix with the i-th block is given by $\nabla^2\phi_i(x_i)$. Letting $W = (\nabla^2\phi(x))^{-1}$, we can solve this:

$$\begin{aligned} \delta_y^{\mathrm{ideal}} &= -t \cdot \left(AWA^\top\right)^{-1} AWh, \\ \delta_s^{\mathrm{ideal}} &= t \cdot A^\top \left(AWA^\top\right)^{-1} AWh, \\ \delta_x^{\mathrm{ideal}} &= Wh - WA^\top \left(AWA^\top\right)^{-1} AWh. \end{aligned}$$

We define projection matrix $P \in \mathbb{R}^{n \times n}$ as follows

$$P = W^{1/2}A^\top \left(AWA^\top\right)^{-1} AW^{1/2}$$

and then we rewrite them

$$\delta_x^{\mathrm{ideal}} = W^{1/2}(I - P)W^{1/2}\delta_\mu, \tag{4}$$

$$\delta_s^{\mathrm{ideal}} = tW^{-1/2}PW^{1/2}\delta_\mu. \tag{5}$$

One standard way to analyze the central path is to measure the error by $\|\mu\|_{\nabla^2\phi(x)^{-1}}$ and uses the step induced by $h = -\mu$. One can easily prove that if $\|\mu\|_{\nabla^2\phi(x)^{-1}} < \frac{1}{10}$, one step of Newton step decreases the norm by a constant factor. Therefore, one can alternatively decrease t and do a Newton step to follow the path.

4.2. Robust Central Path Method

In this section, we develop a central path method that is robust under certain ℓ_∞ perturbations. Due to the ℓ_∞ perturbation, we measure the error μ by a soft max instead of the ℓ_2 type potential:

Definition 6 *For each $i \in [m]$, let $\mu_i^t(x,s) \in \mathbb{R}^{n_i}$ and $\gamma_i^t(x,s) \in \mathbb{R}$ be defined as follows:*

$$\mu_i^t(x,s) = s_i/t + \nabla\phi_i(x_i), \tag{6}$$
$$\gamma_i^t(x,s) = \|\mu_i^t(x,s)\|_{\nabla^2\phi_i(x_i)^{-1}}, \tag{7}$$

and we define potential function Φ as follows:

$$\Phi^t(x,s) = \sum_{i=1}^m \exp(\lambda\gamma_i^t(x,s))$$

where $\lambda = O(\log m)$.

The *robust central path* is the region (x,s) that satisfies $\Phi^t(x,s) \leq O(m)$. To run our convergence argument, we will be setting λ appropriately so that staying on the robust central path will guarantee a ℓ_∞ bound on γ. Then, we will show how to maintain $\Phi^t(x,s)$ to be small throughout the algorithm while decreasing t, always staying on the robust central path. This is broken into a two step analysis: the progress step (decreasing t) and the centering step (moving x, s to decrease γ).

It is important to note that to follow the robust central path, we no longer pick the standard Newton direction by setting $h = -\mu$. To explain how we pick our centering step, suppose we can move $\mu \to \mu + h$ arbitrarily with the only restriction on the distance $\|h\|_{\nabla^2\phi(x)^{-1}} = \alpha$. Then, the natural step would be

$$h = \arg\min_{\|h\|_{\nabla^2\phi(x)^{-1}}=\alpha} \langle \nabla f(\mu(x,s)), h\rangle$$

where $f(\mu) = \sum_{i=1}^m \exp(\lambda\|\mu\|_{\nabla^2\phi_i(x_i)^{-1}})$. Note that

$$\nabla f(\mu^t(x,s))_i = \lambda\exp(\lambda\gamma_i^t(x,s))/\gamma_i^t(x,s)\cdot\nabla^2\phi_i(x_i)^{-1}\mu_i^t(x,s).$$

Therefore, the solution for the minimization problem is

$$h_i^{\text{ideal}} = -\alpha\cdot c_i^t(x,s)^{\text{ideal}}\mu_i^t(x,s) \in \mathbb{R}^{n_i},$$

where $\mu_i^t(x,s) \in \mathbb{R}^{n_i}$ is defined as Eq. (6) and $c_i^t(x,s) \in \mathbb{R}$ is defined as

$$c_i^t(x,s)^{\text{ideal}} = \frac{\exp(\lambda\gamma_i^t(x,s))/\gamma_i^t(x,s)}{(\sum_{i=1}^m \exp(2\lambda\gamma_i^t(x,s)))^{1/2}}.$$

Eq. (4) and Eq. (5) gives the corresponding ideal step on x and s.

Now, we discuss the perturbed version of this algorithm. Instead of using the exact x and s in the formula of h, we use a $\overline{x}$ which is approximately close to x and a $\overline{s}$ which is close to s. Precisely, we have

$$h_i = -\alpha\cdot c_i^t(\overline{x},\overline{s})\mu_i^t(\overline{x},\overline{s}) \tag{8}$$

where

$$c_i^t(x,s) = \begin{cases} \frac{\exp(\lambda\gamma_i^t(x,s))/\gamma_i^t(x,s)}{(\sum_{i=1}^m \exp(2\lambda\gamma_i^t(x,s)))^{1/2}} & \text{if } \gamma_i^t(x,s) \geq 96\sqrt{\alpha} \\ 0 & \text{otherwise} \end{cases}. \tag{9}$$

Note that our definition of c_i^t ensures that $c_i^t(x,s) \leq \frac{1}{96\sqrt{\alpha}}$ regardless of the value of $\gamma_i^t(x,s)$. This makes sure we do not move too much in any coordinates and indeed when γ_i^t is small, it is fine to set $c_i^t = 0$. Furthermore, for the formula on δ_x and δ_s, we use some matrix $\widetilde{V}$ that is close to $(\nabla^2\phi(x))^{-1}$. Precisely, we have

$$\delta_x = \widetilde{V}^{1/2}(I - \widetilde{P})\widetilde{V}^{1/2}h, \tag{10}$$

$$\delta_s = t \cdot \widetilde{V}^{-1/2}\widetilde{P}\,\widetilde{V}^{1/2}h. \tag{11}$$

where

$$\widetilde{P} = \widetilde{V}^{1/2}A^\top(A\widetilde{V}A^\top)^{-1}A\widetilde{V}^{1/2}.$$

Here we give a quick summary of our algorithm. (The more detailed of our algorithm can be found in Algorithm 5 and 6 in Section C.)

- ROBUSTIPM(A, b, c, ϕ, δ)
  - $\lambda = 2^{16}\log(m)$, $\alpha = 2^{-20}\lambda^{-2}$, $\kappa = 2^{-10}\alpha$.
  - $\delta = \min(\frac{1}{\lambda}, \delta)$.
  - $\nu = \sum_{i=1}^m \nu_i$ where ν_i are the self-concordant parameters of ϕ_i.
  - Modify the convex problem and obtain an initial x and s according to Lemma 41.
  - $t = 1$.
  - While $t > \frac{\delta^2}{4\nu}$
    - Find $\overline{x}$ and $\overline{s}$ such that $\|\overline{x}_i - x_i\|_{\overline{x}_i} < \alpha$ and $\|\overline{s}_i - s_i\|^*_{\overline{x}_i} < t\alpha$ for all i.
    - Find $\widetilde{V}_i$ such that $(1-\alpha)(\nabla^2\phi_i(\overline{x}_i))^{-1} \preceq \widetilde{V}_i \preceq (1+\alpha)(\nabla^2\phi_i(\overline{x}_i))^{-1}$ for all i.
    - Compute $h = -\alpha \cdot c_i^t(\overline{x},\overline{s})\mu_i^t(\overline{x},\overline{s})$ where

$$c_i^t(\overline{x},\overline{s}) = \begin{cases} \frac{\exp(\lambda\gamma_i^t(\overline{x},\overline{s}))/\gamma_i^t(\overline{x},\overline{s})}{(\sum_{i=1}^m \exp(2\lambda\gamma_i^t(\overline{x},\overline{s})))^{1/2}} & \text{if } \gamma_i^t(\overline{x},\overline{s}) \geq 96\sqrt{\alpha} \\ 0 & \text{otherwise} \end{cases}.$$

      and $\mu_i^t(\overline{x},\overline{s}) = \overline{s}_i/t + \nabla\phi_i(\overline{x}_i)$ and $\gamma_i^t(\overline{x},\overline{s}) = \|\mu_i^t(\overline{x},\overline{s})\|_{\nabla^2\phi_i(\overline{x}_i)^{-1}}$
    - Let $\widetilde{P} = \widetilde{V}^{1/2}A^\top(A\widetilde{V}A^\top)^{-1}A\widetilde{V}^{1/2}$.
    - Compute $\delta_x = \widetilde{V}^{1/2}(I - \widetilde{P})\widetilde{V}^{1/2}h$ and $\delta_s = t \cdot \widetilde{V}^{-1/2}\widetilde{P}\,\widetilde{V}^{1/2}h$.
    - Move $x \leftarrow x + \delta_x$, $s \leftarrow s + \delta_s$.
    - $t^{\text{new}} = (1 - \frac{\kappa}{\sqrt{\nu}})t$.
  - Return an approximation solution of the convex problem according to Lemma 41.

Theorem 7 (Robust Interior Point Method) *Consider a convex problem* $\min_{Ax=b, x\in\prod_{i=1}^m K_i} c^\top x$ *where* K_i *are compact convex sets. For each* $i \in [m]$*, we are given a* ν_i*-self concordant barrier function* ϕ_i *for* K_i*. Let* $\nu = \sum_{i=1}^m \nu_i$*. Also, we are given* $x^{(0)} = \arg\min_x \sum_{i=1}^m \phi_i(x_i)$*. Assume that*

1. *Diameter of the set: For any* $x \in \prod_{i=1}^m K_i$*, we have that* $\|x\|_2 \leq R$*.*
2. *Lipschitz constant of the program:* $\|c\|_2 \leq L$*.*

Then, the algorithm ROBUSTIPM *finds a vector* x *such that*

$$c^\top x \leq \min_{Ax=b, x \in \prod_{i=1}^m K_i} c^\top x + LR \cdot \delta,$$
$$\|Ax - b\|_1 \leq 3\delta \cdot \left(R \sum_{i,j} |A_{i,j}| + \|b\|_1 \right),$$
$$x \in \prod_{i=1}^m K_i.$$

in $O(\sqrt{\nu} \log^2 m \log(\frac{\nu}{\delta}))$ *iterations.*

Proof Lemma 41 shows that the initial x and s satisfies

$$\|s + \nabla \phi(x)\|_x^* \leq \delta \leq \frac{1}{\lambda}$$

where the last inequality is due to our step $\delta \leftarrow \min(\frac{1}{\lambda}, \delta)$. This implies that $\gamma_i^1(x,s) = \|s_i + \nabla \phi_i(x_i)\|_{x_i}^* \leq \frac{1}{\lambda}$ and hence $\Phi^1(x,s) \leq e \cdot m \leq 80\frac{m}{\alpha}$ for the initial x and s. Apply Lemma 15 repetitively, we have that $\Phi^t(x,s) \leq 80\frac{m}{\alpha}$ during the whole algorithm. In particular, we have this at the end of the algorithm. This implies that

$$\|s_i + \nabla \phi_i(x_i)\|_{x_i}^* \leq \frac{\log(80\frac{m}{\alpha})}{\lambda} \leq 1$$

at the end. Therefore, we can apply Lemma 42 to show that

$$\langle c, x \rangle \leq \langle c, x^* \rangle + 4t\nu \leq \langle c, x^* \rangle + \delta^2$$

where we used the stop condition for t at the end. Note that this guarantee holds for the modified convex program. Since the error is δ^2, Lemma 41 shows how to get an approximate solution for the original convex program with error $LR \cdot \delta$.

The number of steps follows from the fact we decrease t by $1 - \frac{1}{\sqrt{\nu} \log^2 m}$ factor every iteration. ■

Acknowledgments

The authors would like to thank Haotian Jiang, Swati Padmanabhan, Ruoqi Shen, Zhengyu Wang, Xin Yang and Peilin Zhong for useful discussions.

References

Deeksha Adil, Rasmus Kyng, Richard Peng, and Sushant Sachdeva. Iterative refinement for ℓ_p-norm regression. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1405–1424. SIAM, 2019.

Naman Agarwal, Sham Kakade, Rahul Kidambi, Yin Tat Lee, Praneeth Netrapalli, and Aaron Sidford. Leverage score sampling for faster accelerated regression and erm. *arXiv preprint arXiv:1711.08426*, 2017.

Zeyuan Allen-Zhu. Natasha: Faster Non-Convex Stochastic Optimization via Strongly Non-Convex Parameter. In *Proceedings of the 34th International Conference on Machine Learning*, ICML '17, 2017a. Full version available at http://arxiv.org/abs/1702.00763.

Zeyuan Allen-Zhu. Katyusha: The first direct acceleration of stochastic gradient methods. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1200–1205. ACM, 2017b.

Zeyuan Allen-Zhu. Natasha 2: Faster Non-Convex Optimization Than SGD. In *Proceedings of the 32nd Conference on Neural Information Processing Systems*, NIPS '18, 2018a. Full version available at http://arxiv.org/abs/1708.08694.

Zeyuan Allen-Zhu. Katyusha X: Practical Momentum Method for Stochastic Sum-of-Nonconvex Optimization. In *Proceedings of the 35th International Conference on Machine Learning*, ICML '18, 2018b. Full version available at http://arxiv.org/abs/1802.03866.

Zeyuan Allen-Zhu and Elad Hazan. Variance Reduction for Faster Non-Convex Optimization. In *Proceedings of the 33rd International Conference on Machine Learning*, ICML '16, 2016. Full version available at http://arxiv.org/abs/1603.05643.

Zeyuan Allen-Zhu and Yang Yuan. Improved SVRG for Non-Strongly-Convex or Sum-of-Non-Convex Objectives. In *Proceedings of the 33rd International Conference on Machine Learning*, ICML '16, 2016. Full version available at http://arxiv.org/abs/1506.01972.

Josh Alman. Limits on the universal method for matrix multiplication. *arXiv preprint arXiv:1812.08731*, 2018.

Josh Alman and Virginia Vassilevska Williams. Further limitations of the known approaches for matrix multiplication. In *ITCS*. arXiv preprint arXiv:1712.07246, 2018a.

Josh Alman and Virginia Vassilevska Williams. Limits on all known (and some unknown) approaches to matrix multiplication. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2018b.

Peter L Bartlett, Olivier Bousquet, and Shahar Mendelson. Local rademacher complexities. *The Annals of Statistics*, 33(4):1497–1537, 2005.

Léon Bottou and Olivier Bousquet. The tradeoffs of large scale learning. In *Advances in neural information processing systems*, pages 161–168, 2008.

Sébastien Bubeck, Michael B Cohen, Yin Tat Lee, and Yuanzhi Li. An homotopy method for ℓ_p regression provably beyond self-concordance and in input-sparsity time. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 1130–1137. ACM, 2018.

Xue Chen, Daniel M Kane, Eric Price, and Zhao Song. Fourier-sparse interpolation without a frequency gap. In *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*, pages 741–750. IEEE, 2016.

Kenneth L Clarkson. Subgradient and sampling algorithms for ℓ_1 regression. In *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms (SODA)*, pages 257–266, 2005.

Michael B Cohen, Yin Tat Lee, and Zhao Song. Solving linear programs in the current matrix multiplication time. In *STOC*. https://arxiv.org/pdf/1810.07896.pdf, 2019.

James W Cooley and John W Tukey. An algorithm for the machine calculation of complex Fourier series. *Mathematics of computation*, 19(90):297–301, 1965.

Don Coppersmith and Shmuel Winograd. Matrix multiplication via arithmetic progressions. In *Proceedings of the nineteenth annual ACM symposium on Theory of computing(STOC)*, pages 1–6. ACM, 1987.

Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

David R Cox. The regression analysis of binary sequences. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 215–242, 1958.

Dominik Csiba. Data sampling strategies in stochastic algorithms for empirical risk minimization. *arXiv preprint arXiv:1804.00437*, 2018.

Anirban Dasgupta, Petros Drineas, Boulos Harb, Ravi Kumar, and Michael W Mahoney. Sampling algorithms and coresets for ℓ_p regression. *SIAM Journal on Computing*, 38(5):2060–2078, 2009.

Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in neural information processing systems*, pages 1646–1654, 2014.

Alexandre Défossez and Francis Bach. Constant step size least-mean-square: Bias-variance trade-offs and optimal sampling distributions. *arXiv preprint arXiv:1412.0156*, 2014.

Aymeric Dieuleveut and Francis Bach. Nonparametric stochastic approximation with large step-sizes. *The Annals of Statistics*, 44(4):1363–1399, 2016.

Vitaly Feldman, Venkatesan Guruswami, Prasad Raghavendra, and Yi Wu. Agnostic learning of monomials by halfspaces is hard. *SIAM Journal on Computing*, 41(6):1558–1590, 2012.

Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.

Roy Frostig, Rong Ge, Sham M Kakade, and Aaron Sidford. Competing with the empirical risk minimizer in a single pass. In *Conference on learning theory (COLT)*, pages 728–763, 2015.

Alon Gonen and Shai Shalev-Shwartz. Fast rates for empirical risk minimization of strict saddle problems. *arXiv preprint arXiv:1701.04271*, 2017.

Haitham Hassanieh, Piotr Indyk, Dina Katabi, and Eric Price. Nearly optimal sparse fourier transform. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 563–578. ACM, 2012a.

Haitham Hassanieh, Piotr Indyk, Dina Katabi, and Eric Price. Simple and practical algorithm for sparse Fourier transform. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pages 1183–1194. SIAM, 2012b.

David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant. *Applied logistic regression*, volume 398. John Wiley & Sons, 2013.

Piotr Indyk and Michael Kapralov. Sample-optimal fourier sampling in any constant dimension. In *Foundations of Computer Science (FOCS), 2014 IEEE 55th Annual Symposium on*, pages 514–523. IEEE, 2014.

Piotr Indyk, Michael Kapralov, and Eric Price. (Nearly) Sample-optimal sparse Fourier transform. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 480–499. SIAM, 2014.

Chi Jin, Lydia T Liu, Rong Ge, and Michael I Jordan. On the local minima of the empirical risk. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 4901–4910, 2018.

Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in neural information processing systems*, pages 315–323, 2013.

Michael Kapralov. Sparse Fourier transform in any constant dimension with nearly-optimal sample complexity in sublinear time. In *Symposium on Theory of Computing Conference, STOC'16, Cambridge, MA, USA, June 19-21, 2016*, 2016.

Michael Kapralov. Sample efficient estimation and recovery in sparse fft via isolation on average. In *Foundations of Computer Science, 2017. FOCS'17. IEEE 58th Annual IEEE Symposium on*. https://arxiv.org/pdf/1708.04544, 2017.

Roger Koenker. Galton, edgeworth, frisch, and prospects for quantile regression in econometrics. *Journal of Econometrics*, 95(2):347–374, 2000.

Roger Koenker. *Quantile Regression*. Cambridge University Press, 2005.

Roger Koenker and Kevin F Hallock. Quantile regression. *Journal of economic perspectives*, 15 (4):143–156, 2001.

Beatrice Laurent and Pascal Massart. Adaptive estimation of a quadratic functional by model selection. *Annals of Statistics*, pages 1302–1338, 2000.

François Le Gall. Powers of tensors and fast matrix multiplication. In *Proceedings of the 39th international symposium on symbolic and algebraic computation(ISSAC)*, pages 296–303. ACM, 2014.

Francois Le Gall and Florent Urrutia. Improved rectangular matrix multiplication using powers of the coppersmith-winograd tensor. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms(SODA)*, pages 1029–1046. SIAM, 2018.

Nicolas Le Roux, Mark W Schmidt, and Francis R Bach. A stochastic gradient method with an exponential convergence rate for finite training sets. In *NIPS*, pages 2672–2680, 2012.

Yin Tat Lee. Uniform sampling and inverse maintenance. In *Talk at Michael Cohen Memorial Symposium*. Available at: https://simons.berkeley.edu/talks/welcome-andbirds-eye-view-michaels-work., 2017.

Lihua Lei, Cheng Ju, Jianbo Chen, and Michael I Jordan. Non-convex finite-sum optimization via scsg methods. In *Advances in Neural Information Processing Systems*, pages 2348–2358, 2017.

Hongzhou Lin, Julien Mairal, and Zaid Harchaoui. A universal catalyst for first-order optimization. In *Advances in Neural Information Processing Systems*, pages 3384–3392, 2015.

Hongzhou Lin, Julien Mairal, and Zaid Harchaoui. Catalyst acceleration for first-order convex optimization: from theory to practice. *arXiv preprint arXiv:1712.05654*, 2017.

Yichao Lu, Paramveer Dhillon, Dean P Foster, and Lyle Ungar. Faster ridge regression via the subsampled randomized hadamard transform. In *Advances in neural information processing systems*, pages 369–377, 2013.

Zhuang Ma, Yichao Lu, and Dean Foster. Finding linear structure in large datasets with scalable canonical correlation analysis. In *International Conference on Machine Learning*, pages 169–178, 2015.

Eric Moulines and Francis R Bach. Non-asymptotic analysis of stochastic approximation algorithms for machine learning. In *Advances in Neural Information Processing Systems*, pages 451–459, 2011.

Tomoya Murata and Taiji Suzuki. Doubly accelerated stochastic variance reduced dual averaging method for regularized empirical risk minimization. In *Advances in Neural Information Processing Systems*, pages 608–617, 2017.

Elizbar A Nadaraya. On estimating regression. *Theory of Probability & Its Applications*, 9(1): 141–142, 1964.

Vasileios Nakos, Zhao Song, and Zhengyu Wang. (Nearly) sample-optimal sparse Fourier transform in any dimension; RIPless and Filterless. In *manuscript*, 2019.

Arkadi Nemirovski, Anatoli Juditsky, Guanghui Lan, and Alexander Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on optimization*, 19(4):1574–1609, 2009.

Yurii Nesterov. Introductory lectures on convex programming volume i: Basic course. *Lecture notes*, 1998.

Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2004.

Yurii Nesterov and Sebastian U Stich. Efficiency of the accelerated coordinate descent method on structured optimization problems. *SIAM Journal on Optimization*, 27(1):110–123, 2017.

Yurii E Nesterov. A method for solving the convex programming problem with convergence rate $O(1/k^2)$. In *Dokl. Akad. Nauk SSSR*, volume 269, pages 543–547, 1983.

Mert Pilanci and Martin J Wainwright. Iterative hessian sketch: Fast and accurate solution approximation for constrained least-squares. *The Journal of Machine Learning Research*, 17(1): 1842–1879, 2016.

Mert Pilanci and Martin J Wainwright. Newton sketch: A near linear-time optimization algorithm with linear-quadratic convergence. *SIAM Journal on Optimization*, 27(1):205–245, 2017.

Boris T Polyak and Anatoli B Juditsky. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, 30(4):838–855, 1992.

Eric Price and Zhao Song. A robust sparse Fourier transform in the continuous setting. In *Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on*, pages 583–600. IEEE, 2015.

Eric Price, Zhao Song, and David P. Woodruff. Fast regression with an ℓ_∞ guarantee. In *International Colloquium on Automata, Languages, and Programming (ICALP)*, 2017.

Eric C. Price. *Sparse recovery and Fourier sampling*. PhD thesis, Massachusetts Institute of Technology, 2013.

Sashank J Reddi, Ahmed Hefny, Suvrit Sra, Barnabas Poczos, and Alex Smola. Stochastic variance reduction for nonconvex optimization. In *International conference on machine learning*, pages 314–323, 2016.

Mark Schmidt, Nicolas Le Roux, and Francis Bach. Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, 162(1-2):83–112, 2017.

Shai Shalev-Shwartz. SDCA without duality, regularization, and individual convexity. In *International Conference on Machine Learning*, pages 747–754, 2016.

Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

Shai Shalev-Shwartz and Tong Zhang. Stochastic dual coordinate ascent methods for regularized loss minimization. *Journal of Machine Learning Research*, 14(Feb):567–599, 2013.

Shai Shalev-Shwartz and Tong Zhang. Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization. In *International Conference on Machine Learning*, pages 64–72, 2014.

Fanhua Shang, Yuanyuan Liu, James Cheng, KW Ng, and Yuichi Yoshida. Variance reduced stochastic gradient descent with sufficient decrease. *arXiv preprint arXiv:1703.06807*, 2017.

Volker Strassen. Gaussian elimination is not optimal. *Numerische Mathematik*, 13(4):354–356, 1969.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

Pravin M Vaidya. Speeding-up linear programming using fast matrix multiplication. In *FOCS*. IEEE, 1989.

Vladimir Vapnik. Principles of risk minimization for learning theory. In *Advances in neural information processing systems*, pages 831–838, 1992.

Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 2013.

Geoffrey S Watson. Smooth regression analysis. *Sankhyā: The Indian Journal of Statistics, Series A*, pages 359–372, 1964.

Virginia Vassilevska Williams. Multiplying matrices faster than coppersmith-winograd. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing (STOC)*, pages 887–898. ACM, 2012.

Lin Xiao and Tong Zhang. A proximal stochastic gradient method with progressive variance reduction. *SIAM Journal on Optimization*, 24(4):2057–2075, 2014.

Eric P Xing, Michael I Jordan, and Stuart Russell. A generalized mean field algorithm for variational inference in exponential families. In *Proceedings of the Nineteenth conference on Uncertainty in Artificial Intelligence*, pages 583–591. Morgan Kaufmann Publishers Inc., 2002.

Lijun Zhang, Tianbao Yang, and Rong Jin. Empirical risk minimization for stochastic convex optimization: $O(1/n)$-and $O(1/n^2)$-type of risk bounds. *arXiv preprint arXiv:1702.02030*, 2017.

Yuchen Zhang and Lin Xiao. Stochastic primal-dual coordinate method for regularized empirical risk minimization. *The Journal of Machine Learning Research*, 18(1):2939–2980, 2017.

Shun Zheng, Jialei Wang, Fen Xia, Wei Xu, and Tong Zhang. A general distributed dual coordinate optimization framework for regularized loss minimization. *The Journal of Machine Learning Research*, 18(1):4096–4117, 2017.

Hui Zou and Trevor Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(2):301–320, 2005.