Abstract

In distributed statistical learning, $N$ samples are split across $m$ machines and a learner wishes to use minimal communication to learn as well as if the examples were on a single machine. This model has received substantial interest in machine learning due to its scalability and potential for parallel speedup. However, in high-dimensional settings, where the number examples is smaller than the number of features (“dimension”), the speedup afforded by distributed learning may be overshadowed by the cost of communicating a single example. This paper investigates the following question: When is it possible to learn a $d$-dimensional model in the distributed setting with total communication sublinear in $d$? Starting with a negative result, we observe that for learning $\ell_1$-bounded or sparse linear models, no algorithm can obtain optimal error until communication is linear in dimension. Our main result is that by slightly relaxing the standard boundedness assumptions for linear models, we can obtain distributed algorithms that enjoy optimal error with communication logarithmic in dimension. This result is based on a family of algorithms that combine mirror descent with randomized sparsification/quantization of iterates, and extends to the general stochastic convex optimization model.

1. Introduction

In statistical learning, a learner receives examples $z_1, \ldots, z_N$ i.i.d. from an unknown distribution $D$. Their goal is to output a hypothesis $h \in \mathcal{H}$ that minimizes the prediction error $L_D(h) := \mathbb{E}_{z \sim D} \ell(h, z)$, and in particular to guarantee that excess risk of the learner is small, i.e.

$$L_D(h) - \inf_{h \in \mathcal{H}} L_D(h) \leq \varepsilon(\mathcal{H}, N),$$

where $\varepsilon(\mathcal{H}, N)$ is a decreasing function of $N$. This paper focuses on distributed statistical learning. We consider a distributed setting, where the $N$ examples are split evenly across $m$ machines, with $n := N/m$ examples per machine, and the learner wishes to achieve an excess risk guarantee such as (1) with minimal overhead in computation or communication.

Distributed learning has been the subject of extensive investigation due to its scalability for processing massive data: We may wish to efficiently process datasets that are spread across multiple data-centers, or we may want to distribute data across multiple machines to allow for parallelization of learning procedures. The question of parallelizing computation via distributed learning is a well-explored problem (Bekkerman et al., 2011; Recht et al., 2011; Dekel et al., 2012; Chaturapruk et al., 2015). However, one drawback that limits the practical viability of these approaches is that the communication cost between machines may overshadow gains in parallel speedup (Bijral et al., 2016). Indeed, for high-dimensional statistical inference tasks where $N$ could be much smaller than the dimension $d$, or in modern deep learning where the number of model parameters exceeds the number of examples (e.g. (He et al., 2016)), communicating a single gradient or sending the raw model parameters between machines constitutes a significant overhead.

Algorithms with reduced communication complexity in distributed learning have received significant recent development (Seide et al., 2014; Alistarh et al., 2017; Suresh et al., 2017; Bernstein et al., 2018; Tang et al., 2018), but typical results here take as a given that when gradients or examples live in $d$ dimensions, communication will scale as $\Omega(d)$. Our goal is to revisit this tacit assumption and understand when it can be relaxed. We explore the question of sublinear communication:

Suppose a hypothesis class $\mathcal{H}$ has $d$ parameters. When is it possible to achieve optimal excess risk for $\mathcal{H}$ in the distributed setting using $o(d)$ communication?

1.1. Sublinear Communication for Linear Models?

Let us first focus on linear models, which are a special case of the general learning setup (1). We restrict to linear hypotheses of the form $h_w(x) = \langle w, x \rangle$ where $w, x \in \mathbb{R}^d$ and write $\ell(h_w, z) = \phi(\langle w, x \rangle, y)$, where $\phi(\cdot, y)$ is a fixed
We see that when
where
\( (ERM) \), is
To be precise, we would like to develop algorithms
that we can obtain nontrivial excess risk when
Hence, we can get nontrivial excess risk even when the
setting is
1
Recall the definition of the \( \ell_p \) norm: \( \|w\|_p = \left( \sum_{i=1}^{d} |w_i|^p \right)^{1/p} \).

1.2. Contributions
We provide new communication-efficient distributed learning algorithms and lower bounds for \( \ell_p/\ell_q \)-bounded linear models, and more broadly, stochastic convex optimization. We make the following observations:

- For \( \ell_2/\ell_2 \)-bounded linear models, sublinear communication is achievable, and is obtained by using a derandomized Johnson-Lindenstrauss transform to compress examples and weights.

- For \( \ell_1/\ell_{\infty} \)-bounded linear models, no distributed algorithm can obtain optimal excess risk until communication is linear in dimension.

These observations lead to our main result. We show that by relaxing the \( \ell_1/\ell_\infty \)-boundedness assumption and instead learning \( \ell_1/\ell_q \)-bounded models for a constant \( q < \infty \), one unlocks a plethora of new algorithmic tools for sublinear distributed learning:

1. We give an algorithm with optimal rates matching (3), with total communication \( \text{poly}(N, m^q, \log d) \).

2. We extend the sublinear-communication algorithm to give refined guarantees, including instance-dependent small loss bounds for smooth losses, fast rates for strongly convex losses, and optimal rates for matrix learning problems.

Our main algorithm is a distributed version of mirror descent that uses randomized sparsification of weight vectors to reduce communication. Beyond learning in linear models, the algorithm enjoys guarantees for the more general distributed stochastic convex optimization model.

To elaborate on the fast rates mentioned above, another important case where learning is possible when \( N \ll d \) is the sparse high-dimensional linear model setup, centered on compressed sensing. Here, the standard result is that when \( \phi \) is strongly convex and the benchmark class consists of \( k \)-sparse linear predictors, i.e. \( \mathcal{W}_0 = \{ w \in \mathbb{R}^d \mid \|w\|_0 \leq k \} \), one can guarantee

\[
L_{\mathcal{D}}(\mathcal{W}) - \inf_{w \in \mathcal{W}_0} L_{\mathcal{D}}(w) = \Theta\left( \frac{k \log (d/k)}{N} \right).
\]

With \( \ell_\infty \)-bounded features, no algorithm can obtain optimal excess risk for this setting until communication is linear in dimension, even under compressed sensing-style assumptions. When features are \( \ell_q \)-bounded however, our general machinery gives optimal fast rates matching (5) under Lasso-style assumptions, with communication \( \text{poly}(N^q, \log d) \).
sparsified mirror descent algorithms that leads to our main results and sketch the analysis.

1.3. Related Work

Much of the work in algorithm design for distributed learning and optimization does not explicitly consider the number of bits used in communication per messages, and instead tries to make communication efficient via other means, such as decreasing the communication frequency or making learning robust to network disruptions (Duchi et al., 2012; Zhang et al., 2012). Other work reduces the number of bits of communication, but still requires that this number be linear in the dimension $d$. One particularly successful line of work in this vein is low-precision training, which represents the numbers used for communication and elsewhere within the algorithm using few bits (Alistarh et al., 2017; Zhang et al., 2017; Seide et al., 2014; Bernstein et al., 2018; Tang et al., 2018; Stich et al., 2018; Alistarh et al., 2018). Although low-precision methods have seen great success and adoption in neural network training and inference, low-precision methods are fundamentally limited to use bits proportional to $d$; once they go down to one bit per number there is no additional benefit from decreasing the precision. Some work in this space tries to use sparsification to further decrease the communication cost of learning, either on its own or in combination with a low-precision representation for numbers (Alistarh et al., 2017; Wangni et al., 2018; Wang et al., 2018). While the majority of these works apply low-precision and sparsification to gradients, a number of recent works apply sparsification to model parameters (Tang et al., 2018; Stich et al., 2018; Alistarh et al., 2018); We also adopt this approach. The idea of sparsifying weights is not new (Shalev-Shwartz et al., 2010), but our work is the first to provably give communication logarithmic in dimension. To achieve this, our assumptions and analysis are quite a bit different from the results mentioned above, and we crucially use mirror descent, departing from the gradient descent approaches in (Tang et al., 2018; Stich et al., 2018; Alistarh et al., 2018).

Lower bounds on the accuracy of learning procedures with limited memory and communication have been explored in several settings, including mean estimation, sparse regression, learning parities, detecting correlations, and independence testing (Shamir, 2014; Duchi et al., 2014; Garg et al., 2014; Steinhardt & Duchi, 2015; Braverman et al., 2016; Steinhardt et al., 2016; Acharya et al., 2019a; Raz, 2018; Han et al., 2018; Sahasranand & Tyagi, 2018; Dagan & Shamir, 2018; Dagan et al., 2019). In particular, the results of (Steinhardt & Duchi, 2015) and (Braverman et al., 2016) imply that optimal algorithms for distributed sparse regression need communication much larger than the sparsity level under various assumptions on the number of machines and the communication protocol.

2. Linear Models: Basic Results

In this section we develop basic upper and lower bounds for communication in $\ell_2/\ell_2$- and $\ell_1/\ell_\infty$-bounded linear models. Our goal is to highlight that the communication complexity of distributed learning and the statistical complexity of centralized learning do not in general coincide, and to motivate the $\ell_1/\ell_\infty$-boundedness assumption under which we derive communication-efficient algorithms in Section 3.

2.1. Preliminaries

We formulate our results in a distributed communication model following Shamir (2014). Recalling that $n = N/m$, the model is as follows.

- For machine $i = 1, \ldots, m$:
  - Receive $n$ i.i.d. examples $S_i := z_{i1}^1, \ldots, z_{in}^1$.
  - Compute message $W_i = f_i(S_i; W_1, \ldots, W_{i-1})$, where $W_i$ is at most $b_i$ bits.
- Return $W = f(W_1, \ldots, W_m)$.

We refer to $\sum_{i=1}^m b_i$ as the total communication, and we refer to any protocol with $b_i \leq b \forall i$ as a $(b, n, m)$ protocol. As a special case, this model captures a serial distributed learning setting where machines proceed one after another: Each machine does some computation on their data $z_{i1}^1, \ldots, z_{in}^1$ and previous messages $W_1, \ldots, W_{i-1}$, then broadcasts their own message $W_i$ to all subsequent machines, and the final model in (1) is computed from $W$, either on machine $m$ or on a central server. The model also captures protocols in which each machine independently computes a local estimator and sends it to a central server, which aggregates the local estimators to produce a final estimator (Zhang et al., 2012). All of our upper bounds have the serial structure above, and our lower bounds apply to any $(b, n, m)$ protocol.

2.2. $\ell_2/\ell_2$-Bounded Models

In the $\ell_2/\ell_2$-bounded setting, we can achieve sample optimal learning with sublinear communication by using dimensionality reduction. The idea is to project examples into $k = O(N)$ dimensions using the Johnson-Lindenstrauss transform, then perform a naive distributed implementation of any standard learning algorithm in the projected space. Here we implement the approach using stochastic gradient descent.

To project the examples onto the same subspace, the machines need to agree on a JL transformation matrix. To do so with little communication, we consider the derandomized sparse JL transform of Kane & Nelson (2010), which constructs a collection $\mathcal{A}$ of matrices in $\mathbb{R}^{k \times d}$ with $|\mathcal{A}| = \exp(O(\log(k/\delta) \cdot \log d))$ for confidence parameter $\delta$. The first machine randomly selects an entry of $\mathcal{A}$ and sends the identity of this matrix using $O(\log(k/\delta) \cdot \log d)$ bits to
the other $m-1$ machines. The dimension $k$ and parameter $\delta$ are chosen as a function of $N$.

Each machine uses the matrix $A$ to project its features down to $k$ dimensions. Letting $x'_t = Ax_t$ denote the projected features, the first machine starts with a $k$-dimensional weight vector $u_1 = 0$ and performs the online gradient descent update (Zinkevich, 2003; Cesa-Bianchi & Lugosi, 2006) over its $n$ projected samples as:

$$u_t \leftarrow u_{t-1} - \eta \nabla \phi((u_t, x'_t), y_t),$$

where $\eta > 0$ is the learning rate. Once the first machine has passed over all its samples, it broadcasts the last iterate $u_{n+1}$ as well the average $\frac{1}{n} \sum_{t=1}^{n} u_t$, which takes $O(k)$ communication. The next machine performs the same sequence of gradient updates on its own data using $u_{n+1}$ as the initialization, then passes its final iterate and the updated average to the next machine. This repeats until we arrive at the $m$th machine. The $m$th machine computes the $k$-dimensional vector $\tilde{u} := \frac{1}{n} \sum_{t=1}^{n} u_t$, and returns $\overline{w} = A^T \tilde{u}$ as the solution.

**Theorem 1.** When $\phi$ is $L$-Lipschitz and $k = O(N \log (dN))$, the strategy above guarantees that

$$E_S E_A [L_D(\overline{w})] - \inf_{w \in W_2} L_D(w) \leq O\left(\sqrt{\frac{L^2 B^2_2 R^2_2}{N}}\right),$$

where $E_S$ denotes expectation over samples and $E_A$ denotes expectation over the algorithm’s randomness. The total communication is $O(mN \log (dN) \log (LB_2 R_2 N) + m \log (dN) \log d)$ bits.

### 2.3. $\ell_1/\ell_\infty$-Bounded Models: Model Compression

While the results for the $\ell_2/\ell_2$-bounded setting are encouraging, they are not useful in the common situation where features are dense. When features are $\ell_\infty$-bounded, Equation (4) shows that one can obtain nearly dimension-independent excess risk so long as they restrict to $\ell_1$-bounded weights. This $\ell_1/\ell_\infty$-bounded setting is particularly important because it captures the fundamental problem of learning from a finite hypothesis class, or aggregation (Tsybakov, 2003): Given a class $\mathcal{H}$ of $\{\pm 1\}$-valued predictors with $|\mathcal{H}| < \infty$ we can set $x = (h(z))_{h \in \mathcal{H}} \in \mathbb{R}^{|\mathcal{H}|}$, in which case (4) turns into the familiar finite class bound

$$\sqrt{\log |\mathcal{H}| / N} \quad \text{(Shalev-Shwartz & Ben-David, 2014).}$$

Thus, algorithms with communication sublinear in dimension for the $\ell_1/\ell_\infty$ setting would lead to positive results in the general setting (1).

As first positive result in this direction, we observe that by using the well-known technique of randomized sparsification or Maurey sparsification, we can compress models to require only logarithmic communication while preserving excess risk. The method is simple: Suppose we have a weight vector $w$ that lies in the simplex $\Delta_d$. We sample $s$ elements of $[d]$ i.i.d. according to $w$ and return the empirical distribution, which we will denote $Q^s(w)$. The empirical distribution is always $s$-sparse and can be communicated using at most $O(s \log (ed/s))$ bits when $s \leq d$. It follows from standard concentration tools that by taking $s$ large enough the empirical distribution will approximate the true vector $w$ arbitrarily well.

The following lemma shows that Maurey sparsification indeed provides a dimension-independent approximation to the excess risk in the $\ell_1/\ell_\infty$-bounded setting. It applies to a version of the Maurey technique for general vectors, which is given in Algorithm 1.

**Lemma 1.** Let $w \in \mathbb{R}^d$ be fixed and suppose features belong to $\mathcal{X}_\infty$. When $\phi$ is $L$-Lipschitz, Algorithm 1 guarantees that

$$\mathbb{E} L_D(Q^s(w)) \leq L_D(w) + \left(\frac{2L^2 B^2_2 |w|^2}{s}\right)^{1/2},$$

where the expectation is with respect to the algorithm’s randomness. Furthermore, when $\phi$ is $\beta$-smooth $^4$ Algorithm 1 guarantees:

$$\mathbb{E} L_D(Q^s(w)) \leq L_D(w) + \beta \frac{s}{|w|^2}.$$

The number of bits required to communicate $Q^s(w)$, including sending the scalar $|w|^2$, up to numerical precision, is at most $O(s \log (ed/s) + \log (LB_1 R_2 s))$. Thus, if any single machine is able to find an estimator $\overline{w}$ with good excess risk, they can communicate it to any other machine while preserving the excess risk with sublinear communication. In particular, to preserve the optimal excess risk guarantee in (4) for a Lipschitz loss such as absolute or hinge, the total bits of communication required is only

We refer to the method as Maurey sparsification in reference to Maurey’s early use of the technique in Banach spaces (Pisier, 1980), which predates its long history in learning theory (Jones, 1992; Barron, 1993; Zhang, 2002).

$^4$That $O(s \log (ed/s))$ bits rather than, e.g., $O(s \log d)$ bits suffice is a consequence of the usual “stars and bars” counting argument. We expect one can bring the expected communication down further using an adaptive scheme such as Elias coding, as in Alistarh et al. (2017).
weaker guarantee from Maurey. Alas, we have only shown that Theorem 2.

The value of \( \ell_1/\ell_\infty \) bounds is almost immediately from reduc-

The lower bound also extends to the case of multiple ex-

Theorem 2. Consider the problem of learning with linear

Consider square loss regression in the \((b,1,N)\) model. For any algorithm \( \bar{w} \) there exists a dis-

Proposition 2. Consider square loss regression in the

\( \mathcal{W} \) is a convex constraint set, and subgradients \( \partial \ell(w,z) \) are assumed to belong to \( \mathcal{X}_q^* = \{ x \in \mathbb{R}^d \mid \Vert x \Vert_q \leq R_q \} \). This setting captures linear models with \( \ell_1 \)-bounded weights and \( \ell_q \)-bounded features as a special case, but is more general, since the loss can be any Lipschitz function of \( w \).

We have already shown that one cannot expect sublinear-

Statistical complexity and nontriviality. For the dual \( \ell_1/\ell_\infty \) setup in (2) the optimal rate is \( \Theta(\sqrt{\log d/N}) \). While our goal is to find minimal assumptions that allow for distributed learning with sublinear communication, the reader

3See Appendix B.2 for additional discussion of compressed sensing based assumptions under which sublinear communication may be possible.

\( \mathcal{W} \) is a convex constraint set, and subgradients \( \partial \ell(w,z) \) are assumed to belong to \( \mathcal{X}_q^* = \{ x \in \mathbb{R}^d \mid \Vert x \Vert_q \leq R_q \} \). This setting captures linear models with \( \ell_1 \)-bounded weights and \( \ell_q \)-bounded features as a special case, but is more general, since the loss can be any Lipschitz function of \( w \).

We have already shown that one cannot expect sublinear-

Statistical complexity and nontriviality. For the dual \( \ell_1/\ell_\infty \) setup in (2) the optimal rate is \( \Theta(\sqrt{\log d/N}) \). While our goal is to find minimal assumptions that allow for distributed learning with sublinear communication, the reader

\( \mathcal{W} \) is a convex constraint set, and subgradients \( \partial \ell(w,z) \) are assumed to belong to \( \mathcal{X}_q^* = \{ x \in \mathbb{R}^d \mid \Vert x \Vert_q \leq R_q \} \). This setting captures linear models with \( \ell_1 \)-bounded weights and \( \ell_q \)-bounded features as a special case, but is more general, since the loss can be any Lipschitz function of \( w \).

We have already shown that one cannot expect sublinear-

Statistical complexity and nontriviality. For the dual \( \ell_1/\ell_\infty \) setup in (2) the optimal rate is \( \Theta(\sqrt{\log d/N}) \). While our goal is to find minimal assumptions that allow for distributed learning with sublinear communication, the reader
may wonder at this point whether we have made the problem easier statistically by moving to the $\ell_1/\ell_q$ assumption. The answer is “yes, but only slightly.” When $q$ is constant the optimal rate for $\ell_1/\ell_q$-bounded models is $\Theta(1/N)^q$,7 and so the effect of this assumption is to shave off the $\log d$ factor that was present in (4).

3.1. Lipschitz Losses

Our main algorithm is called sparsified mirror descent (Algorithm 2). The idea is to run the online mirror descent algorithm (Ben-Tal & Nemirovski, 2001; Hazan, 2016) in serial across the machines and sparsify the iterates whenever we move from one machine to the next.

In a bit more detail, Algorithm 2 proceeds from machine to machine sequentially. On each machine, the algorithm generates a sequence of iterates $w_1^i, \ldots, w_n^i$ by doing a single pass over the machine’s $n$ examples $x_1^i, \ldots, x_n^i$ using the mirror descent update with regularizer $\mathcal{R}(w) = \frac{1}{2} \|w\|_p^2$, where $\frac{1}{p} + \frac{1}{q} = 1$, and using stochastic gradients $\nabla_i v^t \in \partial \mathcal{L}(w^t, z^t)$. After the last example is processed on machine $i$, we compress the last iterate using Maurey sparsification (Algorithm 1) and send it to the next machine, where the process is repeated.

To formally describe the algorithm, we recall the definition of the Bregman divergence. Given a convex regularization function $\mathcal{R} : \mathbb{R}^d \to \mathbb{R}$, the Bregman divergence for $\mathcal{R}$ is defined as $D_\mathcal{R}(w, w') = \mathcal{R}(w) - \mathcal{R}(w') - \langle \nabla \mathcal{R}(w'), w - w' \rangle$.

For the $\ell_1/\ell_q$ setting we exclusively use the regularizer $\mathcal{R}(w) = \frac{1}{2} \|w\|_p^2$, where $\frac{1}{p} + \frac{1}{q} = 1$.

The main guarantee for Algorithm 2 is as follows.

**Theorem 3.** Let $q \geq 2$ be fixed. Suppose that subgradients belong to $X^2_q$, and that $\mathcal{W} \subseteq \mathcal{W}_1$. If we run Algorithm 2 with $\eta = \frac{B_1}{R_q} \sqrt{\frac{1}{C_q N}}$ and with initial point $\bar{w} = 0$, then whenever $s = \Omega(m^{2(q-1)})$ and $s_0 = \Omega(N^{\frac{3}{2}})$ the algorithm guarantees

$$\mathbb{E}[L_D(\bar{w})] - L_D(w^*) \leq O\left(\sqrt{\frac{B_1^2 R_q^2 C_q}{N}}\right),$$

where $C_q = q - 1$ is a constant depending only on $q$.

The total number of bits sent by each machine—besides communicating the final iterate $\bar{w}$—is at most $O(m^{2(q-1)} \log(d/m) + \log(B_1 R_q N))$, and so the total number of bits communicated globally is at most

$$O\left(N^{\frac{3}{2}} \log(d/N) + m^{2q-1} \log(d/m) + m \log(B_1 R_q N)\right).$$

In the linear model setting (2) with 1-Lipschitz loss $\phi$ it suffices to set $s_0 = \Omega(N)$, so the total bits of communication is $O(N \log(d/N) + m^{2q-1} \log(d/m) + m \log(B_1 R_q N))$.

We see that the communication required by sparsified mirror descent is exponential in the norm parameter $q$. This means that whenever $q$ is constant, the overall communication is polylogarithmic in dimension. When $q = \log d$, $\|x\|_q \approx \|x\|_\infty$ up to a multiplicative constant. In this case the communication from Theorem 3 becomes polynomial in dimension, which we know from Section 2.4 is necessary.

The guarantee of Algorithm 2 extends beyond the statistical learning model to the first-order stochastic convex optimization model, as well as the online convex optimization model.

**Proof sketch.** They basic premise behind the algorithm and its analysis is that by using the same learning rate across all machines, we can pretend as though we are running a single instance of mirror descent on a centralized machine. The key difference from the usual analysis is that we need to bound the error incurred by sparsification between successive machines. Here, the choice of the regularizer is crucial. A fundamental property used in the analysis of mirror descent is *strong convexity* of the regularizer. To give convergence rates that do not depend on dimension (such as (3)) it is essential that the regularizer be $\Omega(1)$-strongly convex. Our regularizer $\mathcal{R}$ indeed has this property.

**Proposition 3** (Ball et al. (1994)). For $p \in (1, 2]$, $\mathcal{R}$ is $(p-1)$-strongly convex with respect to $\|\cdot\|_p$. Equivalently, $D_\mathcal{R}(w, w') \geq \frac{p-1}{2} \|w - w'\|_p^2 \forall w, w' \in \mathbb{R}^d$.

On the other hand, to argue that sparsification has negligible impact on convergence, our analysis leverages *smoothness...*
of the regularizer. Strong convexity and smoothness are at odds with each other: It is well known that in infinite dimension, any norm that is both strongly convex and smooth is isomorphic to a Hilbert space (Pisier, 2011). What makes our analysis work is that while the regularizer \( R \) is not smooth, it is Hölder-smooth for any finite \( q \). This is sufficient to bound the approximation error from sparsification.

To argue that the excess risk achieved by mirror descent with the \( \ell_p \) regularizer \( R \) is optimal, however, it is essential that the gradients are \( \ell_q \)-bounded rather than \( \ell_\infty \)-bounded.

In more detail, the proof has three components:

- **Telescoping.** Mirror descent gives a regret bound that telescopes across all \( m \) machines up to the error introduced by sparsification. To match the optimal centralized regret, we only need to bound \( m \) error terms of the form \( D_R(w^* \| Q^*(w_{n+1}^i)) - D_R(w^* \| w_{n+1}^i) \).

- **Hölder-smoothness.** We prove (Theorem 7) that the difference above is of order

\[
B_1 ||Q^*(w_{n+1}^i) - w_{n+1}||_p + B_2^{1/p} ||Q^*(w_{n+1}^i) - w_{n+1}||_p^{1-q}.
\]

- **Maurey for \( \ell_p \) norms.** We prove (Theorem 6) that

\[
||Q^*(w_{n+1}^i) - w_{n+1}||_p \leq \left( \frac{1}{\delta} \right)^{1/p} \text{ and likewise that}

||Q^*(w_{n+1}^i) - w_{n+1}||_\infty \leq \left( \frac{1}{\delta} \right)^{1/2}.
\]

With a bit more work these inequalities yield Theorem 3. We close this section with a few more notes about Algorithm 2 and its performance.

**Remark 1.** For the special case of \( \ell_1/\ell_q \)-bounded linear models with Lipschitz link function, it is straightforward to show that the following strategy also leads to sublinear communication: Truncate each feature vector to the top \( \Theta(N^{q/2}) \) coordinates, then send all the truncated examples to a central server, which returns the empirical risk minimizer. This strategy matches the risk of Theorem 3 with total communication \( \tilde{O}(N^{q/2+1}) \), but has two deficiencies. First, the total communication is larger than the \( \tilde{O}(N + m^{2r-1}) \) bound achieved by Theorem 3, unless \( m \) is very large. Second, it does not extend to the general optimization setting.

### 3.2. Smooth Losses

We can improve the statistical guarantee and total communication further for the case where \( L_D \) is smooth with respect to \( \ell_q \), rather than just Lipschitz. We assume that \( \ell \) has \( \beta_q \)-Lipschitz gradients, in the sense that for all \( w, w' \in \mathcal{W} \), for all \( z \), \( \| \nabla \ell(w, z) - \nabla \ell(w', z) \|_q \leq \beta_q \| w - w' \|_p \), where \( p \) is such that \( \frac{1}{p} + \frac{1}{q} = 1 \).

**Theorem 4.** Suppose in addition to the assumptions of Theorem 3 that \( \ell(\cdot, z) \) is non-negative and has \( \beta_q \)-Lipschitz gradients with respect to \( \ell_q \). Let \( L^* = \inf_{w \in \mathcal{W}} L_D(w) \). If we run Algorithm 2 with learning rate \( \eta = \sqrt{\frac{B^2}{C_q \beta_q L^* R} \wedge \frac{1}{4C_q^{1/q}}} \) and \( w = 0 \) then, if \( s = \Omega(m^{2(q-1)}) \) and \( s_0 = \sqrt{\frac{\beta_q B^2 N}{C_q L^*} \wedge \frac{N}{C_q}} \), the algorithm guarantees

\[
\mathbb{E}[L_D(\tilde{w})] - L^* \leq O\left( \sqrt{\frac{C_q B^2 L^*}{N}} + \frac{C_q \beta_q B^2 N}{N} \right).
\]

The total number of bits sent by each machine—besides communicating the final iterate \( \tilde{w} \)—is at most \( O(m^{2(q-1)} \log(d/m)) \), and so the total number of bits communicated globally is at most \( O(m \log(\beta_q B_1 N)) + O\left( \sqrt{\frac{\beta_q B^2 N}{C_q L^*} \wedge \frac{N}{C_q}} \right) \log(d/N) + m^{2r-1} \log(d/m) \).

Compared to the previous theorem, this result provides a so-called “small-loss bound” (Srebro et al., 2010), with the main term scaling with the optimal loss \( L^* \). The dependence on \( N \) in the communication cost can be as low as \( O(\sqrt{N}) \) depending on the value of \( L^* \).

### 3.3. Fast Rates under Restricted Strong Convexity

So far all of the algorithmic results we have present scale as \( O(N^{1/2}) \). While this is optimal for generic Lipschitz losses, we mentioned in Section 2 that for strongly convex losses the rate can be improved in a nearly-dimension independent fashion to \( O(N^{-1}) \) for sparse high-dimensional linear models. As in the generic Lipschitz loss setting, we show that making the assumption of \( \ell_1/\ell_q \)-boundedness is sufficient to get statistically optimal distributed algorithms with sublinear communication, thus providing a way around the lower bounds for fast rates in Section 2.4. The key assumption for the results in this section is that the population risk satisfies a form of restricted strong convexity.

**Assumption 1.** There is a constant \( \gamma_q \) such that \( \forall w \in \mathcal{W}, L_D(w) - L_D(w^*) - \langle \nabla L_D(w^*), w - w^* \rangle \geq \frac{\gamma_q}{2} \| w - w^* \|_p^2 \).

In a moment we will show how to relate this property to the standard restricted eigenvalue property in high-dimensional statistics (Negahban et al., 2012) and apply it to sparse regression.

Our main algorithm for strongly convex losses is Algorithm 3, which is stated in Appendix C due to space constraints. The algorithm does not introduce any new tricks for distributed learning over Algorithm 2; rather, it invokes Algorithm 2 repeatedly in an inner loop, relying on these invocations to take care of communication. This reduction is based on techniques developed in (Juditsky & Nesterov, 2014), whereby restricted strong convexity is used to establish that error decreases geometrically as a function of the number of invocations to the sub-algorithm. We refer the reader to Appendix C for additional details.

**Theorem 5.** Suppose Assumption 1 holds, that subgradients belong to \( \mathcal{X}_q \) for \( q \geq 2 \), and that \( \mathcal{W} \subset \mathcal{W}_1 \). When the
parameter $c > 0$ is a sufficiently large absolute constant, Algorithm 3 guarantees that
\[
\mathbb{E}[L_D(\tilde{w}_T)] - L_D(w^*) \leq O\left(\frac{C_q R_2^2}{\gamma \sqrt{N}}\right).
\]
The total numbers of bits communicated is
\[
O\left(N^{2(q-1)}m^{2q-1} \left(\frac{\gamma_q R_2^2}{C_q \eta_q^2}\right)^2(q-1) + N^q \left(\frac{\gamma_q B_1}{C_q \eta_q^2}\right)^q \log d\right),
\]
plus $O(m \log(B_1 R_q N))$. Treating scale parameters as constants, the total communication simplifies to $O\left(N^{2q-2}m^{2q-1} \log d\right)$.

**Application: Sparse Regression.** As an application of Algorithm 3, we consider the sparse regression setting (5), where $L_D(w) = \mathbb{E}_{x,y}[(w, x - y)^2]$. We assume $\|x\|_q \leq R_q$ and $|y| \leq 1$. We let $w^* = \arg \min_{w \in \mathcal{W}} L_D(w)$, so $\|w^*\|_1 \leq B_1$. We assume $w^*$ is $k$-sparse with support set $S \subset [d]$.

We invoke Algorithm 3 constraint set $\mathcal{W} := \{w \in \mathbb{R}^d : \|w\|_1 \leq \|w^*\|_1\}$ and let $\Sigma = \mathbb{E}[(x^* x)]$. Our bound depends on the restricted eigenvalue parameter: $\gamma := \inf_{w \in \mathcal{W} \setminus w^* \{0\}} \frac{\Sigma^{1/2} \|w\|^2_2}{\|w\|_2^2}$.

**Proposition 4.** Algorithm 3, with constraint set $\mathcal{W}$ and appropriate choice of parameters, guarantees
\[
\mathbb{E}[L_D(\tilde{w}_T)] - L_D(w^*) \leq O\left(C_q B_1^2 R_2^2 \cdot \frac{k}{\gamma N}\right).
\]
Suppressing problem-dependent constants, total communication is of order $O\left(N^{2q-2}m^{2q-1} \log d\right) / k^{q-4}$.

**3.4. Extension: Matrix Learning and Beyond**

The basic idea behind sparsified mirror descent—that by assuming $\ell_q$-boundedness one can get away with using a Hölder-smooth regularizer that behaves well under sparsification—is not limited to the $\ell_1/\ell_q$ setting. To extend the algorithm to more general geometry, all that is required is the following:

- The constraint set $\mathcal{W}$ can be written as the convex hull of a set of atoms $\mathcal{A}$ that has sublinear bit complexity.
- The data should be bounded in some norm $\|\cdot\|_\tau$ such that the dual $\|\cdot\|_\tau^*$ admits a regularizer $\mathcal{R}$ that is strongly convex and Hölder-smooth with respect to $\|\cdot\|_\tau$.
- $\|\cdot\|_\tau$ is preserved under sparsification. We remark in passing that this property and the previous one are closely related to the notions of type and cotype in Banach spaces (Pisier, 2011).

Here we deliver on this potential and sketch how to extend the results so far to matrix learning problems where $\mathcal{W} \subseteq \mathbb{R}^{d \times d}$ is a convex set of matrices. As in Section 3.1 we work with a generic Lipschitz loss $L_D(W) = \mathbb{E}_z ((Wz)^2)$. Letting $\|W\|_{S_p} = \text{tr}((WW^*)^{\frac{p}{2}})$ denote the Schatten $p$-norm, we make the following spectral analogue of the $\ell_1/\ell_q$-boundedness assumption: $W \in \mathcal{W}_{S_1} := \{W \in \mathbb{R}^{d \times d} : \|W\|_{S_1} \leq B_1\}$ and subgradients $\partial \ell(z)$ belong to $\mathcal{X}_{S_q} := \{X \in \mathbb{R}^{d \times d} : \|X\|_{S_q} \leq R_q\}$, where $q \geq 2$.

Recall that $S_1$ and $S_\infty$ are the nuclear norm and spectral norm, respectively. The $S_1/S_\infty$ setup has many applications in learning (Hazan et al., 2012).

We make the following key changes to Algorithm 2:

- Use the Schatten regularizer $\mathcal{R}(W) = \frac{1}{2} \|W\|_{S_p}^2$.
- Use the following spectral version of the Maurey operator $Q^s(W)$: Let $W$ have singular value decomposition $W = \sum_{i=1}^s \sigma_i u_i v_i^\top$ with $\sigma_i \geq 0$ and define $P \in \Delta_d$ via $P \propto \sigma_i$. Sample $i_1, \ldots, i_s$ i.i.d. from $P$ and return $Q^s(W) = \|W\|_{S_1} \sigma_{i_1} \ldots \sigma_{i_s} u_{i_1} v_{i_s}^\top$.
- Encode and transmit $Q^s(W)$ as the sequence $(u_{i_1}, v_{i_1}), \ldots, (u_{i_s}, v_{i_s})$, plus the scalar $\|W\|_{S_1}$. This takes $O(sd)$ bits.

**Proposition 5.** Let $q \geq 2$ be fixed, and suppose that subgradients belong to $\mathcal{X}_{S_q}$, and that $W \in \mathcal{W}_{S_1}$. If we run the variant of Algorithm 2 described above with learning rate $\eta = \frac{B_1}{R_q} \sqrt{\frac{1}{C_q N}}$ and initial point $\tilde{W} = 0$, then whenever $s = O((N^2(q-1))$ and $s_0 = O((N^2)^\frac{1}{2})$, the algorithm guarantees
\[
\mathbb{E}[L_D(\tilde{W})] - \inf_{W \in \mathcal{W}} L_D(W) \leq O\left(\sqrt{\frac{B_1 R_2^2 C_q}{N}}\right),
\]
where $C_q = q - 1$. The total number of bits communicated globally is at most $O(m^{2q-2}d + N^2 d)$.

In the matrix setting, the number of bits required to naively send weights $W \in \mathbb{R}^{d \times d}$ or subgradients $\partial \ell(W, z) \in \mathbb{R}^{d \times d}$ is $O(d^2)$. The communication required by our algorithm scales only as $O(d)$, so it is indeed sublinear. The proof of Proposition 5 is sketched in Appendix C.

**4. Discussion**

We hope our work will lead to further development of algorithms with sublinear communication. A few immediate questions:

- Can we get matching upper and lower bounds for communication in terms of $m, N, \log d$, and $q$?
- Currently all of our algorithms work serially. Can we extend the techniques to give parallel speedup?
- Returning to the general setting (1), what abstract properties of the hypothesis class $\mathcal{H}$ are required to guarantee that learning with sublinear-communication is possible?

\footnote{We may assume $\sigma_i \geq 0$ without loss of generality.}
Acknowledgements

Part of this work was completed while DF was a student at Cornell University and supported by the Facebook PhD fellowship.

References


Distributed Learning with Sublinear Communication


