Better generalization with less data using robust gradient descent

Matthew J. Holland [1] **Kazushi Ikeda** [2]

Abstract

For learning tasks where the data (or losses) may be heavy-tailed, algorithms based on empirical risk minimization may require a substantial number of observations in order to perform well off-sample. In pursuit of stronger performance under weaker assumptions, we propose a technique which uses a cheap and robust iterative estimate of the risk gradient, which can be easily fed into any steepest descent procedure. Finite-sample risk bounds are provided under weak moment assumptions on the loss gradient. The algorithm is simple to implement, and empirical tests using simulations and real-world data illustrate that more efficient and reliable learning is possible without prior knowledge of the loss tails.

1. Introduction

Machine learning requires reliable statistical inference which can be implemented to be computationally efficient. To capture this more formally, it is typical to use a risk $R(\boldsymbol{w}) := \mathbf{E}\, l(\boldsymbol{w}; \boldsymbol{z})$, induced by a loss l, where $\boldsymbol{w}$ is the parameter to be specified, and expectation is with respect to the data $\boldsymbol{z}$. Given a sample $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n$, if a learning algorithm outputs $\widehat{\boldsymbol{w}}$ such that $R(\widehat{\boldsymbol{w}})$ is sufficiently small with large confidence over the sample distribution, one takes this as theoretical evidence for good generalization, albeit up to conditions placed on the underlying data distribution. The inferential side of the task is important since the risk can never be observed, and the computational side is important for minimizing the gap between the $\widehat{\boldsymbol{w}}$ we study formally and the $\widehat{\boldsymbol{w}}$ we actually obtain in practice.

The empirical mean of the risk, given by $n^{-1} \sum_{i=1}^{n} l(\cdot; \boldsymbol{z}_i)$, offers a natural objective function to be minimized based on the data, and indeed empirical risk minimization (ERM) is the *de facto* standard learning strategy for tackling most machine learning problems (Kearns & Schapire, 1994; Bartlett et al., 1996; Alon et al., 1997; Bartlett & Mendelson, 2003). While ERM is an established strategy, it has many limitations which have been elucidated in recent years. Since ERM admits any minimizer of the empirical risk, general analysis of ERM learners abstracts away the actual implementation of the algorithm. Recent work, however, has shown how sensitive ERM can be to the actual implementation (Daniely & Shalev-Shwartz, 2014; Feldman, 2016), where highly sub-optimal performance can be observed in classification tasks with more than two classes, let alone the case of unbounded, possibly heavy-tailed losses. A related problem is highlighted clearly by Lin & Rosasco (2016), where ERM implemented using gradient descent (GD) only enjoys strong guarantees when the data is sharply concentrated around the mean. Such results are of conceptual and practical importance given the ubiquity of gradient-based methods in modern machine learning. These results also imply that typical learning algorithms implementing ERM are liable to become highly inefficient if the data is "inconvenient" in the heavy-tailed sense.

As heavy-tailed data abounds in practice (Finkenstädt & Rootzén, 2003), it is naturally of interest to study new algorithms which, like GD-based ERM (henceforth, ERM-GD) are easy to implement, but which are more robust to the underlying data distribution. In this paper, we propose a new learning algorithm which utilizes robust and inexpensive estimates of the risk gradient for use in a first-order update procedure.

Review of related work Seminal work by Lin & Rosasco (2016) looks at the generalization of ERM-GD for sub-Gaussian observations. Since ERM-GD is a key benchmark to be compared against, it is of interest to find a technique that is competitive with ERM-GD when it is optimal, but which behaves better when sub-Gaussianity cannot be assumed.

Our main interest is with robustness to the underlying distribution, and important work in this direction has appeared in recent years. One approach is to use all the data to construct robust estimates $\widehat{R}(\boldsymbol{w})$ of the risk $R(\boldsymbol{w})$ for each $\boldsymbol{w}$ to be checked, and subsequently minimize $\widehat{R}$ as an alternative objective. A strategy using M-estimates of R was introduced

[1]Institute of Scientific and Industrial Research, Osaka University [2]Division of Information Science, Nara Institute of Science and Technology. Correspondence to: Matthew J. Holland <matthew-h@ar.sanken.osaka-u.ac.jp>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

by Brownlees et al. (2015), based on results due to Catoni (2009; 2012). Statistical guarantees are near-optimal under very weak assumptions on the data, but unfortunately $\widehat{R}$ is defined implicitly, which makes computation a challenge. Even if R is convex, the estimate $\widehat{R}$ need not be, and the non-linear optimization required by this method does not scale well to models with many parameters.

Another notable line of work looks at generalizations of the "median of means" procedure, in which one constructs candidates on disjoint partitions of the data, and aggregates them such that anomalous candidates are effectively ignored (Hsu & Sabato, 2016; Minsker & Strawn, 2017). Conceptually the closest recent work to this paper is that of "robust gradient descent" algorithms implemented using a median of means sub-routine. Chen et al. (2017); Prasad et al. (2018) apply this routine directly to the loss gradients, while Lecué & Lerasle (2017); Lecué et al. (2018) take the mean loss gradient of the subset corresponding to the median of means risk estimate. Both approaches have strong statistical guarantees and can be easily implemented, but their performance is marginal: when sample size n is small relative to the complexity of the model, very few subsets can be created, and robustness is poor; conversely, when n is large enough to make many candidates, cheaper and less sophisticated methods often suffice. While demonstrably robust, the difficulty of proper partitioning in practice is demonstrated clearly by the empirical tests of Lecué et al. (2018). Methodologically, these algorithms differ entirely from ours in terms of the robust gradient estimator used, since we use M-estimates of the risk gradient computed using a fixed-point scheme.

Our contributions To overcome the limitations of ERM-GD, and existing robust alternatives highlighted above, our key idea is to make use of robust estimates of the risk gradient, instead of the risk itself. This estimates can then be fed directly into gradient-based iterative updates. While computational overhead arises due to the need to robustify in high dimensions, the optimization side of the problem is much more straightforward, and we obtain strong formal guarantees that hold over a wide class of data distributions. Our main contributions:

- A new learning algorithm which is easy to implement, mimics ERM-GD under sub-Gaussian data, while being more robust to heavy-tailed data.
- Finite sample bounds on the excess risk incurred by our proposed procedure which hold under weak moment assumptions on the data distribution.
- Empirical tests using both simulations and real-world benchmark data reinforce the practical utility of our procedure and the robustness implied by the theory.

2. Robust gradient descent

Were the risk to be known, we could update using

$$\boldsymbol{w}^*_{(t+1)} := \boldsymbol{w}^*_{(t)} - \alpha_{(t)} \boldsymbol{g}(\boldsymbol{w}^*_{(t)}) \tag{1}$$

where $\boldsymbol{g}(\boldsymbol{w}) := R'(\boldsymbol{w})$, an ideal procedure. Any learning algorithm in practice will not have access to R or $\boldsymbol{g}$, and thus must approximate this update with

$$\widehat{\boldsymbol{w}}_{(t+1)} := \widehat{\boldsymbol{w}}_{(t)} - \alpha_{(t)} \widehat{\boldsymbol{g}}(\widehat{\boldsymbol{w}}_{(t)}), \tag{2}$$

where $\widehat{\boldsymbol{g}}$ represents some sample-based estimate of $\boldsymbol{g}$. Setting $\widehat{\boldsymbol{g}}$ to the sample mean reduces to ERM-GD, and conditioned on $\widehat{\boldsymbol{w}}_{(t)}$, $\mathbf{E}\, \widehat{\boldsymbol{g}}(\widehat{\boldsymbol{w}}_{(t+1)}) = \boldsymbol{g}(\widehat{\boldsymbol{w}}_{(t+1)})$, a property used throughout the literature (Rakhlin et al., 2012; Le Roux et al., 2012; Johnson & Zhang, 2013; Shalev-Shwartz & Zhang, 2013; Frostig et al., 2015; Murata & Suzuki, 2016). While convenient from a technical standpoint, there is no conceptual necessity for $\widehat{\boldsymbol{g}}$ to be unbiased. More realistically, as long as $\widehat{\boldsymbol{g}}$ is sharply distributed around $\boldsymbol{g}$, then an approximate first-order procedure should not deviate too far from the ideal, even if these estimators are biased. An outline of such a routine is given in Algorithm 1.

Algorithm 1 Robust gradient descent outline

inputs: $\widehat{\boldsymbol{w}}_0, T > 0$
for $t = 0, 1, \ldots, T-1$ **do**
  $D_{(t)} \leftarrow \{l'(\widehat{\boldsymbol{w}}_{(t)}; \boldsymbol{z}_i)\}_{i=1}^n$ {*Update loss gradients.*}
  $\widehat{\boldsymbol{\sigma}}_{(t)} \leftarrow \text{RESCALE}(D_{(t)})$ {*Eqn. (4).*}
  $\widehat{\boldsymbol{\theta}}_{(t)} \leftarrow \text{LOCATE}(D_{(t)}, \widehat{\boldsymbol{\sigma}}_{(t)})$ {*Eqns. (3), (5).*}
  $\widehat{\boldsymbol{w}}_{(t+1)} \leftarrow \widehat{\boldsymbol{w}}_{(t)} - \alpha_{(t)} \widehat{\boldsymbol{\theta}}_{(t)}$ {*Plug in to update.*}
end for
return: $\widehat{\boldsymbol{w}}_{(T)}$

Let us flesh out the key sub-routines used in a single iteration, for the $\boldsymbol{w} \in \mathbb{R}^d$ case. When the data is prone to outliers, a "soft" truncation of errant values is a prudent alternative to discarding valuable data. This can be done systematically using a convenient class of M-estimators of location and scale (van der Vaart, 1998; Huber & Ronchetti, 2009). The LOCATE sub-routine entails taking a convex, even function ρ, and for each coordinate, computing $\widehat{\boldsymbol{\theta}} = (\widehat{\theta}_1, \ldots, \widehat{\theta}_d)$ as

$$\widehat{\theta}_j \in \operatorname*{arg\,min}_{\theta \in \mathbb{R}} \sum_{i=1}^n \rho\left(\frac{l'_j(\boldsymbol{w}; \boldsymbol{z}_i) - \theta}{s_j}\right), \quad j = 1, \ldots, d. \tag{3}$$

Note that if $\rho(u) = u^2$, then $\widehat{\theta}_j$ reduces to the sample mean of $\{l'_j(\boldsymbol{w}; \boldsymbol{z}_i)\}_{i=1}^n$, thus to reduce the impact of extreme observations, it is useful to take $\rho(u) = o(u^2)$ as $u \to \pm\infty$. Here the $s_j > 0$ factors are used to ensure that consistent estimates take place irrespective of the order of magnitude of the observations. We set the scaling factors in two steps.

First is RESCALE, in which a rough dispersion estimate of the data is computed for each j using

$$\widehat{\sigma}_j \in \left\{ \sigma > 0 : \sum_{i=1}^{n} \chi\left(\frac{l_j'(\boldsymbol{w}; \boldsymbol{z}_i) - \gamma_j}{\sigma} \right) = 0 \right\}. \quad (4)$$

Here $\chi : \mathbb{R} \to \mathbb{R}$ is an even function, satisfying $\chi(0) < 0$, and $\chi(u) > 0$ as $u \to \pm\infty$ to ensure that the resulting $\widehat{\sigma}_j$ is an adequate measure of the dispersion of $l_j'(\boldsymbol{w}; \boldsymbol{z})$ about a pivot point, say $\gamma_j = \sum_{i=1}^{n} l_j'(\boldsymbol{w}; \boldsymbol{z}_i)/n$. Second, we adjust this estimate based on the available sample size and desired confidence level, as

$$s_j = \widehat{\sigma}_j \sqrt{n / \log(2\delta^{-1})} \quad (5)$$

where $\delta \in (0, 1)$ specifies the desired confidence level $(1 - \delta)$, and n is the sample size. This last step appears rather artificial, but can be derived from a straightforward theoretical argument, given in section 3.1. This concludes all the steps[1] in one full iteration of Algorithm 1 on $\mathbb{R}^d$.

In the remainder of this paper, we shall investigate the learning properties of this procedure, through analysis of both a theoretical (section 3) and empirical (section 4) nature. As an example, in the strongly convex risk case, our formal argument yields excess risk bounds of the form

$$O\left(\frac{d\left(\log(d\delta^{-1}) + d\log(n)\right)}{n} \right) + O\left((1 - \alpha\beta)^T\right)$$

with probability no less than $1 - \delta$ over the sample, for small enough $\alpha_{(t)} = \alpha$ over T iterations. Here $\beta > 0$ is a constant that depends only on R, and analogous results hold without strong convexity. Of the underlying distribution, all that is assumed is a bound on the variance of $l'(\cdot; \boldsymbol{z})$, suggesting formally that the procedure should be competitive over a diverse range of data distributions.

3. Theoretical analysis

Here we analyze the performance of Algorithm 1 on hypothesis class $\mathcal{W} \subseteq \mathbb{R}^d$, as measured by the risk achieved, which we estimate using upper bounds that depend on key parameters of the learning task. A general sketch is given, followed by some key conditions, representative results, and discussion. All proofs are relegated to supplementary materials.

[1]For concreteness, unless specified otherwise, in all empirical tests to follow we use the Gudermannian function (Abramowitz & Stegun, 1964), $\rho(u) = \int_0^u \psi(x)\,dx$ where $\psi(u) = 2\,\mathrm{atan}(\exp(u)) - \pi/2$, and $\chi(u) = u^2/(1 + u^2) - c$, for a constant $c > 0$. General conditions on ρ, as well as standard methods for computing the M-estimates, namely the $\widehat{\theta}_j$ and $\widehat{\sigma}_j$, are given in the supplementary materials.

Notation For integer k, write $[k] := \{1, \ldots, k\}$ for all the positive integers from 1 to k. Let μ denote the data distribution, with $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n$ independent observations from μ, and $\boldsymbol{z} \sim \mu$ an independent copy. Risk is then $R(\boldsymbol{w}) := \mathbf{E}_\mu\, l(\boldsymbol{w}; \boldsymbol{z})$, its gradient $\boldsymbol{g}(\boldsymbol{w}) := R'(\boldsymbol{w})$, and $R^* := \inf_{\boldsymbol{w} \in \mathcal{W}} R(\boldsymbol{w})$. $\mathbf{P}$ denotes a generic probability measure, typically the product measure induced by the sample. We write $\|\cdot\|$ for the usual (ℓ_2) norm on $\mathbb{R}^d$. For function F on $\mathbb{R}^d$ with partial derivatives defined, write the gradient as $F'(\boldsymbol{u}) := (F_1'(\boldsymbol{u}), \ldots, F_d'(\boldsymbol{u}))$ where for short, we write $F_j'(\boldsymbol{u}) := \partial F(\boldsymbol{u})/\partial u_j$.

3.1. Argument sketch

The analysis here requires only two steps: (i) A good estimate $\widehat{\boldsymbol{g}} \approx \boldsymbol{g}$ implies that approximate update (2) is near the optimal update. (ii) Under variance bounds, coordinate-wise M-estimation yields a good gradient estimate. We are then able to conclude that with enough samples and iterations (but not too many iterations), the output of Algorithm 1 can achieve an arbitrarily small excess risk. Here we spell out the key facts underlying this approach.

For the first step, let $\boldsymbol{w}^* \in \mathbb{R}^d$ be a minimizer of R. When the risk R is strongly convex, then using well-established convex optimization theory (Nesterov, 2004), we can easily control $\|\boldsymbol{w}^*_{(t+1)} - \boldsymbol{w}^*\|$ as a function of $\|\boldsymbol{w}^*_{(t)} - \boldsymbol{w}^*\|$ for any step $t \geq 0$. Thus to control $\|\widehat{\boldsymbol{w}}_{(t+1)} - \boldsymbol{w}^*\|$, in comparing the approximate case and optimal case, all that matters is the difference between $\boldsymbol{g}(\widehat{\boldsymbol{w}}_{(t)})$ and $\widehat{\boldsymbol{g}}(\widehat{\boldsymbol{w}}_{(t)})$ (Lemma 4). For the general case of convex R, since one cannot easily control the distance of the optimal update from any potential minimum, one must directly compare the trajectories of $\widehat{\boldsymbol{w}}_{(t)}$ and $\boldsymbol{w}^*_{(t)}$ over $t = 0, 1, \ldots, T$, which once again amounts to a comparison of $\boldsymbol{g}$ and $\widehat{\boldsymbol{g}}$. This inevitably leads to more error propagation and thus a stronger dependence on T, but the essence of the argument is the same.

For the second step, since $\widehat{\boldsymbol{g}}$ is based on a random sample $\{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n\}$, we need an estimation technique which admits guarantees for any choice of $\boldsymbol{w}$, with high probability over the random draw of the sample. A basic requirement is that

$$\mathbf{P}\left\{ \max_{t \leq T} \|\widehat{\boldsymbol{g}}(\widehat{\boldsymbol{w}}_{(t)}) - \boldsymbol{g}(\widehat{\boldsymbol{w}}_{(t)})\| \leq \varepsilon \right\} \geq 1 - \delta. \quad (6)$$

Of course this must be proved (see Lemmas 3 and 8), but if valid, then running Algorithm 1 for T steps, we can invoke (6) to get a high-probability event on which $\widehat{\boldsymbol{w}}_{(T)}$ closely approximates the optimal GD output, up to the accuracy specified by ε. Naturally this ε will depend on confidence level δ, which implies that to get $1 - \delta$ confidence intervals, the upper bound in (6) will increase as δ gets smaller.

In the LOCATE sub-routine of Algorithm 1, we construct a more robust estimate of the risk gradient than can be

provided by the empirical mean, using an ancillary estimate of the gradient variance. This is conducted using a smooth truncation scheme, as follows. One important property of ρ in (3) is that for any $u \in \mathbb{R}$, one has

$$-\log(1-u+Cu^2) \le \rho'(u) \le \log(1+u+Cu^2) \quad (7)$$

for a fixed $C > 0$, a simple generalization of the key property utilized by Catoni (2012). For the Gudermannian function (section 2 footnote), we can take $C \le 2$, with the added benefit that ρ' is bounded and increasing. As to the quality of these estimates, note that they are distributed sharply around the risk gradient, as follows.

Lemma 1 (Concentration of M-estimates)**.** *For each coordinate $j \in [d]$, the estimates $\widehat{\theta}_j$ of (3) satisfy*

$$\frac{1}{2}|\widehat{\theta}_j - g_j(\boldsymbol{w})| \le \frac{C \operatorname{var}_\mu l_j'(\boldsymbol{w};\boldsymbol{z})}{s_j} + \frac{s_j \log(2\delta^{-1})}{n} \quad (8)$$

with probability no less than $1-\delta$, given large enough n and s_j.

To get the tightest possible confidence interval as a function of $s_j > 0$, we must set

$$s_j^2 = \frac{Cn \operatorname{var}_\mu l_j'(\boldsymbol{w};\boldsymbol{z})}{\log(2\delta^{-1})},$$

from which we derive (5), with $\widehat{\sigma}_j^2$ corresponding to a computable estimate of $\operatorname{var}_\mu l_j'(\boldsymbol{w};\boldsymbol{z})$. If the variance over all choices of $\boldsymbol{w}$ is bounded by some $V < \infty$, then up to the variance estimates, we have $\|\widehat{\boldsymbol{g}}(\boldsymbol{w}) - \boldsymbol{g}(\boldsymbol{w})\| \le O(\sqrt{dV\log(2d\delta^{-1})/n})$, with $\widehat{\boldsymbol{g}} = \widehat{\boldsymbol{\theta}}$ from Algorithm 1, yielding a bound for (6) free of $\boldsymbol{w}$.

Remark 2 (Comparison with ERM-GD). As a reference example, assume we were to run ERM-GD, namely using an empirical mean estimate of the gradient. Using Chebyshev's inequality, with probability $1-\delta$ all we can guarantee is $\varepsilon \le O(\sqrt{d/(n\delta)})$. On the other hand, using the location estimate of Algorithm 1 provides guarantees with $\log(1/\delta)$ dependence on the confidence level, realizing an exponential improvement over the $1/\delta$ dependence of ERM-GD, and an appealing formal motivation for using M-estimates of location as a novel strategy.

3.2. Conditions and results

On the learning task, we make the following assumptions.

A1. Minimize risk $R(\cdot)$ over a closed, convex $\mathcal{W} \subset \mathbb{R}^d$ with diameter $\Delta < \infty$.

A2. $R(\cdot)$ and $l(\cdot;\boldsymbol{z})$ (for all $\boldsymbol{z}$) are λ-smooth, convex, and continuously differentiable on $\mathcal{W}$.

A3. There exists $\boldsymbol{w}^* \in \mathcal{W}$ at which $\boldsymbol{g}(\boldsymbol{w}^*) = 0$.

A4. Distribution μ satisfies $\operatorname{var}_\mu l_j'(\boldsymbol{w};\boldsymbol{z}) \le V < \infty$, for all $\boldsymbol{w} \in \mathcal{W}$, $j \in [d]$.

Algorithm 1 is run following (3), (4), and (5) as specified in section 2. For RESCALE, the choice of χ is only important insofar as the scale estimates (the $\widehat{\sigma}_j$) should be moderately accurate. To make the dependence on this accuracy precise, take constants $c_{min}, c_{max} > 0$ such that

$$c_{min}^2 \le \frac{\widehat{\sigma}_j}{\operatorname{var}_\mu l_j'(\boldsymbol{w};\boldsymbol{z})} \le c_{max}^2, \quad j \in [d] \quad (9)$$

for all choices of $\boldsymbol{w} \in \mathcal{W}$, and write $c_0 := (c_{max} + C/c_{min})$. For $1-\delta$ confidence, we need a large enough sample; more precisely, for each $\boldsymbol{w}$, it is sufficient if for each j,

$$\frac{1}{4} \ge \frac{C\log(2\delta^{-1})}{n}\left(1 + \frac{C \operatorname{var}_\mu l_j'(\boldsymbol{w};\boldsymbol{z})}{\widehat{\sigma}_j^2}\right). \quad (10)$$

For simplicity, fix a small enough step size,

$$\alpha_{(t)} = \alpha \in (0, 2/\lambda), \quad t \in \{0,\dots,T-1\}. \quad (11)$$

Dependence on initialization is captured by two related factors $R_0 := R(\boldsymbol{w}_{(0)}^*) - R^*$, and $D_0 := \|\boldsymbol{w}_{(0)}^* - \boldsymbol{w}^*\|$. Under this setup, we can control the estimation error.

Lemma 3 (Accuracy of gradient estimates)**.** *For each step $t = 0,\dots,T-1$ of Algorithm 1, we have*

$$\|\widehat{\boldsymbol{\theta}}_{(t)} - \boldsymbol{g}(\widehat{\boldsymbol{w}}_{(t)})\| \le \frac{\widetilde{\varepsilon}}{\sqrt{n}}$$
$$:= \frac{\lambda(\sqrt{d}+1)}{\sqrt{n}} + 2c_0\sqrt{\frac{dV(\log(2d\delta^{-1}) + d\log(3\Delta\sqrt{n}/2))}{n}}$$

with probability no less than $1-\delta$.

Under strongly convex risk In addition to assumptions (A1)–(A4), assume that R is κ-strongly convex. In this case, $\boldsymbol{w}^*$ in (A3) is the unique minimum. First, we control the estimation error by showing that the approximate update (2) does not differ much from the optimal update (1).

Lemma 4 (Minimizer control)**.** *Consider the general approximate GD update (2), with $\alpha_{(t)} = \alpha$ such that $0 < \alpha < 2/(\kappa+\lambda)$. Assume that (6) holds with bound ε. Write $\beta := 2\kappa\lambda/(\kappa+\lambda)$. Then, with probability no less than $1-\delta$, we have*

$$\|\widehat{\boldsymbol{w}}_{(T)} - \boldsymbol{w}^*\| \le (1-\alpha\beta)^{T/2} D_0 + \frac{2\varepsilon}{\beta}.$$

Since Algorithm 1 indeed satisfies (6), as proved in Lemma 3, we can use the control over the parameter deviation provided by Lemma 4 and the smoothness of R to prove a finite-sample excess risk bound.

Theorem 5 (Excess risk bounds)**.** *Write $\widehat{\boldsymbol{w}}_{(T)}$ for the output of Algorithm 1 after T iterations, run such that (10)–(11) hold, with step size $\alpha_{(t)} = \alpha$ for all $0 < t < T$, as in Lemma 4. It follows that*

$$R(\widehat{\boldsymbol{w}}_{(T)}) - R^* \leq \lambda(1-\alpha\beta)^T D_0^2 + \frac{4\lambda\widetilde{\varepsilon}}{\beta^2 n}$$

with probability no less than $1-\delta$, where $\widetilde{\varepsilon}$ is as given in Lemma 3.

Remark 6 (Interpretation of bounds). There are two terms in the upper bound of Theorem 5, an optimization term decreasing in T, and an estimation term decreasing in n. The optimization error decreases at the usual gradient descent rate, and due to the uniformity of the bounds obtained, the statistical error is not hurt by taking T arbitrarily large, thus with enough samples we can guarantee arbitrarily small excess risk. Finally, the most important assumption on the distribution is weak: finite second-order moments. If we assume finite kurtosis, the argument of Catoni (2012) can be used to create analogous guarantees for an explicit scale estimation procedure, yielding guarantees whether the data is sub-Gaussian or heavy-tailed an appealing robustness to the data distribution.

Remark 7 (Doing projected descent). The above analysis proceeds on the premise that $\widehat{\boldsymbol{w}}_{(t)} \in \mathcal{W}$ holds after all the updates, $t \in [T]$. To enforce this, a standard variant of Algorithm 1 is to update as

$$\widehat{\boldsymbol{w}}_{(t+1)} \leftarrow \pi_{\mathcal{W}}\left(\widehat{\boldsymbol{w}}_{(t)} - \alpha_{(t)}\widehat{\boldsymbol{\theta}}_{(t)}\right), \quad t \in \{0,\dots,T-1\}$$

where $\pi_{\mathcal{W}}(\boldsymbol{u}) := \arg\min_{\boldsymbol{v}\in\mathcal{W}} \|\boldsymbol{u}-\boldsymbol{v}\|$. By (A1), this projection is well-defined (Luenberger, 1969, Sec. 3.12, Thm. 3.12). Using this fact, it follows that $\|\pi_{\mathcal{W}}(\boldsymbol{u}) - \pi_{\mathcal{W}}(\boldsymbol{v})\| \leq \|\boldsymbol{u}-\boldsymbol{v}\|$ for all $\boldsymbol{u},\boldsymbol{v} \in \mathcal{W}$, by which we can immediately show that Lemma 4 holds for the *projected robust gradient descent* version of Algorithm 1.

With prior information An interesting concept in machine learning is that of the relationship between learning efficiency, and the task-related prior information available to the learner. In the previous results, the learner is assumed to have virtually no information beyond the data available, and the ability to set a small enough step-size. What if, for example, just the gradient variance was known? A classic example from decision theory is the dominance of the estimator of James and Stein over the maximum likelihood estimator, in multivariate Normal mean estimation using prior variance information. In our more modern and non-parametric setting, the impact of rough, data-driven scale estimates was made explicit by the factor c_0. Here we give complementary results that show how partial prior information on the distribution μ can improve learning.

Lemma 8 (Accuracy with variance information)**.** *Conditioning on $\widehat{\boldsymbol{w}}_{(t)}$ and running one scale-location sequence of Algorithm 1, with $\widehat{\boldsymbol{\sigma}}_{(t)} = (\widehat{\sigma}_1,\dots,\widehat{\sigma}_d)$ modified to satisfy $\widehat{\sigma}_j^2 = C\operatorname{var}_\mu l_j'(\widehat{\boldsymbol{w}}_{(t)};\boldsymbol{z})$, $j \in [d]$. It follows that*

$$\|\widehat{\boldsymbol{\theta}}_{(t)} - \boldsymbol{g}(\widehat{\boldsymbol{w}}_{(t)})\| \leq 4\left(\frac{C\operatorname{trace}(\Sigma_{(t)})\log(2d\delta^{-1})}{n}\right)^{1/2}$$

with probability no less than $1-\delta$, where $\Sigma_{(t)}$ is the covariance matrix of $l'(\widehat{\boldsymbol{w}}_{(t)};\boldsymbol{z})$.

One would expect that with sharp gradient estimates, the variance of the updates should be small with a large enough sample. Here we show that the procedure stabilizes quickly as the estimates get closer to an optimum.

Theorem 9 (Control of update variance)**.** *Run Algorithm 1 as in Lemma 8, with arbitrary step-size $\alpha_{(t)}$. Then, for any $t < T$, taking expectation with respect to the sample $\{\boldsymbol{z}_i\}_{i=1}^n$, conditioned on $\widehat{\boldsymbol{w}}_{(t)}$, we have*

$$\mathbf{E}\,\|\widehat{\boldsymbol{w}}_{(t+1)} - \widehat{\boldsymbol{w}}_{(t)}\|^2 \leq 2\alpha_{(t)}^2\left(\frac{32Cd\operatorname{trace}(\Sigma_{(t)})}{n} + \|\boldsymbol{g}(\widehat{\boldsymbol{w}}_{(t)})\|^2\right).$$

In addition to these results, one can prove an improved version of Theorem 5 in a perfectly analogous fashion, using Lemma 8.

4. Empirical analysis

The chief goal of our numerical experiments is to elucidate the relationship between factors of the learning task (e.g., sample size, model dimension, underlying data distribution) and the behaviour of the robust gradient procedure proposed in Algorithm 1. We are interested in how these factors influence performance, both in an absolute sense and relative to the key competitors.

4.1. Controlled tests

Noisy convex minimization In this experiment, we construct a risk function taking a canonical quadratic form, setting $R(\boldsymbol{w}) = \langle\Sigma\boldsymbol{w},\boldsymbol{w}\rangle/2 + \langle\boldsymbol{w},\boldsymbol{u}\rangle + c$, for pre-fixed constants $\Sigma \in \mathbb{R}^{d\times d}$, $\boldsymbol{u} \in \mathbb{R}^d$, and $c \in \mathbb{R}$. The task is to minimize $R(\cdot)$ without knowledge of R itself, but rather only access to n random function observations $r_1,\dots,r_n$. These $r : \mathbb{R}^d \to \mathbb{R}$ are generated independently from a common distribution, satisfying the property $\mathbf{E}\,r(\boldsymbol{w}) = R(\boldsymbol{w})$ for all $\boldsymbol{w} \in \mathbb{R}^d$. In particular, here we generate observations $r_i(\boldsymbol{w}) = (\langle\boldsymbol{w}^* - \boldsymbol{w},\boldsymbol{x}_i\rangle + \epsilon_i)^2/2$, $i \in [n]$, with $\boldsymbol{x}$ and ϵ independent of each other. Here $\boldsymbol{w}^*$ denotes the minimum, and we have that $\Sigma = \mathbf{E}\,\boldsymbol{x}\boldsymbol{x}^T$. The inputs $\boldsymbol{x}$ shall follow an isotropic d-dimensional Gaussian distribution throughout all the following experiments, meaning Σ is positive definite, and R is strongly convex.

For these first tests, we run three procedures. First is ideal gradient descent, denoted `oracle`, which assumes the objective function R known. This corresponds to (1). Second, as a standard approximate procedure (2), we use ERM-GD, denoted `erm` and discussed at the start of section 2, which approximates the optimal procedure using the empirical risk. Against these two benchmarks, we compare our Algorithm 1, denoted `rgd`, as a robust alternative for (2).

Let us examine the results. We begin with a simple question: are there natural learning settings in which `rgd` outperforms ERM-GD? How does the same algorithm fare in situations where ERM is optimal? Under Gaussian noise, ERM-GD is effectively optimal (Lin & Rosasco, 2016, Appendix C). We thus consider the case of Gaussian noise (mean 0, standard deviation 20) as a baseline, and use centered log-Normal noise (log-location 0, log-scale 1.75) as an archetype of asymmetric heavy-tailed data. Risk results for the two routines are given alongside training error in Figure 1.

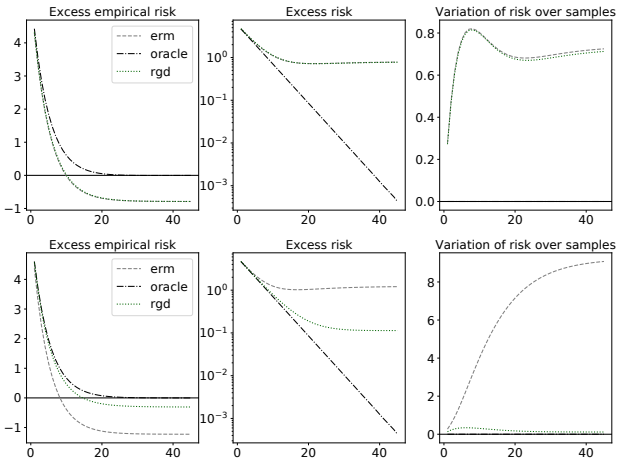

Figure 1. Performance metrics as a function of iterative updates. Top row: Normal noise. Bottom row: log-Normal noise. Settings: $n = 500, d = 2, \alpha_{(t)} = 0.1$ for all t.

In the situation favorable to `erm`, differences in performance are basically negligible. On the other hand, in the heavy-tailed setting, the performance of `rgd` is superior in terms of quality of the solution found and the variance of the estimates. Furthermore, we see that at least in the situation of small d and large n, taking T beyond numerical convergence has minimal negative effect on `rgd` performance; on the other hand `erm` is more sensitive. Comparing true risk with sample error, we see that while there is some unavoidable overfitting, in the heavy-tailed setting `rgd` departs from the ideal routine at a slower rate, a desirable trait.

Comparison with robust loss minimizer Another interesting question: instead of paying the overhead to robustify gradient estimates (d dimensions to handle), why not just make robust estimates of the risk itself, and use those estimates to fuel an iterative optimizer? Just such a procedure is analyzed by Brownlees et al. (2015) (denoted `bjl` henceforth). To compare our gradient-centric approach with their loss-centric approach, we implement `bjl` using the non-linear conjugate gradient method of Polak and Ribière (Nocedal & Wright, 1999), which is provided by `fmin_cg` in the `optimize` module of the SciPy scientific computation library (default maximum number of iterations is $200d$). This gives us a standard first-order general-purpose optimizer for minimizing the `bjl` objective. To see how well our procedure can compete with a pre-fixed max iteration number, we set $T = 25$ for all settings. Computation time is computed using the Python `time` module. To give a simple comparison between `bjl` and `rgd`, we run multiple independent trials of the same task, starting both routines at the same (random) initial value each time, generating a new sample, and repeating the whole process for different settings of $d = 2, 4, 8, 16, 32, 64$. Median times taken over all trials (for each d setting) are recorded, and presented in Figure 2 alongside performance results.

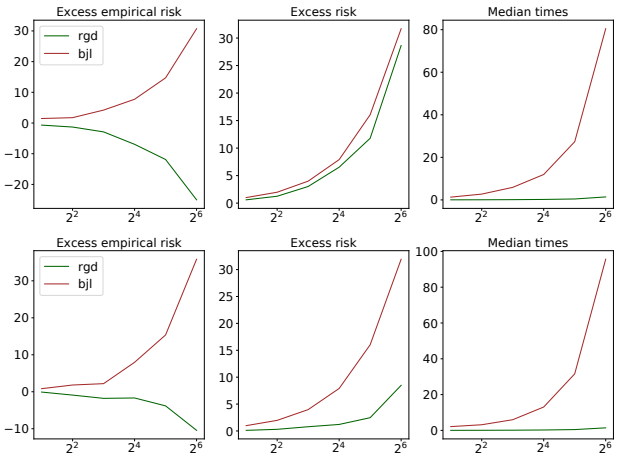

Figure 2. Comparison of our robust gradient-based approach with the robust objective-based approach. Top: Normal noise. Bottom: log-Normal noise. Performance is given as a function of the number of d, the number of parameters to optimize, given in $\log_2$ scale. Settings: $n = 500, \alpha_{(t)} = 0.1$ for all t.

From the results, we can see that while the performance of both methods is similar in low dimensions and under Gaussian noise, in higher dimensions and under heavy-tailed noise, our proposed `rgd` realizes much better performance in much less time. Regarding excess empirical risk, random deviations in the sample cause the minimum of the empirical risk function to deviate away from $\boldsymbol{w}^*$, causing the `rgd` solution to be closer to the ERM solution in higher dimensions. On the other hand, `bjl` is minimizing a different objective function. It should be noted that there are assuredly other ways of approaching the `bjl` optimization

task, but all of which require minimizing an implicitly defined objective which need not be convex. We believe that `rgd` provides a simple and easily implemented alternative, while still utilizing the same statistical principles.

Application to regression In this experiment, we apply our algorithm to a general regression task, under a wide variety of data distributions, and compare its performance against standard regression algorithms, both classical and modern. For each experimental condition, and for each trial, we generate n training observations of the form $y_i = \boldsymbol{x}_i^T \boldsymbol{w}^* + \epsilon_i, i \in [n]$. Distinct experimental conditions are delimited by the setting of (n, d) and μ. Inputs $\boldsymbol{x}$ are assumed to follow a d-dimensional isotropic Gaussian distribution, and thus our setting of μ will be determined by the distribution of noise ϵ. In particular, we look at several families of distributions, and within each family look at 15 distinct noise levels, namely parameter settings designed such that $\mathrm{sd}_\mu(\epsilon)$ monotonically increases over the range 0.3–20.0, approximately linearly over the levels.

To capture a range of signal/noise ratios, for each trial, $\boldsymbol{w}^* \in \mathbb{R}^d$ is randomly generated as follows. Defining the sequence $w_k := \pi/4 + (-1)^{k-1}(k-1)\pi/8, k = 1, 2, \ldots$ and uniformly sampling $i_1, \ldots, i_d \in [d_0]$ with $d_0 = 500$, we set $\boldsymbol{w}^* = (w_{i_1}, \ldots, w_{i_d})$. Computing $\mathrm{SN}_\mu = \|\boldsymbol{w}^*\|_2^2 / \mathrm{var}_\mu(\epsilon)$, we have $0.2 \leq SN_\mu \leq 1460.6$. Noise families: log-logistic (denoted `llog` in figures), log-Normal (`lnorm`), Normal (`norm`), and symmetric triangular (`tri_s`). Even with just these four, we capture both bounded and unbounded sub-Gaussian noise, and heavy-tailed data both with and without finite higher-order moments. Prediction error is the average error computed on an independent large testing set, and averaged over 250 trials.

Regarding the competing methods, classical choices are ordinary least squares (ℓ_2-ERM, denoted `OLS`) and least absolute deviations (ℓ_1-ERM, `LAD`). We also look at two recent methods of practical and theoretical importance described in section 1, namely the robust regression routines of Hsu & Sabato (2016) (`HS`) and Minsker (2015) (`Minsker`). For the former, we used the source published online by the authors. For the latter, on each subset the `OLS` solution is computed, and solutions are aggregated using the geometric median (in ℓ_2 norm), computed using the well-known algorithm of Vardi & Zhang (2000, Eqn. 2.6), and the number of partitions is set to $\max(2, \lfloor n/(2d) \rfloor)$. For comparison to this, writing `RGD` for Algorithm 1 in the regression case, we initialize `RGD` to the `OLS` solution, with confidence $\delta = 0.005$, and $\alpha_{(t)} = 0.1$ for all iterations. Maximum number of iterations is $T \leq 100$; the routine finishes after hitting this maximum or when the absolute value of the gradient falls below 0.001 for all conditions. Illustrative results are given in Figure 3.

Here we have fixed the dimension and noise level, and look at different values for n, the sample size. We see that regardless of distribution, `RGD` effectively matches the optimal convergence of `OLS` in the `norm` and `tri_s` cases, and is resilient to the remaining two scenarios where `OLS` breaks down. There are clear issues with the median of means based methods at very small sample sizes, though the stronger geometric median based method does eventually at least surpass `OLS` in the `llog` and `lnorm` cases.

4.2. Application to real-world benchmarks

To gain some additional perspective on algorithm performance, we shift our focus to real-world benchmark data sets, and classification tasks (binary and multi-class). The model assumed is standard multi-class logistic regression: if the number of classes is C, and the number of input features is F, then the total number of parameters to be determined is $d = (C-1)F$. The loss function is convex in the parameters, and its partial derivatives all exist, so the model aligns well with our problem setting of interest. All learning algorithms are given a fixed budget of gradient computations, set here to $20n$, where n is the size of the training set made available to the learner.

We use three well-known data sets for benchmarking: the CIFAR-10 data set of tiny images (ten classes), the MNIST data set of handwritten digits (ten classes), and the protein homology dataset (two classes) made popular by its inclusion in the KDD Cup. For all data sets, we carry out 10 independent trials, with training and testing tests randomly sampled as will be described shortly. For all datasets, we normalize the input features to the unit interval $[0, 1]$ in a per-dimension fashion. For CIFAR-10, we average the RGD color channels to obtain a single greyscale channel. As a result, $F = 1024$. Note that the protein dataset has highly unbalanced labels, with only 1296 positive labels out of over 145,000 observations. We take random samples such that the training and test sets are balanced, ending up with $n = 2000$.

Regarding the competing methods used, we test out a random mini-batch version of robust gradient descent given in Algorithm 1, with mini-batch sizes ranging over $\{5, 10, 15, 20\}$, roughly on the order of $n^{-1/4}$ for the largest datasets. We also consider a mini-batch in the sense of randomly selecting coordinates to robustify: select $\min(100, d)$ indices randomly at each iteration, and run the RGD subroutine on just these coordinates, using the sample mean for all the rest. Furthermore, we considered several minor alterations to speed up the original routine, including using $\log\cosh(\cdot)$ instead of the Gudermannian function for ρ, updating the scale much less frequently (compared to every iteration), and different choices of χ for rescaling. We compare our proposed algorithm with stochastic

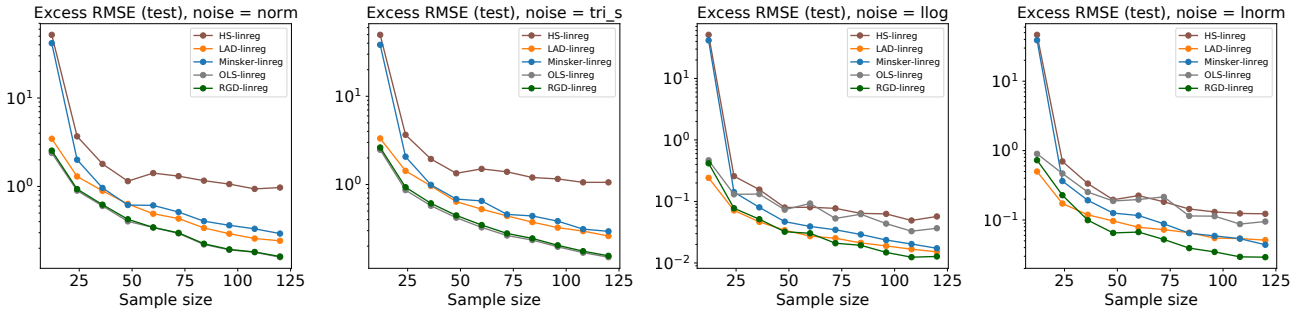

Figure 3. Prediction error over sample size $12 \leq n \leq 122$, fixed $d = 5$, noise level = 8.

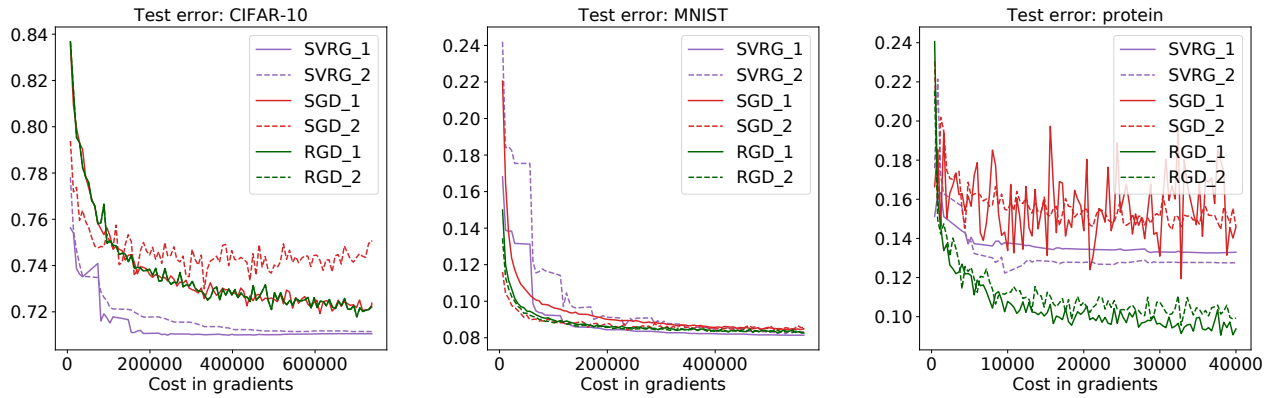

Figure 4. Test error (misclassification rate) over budget spent, as measured by gradient computations, for the top two performers within each method class. Each plot corresponds to a distinct dataset.

gradient descent (SGD), and stochastic variance-reduced gradient descent (SVRG) proposed by Johnson & Zhang (2013). For each method, pre-fixed step sizes ranging over $\{0.0001, 0.001, 0.01, 0.05, 0.10, 0.15, 0.20\}$ are tested. SGD uses mini-batches of size 1, as does the inner loop of SVRG. The outer loop of SVRG continues until the budget is spent, with the inner loop repeating $n/2$ times. Representative results are given in Figure 4.

For each of the three methods of interest, and each dataset, we chose the top two performance settings, displayed using the suffix `*_1` and `*_2` respectively. Here "top performance" is measured by the median value of the last five iterations. We see that in general, robust gradient descent is competitive with the best settings of these well-known routines, has minimal divergence between the performance of its first- and second-best settings, and in the case of smaller data sets (protein homology), indeed significantly outperforms the competitors. While these are simply nascent applications of robust gradient descent, the strong initial performance suggests that further investigation of efficient strategies under high-dimensional data is a promising direction.

5. Concluding remarks

In this paper we analyzed a new learning algorithm which tries to achieve distribution-robust learning guarantees in a computationally efficient fashion by introducing a simple sub-routine to existing gradient-based learning procedures, which utilizes robust estimates of the gradient of the underlying risk. The idea is to integrate reliable statistical estimation and practical implementation into a single cohesive process. The price paid is computational overhead due to doing robust estimation in high dimensions and biased estimates. Since excess risk bounds can be obtained under weak moment assumptions on the data, and initial empirical tests reinforce these theoretical insights, we believe this price is indeed worth paying, since all factors considered, the proposed procedure has been shown to generalize better, using less samples, over more distributions than its competitors.

Acknowledgements

This work was partially supported by the Grant-in-Aid for JSPS Research Fellows.

References

Abramowitz, M. and Stegun, I. A. *Handbook of Mathematical Functions With Formulas, Graphs, and Mathematical Tables*, volume 55 of *National Bureau of Standards Applied Mathematics Series*. US National Bureau of Standards, 1964.

Alon, N., Ben-David, S., Cesa-Bianchi, N., and Haussler, D. Scale-sensitive dimensions, uniform convergence, and learnability. *Journal of the ACM*, 44(4):615–631, 1997.

Bartlett, P. L. and Mendelson, S. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2003.

Bartlett, P. L., Long, P. M., and Williamson, R. C. Fat-shattering and the learnability of real-valued functions. *Journal of Computer and System Sciences*, 52(3):434–452, 1996.

Brownlees, C., Joly, E., and Lugosi, G. Empirical risk minimization for heavy-tailed losses. *Annals of Statistics*, 43(6):2507–2536, 2015.

Catoni, O. High confidence estimates of the mean of heavy-tailed real random variables. *arXiv preprint arXiv:0909.5366*, 2009.

Catoni, O. Challenging the empirical mean and empirical variance: a deviation study. *Annales de l'Institut Henri Poincaré, Probabilités et Statistiques*, 48(4):1148–1185, 2012.

Chen, Y., Su, L., and Xu, J. Distributed statistical machine learning in adversarial settings: Byzantine gradient descent. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 1(2):44, 2017.

Daniely, A. and Shalev-Shwartz, S. Optimal learners for multiclass problems. In *27th Annual Conference on Learning Theory*, volume 35 of *Proceedings of Machine Learning Research*, pp. 287–316, 2014.

Feldman, V. Generalization of ERM in stochastic convex optimization: The dimension strikes back. In *Advances in Neural Information Processing Systems 29*, pp. 3576–3584, 2016.

Finkenstädt, B. and Rootzén, H. (eds.). *Extreme Values in Finance, Telecommunications, and the Environment*. CRC Press, 2003.

Frostig, R., Ge, R., Kakade, S. M., and Sidford, A. Competing with the empirical risk minimizer in a single pass. *arXiv preprint arXiv:1412.6606*, 2015.

Hsu, D. and Sabato, S. Loss minimization and parameter estimation with heavy tails. *Journal of Machine Learning Research*, 17(18):1–40, 2016.

Huber, P. J. and Ronchetti, E. M. *Robust Statistics*. John Wiley & Sons, 2nd edition, 2009.

Johnson, R. and Zhang, T. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems 26*, pp. 315–323, 2013.

Kearns, M. J. and Schapire, R. E. Efficient distribution-free learning of probabilistic concepts. *Journal of Computer and System Sciences*, 48:464–497, 1994.

Le Roux, N., Schmidt, M., and Bach, F. R. A stochastic gradient method with an exponential convergence rate for finite training sets. In *Advances in Neural Information Processing Systems 25*, pp. 2663–2671, 2012.

Lecué, G. and Lerasle, M. Learning from MOM's principles. *arXiv preprint arXiv:1701.01961*, 2017.

Lecué, G., Lerasle, M., and Mathieu, T. Robust classification via MOM minimization. *arXiv preprint arXiv:1808.03106*, 2018.

Lin, J. and Rosasco, L. Optimal learning for multi-pass stochastic gradient methods. In *Advances in Neural Information Processing Systems 29*, pp. 4556–4564, 2016.

Luenberger, D. G. *Optimization by Vector Space Methods*. John Wiley & Sons, 1969.

Minsker, S. Geometric median and robust estimation in Banach spaces. *Bernoulli*, 21(4):2308–2335, 2015.

Minsker, S. and Strawn, N. Distributed statistical estimation and rates of convergence in normal approximation. *arXiv preprint arXiv:1704.02658*, 2017.

Murata, T. and Suzuki, T. Stochastic dual averaging methods using variance reduction techniques for regularized empirical risk minimization problems. *arXiv preprint arXiv:1603.02412*, 2016.

Nesterov, Y. *Introductory Lectures on Convex Optimization: A Basic Course*. Springer, 2004.

Nocedal, J. and Wright, S. *Numerical Optimization*. Springer Series in Operations Research. Springer, 1999.

Prasad, A., Suggala, A. S., Balakrishnan, S., and Ravikumar, P. Robust estimation via robust gradient estimation. *arXiv preprint arXiv:1802.06485*, 2018.

Rakhlin, A., Shamir, O., and Sridharan, K. Making gradient descent optimal for strongly convex stochastic optimization. In *Proceedings of the 29th International Conference on Machine Learning*, pp. 449–456, 2012.

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

van der Vaart, A. W. *Asymptotic Statistics*. Cambridge University Press, 1998.

Vardi, Y. and Zhang, C.-H. The multivariate L_1-median and associated data depth. *Proceedings of the National Academy of Sciences*, 97(4):1423–1426, 2000.