Robust descent using smoothed multiplicative noise

Matthew J. Holland
Osaka University

Abstract

In this work, we propose a novel robust gradient descent procedure which makes use of a smoothed multiplicative noise applied directly to observations before constructing a sum of soft-truncated gradient coordinates. We show that the procedure has competitive theoretical guarantees, with the major advantage of a simple implementation that does not require an iterative sub-routine for robustification. Empirical tests reinforce the theory, showing more efficient generalization over a much wider class of data distributions.

1 Introduction

The risk minimization model of learning is ubiquitous in machine learning, and it effectively captures the key facets of any effective learning algorithm: we must have reliable statistical inference procedures, and practical implementations of these procedures. Formulated using the expected loss, or risk $R(\boldsymbol{w}) := \mathbf{E}\, l(\boldsymbol{w}; \boldsymbol{z})$, induced by a loss l, where $\boldsymbol{w}$ is the parameter (vector, function, set, etc.) to be learned, and expectation is taken with respect to $\boldsymbol{z}$. In practice, all we are given is data $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n$, and based on this the algorithm outputs some candidate $\widehat{\boldsymbol{w}}$. If $R(\widehat{\boldsymbol{w}})$ is small with high confidence over the random sample, it provides some evidence for good generalization, subject to the assumptions placed on the underlying distribution. The statistical side is important because the risk R is always unknown, and the implementation is important since the only $\widehat{\boldsymbol{w}}$ we ever have in practice is one we can actually compute given finite data, time, and memory.

The vast majority of popular algorithms used today can be viewed as different implementations of empirical risk minimization (ERM), which admits any minimizer of $n^{-1}\sum_{i=1}^n l(\cdot; \boldsymbol{z}_i)$. From an algorithmic perspective, ERM is ambiguous; there are countless ways to implement the ERM procedure, and important work in recent years has highlighted the fact that a tremendous gap exists between the quality of good and bad ERM solutions (Feldman, 2016), for tasks as simple as multi-class pattern recognition (Daniely and Shalev-Shwartz, 2014), let alone tasks with unbounded losses. Furthermore, even tried-and-true implementations such as ERM by gradient descent (ERM-GD) only have appealing guarantees when the data is distributed sharply around the mean in a sub-Gaussian sense, as demonstrated in important work by Lin and Rosasco (2016). These facts are important because ERM is ubiquitous in modern learning algorithms, and heavy-tailed data by no means exceptional (Finkenstädt and Rootzén, 2003). Furthermore, these works suggest that procedures which have been designed to deal with finite samples of heavy-tailed data may be much more efficient than traditional ERM-based approaches, and indeed the theoretical promise of robust learning algorithms is being studied rigorously (Lecué and Lerasle, 2017; Lugosi and Mendelson, 2016, 2017a,b).

Review of related work Here we review the technical literature most closely related to our work. The canonical benchmark to be compared against is ERM-GD, for which Lin and Rosasco (2016) in pathbreaking work provide generalization guarantees under sub-Gaussian data. There are naturally two points of interest: (1) How do competing algorithms perform in settings when ERM is optimal? (2) What about robustness to settings in which ERM is sub-optimal? Many interesting robust learning algorithms have been studied in the past few years. One important procedure is from Brownlees et al. (2015), based on fundamental results due to Catoni (2012). The basic idea is to minimize an M-estimator of the risk, namely

$$\widehat{\boldsymbol{w}} = \arg\min_{\boldsymbol{w}} \widehat{l}(\boldsymbol{w})$$
$$\widehat{l}(\boldsymbol{w}) = \arg\min_{\theta\in\mathbb{R}} \sum_{i=1}^n \rho\left(l(\boldsymbol{w}; \boldsymbol{z}_i) - \theta\right).$$

Proceedings of the 22nd International Conference on Artificial Intelligence and Statistics (AISTATS) 2019, Naha, Okinawa, Japan. PMLR: Volume 89.

While the statistical guarantees are near-optimal under weak assumptions on the data, and the proxy loss $\widehat{l}$ can be computed accurately by an iterative procedure, its definition is implicit, and leads to rather significant computational roadblocks. Even if l and R and convex, the proxy loss need not be, and the non-linear optimization required by this method can be both unstable and costly in high dimensions.

Another important body of work looks at generalization of the classical "median of means" technique to higher dimensions. From Minsker (2015) and Hsu and Sabato (2016), the core idea is to partition the data into k disjoint subsets $\mathcal{D}_1 \cup \cdots \cup \mathcal{D}_k = \{1, 2, \ldots, n\}$, obtain ERM solutions on each subset, and then robustly aggregate these solutions such that poor candidates are effectively ignored. For example, using the geometric median approach of aggregation, we have

$$\widehat{\boldsymbol{w}} = \arg\min_{\boldsymbol{w}} \sum_{m=1}^{k} \|\boldsymbol{w} - \widetilde{\boldsymbol{w}}_m\|$$
$$\widetilde{\boldsymbol{w}}_m = \arg\min_{\boldsymbol{w}} \sum_{i \in \mathcal{D}_m} l(\boldsymbol{w}; \boldsymbol{z}_i), \quad m = 1, \ldots, k.$$

These robust aggregation methods can be implemented (Vardi and Zhang, 2000), and have appealing formal properties. An application of this technique to construct a robust loss was very recently proposed by Lecué et al. (2018). The main limitation of all these approaches is practical: when sample size n is small relative to the number of parameters to be determined, very few subsets can be created, and significant error due to bias occurs; conversely, when n is large enough to make many candidates, cheaper and less sophisticated methods often suffice. Furthermore, in the case of Lecué et al. (2018) where an expensive sub-routine must be run at every iteration, the computational overhead is substantial.

Also in the recent literature, interesting work has begun to appear looking at "robust gradient descent" algorithms, which is to say steepest descent procedures which utilize a robust estimate of the gradient vector of the risk (Chen et al., 2017a,b; Prasad et al., 2018). The basic idea is as follows. Assuming partial derivatives exist, writing $\boldsymbol{g}(\boldsymbol{w}) \coloneqq (\partial_1 R(\boldsymbol{w}), \ldots, \partial_d R(\boldsymbol{w}))$ for the risk gradient, we could iteratively solve this task by the following update:

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

Naturally, this procedure is ideal, since the underlying distribution is never known in practice, meaning R is always unknown. As such, we must approximate this objective function and optimize it with incomplete information. In taking a steepest descent approach, all that is required is an accurate approximation of $\boldsymbol{g}$. Instead of first approximating R and then using that approximation to infer $\boldsymbol{g}$, computational resources are better spent approximating $\boldsymbol{g}$ directly with some data-dependent $\widehat{\boldsymbol{g}}$ constructed using the loss gradients $\boldsymbol{l}'(\boldsymbol{w}; \boldsymbol{z}_1), \ldots, \boldsymbol{l}'(\boldsymbol{w}; \boldsymbol{z}_n)$, and plugging this in to the iterative update, as

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

Once again here, the median-of-means idea pops up in the literature, with Prasad et al. (2018) using a robust aggregation of empirical mean estimates of the gradient. That is, after partitioning the data into k subsets as before, the estimate vector $\widehat{\boldsymbol{g}}$ is constructed as

$$\widehat{\boldsymbol{g}}(\boldsymbol{w}) = \arg\min_{\boldsymbol{u}} \sum_{m=1}^{k} \|\boldsymbol{u} - \widetilde{\boldsymbol{g}}_m(\boldsymbol{w})\|$$
$$\widetilde{\boldsymbol{g}}_m(\boldsymbol{w}) = \frac{1}{|\mathcal{D}_m|} \sum_{i \in \mathcal{D}_m} \boldsymbol{l}'(\boldsymbol{w}; \boldsymbol{z}_i), \quad m = 1, \ldots, k.$$

and substituted within the gradient update (2). While conceptually a very appealing new proposition, computing $\widehat{\boldsymbol{g}}$ via a geometric median sub-routine introduces the exact same overhead and bias issues as the procedures of Hsu and Sabato (2016) just discussed, only that this time these costs are incurred at *each step* of the gradient descent procedure, and thus these costs and errors accumulate, and can propagate over time. Iterative approximations at each update take time and are typically distribution-dependent, while fast approximations leave a major gap between the estimators studied in theory and those used in practice.

Our contributions To address the limitations of both ERM-GD and the robust alternatives discussed above, we take an approach that allows us to obtain a robust gradient estimate directly, removing the need for iterative approximations, without losing the theoretical guarantees. In this paper, we provide both theoretical and empirical evidence that using the proposed procedure, paying a small price in terms of bias and computational overhead is worth it when done correctly, leading to a large payout in terms of distributional robustness. Key contributions are as follows:

- A practical learning algorithm which can closely mimic ERM-GD when ERM is optimal, but which performs far better under heavy-tailed data when ERM deteriorates.
- Finite-sample risk bounds that hold with high probability for the proposed procedure, under weak moment assumptions on the distribution of the loss gradient.

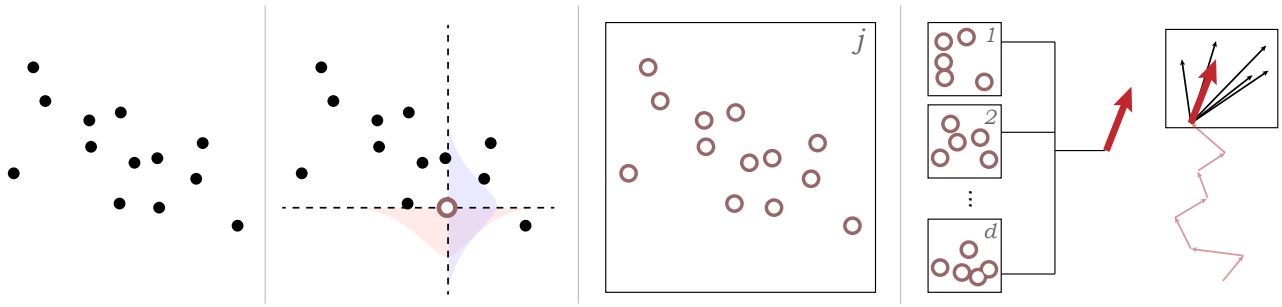

Figure 1: Illustration of the key elements of our proposed algorithm. From the far left, points represent loss gradient coordinates evaluated at different observations in our sample. To each point, we consider multiplication by Gaussian noise centered at 1. This noise is smoothed out by integration over the noise distribution, and applied to each gradient coordinate to generate a robust update direction.

- We demonstrate the ease of use and flexibility of our procedure in a series of experiments, testing performance using both controlled simulations and real-world datasets, and compared with numerous standard competitors.

2 Overview of proposed algorithm

Our proposed procedure can be derived in a few simple steps. Let us begin with a one-dimensional example, in which for random variable x we try to estimate $\mathbf{E}\, x$ based on sample $x_1, \ldots, x_n$. Our problem of interest is the setting in which the underlying distribution may be heavy-tailed, but it also may not be, and this information is not available to the learner *a priori*.

Scaling and truncation The first step involves a very primitive technique for ensuring the bias is small under well-behaved data, all while constraining the impact of outlying points. We re-scale, apply a soft truncation ψ, and then put the truncated arithmetic mean back in the original scale, namely

$$\frac{s}{n}\sum_{i=1}^{n}\psi\left(\frac{x_i}{s}\right) \approx \mathbf{E}\, x.$$

Here ψ should have the symmetry of an odd function ($\psi(-u) = -\psi(u)$), be non-decreasing on $\mathbb{R}$, with a slope of $\psi'(u) \to 1$ as $u \to 0$, and be bounded on $\mathbb{R}$. A simple example is the hyperbolic tangent function, $\tanh(u)$, but we shall consider other examples shortly. If the scale $s > 0$ is set such that $|x_i|/s$ is near zero for all but errant observations, the impact of the non-deviant terms to the arithmetic mean will be approximately equal, while the deviant points will have a disproportionately small impact.

Noise multiplication The second step involves applying multiplicative noise, albeit the purpose is rather unique. Let $\epsilon_1, \ldots, \epsilon_n$ be our independent random noise, generated from a common distribution $\epsilon \sim \nu$ with $\mathbf{E}_\nu\, \epsilon = 0$. We multiply each datum by $1+\epsilon$, and then pass each modified datum $x_i(1+\epsilon_i) = x_i + x_i\epsilon_i$ through the truncation function as above, yielding

$$\widetilde{x}(\boldsymbol{\epsilon}) = \frac{s}{n}\sum_{i=1}^{n}\psi\left(\frac{x_i + \epsilon_i x_i}{s}\right).$$

Multiplicative noise has received much attention in recent years in the machine learning literature, in particular with "dropout" in deep neural networks via Bernoulli random variables (Srivastava et al., 2014), and more recent investigations using Gaussian multiplicative noise (Nalisnick et al., 2015). In using multiplicative noise with mean 1, the basic idea is as follows. For typical points, an increase or decrease of a certain small fraction should not change the estimator output much. On the other hand, for wildly deviant points, a push further in the wrong direction is likely to be harmless due to ψ, while a push in the right direction could earn an additional valid point for the estimator.

Noise smoothing In the third and final step, we smooth out the multiplicative noise by taking the expectation of this estimator with respect to the noise distribution. This smoothed version of the estimator, still a random variable dependent on the original sample, is the final estimator of interest, defined

$$\widehat{x} := \mathbf{E}\, \widetilde{x}(\boldsymbol{\epsilon}) = \frac{s}{n}\sum_{i=1}^{n}\int \psi\left(\frac{x_i + \epsilon_i x_i}{s}\right) d\nu(\epsilon_i). \quad (3)$$

Computationally, in order to obtain $\widehat{x}$ to approximate $\mathbf{E}\, x$, we will not actually have to generate the ϵ_i and multiply the x_i by $(1+\epsilon_i)$, but instead will have to evaluate the integral.

Computational matters Before we move to the high-dimensional setting of interest, how can we ac-

tually compute this $\widehat{x}$? Numerical integration is not appealing as the overhead will be too much for a subroutine to be repeated many times. Naturally, the computational approach will depend on the noise distribution ν, and the truncation function ψ. Using recent results in the statistical literature from Catoni and Giulini (2017), if we set the truncation function to be

$$\psi(u) := \begin{cases} u - u^3/6, & -\sqrt{2} \le u \le \sqrt{2} \\ 2\sqrt{2}/3, & u > \sqrt{2} \\ -2\sqrt{2}/3, & u < -\sqrt{2} \end{cases} \quad (4)$$

and set the noise distribution to be $\nu = N(0, 1/\beta)$, then the integral of interest can be given in an explicit form that is simple to compute, requiring no numerical integration or approximation. Written in a general form with shift parameter $a \in \mathbb{R}$ and scale parameter $b > 0$, we can express the integral as

$$\mathbf{E}_\nu \, \psi\left(a + b\sqrt{\beta}\epsilon\right) = a\left(a - \frac{b^2}{2}\right) - \frac{a^3}{6} + C(a, b) \quad (5)$$

where $C(a, b)$ is a correction term that is complicated to write, but extremely simple to implement (see supplement for exact form).

Proposed learning algorithm Let us now return to the high-dimensional setting of interest. At any candidate $\boldsymbol{w}$, we can evaluate the $l(\boldsymbol{w}; \boldsymbol{z}_i)$ and $\boldsymbol{l}'(\boldsymbol{w}; \boldsymbol{z}_i)$ for all points $i = 1, \ldots, n$. The heart of our proposal: apply the sub-routine specified in (3) to each coordinate of the loss gradients, which can be computed directly using (5), and plug the resulting "robust gradient estimate" into the usual first-order update (2). Pseudocode for the proposed procedure is provided in Algorithm 1. All operations on vectors in the pseudo-code are element-wise, e.g., $\boldsymbol{u}^2 = (u_1^2, \ldots, u_d^2)$, $|\boldsymbol{u}| = (|u_1|, \ldots, |u_d|)$, $\boldsymbol{u}/\boldsymbol{v} = (u_1/v_1, \ldots, u_d/v_d)$, and so forth. For readability, we abbreviate $\boldsymbol{l}_i'(\boldsymbol{w}) := \boldsymbol{l}'(\boldsymbol{w}; \boldsymbol{z}_i)$.

As a simple example of the guarantees that are available for this procedure, assuming just finite variance of the gradients, and setting $\alpha_{(t)} = \alpha$ for simplicity, we have that $R(\widehat{\boldsymbol{w}}_{(T)}) - R^*$ is bounded above by

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

with probability at least $1 - \delta$ over the random draw of the sample, where d is the dimension of the space the gradient lives in, Δ is the diameter of $\mathcal{W}$, and the constant γ depends only on $R(\cdot)$. Theoretically, these results are competitive with existing state of the art methods cited in the previous section, but with the computational benefits of zero computational error, direct computability, and the fact that per-step computation time is independent of the underlying distribution.

3 Theoretical analysis

In this section, we carry out some formal analysis of the generalization performance of Algorithm 1. More concretely, we provide guarantees in the form of high-probability upper bounds on the excess risk achieved by the proposed procedure, given a finite sample of n observations, and finite budget of T iterations. Our approach can be broken down into three straightforward steps: (a) Obtain pointwise error bounds for $\widehat{\boldsymbol{g}}(\boldsymbol{w}) \approx \boldsymbol{g}(\boldsymbol{w})$. (b) Extend step (a) to obtain error bounds uniform in $\boldsymbol{w} \in \mathcal{W}$. (c) Control distance of $\widehat{\boldsymbol{w}}_{(t)}$ from minimizer at each step. All detailed proofs are given in the supplemental materials.

Notation Here we organize the key notation used in the remainder of our theoretical analysis and associated proofs (some are re-statements of definitions above). The observable loss is $l : \mathcal{W} \times \mathcal{Z} \to \mathbb{R}$, where $\mathcal{W}$ is the model from which the learning machine can select parameters $\boldsymbol{w} \in \mathcal{W}$, and $\mathcal{Z}$ is the space housing the data sample, $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n$. The data distribution is denoted $\boldsymbol{z} \sim \mu$, and the noise distribution featured in our algorithm is $\epsilon \sim \nu$. The risk to be minimized is $R(\boldsymbol{w}) := \mathbf{E}_\mu \, l(\boldsymbol{w}; \boldsymbol{z})$. The risk and loss gradients are respectively $\boldsymbol{g}$ and $\boldsymbol{l}'$. Estimates of $\boldsymbol{g}$ based on observations of $\boldsymbol{l}'$ are denoted $\widehat{\boldsymbol{g}}$. We frequently use $\mathbf{P}$ to denote a generic probability measure, typically the product measure induced by the sample, which should be clear from the context. Unless specified otherwise, $\|\cdot\|$ shall denote the usual ℓ_2 norm on Euclidean space. For integer $k > 0$, write $[k] := \{1, \ldots, k\}$.

Assumptions No algorithm can achieve arbitrarily good performance across all possible distributions. To obtain meaningful results, we must place conditions on the data and algorithm. We give concrete examples to illustrate that these assumptions are reasonable, and that they include scenarios that allow for both sub-Gaussian and heavy-tailed data.

A0. $\mathcal{W}$ is a closed, convex subset of $\mathbb{R}^d$, with diameter $\Delta := \sup\{\|\boldsymbol{u} - \boldsymbol{v}\| : \boldsymbol{u}, \boldsymbol{v} \in \mathcal{W}\} < \infty$.

A1. Loss function $l(\,\cdot\,; \boldsymbol{z})$ is λ-smooth on $\mathcal{W}$.

A2. $R(\cdot)$ is λ-smooth, and continuously differentiable on $\mathcal{W}$.

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

A4. $R(\cdot)$ is κ-strongly convex on $\mathcal{W}$.

Algorithm 1 Outline of robust gradient descent learning algorithm

inputs: $\widehat{\boldsymbol{w}}_{(0)}$, $T > 0$, $\delta \in (0,1)$, $\beta = \sqrt{2\log(\delta^{-1})}$
for $t = 0, 1, \ldots, T-1$ **do**

Scale set to minimize error bound, via Remark 3 and Lemma 5.

$$\boldsymbol{s}_{(t)} = \sqrt{n\boldsymbol{v}_{(t)}/2\log(\delta^{-1})}, \text{ where } \boldsymbol{v}_{(t)} \geq \mathbf{E}_\mu \boldsymbol{l}_i'(\widehat{\boldsymbol{w}}_{(t)})^2$$

Corrected gradient estimate, via (3) and (5):

$$\widehat{\boldsymbol{g}}_{(t)} = \frac{1}{n}\sum_{i=1}^n \left(\boldsymbol{l}_i'(\widehat{\boldsymbol{w}}_{(t)}) \left(1 - \frac{\boldsymbol{l}_i'(\widehat{\boldsymbol{w}}_{(t)})^2}{2\boldsymbol{s}_{(t)}^2\beta} \right) - \frac{\boldsymbol{l}_i'(\widehat{\boldsymbol{w}}_{(t)})^3}{6\boldsymbol{s}_{(t)}^2} \right) + \frac{1}{n}\sum_{i=1}^n C\left(\frac{\boldsymbol{l}_i'(\widehat{\boldsymbol{w}}_{(t)})}{\boldsymbol{s}_{(t)}}, \frac{|\boldsymbol{l}_i'(\widehat{\boldsymbol{w}}_{(t)})|}{\boldsymbol{s}_{(t)}\sqrt{\beta}} \right)$$

Plug in to gradient-based update (2).

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

end for
return: $\widehat{\boldsymbol{w}}_{(T)}$

A5. There exists $v < \infty$ such that $\mathbf{E}_\mu(l_j'(\boldsymbol{w};\boldsymbol{z}))^2 \leq v$, for all $\boldsymbol{w} \in \mathcal{W}$, $j \in [d]$.

Of these assumptions, assuredly A0 is simplest: any ball (here in the ℓ_2 norm) with finite radius will suffice, though far more exotic examples are assuredly possible. The remaining assumptions require some checking, but hold under very weak assumptions on the underlying distribution (additional examples in the supplement).

Example 1 (Concrete example of assumption A5). Consider a linear regression model $y = \langle \boldsymbol{w}^*, \boldsymbol{x}\rangle + \eta$, under squared error $l(\boldsymbol{w};\boldsymbol{z}) = (\langle \boldsymbol{w}, \boldsymbol{x}\rangle - y)^2$. Assume that $\mathbf{E}\,\boldsymbol{x} = 0$, that the noise η and input $\boldsymbol{x}$ are independent, and that the components of $\boldsymbol{x} = (x_1, \ldots, x_d)$ are independent of each other. Some straightforward algebra shows that

$$\begin{aligned}\mathbf{E}(l_j'(\boldsymbol{w};\boldsymbol{z}))^2 &= 4\left(\mathbf{E}\, x_j^2 \langle \boldsymbol{w} - \boldsymbol{w}^*, \boldsymbol{x}\rangle^2 + \mathbf{E}\,\eta^2\, \mathbf{E}\, x_j^2\right) \\ &\leq 4\left(\|\boldsymbol{w} - \boldsymbol{w}^*\|^2\, \mathbf{E}\, x_j^2 \|\boldsymbol{x}\|^2 + \mathbf{E}\,\eta^2\, \mathbf{E}\, x_j^2\right).\end{aligned}$$

It follows that as long as the noise η has finite variance ($\mathbf{E}\,\eta^2 < \infty$), and all inputs have finite fourth moments $\mathbf{E}\, x_j^4 < \infty$, then using assumption A0, we get

$$\mathbf{E}(l_j'(\boldsymbol{w};\boldsymbol{z}))^2 \leq 4\left(\Delta^2\, \mathbf{E}\, x_j^2 \|\boldsymbol{x}\|^2 + \mathbf{E}\,\eta^2\, \mathbf{E}\, x_j^2\right) < \infty.$$

This holds for all $\boldsymbol{w} \in \mathcal{W}$, satisfying A5.

In the analysis that follows, A0–A5 are assumed to hold.

Analysis of Algorithm 1 with discussion Here we consider the learning performance of the proposed procedure given by Algorithm 1. Almost every step of the procedure is given explicitly, save for the means of setting the moment bound $\boldsymbol{v}_{(t)}$ (see section 4), and the step size setting of $\alpha_{(t)}$. In the subsequent analysis of section 3, we shall specify exact settings of $\alpha_{(t)}$, and show how these settings impact the final guarantees that can be made.

We begin with a general fact that shows the subroutine used to estimate each element of the risk gradient has sharp guarantees under weak assumptions.

Lemma 2 (Pointwise accuracy)**.** *Consider data $x_1, \ldots, x_n$, with distribution $x \sim \mu$. Assume finite second moments, and a known upper bound $\mathbf{E}_\mu\, x^2 \leq v < \infty$. With probability no less than $1-\delta$, the estimator $\widehat{x}$ defined in (3) using $\nu = N(0, 1/\beta)$ with $\beta = \sqrt{2\log(\delta^{-1})}$, and scaled with $s = \sqrt{nv/(2\log(\delta^{-1}))}$ satisfies*

$$|\widehat{x} - \mathbf{E}_\mu\, x| \leq \sqrt{\frac{2v\log(\delta^{-1})}{n}} + \sqrt{\frac{v}{n}}.$$

Remark 3 (Scaling in Lemma 2). Note that the scale setting $s > 0$ in the above lemma depends on the sample size n and the second moment of the underlying distribution. This can be derived from an exponential tail bound on the deviations of this estimator, namely we have that for any choice of $s > 0$,

$$\mathbf{P}\left\{ |\widehat{x} - \mathbf{E}_\mu\, x| \leq \frac{v}{2s} + \frac{s\log(\delta^{-1})}{n} + \sqrt{\frac{v}{n}} \right\} \geq 1 - \delta.$$

Choosing s to minimize this upper bound yields the final results given in the lemma. Details are given in the proof. As for the setting of β given, this is also described in the proof; an optimal setting dependent on s can be derived in the form $\beta^2 = nv/s^2$, which simplifies to the final form given in the above lemma.

Of critical importance is that Lemma 2 only assumes finite variance, nothing more. Higher-order moments

may be infinite or undefined, and the results still hold. This means the results hold for both Gaussian-like well-behaved data, and heavy-tailed data which are prone to errant observations. Next, we show that this estimator has a natural continuity property.

Lemma 4 (Estimator is Lipschitz)**.** *Considering the estimator $\widehat{x}$ defined in (3) as a function of the data $\boldsymbol{x} := (x_1, \ldots, x_n) \in \mathbb{R}^n$, it satisfies the following Lipschitz property:*

$$|\widehat{x}(\boldsymbol{x}) - \widehat{x}(\boldsymbol{x}')| \leq \frac{c_\nu}{n}\|\boldsymbol{x} - \boldsymbol{x}'\|_1, \quad \text{for all } \boldsymbol{x}, \boldsymbol{x}' \in \mathbb{R}^n$$

where the factor c_ν takes the form

$$c_\nu = 1 - 2\Phi\left(-\sqrt{\beta}\right) + \sqrt{\frac{2}{\beta\pi}}\exp\left(-\frac{\beta}{2}\right)$$

where $\Phi(u) := \mathbf{P}\{N(0,1) \leq u\}$, the cumulative distribution function of the standard Normal distribution.

At each step in an iterative procedure, we have some candidate $\boldsymbol{w}$, at which we can evaluate the loss $l(\boldsymbol{w}; \boldsymbol{z}_i)$ and/or the gradient $\boldsymbol{l}'(\boldsymbol{w}; \boldsymbol{z}_i)$ over some or all data points $i \in [n]$. In traditional ERM-GD, one simply uses the empirical mean of the loss gradients to approximate $\boldsymbol{g}(\boldsymbol{w})$. In our proposed robust gradient descent procedure, instead of just doing summation, we feed the loss gradients as data into the robust procedure (3), highlighted in Lemma 2. Running this subroutine for each dimension results in a novel estimator $\widehat{\boldsymbol{g}}(\boldsymbol{w})$ of the risk gradient $\boldsymbol{g}(\boldsymbol{w})$, to be plugged into (2), constructing a novel steepest descent update. Since the candidate $\boldsymbol{w}$ at any step will depend on the random draw of the data set $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n$, upper bounds on the estimation error must be uniform in $\boldsymbol{w} \in \mathcal{W}$ in order to capture all contingencies. More explicitly, we require for some bound $0 < \varepsilon < \infty$ that

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

Using the following lemma, we can show that such a bound does exist, and its form can be readily characterized.

Lemma 5 (Uniform accuracy)**.** *Consider the risk gradient approximation $\widehat{\boldsymbol{g}} = (\widehat{g}_1, \ldots, \widehat{g}_d)$, defined at $\boldsymbol{w}$ (for $j \in [d]$) with $s_j^2 = nv_j/2\log(\delta^{-1})$ as*

$$\widehat{g}_j(\boldsymbol{w}) := \frac{s_j}{n}\sum_{i=1}^{n}\int \psi\left(\frac{l_j'(\boldsymbol{w}; \boldsymbol{z}_i)(1+\epsilon_i)}{s_j}\right) d\nu(\epsilon_i) \tag{7}$$

with v_j any valid bound satisfying $v_j \geq \mathbf{E}_\mu |l_j'(\boldsymbol{w}; \boldsymbol{z})|^2$, for all $\boldsymbol{w} \in \mathcal{W}$. Then, with probability no less than $1 - \delta$, for any choice of $\boldsymbol{w} \in \mathcal{W}$, we have that

$$\|\widehat{\boldsymbol{g}}(\boldsymbol{w}) - \boldsymbol{g}(\boldsymbol{w})\| \leq \frac{\widetilde{\varepsilon}}{\sqrt{n}},$$

where writing $V := \max_{j \in [d]} v_j$, the error $\widetilde{\varepsilon}$ is

$$\widetilde{\varepsilon} := \sqrt{2dV(\log(d\delta^{-1}) + d\log(3\Delta\sqrt{n}/2))} + \lambda(1 + c_\nu\sqrt{d}) + \sqrt{dV}.$$

We now have that (6) is satisfied by our underlying routine, as just proved in Lemma 5. The last remaining task is to disentangle the underlying optimization problem (minimization of unknown $R(\cdot)$) from the statistical estimation problem (approximating $\boldsymbol{g}$ with $\widehat{\boldsymbol{g}}$), in order to control the distance between the output of Algorithm 1 after T iterations, denoted $\widehat{\boldsymbol{w}}_{(T)}$, and the minimizer $\boldsymbol{w}^*$ of $R(\cdot)$.

Lemma 6 (Distance control)**.** *Consider the general approximate GD update (1), and assume that (6) holds with bound $0 < \varepsilon < \infty$. Then, with probability no less than $1 - \delta$, the following statements hold.*

1. *Setting $\alpha_{(t)} = \alpha/\gamma$, with $0 < \alpha < 1$, we have*

$$\|\widehat{\boldsymbol{w}}_{(T)} - \boldsymbol{w}^*\| \leq (1-\alpha)^{T/2}\|\widehat{\boldsymbol{w}}_{(0)} - \boldsymbol{w}^*\| + \frac{2\varepsilon}{\gamma}.$$

2. *Setting $\alpha_{(t)} = 1/((2+t)\gamma)$, we have*

$$\|\widehat{\boldsymbol{w}}_{(T)} - \boldsymbol{w}^*\| \leq \frac{1}{\sqrt{t+2}}\|\widehat{\boldsymbol{w}}_{(0)} - \boldsymbol{w}^*\| + \frac{\varepsilon}{\gamma}.$$

Our preparatory lemmas are now complete, and we can finally focus on the risk itself. We are considering bounds on the excess risk, namely the difference between the risk achieved by our procedure $R(\widehat{\boldsymbol{w}}_{(T)})$, and $R^* := \inf\{R(\boldsymbol{w}) : \boldsymbol{w} \in \mathcal{W}\} = R(\boldsymbol{w}^*)$, namely the best possible performance using $\mathcal{W}$.

Theorem 7 (Excess risk bounds, fixed step size)**.** *Write $\widehat{\boldsymbol{w}}_{(T)}$ for the output of Algorithm 1 after T iterations, assuming step size $\alpha_{(t)} = \alpha/\gamma$, and moment bounds $\boldsymbol{v}_{(t)} \leq V$ for all t. It follows that*

$$\begin{aligned} R(\widehat{\boldsymbol{w}}_{(T)}) - R^* &\leq (1-\alpha)^T\lambda\|\widehat{\boldsymbol{w}}_{(0)} - \boldsymbol{w}^*\|^2 + \frac{4\lambda\widetilde{\varepsilon}}{\kappa^2 n} \\ &= O\left((1-\alpha)^T\right) + O\left(\frac{d(\log(d\delta^{-1}) + d\log(\Delta n))}{n}\right) \end{aligned}$$

with probability no less than $1 - \delta$ over the random draw of the sample $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_n$, where $\widetilde{\varepsilon}$ and V are as defined in Lemma 5.

Remark 8 (Comparison with other RGD). An immediate observation is that if we let T scale with n such that $T \to \infty$ as $n \to \infty$, we have convergence in probability, and indeed to get arbitrarily good generalization at confidence $1 - \delta$, one requires on the other of $d^2\log(\delta^{-1})$ observations. These rates are directly comparable to those of Chen et al. (2017b) under similar

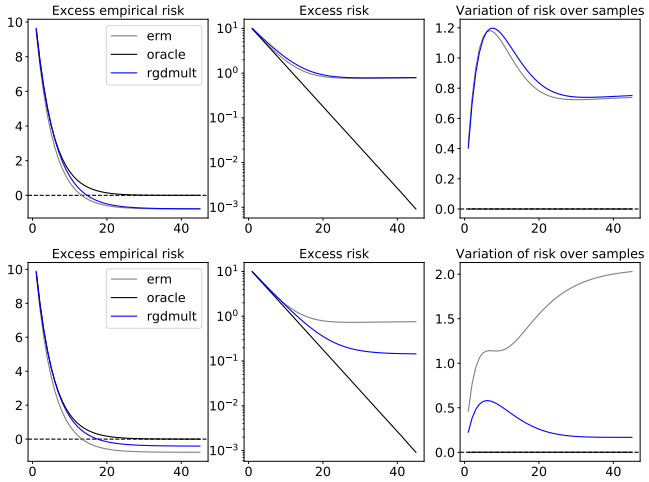

Figure 2: Performance metrics as a function of iterative updates. Top row: Normal noise. Bottom row: log-Normal noise. Settings: $n = 500, d = 2, \alpha_{(t)} = 0.1$ for all t.

assumptions. The rates in Prasad et al. (2018) are of order $\Omega(d)$, but their sample-splitting argument means error terms depend on T, leading to much slower rates when T grows with n.

One would expect that with robust estimates of the risk gradient that over a wide variety of distributions, that the updates of Algorithm 1 should have small variance given enough observations. The following result shows that this is true, with the procedure stabilizing to the best level available under the given sample as the procedure closes in on a valid solution.

Theorem 9 (Control of update variance)**.** *Run Algorithm 1 under the same assumptions as Theorem 7, except with step-size $\alpha_{(t)}$ left arbitrary. Then, for any step $t \geq 0$, taking expectation with respect to the sample $\{\boldsymbol{z}_i\}_{i=1}^n$ conditioned on $\widehat{\boldsymbol{w}}_{(t)}$, we have*

$$\mathbf{E}\,\|\widehat{\boldsymbol{w}}_{(t+1)} - \widehat{\boldsymbol{w}}_{(t)}\|^2 \leq 2\alpha_{(t)}^2 \left(\frac{d^2 V}{n} \left(\sqrt{\frac{\pi}{2}} + \frac{1}{e^2} \right) + \|\boldsymbol{g}(\widehat{\boldsymbol{w}}_{(t)})\|^2 \right).$$

4 Empirical analysis

In the numerical experiments that follow, our primary goal is to elucidate the relationship that exists between factors of the learning task (e.g., sample size, model dimension, initial value, underlying data distribution) and the performance of the robust gradient descent procedure proposed in Algorithm 1. We are interested in how these factors impact algorithm behavior in an absolute sense, as well as performance relative to well-known competitors. Details of the experimental setup are given in the supplementary materials.

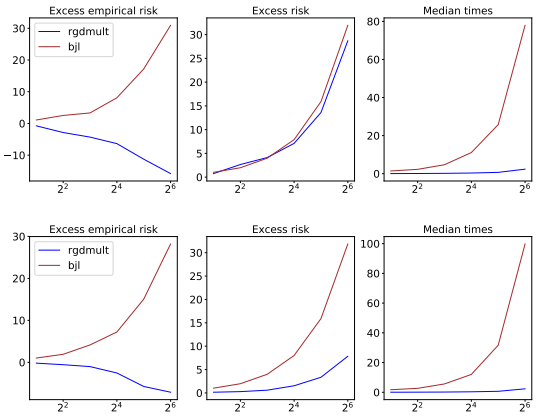

Figure 3: Comparison of our robust gradient-based approach with the robust objective-based approach. Top: Normal noise. Bottom: log-Normal noise. Performance is given as a function of the number of d, the number of parameters to optimize, given in $\log_2$ scale. Settings: $n = 500, \alpha_{(t)} = 0.1$ for all t.

4.1 Controlled tests

Noisy convex minimization Our first inquiry is a basic proof of concept: are there natural problem settings under which using `rgdmult` over ERM-GD is advantageous? How does this procedure perform when ERM-GD is known to be effectively optimal? In linear regression under Gaussian noise, ERM-GD is effectively optimal (Lin and Rosasco, 2016, Appendix C). As a baseline, we start with Gaussian noise (mean 0, standard deviation 20), and then consider centered log-Normal noise (log-location 0, log-scale 1.75) as a representative example of asymmetric, heavy-tailed data. Performance results are given in Figure 2.

Comparison with robust loss minimizer In section 1, we cited the important work of Brownlees et al. (2015), which chiefly considered theoretical analysis of a robust learning procedure that minimizes a robust *objective*, in contrast to our use of a robust update direction. Our proposed procedure enjoys essentially the same theoretical guarantees, and we have claimed that it is more practical. Here we attempt to verify this claim empirically. Denote the method of Brownlees et al. (2015) by `bjl`. Computation times along with performance results are given in Figure 3.

Regression application For our next class of experiments, we look at a more general regression task, under a diverse collection of data distributions. We then compare Algorithm 1 with well-known procedures specialized to regression, both classical and recent. Here we consider four noise families: log-logistic

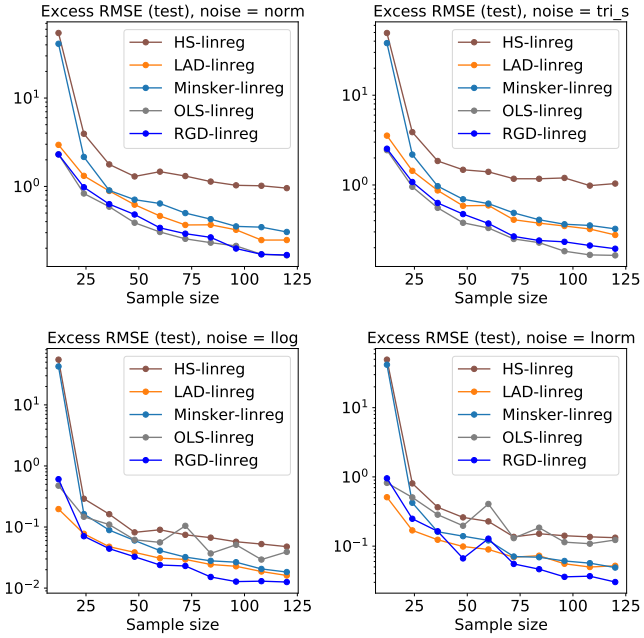

Figure 4: Top: Prediction error over sample size $12 \leq n \leq 122$, fixed $d = 5$, noise level $= 8$. Noise distributions are Normal, symmetric triangular, log-logarithmic, and log-Normal.

(denoted `llog` in figures), log-Normal (`lnorm`), Normal (`norm`), and symmetric triangular (`tri_s`). Even with just these four, we have representative distributions with both bounded and unbounded sub-Gaussian noise, and heavy-tailed data both with and without finite higher-order moments. We consider several methods against which we compare the proposed Algorithm 1. As classical choices, we have ordinary least squares (ERM under the squared error, `ols`) and least absolute deviations (ERM under absolute error, `lad`). For more recent methods, as described in section 1, we consider robust regression routines as given by Minsker (2015) (`geomed`) and Hsu and Sabato (2016) (`hs`). In the former, we partition the data, obtaining the `ols` solution on each subset, and these candidates are aggregated using the geometric median in the ℓ_2 norm (Vardi and Zhang, 2000). In the latter, we used source code published online by the authors. To compare our Algorithm 1 with these routines, we initialize `rgdmult` to the analytical `ols` solution, with step size $\alpha_{(t)} = 0.01$ for all iterations, and $\delta = 0.005$. Variance bounds $\boldsymbol{v}_{(t)}$ are set to the empirical second moments of $\boldsymbol{l}'(\widehat{\boldsymbol{w}}_{(t)}, \boldsymbol{z})$, divided by 2. In total, the number of iterations is constrained by a fixed budget: we allow for $40n$ gradient evaluations in total. Representative results are provided in Figure 4.

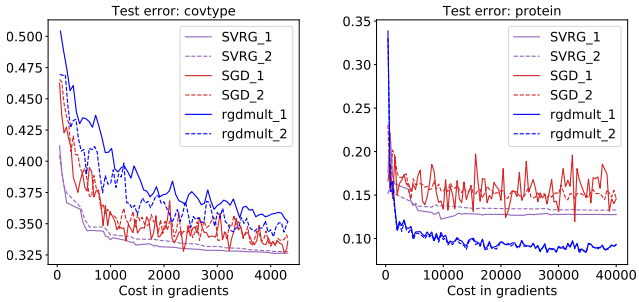

Figure 5: Test error (misclassification rate) over budget spent, as measured by gradient computations, for the top two performers within each method class. Each plot corresponds to a distinct dataset.

4.2 Application to real-world benchmarks

We investigate the utility of a random mini-batch version of Algorithm 1 here, compared with vanilla SGD, and stochastic variance-reduced gradient descent (SVRG) proposed by Johnson and Zhang (2013). The model used is standard logistic regression, common across all methods. We consider binary classification based on the Covertype dataset (UCI respository) and the protein homology dataset (KDD Cup). For each dataset and each method, we chose the top two parameters settings, written `*_1` and `*_2` here. Here the "top two" refers to performance as measured by the median test error for the last five iterations. Representative results are given in Figure 5.

5 Looking ahead

We are particularly interested in moving beyond per-coordinate robustification, and considering operations that operate on the loss gradient vectors themselves as atomic units. The per-coordinate technique is easy to implement and theoretical analysis is also more straightforward, but the risk bounds have an extra d factor that should be removable given more sophisticated procedures. Indeed, the high-dimensional mean estimation discussed by Catoni and Giulini (2017) has such a vector estimator, but unfortunately there is no way to actually compute the estimator they analyze. Bridging this gap is an important next step, from the perspective of both learning theory and machine learning practice.

Acknowledgements

This work was supported by ACT-I, from the Japan Science and Technology Agency (JST).

References

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

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

Catoni, O. and Giulini, I. (2017). Dimension-free PAC-Bayesian bounds for matrices, vectors, and linear least squares regression. *arXiv preprint arXiv:1712.02747*.

Chen, Y., Su, L., and Xu, J. (2017a). Distributed statistical machine learning in adversarial settings: Byzantine gradient descent. *arXiv preprint arXiv:1705.05491*.

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

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

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

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

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

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

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

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

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

Lugosi, G. and Mendelson, S. (2016). Risk minimization by median-of-means tournaments. *arXiv preprint arXiv:1608.00757*.

Lugosi, G. and Mendelson, S. (2017a). Regularization, sparse recovery, and median-of-means tournaments. *arXiv preprint arXiv:1701.04112*.

Lugosi, G. and Mendelson, S. (2017b). Sub-gaussian estimators of the mean of a random vector. *arXiv preprint arXiv:1702.00482*.

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

Nalisnick, E., Anandkumar, A., and Smyth, P. (2015). A scale mixture perspective of multiplicative noise in neural networks. *arXiv preprint arXiv:1506.03208*.

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

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958.

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