Robust gradient estimation in the presence of heavy-tailed noise

Fabian Schaipp
Technical University of Munich
Munich, Germany
fabian.schaipp@tum.de

Umut Şimşekli
Inria, CNRS, DI-ENS, PSL Research University
Paris, France
umut.simsekli@inria.fr

Robert M. Gower
Flatiron Institute, CCM
New York, U.S.
rgower@flatironinstitute.org

Abstract

In applications such as training transformers on NLP tasks, or distributed learning in the presence of corrupted nodes, the stochastic gradients have a heavy-tailed distribution. We argue that in these settings, momentum is not the best suited method for estimating the gradient. Instead, variants of momentum with different forms of clipping are better suited. Our argument is based on the following: in the presence of heavy tailed noise the sample median of the gradient is a better estimate than the sample mean. We then devise new iterative methods for computing the sample median on the fly based on the SPP (stochastic proximal point) method. These SPP methods applied to different definitions of median give rise to known and new type of clipped momentum estimates. We find that these clipped momentum estimates are more robust at estimating the gradient in the presence of noise coming from an α-stable distribution, and for a transformer architecture on the PTB and Wikitext-2 datasets, in particular when the batch size is large.

Consider the problem

$$\min_{\mathbf{w}\in\mathbb{R}^d} \ell(\mathbf{w}), \quad \ell(\mathbf{w}) := \mathbb{E}_{\mathbf{x}\sim\mathcal{P}}\left[\ell(\mathbf{w};\mathbf{x})\right] \tag{1}$$

where $\mathcal{P}$ is the distribution over data and $\ell(\mathbf{w};\mathbf{x})$ is a loss function. Here we are interested in cases where the distribution of $\nabla_{\mathbf{w}}\ell(\mathbf{w};\mathbf{x}) := \nabla\ell(\mathbf{w};\mathbf{x})$ may have significant outliers, is heavy tailed, or just very non-Gaussian. To give a few examples:

Language. For language modelling tasks, Zipf's law [20] states that the k-th most frequent word of a corpus appears with probability proportional to k^{-s} for $s \geq 0$. In other words, natural language roughly follows an inverse powerlaw distribution. Clearly, this is more heavy-tailed than a Gaussian where the tails go to zero exponentially.

Transformer gradients. It has been observed empirically that the gradients for training transformer architectures on language tasks are more heavy-tailed compared to, for example, convolutional models for image data [19, 11]. One possible explanation for this might be that the gradients inherit the heavy-tailed properties of the underlying data distribution $\mathcal{P}$. This is clearly the case in the simple setting of linear least squares

$$\min_{\mathbf{w}\in\mathbb{R}^d} \mathbb{E}_{\mathbf{x},y\sim\mathcal{P}}\left[\frac{1}{2}\|\langle\mathbf{x},\mathbf{w}\rangle - y\|^2\right].$$

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

The gradient is given by $\nabla \ell(\mathbf{w};(\mathbf{x},y)) = \mathbf{x}(\mathbf{x}^\top \mathbf{w} - y)$, and hence the distribution of $\nabla \ell(\mathbf{w}) = \mathbb{E}_{\mathbf{x},y\sim\mathcal{P}}\left[\nabla \ell(\mathbf{w};\mathbf{x})\right]$ is governed by the distribution of the data $(\mathbf{x},y)$. For example, having large outliers in $\mathbf{x}$ will result in large outliers in $\nabla \ell(\mathbf{w};\mathbf{x})$.

Corrupted nodes. In distributed learning, a well-studied scenario is when some of the nodes are malicious and can communicate adversarial updates. Many techniques have been developed in order to make training robust to this setting [4, 8, 10]. This line of research is also closely related to error feedback in federated learning [9, 14] and learning under differential privacy constraints [10].

Problems of form (1) are typically solved with SGD-type methods, that is

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \eta_t \mathbf{g}_t, \tag{SGD}$$

where $\eta_t > 0$ is a learning rate, and $\mathbf{g}_t$ is an estimate of $\nabla \ell(\mathbf{w}_t) = \mathbb{E}\left[\nabla \ell(\mathbf{w}_t;\mathbf{x})\right]$. The standard way to obtain $\mathbf{g}_t$ is to sample a mini-batch of data $\mathbf{x}_t \sim \mathcal{D}$, and select $\mathbf{g}_t = \nabla \ell(\mathbf{w}_t;\mathbf{x}_t)$ or use an exponentially weighted momentum estimate.

In the presence of heavy tails, the sample median is a more robust estimator of the mean (see Fig. 1 for a simple example). Consequently, to estimate the gradient in the presence of heavy-tailed noise we should consider the median. But there are three issues: (i) computing the median is expensive (classical iterative methods are Weiszfeld's algorithm [17] or modifications thereof [16]), (ii) the distribution of the gradient changes with the weights $\mathbf{w}$, and (iii) the gradient is a d-dimensional vector and there are several different notions of median in $\mathbb{R}^d$.

To resolve the first two issues, we consider only online iterative methods for estimating the median. These iterative estimates are cheap to compute, thus resolving the first issue. Furthermore, being iterative and online they can adapt as the distribution of the gradient changes. As for the third issue, we consider the median with respect to a divergence. This includes as a special case the classical notion of the *geometric median*.

The online iterative methods we consider are Stochastic (Sub-)Gradient Descent (SGD), and Stochastic Proximal Point (SPP). By using these two iterative methods, and different notions of median, we recover as a special case several well-known techniques such as heavy-ball momentum [13], clipping [7, 18] (such as Clip21 for distributed learning [10]), and signed gradient methods. In particular the observation that Clip21 is an online estimator of a geometric median appears to be new, and could have consequences for its application in distributed learning. Our proposed framework allows to unify the different motivations for previously developed techniques, but also to derive new robust estimators.

1 Sample median as a robust mean estimator

Let us give some basic definitions and properties of the median. For a real-valued random variable Z, its median is the solution to the problem

$$\texttt{median}(Z) \;:=\; \operatorname*{argmin}_{m\in\mathbb{R}} \mathbb{E}_{z\sim Z}\left[|m-z|\right], \tag{2}$$

where we use $\mathbb{E}_{z\sim Z}\left[\cdot\right]$ to denote taking expectation with respect to whichever distribution governs Z. In contrast to the mean, the median is always defined (though it may not be unique) [3]. If Z has a continuous density, taking the derivative[1] in (2) and setting to zero gives

$$\mathbb{E}_{z\sim Z}\left[\operatorname{sgn}(m-z)\right] = 0,$$

which is satisfied if m is greater than or equal to half of the elements (wrt. the density of Z), and less than or equal to the other half.

Since we are interested in approximating the median of the gradient, we need a notion of median that generalizes to random vectors $Z \in \mathbb{R}^d$. For this, we use the *geometric median* [6, 17, 2] defined as

$$\texttt{median}(Z) \;:=\; \operatorname*{argmin}_{\boldsymbol{m}\in\mathbb{R}^d} \mathbb{E}_{\boldsymbol{z}\sim Z}\left[\|\boldsymbol{m}-\boldsymbol{z}\|_2\right], \tag{3}$$

[1] The case where the set of subgradients is not a singleton is a null set and hence can be disregarded.

where $\|\boldsymbol{z}\|_2 := \sqrt{\sum_{i=1}^d \boldsymbol{z}_i^2}$ is the ℓ_2-norm. Though the ℓ_2-norm is often used in defining the geometric median, other norms might also induce robust gradient estimates. More generally, we are interested in estimators that solve

$$\texttt{median}_{\mathcal{D}}(Z) := \operatorname*{argmin}_{\boldsymbol{m}\in\mathbb{R}^d} \mathbb{E}_{\boldsymbol{z}\sim Z}\left[\mathcal{D}(\boldsymbol{m}-\boldsymbol{z})\right], \tag{4}$$

where $\mathcal{D}:\mathbb{R}^d\to\mathbb{R}_{\geq 0}$ is a function, which can be thought of as a distance measure or divergence. We assume that $\mathcal{D}$ is such that (4) admits a solution. If the function $\mathcal{D}$ heavily penalizes large values then (4) will be sensitive to outliers. This is the case for $\mathcal{D}=\frac{1}{2}\|\cdot\|_2^2$, which grows quadratically. In contrast $\mathcal{D}=\|\cdot\|_2$ and $\mathcal{D}=\|\cdot\|_1$ grow only linearly, and thus (4) will be less sensitive to outliers.

For Cauchy distributions, it is known that the variance of the sample median is finite for sample size $2n+1\geq 5$, while the variance of the sample mean is not finite [1]. Further, the breakdown point of the median estimator is roughly one half, meaning that at least half of the samples need to be outliers to let the estimator become useless, whereas for the mean the breakdown point is one over the sample size [5, 12]. See Fig. 1 for an illustration of the sensitivity of mean and median to outliers.

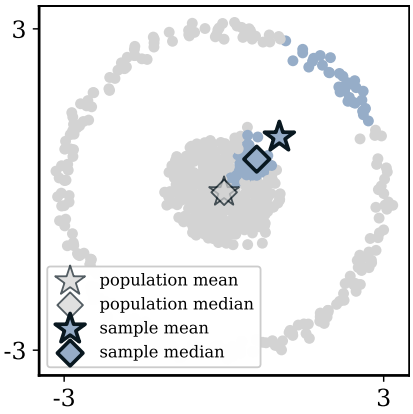

Figure 1: An illustration why the sample median can be a better estimator for the population mean than the sample median. Blue dots mark a sample of the full population. The outliers in the outer circle result in the sample mean moving away from the true mean.

Here we show that by choosing different divergences $\mathcal{D}$, and using standard iterative methods for computing (4), we recover many well known robust gradient estimators. These include estimators from distributed learning with adversarial nodes, heavy tailed problems, and problems with outliers. The iterative methods we use are stochastic (sub)gradient descent (SGD), and stochastic proximal point (SPP). Using different norms we also explore some new directions for robust gradient estimation.

2 Iterative estimator of the median gradient

First we consider the task of estimating the gradient from a fixed distribution. In the context of training machine learning models, $\boldsymbol{g}\in\mathbb{R}^d$ will be a stochastic gradient for a fixed weight, i.e. it is sampled from $\nabla\ell(\mathbf{w};\mathbf{x})$, for some fixed $\mathbf{w}$, where $\mathbf{x}\sim\mathcal{P}$ is sampled from training data. In the next section we will consider what happens when the distribution of $\boldsymbol{g}$ changes, as is the case during optimization since the parameters change.

Let $\mathcal{D}:\mathbb{R}^d\to\mathbb{R}_{\geq 0}$ be a closed, proper, and convex function. As discussed in the previous section, the geometric median is a robust estimator, given by

$$\operatorname*{argmin}_{\boldsymbol{m}\in\mathbb{R}^d} \mathbb{E}_{\boldsymbol{g}}\left[\mathcal{D}(\boldsymbol{m}-\boldsymbol{g})\right]. \tag{5}$$

Since we know the distribution of $\boldsymbol{g}$ will evolve as the parameters $\mathbf{w}$ evolve, we will only consider iterative methods for computing (5); more specifically, we consider either stochastic subgradient descent or stochastic proximal point (SPP). Let $\boldsymbol{m}_t\in\mathbb{R}^d$ be our current gradient estimate.

Stochastic (Sub-)Gradient Descent. We can solve (5) by iteratively taking steps of stochastic subgradient descent, where in each iteration we sample a gradient $\boldsymbol{g}$, then update our current estimate $\boldsymbol{m}_t$ by subtracting a stochastic subgradient as follows:

$$\boldsymbol{m}_{t+1}=\boldsymbol{m}_t-\tau\boldsymbol{u}_t,\quad \boldsymbol{u}_t\in\partial\mathcal{D}(\boldsymbol{m}_t-\boldsymbol{g}), \tag{6}$$

where $\tau>0$ is the learning rate, and $\boldsymbol{u}_t$ is a subgradient. Since $\mathcal{D}$ could be a non-differentiable function such as the ℓ_2- or ℓ_1-norm, we use subgradients instead of gradients. As a first simple

example of (6), consider the case where $\mathcal{D} = \frac{1}{2}\|\cdot\|_2^2$. In this case (6) becomes

$$\boldsymbol{m}_{t+1} = \boldsymbol{m}_t - \tau(\boldsymbol{m}_t - \boldsymbol{g}) = (1-\tau)\boldsymbol{m}_t + \tau\boldsymbol{g}. \tag{7}$$

This is the momentum [13, 15] estimator of the gradient, with momentum parameter $\beta = 1 - \tau$.

Stochastic Proximal Point. Because $\mathcal{D}$ will be a relatively simple function, such as a norm, we can compute its proximal operator and use SPP, given by

$$\boldsymbol{m}_{t+1} \;=\; \operatorname{prox}_{\tau\mathcal{D}}(\boldsymbol{m}_t;\boldsymbol{g}) \;:=\; \operatorname*{argmin}_{\mathbf{y}\in\mathbb{R}^d} \mathcal{D}(\mathbf{y}-\boldsymbol{g}) + \frac{1}{2\tau}\|\mathbf{y}-\boldsymbol{m}_t\|^2. \tag{8}$$

The proximal point method is sometimes referred to as the implicit gradient method, because it can also be written as

$$\boldsymbol{m}_{t+1} = \boldsymbol{m}_t - \tau\boldsymbol{u}_t, \quad \boldsymbol{u}_t \in \partial\mathcal{D}(\boldsymbol{m}_{t+1} - \boldsymbol{g}). \tag{9}$$

Note that the subgradient is evaluated at $\boldsymbol{m}_{t+1}-\boldsymbol{g}$, thus $\boldsymbol{m}_{t+1}$ appears on both sides of the inclusion, which is why this is an *implicit* update, as opposed to an *explicit* update as is the case in (6). Because of this implicit inclusion (9), computing a step of the proximal point method often requires an additional subroutine to solve (9), making it potentially impractical. Fortunately, in our setting the proximal operator of $\mathcal{D}$ will have a closed form solution which can be computed as

$$\begin{aligned}\operatorname{prox}_{\tau\mathcal{D}}(\boldsymbol{m}_t;\boldsymbol{g}) &\overset{\mathbf{y}=\hat{\mathbf{y}}+\boldsymbol{g}}{=} \boldsymbol{g} + \operatorname*{argmin}_{\hat{\mathbf{y}}\in\mathbb{R}^d} \mathcal{D}(\hat{\mathbf{y}}) + \frac{1}{2\tau}\|\hat{\mathbf{y}} - (\boldsymbol{m}_t - \boldsymbol{g})\|^2 \\ &=: \boldsymbol{g} + \operatorname{prox}_{\tau\mathcal{D}}(\boldsymbol{m}_t - \boldsymbol{g}).\end{aligned} \tag{10}$$

As a first simple example of (8), consider again the case where $\mathcal{D} = \frac{1}{2}\|\cdot\|_2^2$. In this case (8) gives

$$\boldsymbol{m}_{t+1} = \left(1 - \frac{\tau}{1+\tau}\right)\boldsymbol{m}_t + \frac{\tau}{1+\tau}\boldsymbol{g}. \tag{11}$$

This is again equivalent to the momentum gradient estimate, where the momentum parameter is given by $\beta = 1 - \tau/(1+\tau) = 1/(1+\tau)$. Though we have again arrived at the momentum method, curiously the momentum parameter β is such that $\beta \in [0,\ 1)$, which is what is used in practice. This is in contrast to (7) where $\beta \in [-\infty, 1)$ since $\tau > 0$. Because of this, and because SPP is generally considered a better method when it can be applied, we will now focus on SPP.

So far, using either (6) or SPP with $\mathcal{D} = \frac{1}{2}\|\cdot\|_2^2$ has resulted in the momentum method. We now consider other divergences, and uncover both known and new methods.

Vectorwise clipping. If we choose $\mathcal{D} := \|\cdot\|_2$ we arrive at a type of clipping that protects against sampled gradient with large norms by shrinking their norm.

Lemma 2.1. For $\mathcal{D} := \|\cdot\|_2$ the update (8) is given by

$$\boldsymbol{m}_{t+1} = \boldsymbol{m}_t + \operatorname{clip}_{\tau,2}(\boldsymbol{g} - \boldsymbol{m}_t) \tag{12}$$

$$= \left(1 - \frac{\tau}{\max\{\tau, \|\boldsymbol{g} - \boldsymbol{m}_t\|_2\}}\right)\boldsymbol{m}_t + \frac{\tau}{\max\{\tau, \|\boldsymbol{g} - \boldsymbol{m}_t\|_2\}}\boldsymbol{g}, \tag{13}$$

where

$$\operatorname{clip}_{\tau,2}(\boldsymbol{v}) := \frac{\tau}{\max\{\tau, \|\boldsymbol{v}\|_2\}}\boldsymbol{v}, \qquad \forall \boldsymbol{v} \in \mathbb{R}^d,\ \tau > 0. \tag{14}$$

Proof. Follows from (10) and Lemma A.2 since

$$\boldsymbol{m}_{t+1} = \boldsymbol{g} + \operatorname*{argmin}_{\hat{\mathbf{y}}\in\mathbb{R}^d} \|\hat{\mathbf{y}}\|_2 + \frac{1}{2\tau}\|\hat{\mathbf{y}} - (\boldsymbol{m}_t - \boldsymbol{g})\|^2 = \boldsymbol{g} + \operatorname{prox}_{\tau\|\cdot\|_2}(\boldsymbol{m}_t - \boldsymbol{g}). \qquad \square$$

Update (12) is identical to Clip21 [10]. In particular, it has also the form of a momentum update, but contrary to before the momentum coefficient now depends on t.

Componentwise clipping. If $\mathcal{D} := \|\cdot\|_1$, we again arrive at a form of clipping, but a clipping that protects against gradients with large individual entries by shrinking the entries independently.

Lemma 2.2. For $\mathcal{D} := \|\cdot\|_1$ the update (8) is given by

$$\boldsymbol{m}_{t+1} = \boldsymbol{m}_t + \mathrm{clip}_{\tau,1}(\boldsymbol{g} - \boldsymbol{m}_t). \tag{15}$$

where we define

$$\mathrm{clip}_{\tau,1}(\boldsymbol{v}) := \min\{\max\{\boldsymbol{v}, -\tau\}, \tau\}, \quad \forall \boldsymbol{v} \in \mathbb{R}^d,\ \tau > 0. \tag{16}$$

In the above, the max and min operators are defined component-wise.

Proof. Follows from (10) and Lemma A.1 since

$$\boldsymbol{m}_{t+1} = \boldsymbol{g} + \operatorname*{argmin}_{\hat{\mathbf{y}} \in \mathbb{R}^d} \|\hat{\mathbf{y}}\|_1 + \frac{1}{2\tau}\|\hat{\mathbf{y}} - (\boldsymbol{m}_t - \boldsymbol{g})\|^2 = \boldsymbol{g} + \mathrm{prox}_{\tau\|\cdot\|_1}(\boldsymbol{m}_t - \boldsymbol{g}). \qquad \square$$

In Appendix B we also consider more applications of stochastic subgradient descent (6). There we show that when $\mathcal{D} = \|\cdot\|_1$ the update (6) results in a type of signed gradient descent. When $\mathcal{D} = \|\cdot\|_2$ the update (6) is a type of momentum with an adaptive momentum coefficient.

3 Gradient estimator in the wild

Our final objective is to use robust gradient estimators within a `SGD`-type method. Because at each iteration the weights $\mathbf{w}_t$ are updated, the distribution of the gradients also changes at each iteration. Our strategy is to interweave updates in the parameters $\mathbf{w}_t$ with updates in the gradient estimators $\boldsymbol{m}_t$. That is, let $\mathcal{C} : \mathbb{R}^d \to \mathbb{R}^d$ be a given operator (e.g. a clipping operator), we consider methods of the form

$$\begin{array}{lcll} \boldsymbol{g}_t & = & \nabla \ell(\mathbf{w}_t; \mathbf{x}_t) & \texttt{(Sample gradient)} \\ \boldsymbol{m}_{t+1} & = & \boldsymbol{m}_t + \mathcal{C}(\mathbf{g}_t - \boldsymbol{m}_t) & \texttt{(Update gradient estimate)} \\ \boldsymbol{w}_{t+1} & = & \boldsymbol{w}_t - \eta_t \boldsymbol{m}_{t+1} & \texttt{(Update parameters)} \end{array} \tag{17}$$

When $\mathcal{C}$ is the identity operator (17) reduces to (`SGD`). If $\mathcal{C}$ is equal to $\mathrm{clip}_{\tau,2}$ or $\mathrm{clip}_{\tau,1}$, the gradient update (17) is equivalent to (12) or (15), respectively.

This method (17) decomposes the problem (1) into two steps: first, do an online update of the gradient estimator $\boldsymbol{m}_t$, and then take a descent step using this estimator $\boldsymbol{m}_{t+1}$. That is, as we can not access the full gradient $\nabla \ell(\mathbf{w}_t)$, we estimate it with $\boldsymbol{m}_{t+1}$, using only samples $\boldsymbol{g}_t = \nabla \ell(\boldsymbol{w}_t; \mathbf{x}_t)$. Second, we update weights $\boldsymbol{w}_t$, which in turn changes the distribution of the stochastic gradients. Thus our gradient estimator $\boldsymbol{m}_t$ is constantly playing catch up.

4 Experiments

Here we use `VClip` to refer to vectorwise clipping (14) and `CClip` for componentwise clipping (16).

4.1 Estimating a Gradient with α-stable noise

In a first experiment, we verify the hypothesis that estimating the median is more stable when the distribution $\mathcal{P}$ is heavy-tailed.

We consider the problem of estimating a fixed vector (resembling the true gradient at fixed weights), under varying degree of heavy-tailedness. For this purpose, we choose $\mathcal{P}$ to be the α-stable distribution $\mathcal{P}_\alpha$ with stability parameter $0 < \alpha \leq 2$ and skeweness parameter 0. The family of α-stable distributions is suitable because α controls the degree of heavy-tailedness: $\mathcal{P}_1$ is equal to the Cauchy distribution, and $\mathcal{P}_2$ is equal to a Gaussian.

Setup. We generate a fixed oracle vector $\hat{\mathbf{g}} \in \mathbb{R}^d$ with $d = 10$ where each component is generated i.i.d standard Gaussian. In each iteration, a sample $\mathbf{g}_t$ is generated as follows: each coordinate of $(\mathbf{g}_t)_i$ is (independently) sampled from $\mathcal{P}_\alpha$ with location $(\hat{\mathbf{g}})_i$ and varying values for $0 < \alpha \leq 2$. Importantly, the median of $\mathcal{P}_\alpha$ is equal to the location (i.e. $\hat{\mathbf{g}}$) for all values of α. On the contrary, for $\alpha \leq 1$ the mean of $\mathcal{P}_\alpha$ is not defined, and otherwise equal to the median.

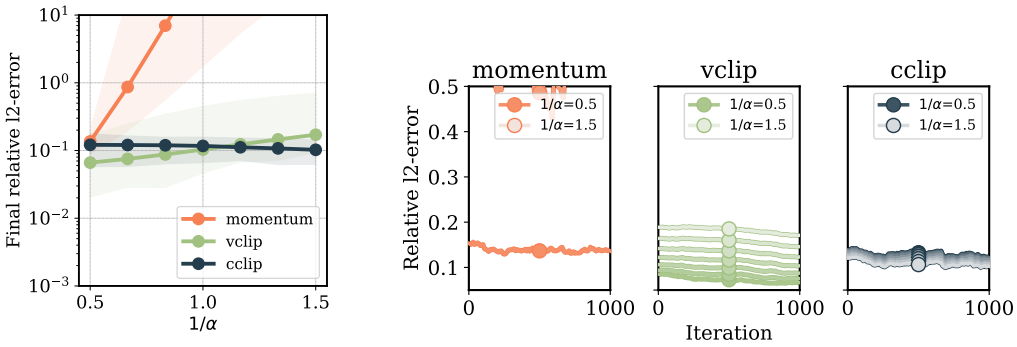

Figure 2: $\tau = 0.01$ **Left:** Final error for varying values of α (from left to right, distributions are more heavy-tailed). Shaded area marks minimal and maximal value over the 50 independent runs. **Right:** Convergence plot for all methods for $\frac{1}{\alpha} \in [0.5, 1.5]$ (higher value of $\frac{1}{\alpha}$ corresponds to heavier tails).

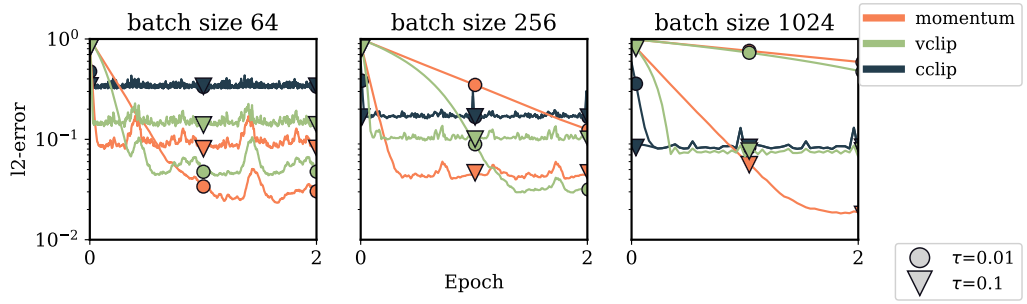

Figure 3: Encoder transformer on PTB dataset, with weights fixed at initialization. We use momentum, $\text{clip}_{\tau,1}$ and $\text{clip}_{\tau,2}$ to estimate the full-batch gradient.

We run several of the methods presented above with a fixed step size $\tau > 0$ and track the relative ℓ_2-error $\frac{\|\boldsymbol{m}_t - \hat{\mathbf{g}}\|_2}{\|\hat{\mathbf{g}}\|_2}$. For each method and distribution, we run 1000 iterations and 50 different seeds.

Discussion. From Fig. 2 we find that as the noise becomes more heavy-tailed (as α decreases), momentum fails to produce accurate estimates of the oracle $\hat{\mathbf{g}}$. On the other hand, both vectorwise clipping (`VClip`) and coordinatewise clipping (`CClip`) are relatively robust to heavy tails. Most pronounced, we observe that the error of `CClip` even slightly decreases with α decreasing. Both `VClip` and `CClip` are iteratively approaching the median instead of the mean - in conclusion, online median estimators are more robust to heavy tails as expected.

4.2 Transformer with Fixed Weights

For our second experiment, we run momentum, vectorwise and componentwise clipping, with the goal of estimating the full gradient for a transformer architecture *with fixed weights*. The weights are fixed at initialization. The setup is identical to the one described in [11], in particular Figure 1 therein: we use a simple transformer for the `PTB` dataset, and try three different batch sizes $\{64, 256, 1024\}$. The same experiment for the `Wikitext-2` setup from [11] can be found in Fig. 4.

Discussion. From Fig. 3, we observe two phenomena: in the long run, momentum attains the lowest error for estimating the full batch gradient. However, the initial decrease of the error is much faster for `CClip`, followed by `VClip`. This is important when using these estimates within a training setup such as (17), where we only do one iteration of gradient estimation, followed by an update of the weight (and hence a change in the full-batch gradient). Secondly, we observe that the difference in convergence speeds is most pronounced when the batch size increases. Hence, being robust to outliers seems not to be solvable only by increasing the batch size and thus decreasing the noise of the mini-batch gradient. In fact, the contrary seems to be the case. This is similar to the observations made in [11].

Acknowledgments

Umut Şimşekli's research is supported by the French government under management of Agence Nationale de la Recherche as part of the "Investissements d'avenir" program, reference ANR-19-P3IA-0001 (PRAIRIE 3IA Institute) and the European Research Council Starting Grant DYNASTY – 101039676.

References

[1] John T. Chu and Harold Hotelling. The moments of the sample median. *The Annals of Mathematical Statistics*, 26(4):593–606, 1955.

[2] Michael B. Cohen, Yin Tat Lee, Gary Miller, Jakub Pachocki, and Aaron Sidford. Geometric median in nearly linear time. In *Proceedings of the Forty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '16, page 9–21, New York, NY, USA, 2016. Association for Computing Machinery.

[3] Harald Cramér. *Mathematical Methods of Statistics (PMS-9), Volume 9*. Princeton University Press, 2016.

[4] Deepesh Data, Linqi Song, and Suhas Diggavi. Data encoding for byzantine-resilient distributed gradient descent. In *2018 56th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 863–870, 2018.

[5] David Donoho and Peter J. Huber. The notion of breakdown point. In *A Festschrift for Erich L. Lehmann*, Wadsworth Statist./Probab. Ser., pages 157–184. Wadsworth, Belmont, CA, 1983.

[6] Maurice R. Fréchet. Les éléments aléatoires de nature quelconque dans un espace distancié. 1948.

[7] Eduard Gorbunov, Marina Danilova, and Alexander Gasnikov. Stochastic optimization with heavy-tailed noise via accelerated gradient clipping. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 15042–15053. Curran Associates, Inc., 2020.

[8] Sai Praneeth Karimireddy, Lie He, and Martin Jaggi. Learning from history for byzantine robust optimization. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 5311–5319. PMLR, 18–24 Jul 2021.

[9] Sai Praneeth Karimireddy, Quentin Rebjock, Sebastian Stich, and Martin Jaggi. Error feedback fixes SignSGD and other gradient compression schemes. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 3252–3261. PMLR, 09–15 Jun 2019.

[10] Sarit Khirirat, Eduard Gorbunov, Samuel Horváth, Rustem Islamov, Fakhri Karray, and Peter Richtárik. Clip21: Error feedback for gradient clipping, 2023.

[11] Frederik Kunstner, Jacques Chen, Jonathan Wilder Lavington, and Mark Schmidt. Noise is not the main factor behind the gap between sgd and adam on transformers, but sign descent might be. In *The Eleventh International Conference on Learning Representations*, 2023.

[12] Hendrik P. Lopuhaa and Peter J. Rousseeuw. Breakdown points of affine equivariant estimators of multivariate location and covariance matrices. *The Annals of Statistics*, 19(1):229–248, 1991.

[13] Boris T. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964.

[14] Peter Richtárik, Igor Sokolov, and Ilyas Fatkhullin. EF21: A new, simpler, theoretically better, and practically faster error feedback. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021.

[15] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 1139–1147, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR.

[16] Yehuda Vardi and Cun-Hui Zhang. The multivariate L_1-median and associated data depth. *Proceedings of the National Academy of Sciences*, 97(4):1423–1426, 2000.

[17] Endre Weiszfeld and Frank Plastria. On the point for which the sum of the distances to n given points is minimum. *Ann. Oper. Res.*, 167(1):7–41, 2009.

[18] Jingzhao Zhang, Tianxing He, Suvrit Sra, and Ali Jadbabaie. Why gradient clipping accelerates training: A theoretical justification for adaptivity. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[19] Jingzhao Zhang, Sai Praneeth Karimireddy, Andreas Veit, Seungyeon Kim, Sashank Reddi, Sanjiv Kumar, and Suvrit Sra. Why are adaptive methods good for attention models? In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NeurIPS 2020, Red Hook, NY, USA, 2020. Curran Associates Inc.

[20] George K. Zipf. *Human Behaviour and the Principle of Least Effort*. Addison-Wesley, Oxford, UK, 1949.

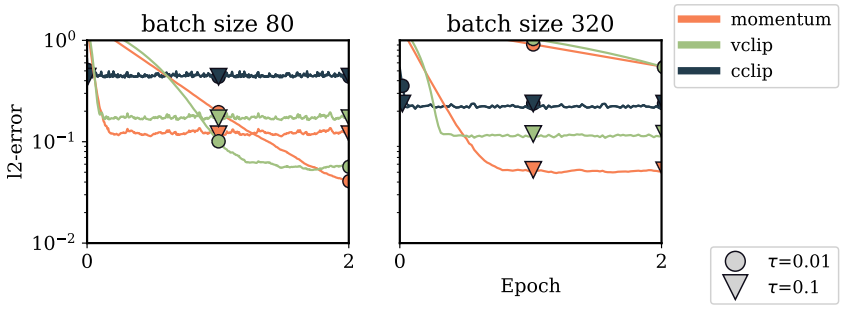

Figure 4: Encoder transformer on `Wikitext-2` dataset, with weights fixed at initialization. We use momentum, $\mathrm{clip}_{\tau,1}$ and $\mathrm{clip}_{\tau,2}$ to estimate the full-batch gradient.

A Auxiliary Lemmas

Lemma A.1. Let $m, g \in \mathbb{R}^d$ and $\tau > 0$. Then, $g + \mathrm{prox}_{\tau\|\cdot\|_1}(m-g) = m + \mathrm{clip}_{\tau,1}(g-m)$.

Proof. For $i \in [d]$, we have

$$\big[g + \mathrm{prox}_{\tau\|\cdot\|_1}(m-g)\big]_i = \begin{cases} g_i + (m_i - g_i - \tau) = m_i - \tau & \text{if } m_i - g_i > \tau, \\ g_i - (g_i - m_i - \tau) = m_i + \tau & \text{if } m_i - g_i < -\tau, \\ g_i & \text{else.} \end{cases}$$

Clearly, this is equal to $m + \mathrm{clip}_{\tau,1}(g-m)$. □

Lemma A.2. Let $m, g \in \mathbb{R}^d$ and $\tau > 0$. Then, $g + \mathrm{prox}_{\tau\|\cdot\|_2}(m-g) = m + \mathrm{clip}_{\tau,2}(g-m)$.

Proof. We have $\mathrm{prox}_{\tau\|\cdot\|_2}(m-g) = (1 - \frac{\tau}{\|m-g\|})(m-g)$ if $\|m-g\| \geq \tau$ else zero. Thus,

$$g + \mathrm{prox}_{\tau\|\cdot\|_2}(m-g) = \begin{cases} g + (1 - \frac{\tau}{\|m-g\|})(m-g) = m + \frac{\tau}{\|g-m\|}(g-m) & \text{if } \|m-g\| \geq \tau, \\ g & \text{else.} \end{cases}$$

Clearly, this is equal to $m + \mathrm{clip}_{\tau,2}(g-m)$. □

B Subgradient update for $\|\cdot\|_1$ and $\|\cdot\|_2$

Signed increment: If we choose $\mathcal{D} := \|\cdot\|_1$, update (6) gives

$$\boldsymbol{m}_{t+1} = \boldsymbol{m}_t + \tau\,\mathrm{sgn}(\boldsymbol{g} - \boldsymbol{m}_t),$$

where the sgn-operator can take any value in $[-1, 1]$ for coordinates where $\boldsymbol{g}_{t,i} = \boldsymbol{m}_{t,i}$.

Adaptive momentum: If we choose $\mathcal{D} := \|\cdot\|_2$, and if $\boldsymbol{g} \neq \boldsymbol{m}_t$, update (6) gives

$$\boldsymbol{m}_{t+1} = \boldsymbol{m}_t - \tau\frac{\boldsymbol{m}_t - \boldsymbol{g}}{\|\boldsymbol{m}_t - \boldsymbol{g}\|} = \big(1 - \frac{\tau}{\|\boldsymbol{m}_t - \boldsymbol{g}\|}\big)\boldsymbol{m}_t + \frac{\tau}{\|\boldsymbol{m}_t - \boldsymbol{g}\|}\boldsymbol{g}.$$

This can be seen as heavy-ball momentum with an adaptive momentum parameter.