Error Feedback Fixes SignSGD and other Gradient Compression Schemes

Sai Praneeth Karimireddy 1  Quentin Rebjock 1  Sebastian U. Stich 1  Martin Jaggi 1

Abstract
Sign-based algorithms (e.g. signSGD) have been proposed as a biased gradient compression technique to alleviate the communication bottleneck in training large neural networks across multiple workers. We show simple convex counter-examples where signSGD does not converge to the optimum. Further, even when it does converge, signSGD may generalize poorly when compared with SGD. These issues arise because of the biased nature of the sign compression operator.

We then show that using error-feedback, i.e. incorporating the error made by the compression operator into the next step, overcomes these issues. We prove that our algorithm (EF-SGD) with arbitrary compression operator achieves the same rate of convergence as SGD without any additional assumptions. Thus EF-SGD achieves gradient compression for free. Our experiments thoroughly substantiate the theory.

1. Introduction
Stochastic optimization algorithms (Bottou, 2010) which are amenable to large-scale parallelization, taking advantage of massive computational resources (Krizhevsky et al., 2012; Dean et al., 2012) have been at the core of significant recent progress in deep learning (Schmidhuber, 2015; LeCun et al., 2015). One such example is the signSGD algorithm and its variants, c.f. (Seide et al., 2014; Bernstein et al., 2018; Balles & Hennig, 2018).

To minimize a continuous (possibly) non-convex function $f : \mathbb{R}^d \to \mathbb{R}$, the classic stochastic gradient algorithm (SGD) (Robbins & Monro, 1951) performs iterations of the form

$$x_{t+1} := x_t - \gamma g_t,$$

(SGD)

where $\gamma \in \mathbb{R}$ is the step-size (or learning-rate) and $g_t$ is the stochastic gradient such that $\mathbb{E}[g_t] = \nabla f(x_t)$.

$^1$EPFL, Switzerland. Correspondence to: Sai Praneeth Karimireddy <sai.karimireddy@epfl.ch>.


Algorithm 1 EF-SIGNSGD (SIGNSGD with Error-Feedback)
1: Input: learning rate $\gamma$, initial iterate $x_0 \in \mathbb{R}^d$, $e_0 = 0$
2: for $t = 0, \ldots, T - 1$ do
3: $g_t := \text{stochasticGradient}(x_t)$
4: $p_t := \gamma g_t + e_t$  \textit{error correction}
5: $\Delta_t := (\|p_t\|_1/d) \text{sign}(p_t)$  \textit{compression}
6: $x_{t+1} := x_t - \Delta_t$  \textit{update iterate}
7: $e_{t+1} := p_t - \Delta_t$  \textit{update residual error}
8: end for

Methods performing updates only based on the sign of each coordinate of the gradient have recently gaining popularity for training deep learning models (Seide et al., 2014; Carlson et al., 2015; Wen et al., 2017; Balles & Hennig, 2018; Bernstein et al., 2018; Zaheer et al., 2018; Liu et al., 2018). For example, the update step of signSGD is given by:

$$x_{t+1} := x_t - \gamma \text{sign}(g_t).$$

(SIGNSGD)

Such sign-based algorithms are particularly interesting since they can be viewed through two lenses: as i) approximations of adaptive gradient methods such as ADAM (Balles & Hennig, 2018), and also as ii) communication efficient gradient compression scheme (Seide et al., 2014). However, we show that a severe handicap of sign-based algorithms is that they do not converge in general. To substantiate this claim, we present in this work simple convex counter-examples where signSGD cannot converge. The main reasons being that the sign operator loses information about, i.e. ‘forgets’, i) the magnitude, as well as ii) the direction of $g_t$. We present an elegant solution that provably fixes these problems of signSGD, namely algorithms with error-feedback.

Error-feedback. We demonstrate that the aforementioned problems of signSGD can be fixed by i) scaling the signed vector by the norm of the gradient to ensure the magnitude of the gradient is not forgotten, and ii) locally storing the difference between the actual and compressed gradient, and iii) adding it back into the next step so that the correct direction is not forgotten. We call our ‘fixed’ method EF-SIGNSGD (Algorithm 1).

In Algorithm 1, $e_t$ denotes the accumulated error from all quantization/compression steps. This residual error is added to the gradient step $\gamma g_t$ to obtain the corrected direction $p_t$. 

\[\phi \rightarrow \psi\]
When compressing $p_t$, the signed vector is again scaled by $\|p_t\|_1$ and hence does not lose information about the magnitude. Note that our algorithm does not introduce any additional parameters and requires only the step-size $\gamma$.

**Our contributions.** We show that naively using biased gradient compression schemes (such as e.g. SIGNSGD) can lead to algorithms which may not generalize or even converge. We show both theoretically and experimentally that simply adding error-feedback solves such problems and recovers the performance of full SGD, thereby saving on communication costs. We state our results for SIGNSGD to ease our exposition but our positive results are valid for general compression schemes, and our counterexamples extend to SIGNSGD with momentum, multiple worker settings, and even other biased compression schemes. More specifically our contributions are:

1. We construct a simple convex non-smooth counterexample where SIGNSGD cannot converge, even with the full batch (sub)-gradient and tuning the step-size. Another counterexample for a wide class of smooth convex functions proves that SIGNSGD with stochastic gradients cannot converge with batch-size one.

2. We prove that by incorporating error-feedback, SIGNSGD—as well as any other gradient compression schemes—always converge. Further, our analysis for non-convex smooth functions recovers the same rate as SGD, i.e. we get gradient compression for free.

3. We show that our algorithm EF-SIGNSGD which incorporates error-feedback approaches the linear span of the past gradients. Therefore, unlike SIGNSGD, EF-SIGNSGD converges to the max-margin solution in over-parameterized least-squares. This provides a theoretical justification for why EF-SIGNSGD can be expected to generalize much better than SIGNSGD.

4. We show extensive experiments on CIFAR10 and CIFAR100 using Resnet and VGG architectures demonstrating that EF-SIGNSGD indeed significantly outperforms SIGNSGD, and matches SGD both on test as well as train datasets while reducing communication by a factor of $\sim 64 \times$.

**2. Significance and Related Work**

**Relation to adaptive methods.** Introduced in (Kingma & Ba, 2015), ADAM has gained immense popularity as the algorithm of choice for adaptive stochastic optimization for its perceived lack of need for parameter-tuning. However since, the convergence (Reddi et al., 2018) as well the generalization performance (Wilson et al., 2017) of such adaptive algorithms has been called into question. Understanding when ADAM performs poorly and providing a principled ‘fix’ for these cases is crucial given its importance as the algorithm of choice for many researchers. It was recently noted by Balles & Hennig (2018) that the behavior of ADAM is in fact identical to that of SIGNSGD with momentum: More formally, the SIGNSGD algorithm (referred to as ‘signum’ by Bernstein et al. (2018; 2019)) adds momentum to the SIGNSGD update as:

$$\begin{align*}
    m_{t+1} &:= g_t + \beta m_t, \\
    x_{t+1} &:= x_t - \gamma \text{sign}(m_{t+1}).
\end{align*}$$

for parameter $\beta > 0$. This connection between signed methods and fast stochastic algorithms is not surprising since sign-based gradient methods were first studied as a way to speed up SGD (Riedmiller & Braun, 1993). Given their similarity, understanding the behavior of SIGNSGD and SIGNSGD can help shed light on ADAM.

**Relation to gradient compression methods.** As the size of the models keeps getting bigger, the training process can often take days or even weeks (Dean et al., 2012). This process can be significantly accelerated by massive parallelization (Li et al., 2014; Goyal et al., 2017). However, at these scales communication of the gradients between the machines becomes a bottleneck hindering us from making full use of the impressive computational resources available in today’s data centers (Chilimbi et al., 2014; Seide et al., 2014; Strom, 2015). A simple solution to alleviate this bottleneck is to compress the gradient and reduce the number of bits transmitted. While the analyses of such methods have largely been restricted to unbiased compression schemes (Alistarh et al., 2017; Wen et al., 2017; Wang et al., 2018), biased schemes which perform extreme compression practically perform much better (Seide et al., 2014; Strom, 2015; Lin et al., 2018)—often without any loss in convergence or accuracy. Of these, (Seide et al., 2014; Strom, 2015; Wen et al., 2017) are all sign-based compression schemes. Interestingly, all the practical works (Seide et al., 2014; Strom, 2015; Lin et al., 2018) use some form of error-feedback.

**Error-feedback.** The idea of error-feedback was, as far as we are aware, first introduced in 1-bit SGD (Seide et al., 2014; Strom, 2015). The algorithm 1-bit SGD is very similar to our EF-SIGNSGD algorithm, but tailored for the specific recurrent network studied there. Though not presented as such, the ‘momentum correction’ used in (Lin et al., 2018) is a variant of error-feedback. However the error-feedback is not on the vanilla SGD algorithm, but on SGD with momentum. Recently, Stich et al. (2018) conducted the first theoretical analysis of error-feedback for compressed gradient methods (they call it ‘memory’) in the strongly convex case. Our convergence results can be seen as an extension of theirs to the non-convex and weakly convex cases.
Generalization of deep learning methods. Deep networks are almost always over-parameterized and are known to be able to fit arbitrary data and always achieve zero training error (Zhang et al., 2017). This ability of deep networks to generalize well on real data, while simultaneously being able to fit arbitrary data has recently received a lot of attention (e.g. Soudry et al. (2018); Dinh et al. (2017); Zhang et al. (2018); Arpit et al. (2017); Kawaguchi et al. (2017)). SIGNSGD and ADAM are empirically known to generalize worse than SGD (Wilson et al., 2017; Balles & Hennig, 2018). A number of recent papers try close this gap for ADAM. Luo et al. (2019) show that by bounding the adaptive step-sizes in ADAM leads to closing the generalization gap. They require new hyper-parameters on top of ADAM to adaptively tune these bounds on the step-sizes. Chen & Gu (2019) interpolate between SGD and ADAM using a new hyper-parameter $\rho$ and show that using this can recover performance of SGD. Zaheer et al. (2018) introduce a new adaptive algorithm which is closer to Adagrad (Duchi et al., 2011). Similarly, well-tuned ADAM (where all the hyper-parameters and not just the learning rate are tuned) is also known to close the generalization gap (Gugger & Howard, 2018). In all of these algorithms, new hyper-parameters are introduced which essentially control the effect of the adaptivity. Thus they require additional tuning while the improvement upon traditional SGD is questionable. We are not aware of other work bridging the generalization gap in sign-based methods.

3. Counterexamples for SIGNSGD

In this section we study the limitations of SIGNSGD. Under benign conditions—for example if i) the function $f$ is smooth, and ii) the stochastic noise is gaussian or an extremely large batch-size is used (equal to the total number of iterations)—the algorithm can be shown to converge (Bernstein et al., 2018; 2019). However, we show that SIGNSGD does not converge under more standard assumptions. We demonstrate this first on a few pedagogic examples and later also for realistic and general sum-structured loss functions.

If we use a fixed step-size $\gamma \geq 0$, SIGNSGD does not converge even for simple one-dimensional linear functions. 

**Counterexample 1.** For $x \in \mathbb{R}$ consider the constrained problem

$$\min_{x \in [-1,1]} \left[f(x) := \frac{1}{4}x^2\right],$$

with minimum at $x^* = -1$. Assume stochastic gradients are given as (note that $f(x) = \frac{1}{4}(4x - x^2 - x^2)$)

$$g = \begin{cases} 4, & \text{with prob. } \frac{1}{4} \\ -1, & \text{with prob. } \frac{3}{4} \end{cases}$$

with $E[g] = \nabla f(x)$. 

For SGD with any step-size $\gamma$,

$$E_t[f(x_{t+1})] = \frac{1}{4}(x_t - \gamma E[g]) = f(x_t) - \frac{\gamma}{16}.$$ 

On the other hand, for SIGNSGD with any fixed $\gamma$,

$$E_t[f(x_{t+1})] = \frac{1}{4}(x_t - \gamma E[\text{sign}(g)]) = f(x_t) + \frac{\gamma}{8},$$

i.e. the objective function increases in expectation for $\gamma \geq 0$.

**Remark 1.** In the above example, we exploit that the sign operator loses track of the magnitude of the stochastic gradient. Also note that our noise is bimodal. The counterexamples for the convergence of ADAM (Reddi et al., 2018; Luo et al., 2019) also use similar ideas. Such examples were previously noted for SIGNSGD by (Bernstein et al., 2019).

In the example above the step-size $\gamma$ was fixed. However increasing batch-size or tuning the step-size may still allow convergence. Next we show that even with adaptive step-sizes (e.g. decreasing, or adaptively chosen optimal step-sizes) SIGNSGD does not converge. This even holds if the full (sub)-gradient is available (non-stochastic case).

**Counterexample 2.** For $x \in \mathbb{R}^2$ consider the following non-smooth convex problem with $x^* = (0,0)^T$:

$$\min_{x \in \mathbb{R}^2} \left[f(x) := \epsilon |x_1 + x_2| + |x_1 - x_2| \right],$$

for parameter $0 < \epsilon < 1$ and subgradient

$$g(x) = \text{sign}(x_1 + x_2) \cdot \epsilon \begin{pmatrix} 1 \\ 1 \end{pmatrix} + \text{sign}(x_1 - x_2) \begin{pmatrix} 1 \\ -1 \end{pmatrix}.$$ 

See Fig. 1. The iterates of SIGNSGD started at $x_0 = (1, 1)^T$ lie along the line $x_1 + x_2 = 2$. Note that for any $x$ s.t. $x_1 + x_2 > 0$, $\text{sign}(g(x)) = (1, -1)^T$, and hence $x_1 + x_2$ remains constant among the iterations of SIGNSGD. Consequently, for any step-size sequence $\gamma_t$, $f(x_t) \geq f(x_0)$.

**Remark 2.** In this example, we exploit the fact that the sign operator is a biased approximation of the gradient—it consistently ignores the direction $e = \epsilon (1, 1)^T$ (see Fig 1). Tuning the step-size would not help either.

One might wonder if the smooth-case is easier. Unfortunately, the previous example can easily be extended to show that SIGNSGD with stochastic gradients may not converge even for smooth functions.
Counterexample 3. For $x \in \mathbb{R}^2$ consider the following least-squares problem with $x^* = (0, 0)^T$:

$$
\min_{x \in \mathbb{R}^2} \left[ f(x) := (\langle a_1, x \rangle)^2 + (\langle a_2, x \rangle)^2 \right],
$$
where

$$
a_{1,2} := \pm (1, -1) + \epsilon (1, 1),
$$

for parameter $0 < \epsilon < 1$ and stochastic gradient $g(x) = \nabla_x ((a_1, x))^2$ with prob. $\frac{1}{2}$ and $g(x) = \nabla_x ((a_2, x))^2$ with prob. $\frac{1}{2}$. The stochastic gradient is then either $ca_1$ or $ca_2$ for some scalar $c$. Exactly as in the non-smooth case, for $x_0 = (1, 1)^T$, the sign of the gradient $\text{sign}(g) = (\pm (1, -1), 1)$. Hence SIGNSGD with any step-size sequence remains stuck along the line $x_1 + x_2 = 2$ and $f(x_t) \geq f(x_0)$ a.s.

We can generalize the above counter-example to arbitrary dimensions and loss functions.

Theorem 1. Suppose that scalar loss functions $\{l_i : \mathbb{R} \to \mathbb{R}\}_{i=1}^n$ and data-points $\{(a_i, x_i)\}_{i=1}^n \in \mathbb{R}^d$ for $d \geq 2$ satisfy: i) $f(x) := \sum_{i=1}^n l_i(\langle a_i, x \rangle)$ has a unique optimum at $x^*$, and ii) there exists $s \in [-1, 1]^d$ such that $\text{sign}(a_i) = \pm s$ for all $i$. Then SIGNSGD with batch-size 1 and stochastic gradients $g(x) = \nabla_x l_i((a_i, x))$ for $i$ chosen uniformly at random does not converge to $x^*$ a.s. for any adaptive sequence of step-sizes, even with random initialization.

4. Convergence of Compressed Methods

We show the rather surprising result that incorporating error-feedback is sufficient to ensure that the algorithm converges at a rate which matches that of SGD. In this section we consider a general gradient compression scheme.

Algorithm 2 EF-SGD (Compr. SGD with Error-Feedback)

1: Input: learning rate $\gamma$, compressor $C(\cdot)$, $x_0 \in \mathbb{R}^d$
2: Initialize: $e_0 = 0 \in \mathbb{R}^d$
3: for $t = 0, \ldots, T-1$ do
4: $g_t := \text{stochasticGradient}(x_t)$
5: $p_t := \gamma g_t + e_t$ \hspace{1cm} $\triangleright$ error correction
6: $\Delta_t := C(p_t)$ \hspace{1cm} $\triangleright$ compression
7: $x_{t+1} := x_t - \Delta_t$ \hspace{1cm} $\triangleright$ update iterate
8: $e_{t+1} := p_t - \Delta_t$ \hspace{1cm} $\triangleright$ update residual error
9: end for

We generalize the notion of a compressor from (Stich et al., 2018).

Assumption A (Compressor). An operator $C : \mathbb{R}^d \to \mathbb{R}^d$ is a $\delta$-approximate compressor over $Q$ for $\delta \in (0, 1]$ if

$$
\|C(x) - x\|_2^2 \leq (1 - \delta)\|x\|_2^2, \forall x \in Q.
$$

Note that $\delta = 1$ implies that $C(x) = x$. Examples of compressors include: i) the sign operator, ii) top-$k$ which selects $k$ coordinates with the largest absolute value while zero-ing out the rest (Lin et al., 2018; Stich et al., 2018), iii) $k$-PCA which approximates a matrix $X$ with its top $k$ eigenvectors (Wang et al., 2018). Randomized compressors satisfying the assumption in expectation are also allowed.

We now state a key lemma that shows that the residual errors maintained in Algorithm 2 do not accumulate too much.

Lemma 3 (Error is bounded). Assume that $\mathbb{E}[\|g_t\|^2] \leq \sigma^2$ for all $t \geq 0$. Then at any iteration $t$ of EF-SGD, the norm of the error $e_t$ in Algorithm 2 is bounded:

$$
\mathbb{E}[\|e_t\|^2] \leq \frac{4(1 - \delta)\gamma^2\sigma^2}{\delta^2}, \forall t \geq 0.
$$

If $\delta = 1$, then $\|e_t\| = 0$ and the error is zero as expected.

We employ standard assumptions of smoothness of the loss function and the variance of the stochastic gradient.

Assumption B (Smoothness). A function $f : \mathbb{R}^d \to \mathbb{R}$ is $L$-smooth if for all $x, y \in \mathbb{R}^d$ the following holds:

$$
|f(y) - f(x) - \langle \nabla f(x), y - x \rangle| \leq \frac{L}{2} \|y - x\|^2_2.
$$

Assumption C (Moment bound). For any $x$, our query for a stochastic gradient returns $g$ such that

$$
\mathbb{E}[g] = \nabla f(x) \quad \text{and} \quad \mathbb{E}[\|g\|^2] \leq \sigma^2.
$$

Given these assumptions, we can formally state our theorem followed by a sketch of the proof.

Theorem II (Non-convex convergence of EF-SGD). Let $\{x_t\}_{t \geq 0}$ denote the iterates of Algorithm 2 for any step-size $\gamma > 0$. Under Assumptions A, B, and C,

$$
\min_{t \in [T]} \mathbb{E}[\|\nabla f(x_t)\|^2] \leq \frac{2f_0}{\gamma(T + 1)} + \frac{\gamma L\sigma^2}{2} + \frac{4\gamma^2 L^2 \sigma^2(1 - \delta)}{\delta^2},
$$

with $f_0 := f(x_0) - f^*$.

Proof Sketch. Intuitively, the condition that $C(\cdot)$ is a $\delta$-approximate compressor implies that at each iteration a $\delta$-fraction of the gradient information is sent. The rest is added to $e_t$ to be transmitted later. Eventually, all the gradient information is transmitted—albeit with a delay which depends on $\delta$. Thus EF-SGD can intuitively be viewed as a delayed gradient method. If the function is smooth, the gradient does not change quickly and so the delay does not significantly matter.

More formally, consider the error-corrected sequence $\tilde{x}_t$ which represents $x_t$ with the ‘delayed’ information added:

$$
\tilde{x}_t := x_t - e_t.
$$

It satisfies the recurrence

$$
\tilde{x}_{t+1} = x_t - e_{t+1} - C(p_t) = x_t - p_t = \tilde{x}_t - \gamma g_t.
$$
If $\mathbf{x}_t$ was exactly equal to $\tilde{\mathbf{x}}_t$ (i.e., there was zero ‘delay’), then we could proceed with the standard proof of SGD. We instead rely on Lemma 3 which shows $\tilde{\mathbf{x}}_t \approx \mathbf{x}_t$ and on the smoothness of $f$ which shows $\nabla f(\mathbf{x}_t) \approx \nabla f(\tilde{\mathbf{x}}_t)$.

**Remark 4.** If we substitute $\gamma := \frac{1}{\sqrt{T+1}}$ in Theorem II, we get

$$\min_{t \in [T]} \mathbb{E}[\|\nabla f(\mathbf{x}_t)\|^2] \leq \frac{4(f(\mathbf{x}_0) - f^*) + \lambda_0^2 + 4L^2\sigma^2(1-\delta)}{2\sqrt{T+1}} + \frac{4L^2\sigma^2}{\delta\gamma(T+1)}.$$

In the above rate, the compression factor $\delta$ only appears in the higher order $O(1/T)$ term. For comparison, SGD under the exact same assumptions achieves

$$\min_{t \in [T]} \mathbb{E}[\|\nabla f(\mathbf{x}_t)\|^2] \leq \frac{2(f(\mathbf{x}_0) - f^*) + \lambda_0^2}{2\sqrt{T+1}}.$$

This means that after $T \geq O(1/\delta^2)$ iterations the second term becomes negligible and the rate of convergence catches up with full SGD—this is usually true after just the first few epochs. Thus we prove that compressing the gradient does not change the asymptotic rate of SGD.$^\dagger$

**Remark 5.** The use of error-feedback was motivated by our counter-examples for biased compression schemes. However, our rates show that even if using an unbiased compression (e.g. QSGD (Alistarh et al., 2017)), using error-feedback gives significantly better rates. Suppose we are given an unbiased compressor $C(\cdot)$ such that $\mathbb{E}[\mathcal{U}(\cdot)] = \mathbf{x}$ and $\mathbb{E}[\|\mathcal{U}(\mathbf{x})\|^2] \leq k\|\mathbf{x}\|^2$. Then without feedback, using standard analysis (e.g. (Alistarh et al., 2017)) the algorithm converges $k$ times slower:

$$\min_{t \in [T]} \mathbb{E}[\|\nabla f(\mathbf{x}_t)\|^2] \leq \frac{(f(\mathbf{x}_0) - f^*) + \lambda_0^2}{2\sqrt{T+1}}.$$

Instead, if we use $C(\mathbf{x}) = \frac{1}{k}\mathcal{U}(\mathbf{x})$ with error-feedback, we would achieve

$$\min_{t \in [T]} \mathbb{E}[\|\nabla f(\mathbf{x}_t)\|^2] \leq \frac{4(f(\mathbf{x}_0) - f^*) + \lambda_0^2 + 2L^2\sigma^2k^2}{2\sqrt{T+1}} + \frac{4L^2\sigma^2k^2}{T+1},$$

thereby pushing the dependence on $k$ into the higher order $O(1/T)$ term.

Our counter-examples showed that biased compressors may not converge for non-smooth functions. Below we prove that adding error-feedback ensures convergence under standard assumptions even for non-smooth functions.

**Theorem III (Non-smooth convergence of EF-SGD).** Let $\{\mathbf{x}_t\}_{t \geq 0}$ denote the iterates of Algorithm 2 for any step-size $\gamma > 0$ and define $\tilde{\mathbf{x}}_t := \frac{1}{T} \sum_{t=0}^{T} \mathbf{x}_t$. Given that $f$ is convex and Assumptions A and C hold,

$$\mathbb{E}[f(\mathbf{x}_t)] - f^* \leq \frac{\|\mathbf{x}_0 - \mathbf{x}^*\|^2}{2\gamma(T+1)} + \gamma\sigma^2 \left( \frac{1}{2} + \frac{2\sqrt{1-\delta}}{\delta} \right).$$

**Remark 6.** By picking the optimal $\gamma = O(1/\sqrt{T})$, we see that

$$\mathbb{E}[f(\mathbf{x}_t)] - f^* \leq \frac{\|\mathbf{x}_0 - \mathbf{x}^*\|\sigma}{2\sqrt{T+1}} \sqrt{1 + \frac{4\sqrt{1-\delta}}{\delta}}.$$

For comparison, the rate of convergence under the same assumptions for SGD is

$$\mathbb{E}[f(\mathbf{x}_t)] - f^* \leq \frac{\|\mathbf{x}_0 - \mathbf{x}^*\|\sigma}{2\sqrt{T+1}}.$$

For non-smooth functions, unlike in the smooth case, the compression quality $\delta$ appears directly in the leading term of the convergence rate. This is to be expected since we can no longer assume that $\nabla f(\tilde{\mathbf{x}}_t) \approx \nabla f(\mathbf{x}_t)$, which formed the crux of our argument for the smooth case.

**Remark 7.** Consider the top-1 compressor which just picks the coordinate with the largest absolute value, and zeroes out everything else. It is obvious that top-1 is a $\frac{1}{T}$-approximate compressor (cf. (Stich et al., 2018, Lemma A.1)). Running EF-SGD with $C$ as top-1 results in a greedy coordinate algorithm. This is the first result we are aware which shows the convergence of a greedy-coordinate type algorithm on non-smooth functions.

If the function is both smooth and convex, we can fall back to the analysis of (Stich et al., 2018) and so we won’t examine it in more detail here.

**Convergence of EF-SIGNSGD**

What do our proven rates imply for EF-SIGNSGD (Algorithm 1), the method of our interest here?

**Lemma 8** (Compressed sign). The operator $C(\mathbf{v}) := \frac{\|\mathbf{v}\|_1}{d} \text{sign}(\mathbf{v})$ is a $\frac{\|\mathbf{v}\|_2}{d\|\mathbf{v}\|_2}$ compressor.

Figure 2: The density $\phi(\cdot)$ for the stochastic gradients $g_t$ and the error-corrected stochastic gradients $g_t + e_t$ for VGG19 on CIFAR10 and batchsize 128. Minimum value is above 0.13. See Sec. 6 for setup.

We refer to the quantity $\phi(\mathbf{v})$ as the density of $\mathbf{v}$. If the vector $\mathbf{v}$ had only one non-zero element, the value of $\delta$ for EF-SIGNSGD could be as bad as $1/d$. However, in deep learning the gradients are usually dense and hence $\phi(\mathbf{v})$ is much larger (see Fig. 2). Note that for our convergence rates, it is not the density of the gradient $g_t$, which matters but the density of the error-corrected gradient $g_t + e_t$.\[\dagger\]

The astute reader would have observed that the asymptotic rate for EF-SGD is in fact 2 times slower than SGD. This is just for simplicity of presentation and can easily be fixed with a tighter analysis.
Faster convergence than SGD? (Kingma & Ba, 2015) and (Bernstein et al., 2018) note that different coordinates of the stochastic gradient \( g \) may have different variances. In standard SGD, the learning rate \( \gamma \) would be reduced to account for the maximum of these coordinate-wise variances since otherwise the path might be dominated by the noise in these sparse coordinates. Instead, using coordinate-wise learning rates like ADAM does, or using only the coordinate-wise sign of \( g \) as SIGNSGD does, might mitigate the effect of such ‘bad’ coordinates by effectively scaling down the noisy coordinates. This is purported to be the reason why ADAM and SIGNSGD can be faster than SGD on training data.

In \( \text{EF-SIGNSGD} \), the noise from the ‘bad’ coordinates gets accumulated in the error term \( e_t \) and is not forgotten or scaled down. Thus, if there are ‘bad’ coordinates whose variance slows down convergence of SGD, \( \text{EF-SIGNSGD} \) should be similarly slow. Confirming this, in a toy experiment with sparse noise (Appendix A.1), SGD and \( \text{SIGNSGD} \) converge at the same slower rate, whereas ADAM and \( \text{SIGNSGD} \) are known to generalize poorly compared with SGD (Wilson et al., 2017; Balles & Hennig, 2018), and (Balles & Hennig, 2018).

5. Generalization of SignSGD

So far our discussion has mostly focused on the convergence of the methods i.e. their performance on training data. However for deep-learning, we actually care about their performance on test data i.e. their generalization. It has been observed that the optimization algorithm being used significantly impacts the properties of the optima reached (Im et al., 2016; Li et al., 2018). For instance, ADAM and SIGNSGD are known to generalize poorly compared with SGD (Wilson et al., 2017; Balles & Hennig, 2018).

The proposed explanation for this phenomenon is that in an over-parameterized setting, SGD reaches the ‘max-margin’ solution whereas ADAM and SIGNSGD do not (Zhang et al., 2017; Wilson et al., 2017), and (Balles & Hennig, 2018). As with the issues in convergence, the issues of SIGNSGD with generalization also turn out to be related to the biased nature of the sign operator. We explore how error-feedback may also alleviate the issues with generalization for any compression operator.

5.1. Distance to gradient span

Like (Zhang et al., 2017; Wilson et al., 2017), we consider an over-parameterized least-squares problem

\[
\min_{x \in \mathbb{R}^d} \left[ f(x) := \|Ax - y\|_2^2 \right],
\]

where \( A \in \mathbb{R}^{n \times d} \) for \( d > n \) is the data matrix and \( y \in \{-1, 1\}^n \) is the set of labels. The set of solutions \( X^* := \{x : f(x) = 0\} \) of this problem forms a subspace in \( \mathbb{R}^d \). Of particular interest is the solution with smallest norm:

\[
\arg\min_{x \in X^*} \|x\|_2 = A^+ y = A^+ (AA^+)^{-1} y,
\]

as this corresponds to the maximum margin solution in the dual.

Maximizing margin is known to have a regularizing effect and is said to improve generalization (Valiant, 1984; Cortes & Vapnik, 1995).

The key property that SGD (with or without momentum) trivially satisfies is that the iterates always lie in the linear span of the gradients.

\[ \text{Lemma 9.} \quad \text{Given any over-parameterized least-squares problem, suppose that the iterates of the algorithm always lie in the linear span of the gradients and it converges to a 0 loss solution. This solution corresponds to the minimum norm/maximum margin solution.} \]

If we instead use a biased compressor (e.g. SIGNSGD), it is clear that the iterate may not lie in the span of the gradients. In fact it is easy to construct examples where this happens for SIGNSGD (Balles & Hennig, 2018), as well as top-\( k \) sparsification (Gunasekar et al., 2018), perhaps explaining the poor generalization of these schemes. Error-feedback is able to overcome this issue as well.

\[ \text{Theorem IV. Suppose that we run Algorithm 2 for } t \text{ iterations starting from } x_0 = 0. \text{ Let } G_t = [g_1^\top, \ldots, g_{t-1}^\top] \in \mathbb{R}^{d \times t} \text{ denote the matrix of the stochastic gradients and let } \Pi_{G_t} : \mathbb{R}^n \rightarrow \text{Im}(G_t) \text{ denote the projection onto the range of } G_t. \]

\[
\|x_t - \Pi_{G_t}(x_t)\|_2^2 \leq \|e_t\|_2^2.
\]

Here \( e_t \) is the error as defined in Algorithm 2. The theorem follows directly from observing that \( (x_{t+1} + e_{t+1}) = (x_0 + \sum_{i=0}^t \gamma g_i) \), and hence lies in the linear span of the gradients.

\[ \text{Remark 10. Theorem IV along with Lemma 3 implies that the iterates of Algorithm 2 are always close to the linear span of the gradients.} \]

\[
\|x_t - \Pi_{G_t}(x_t)\|_2^2 \leq \frac{4\gamma^2(1 - \delta)}{\delta^2} \max_{i \in [t]} \|g_i\|^2.
\]

This distance further reduces as the algorithm progresses since the the step-size \( \gamma \) is typically reduced.

5.2. Simulations

We compare the generalization of the four algorithms with full batch gradient: i) SGD ii) SIGNSGD, iii) SIGNSGDM, and iv) \( \text{EF-SIGNSGD} \). The data is generated as in (Wilson
We run experiments on deep networks to test the validity of our insights. The results confirm that i) EF-SIGNSGD with error feedback always outperforms the standard SIGNSGD and SIGNSGDM, ii) the generalization gap of SIGNSGD and SIGNSGDM vs. SGD gets larger for smaller batch sizes, iii) the performance of EF-SIGNSGD on the other hand is much closer to SGD, and iv) SIGNSGD behaves erratically when using small batch-sizes. Our code is available at github.com/epfml/error-feedback-SGD.

6. Experiments

We run experiments on deep networks to test the validity of our insights. The results confirm that i) EF-SIGNSGD with error feedback always outperforms the standard SIGNSGD and SIGNSGDM, ii) the generalization gap of SIGNSGD and SIGNSGDM vs. SGD gets larger for smaller batch sizes, iii) the performance of EF-SIGNSGD on the other hand is much closer to SGD, and iv) SIGNSGD behaves erratically when using small batch-sizes. Our code is available at github.com/epfml/error-feedback-SGD.

6.1. Experimental setup

All our experiments used the PyTorch framework (Paszke et al., 2017) and is randomly split into test and train. The step-size $\gamma$ and (where applicable) the momentum parameter $\beta$ are tuned to obtain fastest convergence, but the results are representative across parameter choices.

In all four cases (Fig. 3), the train loss quickly goes to 0. The distance to the linear span of gradients is quite large for SIGNSGD and SIGNSGDM. For EF-SIGNSGD, exactly as predicted by our theory, it first increases to a certain limit and then goes to 0 as the algorithm converges. The test error, almost exactly corresponding to the distance of the gradient, iii) (scaled) SIGNSGD has a much smaller gap.

Motivation and intuition: We experimentally compared the following four algorithms: i) SGD which is SGD with momentum, and iv) EF-SIGNSGD (Alg. 1). The scaled SIGNSGD performs the update

$$x_{t+1} := x_t - \gamma \frac{\|g_t\|_1}{d} \text{sign}(g_t).$$

We chose to include this in our experiments since we wanted to isolate the effects of error-feedback from that of scaling. Further we drop the unscaled SIGNSGD from our discussion here since it was observed to perform worse than the scaled version. As is standard in compression schemes (Seide et al., 2014; Lin et al., 2018; Wang et al., 2018), we apply our compression layer-wise. Thus the net communication for the (scaled) SIGNSGD and EF-SIGNSGD is $\sum_{i=1}^t (d_i + 32)$ bits where $d_i$ is the dimension of layer $i$, and $t$ is the total number of layers. If the number of layers is much larger than the number of layers, then the cost of the extra 32 bits is negligible—usually the number of parameters is three orders of magnitude more than the number of layers.

All algorithms are run for 200 epochs. The learning-rate is decimated at 100 epochs and then again at 150 epochs. The initial learning rate is tuned manually (see Appendix A) for all algorithms using batch-size 128. For the smaller batch-sizes, the learning-rate is proportionally reduced as suggested in (Goyal et al., 2017). The momentum parameter $\beta$ (where applicable) was fixed to 0.9 and weight decay was left to the default value of $5 \times 10^{-4}$.

Models: We use the VGG+BN+Dropout network on CIFAR-10 (VGG19) from (Simonyan & Zisserman, 2014) and Resnet+BN+Dropout network (Resnet18) from (He et al., 2016a). We adopt the standard data augmentation scheme and preprocessing scheme (He et al., 2016a,b).

6.2. Results

The results of the experiments for Resnet18 on Cifar100 are shown in Fig. 4 and Table 1. Results for VGG19 on Cifar10 are also similar and can be found in the Appendix. We make four main observations:

**EF-SIGNSGD is faster than SGD on train.** On the train dataset, both the accuracy and the losses (Fig. 6) is better for EF-SIGNSGD than for SGD for all batch-sizes and models (Fig. 7). In fact even SIGNSGD is faster than SGD on the train dataset on VGG19 (Fig. 7) for batch-size 128. As we note in Section 4, the result that the scaled sign

<table>
<thead>
<tr>
<th>Batch-size</th>
<th>SGD</th>
<th>SIGNSGD</th>
<th>SIGNSGDM</th>
<th>EF-SIGNSGD</th>
</tr>
</thead>
<tbody>
<tr>
<td>128</td>
<td>75.35</td>
<td>-2.21</td>
<td>-3.15</td>
<td>-0.92</td>
</tr>
<tr>
<td>32</td>
<td>76.22</td>
<td>-3.04</td>
<td>-3.57</td>
<td>-0.79</td>
</tr>
<tr>
<td>8</td>
<td>74.91</td>
<td>-36.35</td>
<td>-6.6</td>
<td>-0.64</td>
</tr>
</tbody>
</table>

Table 1: Generalization gap on CIFAR-100 using Resnet18 for different batch-sizes. For SGD we report the best mean test accuracy percentage, and for the other algorithms we report their difference to the SGD accuracy (i.e. the generalization gap). EF-SIGNSGD has a much smaller gap.
methods are also faster than SGD (and in fact faster than even the without feedback algorithms) overturns previously understood intuition (cf. (Kingma & Ba, 2015; Bernstein et al., 2018)) for why signSGD and other adaptive methods outperform SGD—i.e. restricting the effect of a some ‘bad’ coordinates with high variance may not be the main reason why sign based methods are faster than SGD on train.

**EF-SIGNSGD almost matches SGDM on test.** On the test dataset (Table 1), the accuracy and the loss is much closer to SGDM than the other sign methods across batch-sizes and models (Tables 3, 4). The generalization gap (both in accuracy and loss) reduces with decreasing batch-size. We believe this is because the learning-rate was scaled proportional to the batch-size and smaller learning-rates lead to smaller generalization gap, as noted in Remark 10.

**SIGNSGD performs poorly for small batch-sizes.** The performance of SIGNSGD is always worse than EF-SIGNSGD indicating that scaling is insufficient and that error-feedback is crucial for performance. Further all metrics (train and test, loss and accuracy) increasingly become worse as the batch-size decreases indicating that SIGNSGD is indeed a brittle algorithm. In fact for batch-size 8, the algorithm becomes extremely unstable.

**SIGNSGDM performs poorly on some datasets and for smaller batch-sizes.** We were surprised that the training performance of SIGNSGDM is significantly worse than even SIGNSGD on CIFAR-100 for batch-sizes 128 and 32. On CIFAR-10, on the other hand, SIGNSGDM manages to be faster than SGDM (though still slower than EF-SIGNSGD). We believe this may be due to SIGNSGDM being sensitive to the weight-decay parameter as was noted in (Bernstein et al., 2018). We do not tune the weight-decay parameter and leave it to its default value for all methods (including EF-SIGNSGD). Further the generalization gap of SIGNSGDM gets worse for decreasing batch-sizes with a significant 6.6% drop in accuracy for batch-size 8.

### 7. Conclusion

We study the effect of biased compressors on the convergence and generalization of stochastic gradient algorithms for non-convex optimization. We have shown that biased compressors if naively used can lead to bad generalization, and even non-convergence. We then show that using error-feedback all such adverse effects can be mitigated. Our theory and experiments indicate that using error-feedback, our compressed gradient algorithm EF-SGD enjoys the same rate of convergence as original SGD—thereby giving compression for free. We believe this should have a large impact in the design of future compressed gradient schemes for distributed and decentralized learning. Further, given the relation between sign-based methods and ADAM, we believe that our results will be relevant for better understanding the performance and limitations of adaptive methods. Finally, in this work we only consider the single worker case. Developing a practical and scalable algorithm for multiple workers is a fruitful direction for future work.
Acknowledgements

We are grateful to Tao Lin, Thijs Vogels, and Negar Foroutan for their help with the experiments. We also thank Jean-Baptiste Cordonnier, Konstantin Mishchenko, Jeremy Bernstein, and anonymous reviewers for their suggestions which helped improve our writeup. We acknowledge funding from SNSF grant 200021_175796, as well as a Google Focused Research Award.

References


