Optimization for Machine Learning

Lecture 8: Distributed Optimization III

Sebastian Stich

CISPA – `https://cms.cispa.saarland/optml24/`

June 11, 2024

Mid-term Exam 2024

Question 8: *5 points.* Consider the following implementation of Gradient Descent with momentum.

Algorithm 1 Gradient descent with momentum

Input: $\mathbf{x}_0 \in \mathbb{R}^d$, $T > 0$, $f: \mathbb{R}^d \to \mathbb{R}$, $\gamma \geq 0$, $\alpha \geq 0$.
Initialization: $\mathbf{m}_0 = \nabla f(\mathbf{x}_0)$
for $t = \{0, \ldots, T-1\}$ **do**
  $\mathbf{m}_{t+1} = (1-\alpha)\mathbf{m}_t + \alpha \nabla f(\mathbf{x}_t)$
  $\mathbf{x}_{t+1} = \mathbf{x}_t - \gamma \mathbf{m}_t$
end for
Output: $\mathbf{x}_T$

lecture !

In the lecture notes you found the following convergence guarantee for this algorithm when used to minimize a convex function $f: \mathbb{R}^d \to \mathbb{R}$ (if run with a good choice of parameters γ, α):

$$\frac{1}{T}\sum_{t=0}^{T-1} f(\mathbf{x}_t) - f^\star \leq \frac{\|\mathbf{x}_0 - \mathbf{x}^\star\|^2 L}{T}.$$

Here $f^\star = f(\mathbf{x}^\star)$, for $\mathbf{x}^\star$ a minimizer of f.

1) How can you fix the algorithm so that it outputs a point for which the convergence guarantee applies?

2) Analyze the time- and memory complexity of your proposed solution. (Big-$\mathcal{O}$ notation suffices.)

3) Is your proposed solution optimal (in order, i.e. ignoring constants)?
If yes, please argue why. If no, can you propose a more efficient solution?

☐ 0 ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5

Mid-term Exam 2024

Ideas: need $\frac{1}{T}\sum_{t=0}^{T-1} f(x_t)$

1) $\frac{1}{T}\sum_{t=0}^{T-1} f(x_t) = E\, f(x_\tau)$ where x_τ is chosen uniformly at random from $\{x_0 \ldots x_{T-1}\}$

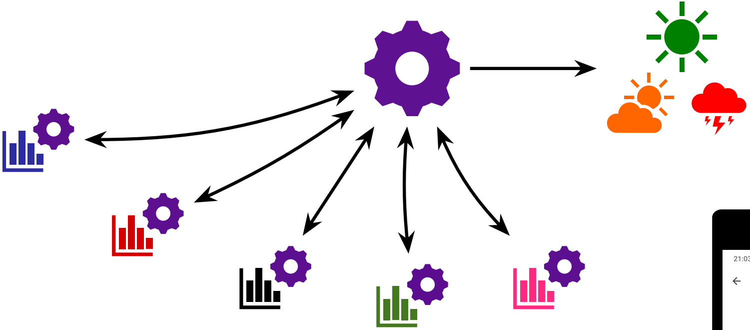
2) convex: $f(\bar{x}_T) \leq \frac{1}{T}\sum_{t=0}^{T-1} f(x_t)$, where $\bar{x}_T = \frac{1}{T}\sum_{i=1}^{T-1} x_t$

$\Rightarrow$ output $\bar{x}_T$

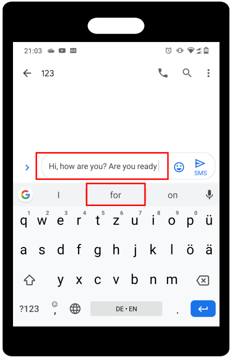
- Naive implementation: store all x_i, compute $\bar{x}_T$
  $O(d \cdot T)$ memory! $O(d \cdot T)$ time.
- Efficient implementation: $\bar{x}_t := \left(1 - \frac{1}{t}\right) \bar{x}_{t-1} + \frac{1}{t} x_t$ $O(d)$ memory $O(d)$ time.

Local SGD

Example: Federated Learning [MMR$^+$17, KMea21]



- private data stays on device
- server coordinates training and aggregates focused updates



Training Objective

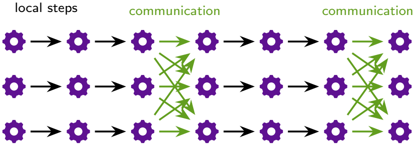
$$\min_{\mathbf{x}\in\mathbb{R}^d} \left[f(\mathbf{x}) := \frac{1}{n}\sum_{i=1}^{n} \underbrace{f_i(\mathbf{x})}_{\text{data } \mathcal{D}_i \text{ on client } i} \right] \qquad f_i(\mathbf{x}) = \begin{cases} \mathbb{E}_{\xi\sim\mathcal{D}_i} F(\mathbf{x},\xi) \\ \frac{1}{m}\sum_{j=1}^{m} f_{ij}(\mathbf{x}) \end{cases}$$

- Collaboratively solve **a (joint)** machine learning problem
- **efficiently**, in terms of:
  - computation (stochastic gradients, mini-batches),
  - communication (server $\leftrightarrow$ client).

Other very relevant scenarios:

• personalization • heterogenity • privacy • robustness

Local SGD



client i:

$x_t^{(i)}$

$x_{t+1}^{(i)} = x_t^{(i)} - \gamma\, g^{(i)}(x_t^{(i)})$

Notation/Setting

- n machines
- $f_i(\mathbf{x})$ denote the function (data) available locally at node i
- local gradient oracle $\mathbb{E}[\mathbf{g}^{(i)}(\mathbf{x})] = \nabla f_i(\mathbf{x})$, $\forall i \in [n]$, with bounded variance:

$$\mathbb{E}\left\|\mathbf{g}^{(i)}(\mathbf{x}) - \nabla f_i(\mathbf{x})\right\|^2 \leq \sigma^2$$

- let $\mathbf{x}_t^{(i)} \in \mathbb{R}^d$ denote the local iterate at node i, $\forall i \in [n]$

Local SGD

input

Input: $\mathbf{x}_0 \in \mathbb{R}^d$, $\mathbf{x}_0^{(i)} = \mathbf{x}_0$, $\forall i \in [n]$, stepsize γ, $\tau \geq 1$ (number of local steps)

At iteration t (in parallel on all nodes $i \in [n]$):

$\mathbf{g}_t^i = \mathbf{g}^{(i)}(\mathbf{x}_t^{(i)})$ (stochastic gradient locally on each node)

if $t+1$ is a multiple of τ :

$$\mathbf{x}_{t+1}^{(i)} = \frac{1}{n}\sum_{i=1}^{n}\left(\mathbf{x}_t^{(i)} - \gamma \mathbf{g}_t^i\right) \qquad \text{(global averaging)}$$

otherwise:

$$\mathbf{x}_{t+1}^{(i)} = \mathbf{x}_t^{(i)} - \gamma \mathbf{g}_t^i \qquad \text{(local step)}$$

independent

Discussion

expectation over the local data!

A very crucial assumption was: ✓

$$f_i(\mathbf{x}) = f_j(\mathbf{x}) \qquad \forall i \neq j$$

Example: $f_i(x) = x^2$

$f_j(x) = \underbrace{(x^2 + x)}_{\text{1/2 prob.}} - \underbrace{x}_{\text{1/2 prob.}}$

– this does not mean that stochastic gradients are the same!

NB: This was also a (hidden) assumption in our analysis of asynchronous SGD. We assumed each worker node can compute unbiased gradients of the objective $f(\mathbf{x})$.

Local SGD on heterogeneous data

Heterogeneous Functions

- We now want/need to drop the assumption that $f_i(\mathbf{x}) = f_j(\mathbf{x})$, $i \neq j$.

$$f(x) = f_1(x) + f_2(x)$$

$T=1 \Rightarrow$ local SGD $\hat{=}$ mini-batch SGD ✓

$T=100$?

Illustration I

$f(x) = \frac{1}{2}\left(f_1(x) + f_2(x)\right)$

Example: Consider $f_1(x) = \frac{1}{2}x^2$, $f_2(x) = (x-1)^2$, with optimal solution

$$\nabla f(x^\star) = \nabla f_1(x^\star) + \nabla f_2(x^\star) = x^\star + 2(x^\star - 1) = 0 \qquad \Leftrightarrow \qquad x^\star = \frac{2}{3}$$

- Note that $\nabla f_i(x^\star) \neq 0$, for each $i \in \{1, 2\}$.
- Consider an update step of mini-batch SGD (for arbitrary batch size), when initialized/starting at $x^\star$:

$=$ local SGD with $\tau = 1$

$$x^\star - \gamma \frac{1}{2} \underbrace{\left(\nabla f_1(x^\star) + \nabla f_2(x^\star)\right)}_{= 0} = x^\star$$

That is, $x^\star$ is a fix point!

Illustration II

- Consider local SGD with $\tau = 2$ local steps, when initialized/starting at $x^\star$.
- Node 1:

$f_1(x) = \frac{1}{2}x^2$

$$x_1^{(1)} = x^\star - \gamma \nabla f_1(x^\star) = x^\star - \gamma x^\star = x^\star(1-\gamma)$$
$$x_2^{(1)'} = x_1^{(1)} - \gamma \nabla f_1(x_1^{(1)}) = x^\star(1-\gamma)^2 = \frac{2}{3}(1-\gamma)^2$$

(Here $x_2^{(i)'}$ denotes the local solution before averaging.)

- Node 2: (by identical calculations, left as exercise)

$$x_2^{(2)'} = 1 + (x^\star - 1)(1-2\gamma)^2 = 1 - \frac{1}{3}(1-2\gamma)^2$$

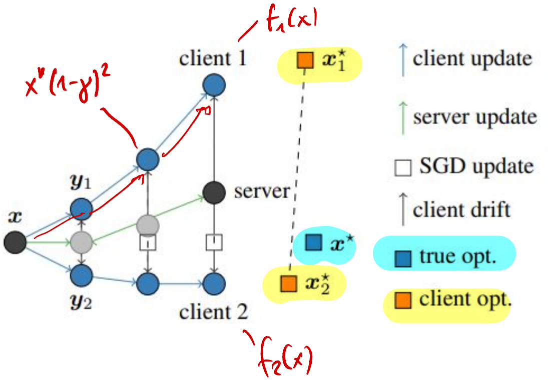
- Therefore

$$\bar{x}_2 = \frac{1}{2}\left(x_2^{(1)'} + x_2^{(2)'}\right) = \frac{2}{3} - \frac{\gamma^2}{3} \neq \frac{2}{3}$$

$x^\star$ is not a fix point, for any $\gamma > 0$.

Illustration III

Data-dissimilarity causes drift when doing local steps.



Measuring Data-Dissimilarity

$f_i(x) \neq f_j(x)$

Definition (Dissimilarity ζ^2)

(The smallest) parameter ζ^2 such that for all $\mathbf{x} \in \mathbb{R}^d$:

$$\frac{1}{n} \sum_{i=1}^{n} \|\nabla f_i(\mathbf{x}) - \nabla f(\mathbf{x})\|^2 \leq \zeta^2 \,.$$

Similar to the definition of the variance, but now we measure the inter-worker variance.

- $\zeta^2 = 0$ if $f_i = f_j \quad \forall i,j$
- ζ^2 can be very large!

Convergence

Theorem (Heterogeneous Case, [KLB+20])

Let $f_i \colon \mathbb{R}^d \to \mathbb{R}$ be L-smooth, $\forall i \in [n]$ and the function heterogenity bounded by ζ^2, with $\Delta = f(\mathbf{x}_0) - f^\star$. Then for there exists a stepsize $\gamma \leq \gamma_{\text{crit}} := \frac{1}{10L\tau}$ such that after T steps (that is, T/τ communication rounds) of Local SGD it holds

$$\min_{t \leq T} \mathbb{E} \left\| \nabla f(\bar{\mathbf{x}}_t) \right\|^2 = \mathcal{O}\left(\frac{\Delta L \tau}{T} + \underbrace{\left(\frac{L\Delta(\tau\zeta + \sqrt{\tau}\sigma)}{T} \right)^{2/3}} + \frac{\sqrt{L\Delta\sigma^2}}{\sqrt{Tn}} \right),$$

with $\bar{\mathbf{x}}_t := \frac{1}{n} \sum_{i=1}^n \mathbf{x}_t^{(i)}$.

✓ (the same as for mini-batch!)

(the same!)

Discussion

x^* is not a fix-point!

- $\zeta^2 > 0$ slows down the convergence.
- The dependency on ζ is optimal [WPS20].
- In general, the convergence is slower than for mini-batch SGD.

Why does then Local SGD behave well on many practical problems?

- Speedup can be proven under different similarity assumptions.

Proof

We will prove a new (Difference) Lemma, the rest of the proof will follow exactly the same pattern as for the homogeneous case.

Lemma (Difference)

For $\gamma \leq \gamma_{\text{crit}} = \frac{1}{20L\tau}$, with the notation for $R_t = \frac{1}{n}\sum_{i=1}^{n} \left\|\bar{\mathbf{x}}_t - \mathbf{x}_t^{(i)}\right\|^2$, it holds

$$\mathbb{E}R_t \leq \frac{1}{10L^2\tau} \sum_{j=(t-1)-k}^{t-1} \mathbb{E}\left\|\nabla f(\bar{\mathbf{x}}_j)\right\|^2 + 5\gamma^2\sigma^2\tau + 40\gamma^2\tau^2\zeta^2$$

where $(t-1)-k$ denotes the index of the last communication round ($k \leq \tau - 1$).

Proof II

Plug the new (Difference) bound into (Decrease), re-arrange, divide by γ, divide by T, and sum over $t = 0, \ldots, T-1$... (left as exercise!)

We will end up with:

$$\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\left\|\nabla f(\bar{\mathbf{x}}_t)\right\|^2 = \mathcal{O}\left(\frac{\Delta}{\gamma T} + \gamma L \frac{\sigma^2}{n} + \gamma^2 L^2 (\tau^2 \zeta^2 + \tau \sigma^2)\right)$$

Now use Exercise 6.1.

Proof of Lemma (Difference) I

For the proof of the difference lemma, we need to be more careful at inequality $(*)$. Recall the inequality $(*)$ from Lecture 7:

$$\mathbb{E}R_{t+1} \leq \left(1 + \frac{1}{\tau}\right)\mathbb{E}R_t + \frac{2\tau\gamma^2}{n}\sum_{i=1}^{n}\mathbb{E}\left(2\left\|\nabla f_i(\bar{\mathbf{x}}_t)\right\|^2 + 2\left\|\nabla f_i(\mathbf{x}_t^{(i)}) - \nabla f_i(\bar{\mathbf{x}}_t)\right\|^2\right) + \gamma^2\sigma^2$$

Previously we used $f_i = f_j$, to simplify. Instead, note that:

$$\begin{aligned}
\tfrac{1}{n}\textstyle\sum_{i=1}^{n}\left\|\nabla f_i(\bar{\mathbf{x}}_t)\right\|^2 &= \tfrac{1}{n}\textstyle\sum_{i=1}^{n}\left(\left\|\underbrace{\nabla f_i(\bar{\mathbf{x}}_t) - \nabla f(\bar{\mathbf{x}}_t)}_{a} + \underbrace{\nabla f(\bar{\mathbf{x}}_t)}_{b}\right\|^2\right) \\
&\leq \tfrac{2}{n}\textstyle\sum_{i=1}^{n}\left(\underbrace{\left\|\nabla f_i(\bar{\mathbf{x}}_t) - \nabla f(\bar{\mathbf{x}}_t)\right\|^2}_{\le \zeta^2} + \left\|\nabla f(\bar{\mathbf{x}}_t)\right\|^2\right) \\
&\leq 2\zeta^2 + 2\left\|\nabla f(\bar{\mathbf{x}}_t)\right\|^2
\end{aligned}$$

where we used $\|\mathbf{a}+\mathbf{b}\|^2 \leq 2\|\mathbf{a}\|^2 + 2\|\mathbf{b}\|^2$ for vectors $\mathbf{a}, \mathbf{b} \in \mathbb{R}^d$.

Proof of Lemma (Difference) II

Therefore;

$$\mathbb{E}R_{t+1} \le \left(1+\frac{1}{\tau}\right)\mathbb{E}R_t + \frac{2\tau\gamma^2}{n}\sum_{i=1}^{n}\mathbb{E}\left(2\left\|\nabla f_i(\bar{\mathbf{x}}_t)\right\|^2 + 2\underbrace{\left\|\nabla f_i(\mathbf{x}_t^{(i)}) - \nabla f_i(\bar{\mathbf{x}}_t)\right\|^2}_{\text{smoothen}}\right) + \gamma^2\sigma^2$$

$$\le \left(1+\frac{1}{\tau}\right)\mathbb{E}R_t + \frac{2\tau\gamma^2}{n}\sum_{i=1}^{n}\mathbb{E}\left(2\left(2\zeta^2 + 2\left\|\nabla f(\bar{\mathbf{x}}_t)\right\|^2\right) + 2L^2\left\|\mathbf{x}_t^{(i)} - \bar{\mathbf{x}}_t\right\|^2\right) + \gamma^2\sigma^2$$

$$\le \left(1+\frac{1}{\tau}\right)\mathbb{E}R_t + 8\tau\gamma^2\zeta^2 + 8\tau\gamma^2\mathbb{E}\left\|\nabla f(\bar{\mathbf{x}}_t)\right\|^2 + 4\tau\gamma^2L^2\mathbb{E}R_t + \gamma^2\sigma^2$$

$\gamma^2 \le \frac{1}{400 \cdot L^2\tau^2} \cdot 8\tau$

And with $\gamma \le \frac{1}{20L\tau}$:

$$\mathbb{E}R_{t+1} \le \left(1+\frac{1}{\tau}\right)\mathbb{E}R_t + 8\tau\gamma^2\zeta^2 + \frac{1}{50L^2\tau}\mathbb{E}\left\|\nabla f(\bar{\mathbf{x}}_t)\right\|^2 + \frac{1}{100\tau}\mathbb{E}R_t + \gamma^2\sigma^2$$

$$\le \left(1+\frac{3}{2\tau}\right)\mathbb{E}R_t + 8\tau\gamma^2\zeta^2 + \frac{1}{50L^2\tau}\mathbb{E}\left\|\nabla f(\bar{\mathbf{x}}_t)\right\|^2 + \gamma^2\sigma^2$$

Proof of Lemma (Difference) II

The lemma now follows by unrolling for k, at most $k \leq \tau - 1$ steps:

$$\mathbb{E} R_t \leq \left(1 + \frac{3}{2\tau}\right)^k \underbrace{\mathbb{E} R_{t-k}}_{=0!} + \sum_{i=0}^{k} \left(1 + \frac{3}{2\tau}\right)^i \left(8\tau\gamma^2\zeta^2 + \frac{1}{50L^2\tau}\mathbb{E}\left\|\nabla f(\bar{\mathbf{x}}_i)\right\|^2 + \gamma^2\sigma^2\right)$$

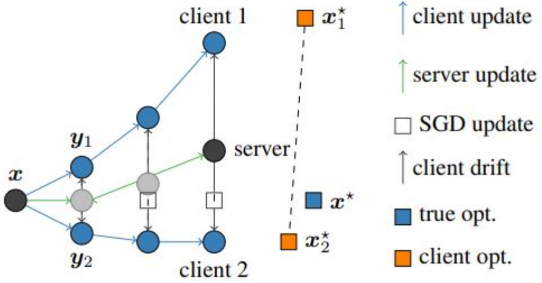
Note that $\left(1 + \frac{3}{2\tau}\right)^j \leq e^{\frac{3}{2\tau}j} \leq e^{\frac{3}{2}} \leq 5$ for all $0 \leq j \leq \tau$. Therefore:

$$\begin{aligned}
\mathbb{E} R_t &\leq 0 + \sum_{i=0}^{k} 5 \left(8\tau\gamma^2\zeta^2 + \frac{1}{50L^2\tau}\mathbb{E}\left\|\nabla f(\bar{\mathbf{x}}_i)\right\|^2 + \gamma^2\sigma^2\right) \\
&\leq 40\tau\gamma^2\zeta^2 + \sum_{i=0}^{k}\frac{1}{10L^2\tau}\mathbb{E}\left\|\nabla f(\bar{\mathbf{x}}_i)\right\|^2 + 5\gamma^2\sigma^2 \,.
\end{aligned}$$

↑ typo, this Σ must stay (as in the lemma statement)

Drift Correction

Client Drift

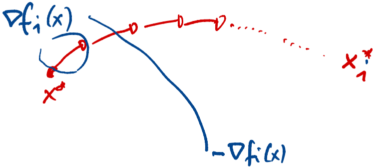


$$\frac{1}{n}\sum_{i=1}^{n}\|\nabla f_i(\mathbf{x}) - \nabla f(\mathbf{x})\|^2 \leq \zeta^2.$$

$$\min_{t\leq T} \mathbb{E}\,\|\nabla f(\bar{\mathbf{x}}_t)\|^2 = \mathcal{O}\left(\frac{\Delta L\tau}{T} + \left(\frac{L\Delta(\tau\zeta + \sqrt{\tau}\sigma)}{T}\right)^{2/3} + \frac{\sqrt{L\Delta\sigma^2}}{\sqrt{Tn}}\right),$$

Q: How can we fix the drift issue?

Main Idea



Bias correction in local update

$$\mathbf{x}^i_{t+1} = \mathbf{x}^i_t - \gamma\Big(\underbrace{\mathbf{g}^i_t}_{\text{normal update}} \underbrace{- \underbrace{\mathbf{c}^i_t}_{\substack{\mathbf{c}^i_t \approx \nabla f_i(\bar{\mathbf{x}}^i_t) \\ \text{local drift}}} + \underbrace{\mathbf{c}_t}_{\substack{\mathbf{c}_t \approx \nabla f(\bar{\mathbf{x}}_t) \\ \text{global drift}}}}_{\textbf{drift correction}}\Big)$$

- correction does not depend on local steps and is unbiased!

Similarities to variance reduction in server-only optimization, like in SVRG, SAGA, SCSG, etc. (see next lecture).

Implementation Sketch: Estimate Bias

- if n small, SVRG/SAGA-type correction **[SCAFFOLD]**
  $\mathbf{c}_t^i = \mathbf{g}_t^i$, $\mathbf{c}_t = \frac{1}{n}\sum_{i=1}^{n} \mathbf{c}_t^i$
- if n is huge, SCSG-type correction **[Mime]**
  $\mathbf{c}_t^i = \mathbf{g}_t^i$, $\mathbf{c}_t = \frac{1}{|\mathcal{S}_t|}\sum_{i \in \mathcal{S}_t} \mathbf{c}_t^i$, for active clients $\mathcal{S}_t \subset [n]$

  sample a set of clients

Here $\mathbf{g}_t^i$ denotes a (stochastic (possibly mini-batch) or full batch) gradient.

Theoretical Results with Bias Correction

algorithm	rounds
mini-batch SGD batch size τ	$\mathcal{O}\left(\frac{\sigma^2}{n\tau\mu\epsilon} + \frac{L}{\mu}\log\frac{1}{\epsilon}\right)$
SCAFFOLD τ local steps + proper init	$\tilde{\mathcal{O}}\left(\frac{\sigma^2}{n\tau\mu\epsilon} + \frac{L}{\mu}\log\frac{1}{\epsilon}\right)$

\"

$\Rightarrow$ does not depend on ζ^2 !

Scaffold

Local iteration:

stochastic gradient

$$\mathbf{x}_{t+1}^i = \mathbf{x}_t^i - \gamma_\ell \left(\mathbf{g}_t^i - \mathbf{c}_t^i + \mathbf{c}_t\right)$$

Algorithm 1 Scaffold (Karimireddy et al., 2019)

```
Initialize x_0, c^i, i = 1, ..., n
for r = 0, 1, ..., R-1 do
    sample clients S_t ⊆ [n]
    for on client i ∈ S_t in parallel do
        initialize local model y_0^i = x_r.
        for k = 0, ... K-1 local steps do
            y_{k+1}^i = y_k^i - γ_ℓ (g^i(y_k^i) - c^i + c)
        end for
        Option I: c^i = g^(i)(x_r)
        Option II: c^i = c^i - c + 1/(K γ_ℓ) (x_t - y_K^i)
    end for
    x_{r+1} = x_r - γ (1/|S_t|) Σ_{i∈S_t} (y_K^i - x_r)
    c = c + (1/n) Σ_{i∈S} (c^i - c_old^i)
end for
```

note: can also be done in local SGD

better!

- client sampling
- two stepsizes γ, γ_ℓ
- cheap iteration cost
- clients need to preserve state $\mathbf{c}^i$
- (difficult to analyze)

Mime (state-free version of Scaffold, with momentum)

$$\mathbf{x}_{t+1}^i = \mathbf{x}_t^i - \gamma_\ell \left((1-\beta)\left(\mathbf{g}_t^i - \mathbf{c}^i + \mathbf{c}\right) + \beta \mathbf{m} \right)$$

Algorithm 2 Mime (Karimireddy et al., 2020)

1: Initialize $\mathbf{x}_0$, momentum $\mathbf{m}_0$,
2: **for** $r = 0, 1, \ldots, R-1$ **do**
3: sample clients $\mathcal{S}_t \subseteq [n]$
4: $\mathbf{c}_r = \frac{1}{|\mathcal{S}_t|} \sum_{i \in \mathcal{S}_t} \nabla f_i(\mathbf{x}_r)$
5: **for** on client $i \in \mathcal{S}_t$ in parallel **do**
6: initialize local model $\mathbf{y}_0^i = \mathbf{x}_r$.
7: **for** $k = 0, \ldots K-1$ local steps **do**
8: $\mathbf{u}_k^i = \nabla f_i(\mathbf{y}_k^i) - \nabla f_i(\mathbf{x}_r) + \mathbf{c}_r$
9: $\mathbf{y}_{k+1}^i = \mathbf{y}_k^i - \gamma_\ell \left((1-\beta)\mathbf{u}_t^i + \beta \mathbf{m}_r \right)$
10: **end for**
11: **end for**
12: $\mathbf{x}_{r+1} = \mathbf{x}_r - \gamma \frac{1}{|\mathcal{S}_t|} \sum_{i \in \mathcal{S}_t} (\mathbf{y}_K^i - \mathbf{x}_r)$
13: $\mathbf{m}_{r+1} = \mathbf{m}_r + (1-\beta)\mathbf{c}_r + \beta \mathbf{m}_r$
14: **end for**

- client sampling
- two stepsizes γ, γ_ℓ
- requires two rounds of communication
  - MimeLite skips line 4, and uses a cheaper update on line 8
- server-only momentum (momentum is fixed during local steps)
- (difficult to analyze)

Numerical Experiment

- drift correction accelerates training on non-IID data!

		non-IID data				IID data	
	Epochs	0% similarity (sorted) Num. of rounds	Speedup	10% similarity Num. of rounds	Speedup	100% similarity (i.i.d.) Num. of rounds	Speedup
SGD	1	317	(1×)	365	(1×)	416	(1×)
SCAFFOLD	1	**77**	(4.1×)	62	(5.9×)	60	(6.9×)
	5	152	(2.1×)	20	(18.2×)	10	(41.6×)
FedAvg	1	258	(1.2×)	74	(4.9×)	83	(5×)
local SGD	5	428	(0.7×)	34	(10.7×)	10	(41.6×)

less rounds with drift correction

no drift correction needed

Communication rounds to reach 0.5 test accuracy for logistic regression on EMNIST.

Discussion

- We have seen two methods to reduce client drift. Both converge as fast as mini-batch SGD (in the worst case), but not faster (in the worst-case).
- (Stateless algorithms matters in applications with huge number of clients)

Beyond mini-batch SGD!

- in order to prove faster convergence than mini-batch SGD, it has been helpful to define additional similarity conditions:

Definition (Hessian similarity)

$$\left\|\nabla^2 f_i(\mathbf{x}) - \nabla^2 f(\mathbf{x})\right\| \leq \delta \qquad \forall i$$

Lecture 8 Recap

- Federated Learning
  - studied the convergence properties of local SGD
  - in practice: FedAvg

local SGD + client sampling

- Convergence proof for local SGD in both:
  - homogeneous (Lecture 7), and
  - heterogeneous setting (Lecture 8)
- Data-dissimilarity can cause drift.
  - example
  - data-dissimilarity measure ζ^2 (δ)
- Carefully designed methods can mitigate drift.
  - Scaffold, Mime (without proof)

Bibliography I

Anastasia Koloskova, Nicolas Loizou, Sadra Boreiri, Martin Jaggi, and Sebastian U. Stich.
A unified theory of decentralized SGD with changing topology and local updates.
In *37th International Conference on Machine Learning (ICML)*. PMLR, 2020.

Peter Kairouz, H. Brendan McMahan, and et al.
Advances and open problems in federated learning.
Foundations and TrendsÂ® in Machine Learning, 14(1–2):1–210, 2021.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas.
Communication-efficient learning of deep networks from decentralized data.
In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1273–1282, 2017.

Blake Woodworth, Kumar Kshitij Patel, and Nathan Srebro.
Minibatch vs local SGD for heterogeneous distributed learning.
arXiv preprint arXiv:2006.04735, 2020.