ProxSkip: Yes! Local Gradient Steps Provably Lead to Communication Acceleration! Finally!

Peter Richtárik
Workshop on Federated Learning
Lagrange Mathematics and Computation Research Center
April 25, 2022

Konstantin Mishchenko, Grigory Malinovsky, Sebastian Stich, Peter Richtárik
ProxSkip: Yes! Local Gradient Steps Provably Lead to Communication Acceleration! Finally!
arXiv:2202.09357, 2022
Please accept our apologies, our excitement apparently spilled over into the title. If we were to choose a more scholarly title for this work, it would be **ProxSkip: Breaking the Communication Barrier of Local Gradient Methods**.
Coauthors

Konstantin Mishchenko
Grigory Malinovsky
Sebastian Stich
Outline of the Talk

1. Introduction
2. Consensus Reformulation
3. Proximal Gradient Descent
4. ProxSkip: Algorithm
5. ProxSkip: Theory
6. Experiments
7. Extensions
Part 1
Introduction
Gradient Descent
Gradient Descent

$$\min_{x \in \mathbb{R}^d} f(x)$$

$$x_{t+1} = x_t - \gamma \nabla f(x_t)$$

Stepsize / Learning rate

$$d = 2$$
Distributed Gradient Descent
Federated Training of a Supervised Machine Learning Model

\[
\min_{x \in \mathbb{R}^d} f(x) \overset{\text{def}}{=} \frac{1}{n} \sum_{i=1}^{n} f_i(x)
\]

Loss on local data \(D_i\) stored on device \(i\)

\[
f_i(x) = \mathbb{E}_{\xi \sim D_i} f_{i,\xi}(x)
\]

The datasets \(D_1, \ldots, D_n\) can be arbitrarily heterogeneous
Distributed Gradient Descent

Optimization problem:

\[
\min_{x \in \mathbb{R}^d} f(x) \overset{\text{def}}{=} \frac{1}{n} \sum_{i=1}^{n} f_i(x)
\]

\[
x_{t+1} = x_t - \gamma \nabla f(x_t)
\]

\[
= x_t - \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(x_t)
\]

\[
f(x) = \frac{1}{n} \sum_{i=1}^{n} f_i(x_t)
\]

\(d\)-dimensional gradient computed by machine \(i\)
Distributed Gradient Descent

(Each worker performs 1 GD step using its local function, and the results are averaged)

**Optimization problem:**
\[
\min_{x \in \mathbb{R}^d} f(x) \overset{\text{def}}{=} \frac{1}{n} \sum_{i=1}^{n} f_i(x)
\]

**Worker 1**
- Receive \( x_t \) from the server
- \( x_{1,t} = x_t \)
- \( x_{1,t+1} = x_{1,t} - \gamma \nabla f_1(x_{1,t}) \)

**Worker 2**
- Receive \( x_t \) from the server
- \( x_{2,t} = x_t \)
- \( x_{2,t+1} = x_{2,t} - \gamma \nabla f_2(x_{2,t}) \)

**Worker 3**
- Receive \( x_t \) from the server
- \( x_{3,t} = x_t \)
- \( x_{3,t+1} = x_{3,t} - \gamma \nabla f_3(x_{3,t}) \)

**Server**
- Broadcast \( x_{t+1} \) to the workers
- \( x_{t+1} = \frac{1}{3} \sum_{i=1}^{3} x_{i,t+1} \)
Distributed Local Gradient Descent
Distributed **Local** Gradient Descent

(Each worker performs $K$ GD steps using its local function, and the results are averaged)

### Optimization problem:

$$\min_{x \in \mathbb{R}^d} f(x) \overset{\text{def}}{=} \frac{1}{n} \sum_{i=1}^{n} f_i(x)$$
From GD to Local GD

1847
Gradient Descent (GD)
Compte Rendu à l'Académie des Sciences
(L. A. Cauchy)

1995
Local GD Proposed
Parallel Gradient Distribution in Unconstrained Optimization
(O. L. Mangasarian)

2017
Federated Averaging: Local GD Plays a Key Role in Federated Learning
Communication-efficient Learning of Deep Networks from Decentralized Data
(H. B. McMahan et al)

2020
First General Theory for Local GD
First Analysis of Local GD on Heterogeneous Data
(Khaled, Mishchenko & R)
What do the Local Steps do?

Plot taken from:

L2-regularized logistic regression
LibSVM mushrooms dataset

Ahmed Khaled, Konstantin Mishchenko, Peter Richtárik
First Analysis of Local GD on Heterogeneous Data
NeurIPS 2019 Workshop on Federated Learning for Data Privacy and Confidentiality, 2019
Linearly Converging Local GD Methods
Local GD with GD-like (=Linear) Convergence

SCAFFOLD
*Scaffold: Stochastic Controlled Averaging for Federated Learning*
(Karimireddy, Kale, Mohri, Reddi, Stich, Suresh)

S-Local-GD
*Local SGD: Unified Theory and New Efficient Methods*
(Gorbunov, Hanzely & R)

FedLin
*Linear Convergence in Federated Learning . . .*
(Mitra, Jaafar, Pappas, Hassani)
Key Theoretical Problem in Federated Learning

Local GD and its variants are of key importance in federated learning. They work better than GD in practice in terms of communication efficiency.

Why?

Do Multiple Local GD Steps Provably Help? No Current Theoretical Result Supports This.
### Federated Learning: ProxSkip vs Baselines

**Table 1.** The performance of federated learning methods employing multiple local gradient steps in the strongly convex regime.

<table>
<thead>
<tr>
<th>method</th>
<th># local steps per round</th>
<th># floats sent per round</th>
<th>stepsize on client $i$</th>
<th>linear rate?</th>
<th># rounds</th>
<th>rate better than GD?</th>
</tr>
</thead>
<tbody>
<tr>
<td>GD (Nesterov, 2004)</td>
<td>1</td>
<td>$d$</td>
<td>$\frac{1}{L}$</td>
<td>✓</td>
<td>$\tilde{O}(\kappa)$ (c)</td>
<td>x</td>
</tr>
<tr>
<td>LocalGD (Khaled et al., 2019; 2020)</td>
<td>$\tau$</td>
<td>$d$</td>
<td>$\frac{1}{\tau L}$</td>
<td>x</td>
<td>$O\left(\frac{G^2}{\mu n \tau \epsilon}\right)$ (d)</td>
<td>x</td>
</tr>
<tr>
<td>Scaffold (Karimireddy et al., 2020)</td>
<td>$\tau$</td>
<td>$2d$</td>
<td>$\frac{1}{\tau L}$</td>
<td>✓</td>
<td>$\tilde{O}(\kappa)$ (c)</td>
<td>x</td>
</tr>
<tr>
<td>S-Local-GD (a) (Gorbunov et al., 2021)</td>
<td>$\tau$</td>
<td>$d &lt; # &lt; 2d$</td>
<td>$\frac{1}{\tau L}$</td>
<td>✓</td>
<td>$\tilde{O}(\kappa)$</td>
<td>x</td>
</tr>
<tr>
<td>FedLin (b) (Mitra et al., 2021)</td>
<td>$\tau_i$</td>
<td>$2d$</td>
<td>$\frac{1}{\tau_i L}$</td>
<td>✓</td>
<td>$\tilde{O}(\kappa)$ (c)</td>
<td>x</td>
</tr>
<tr>
<td>Scaffnew (g) (this work) for any $p \in (0, 1]$</td>
<td>$\frac{1}{p}$ (h)</td>
<td>$d$</td>
<td>$\frac{1}{L}$</td>
<td>✓</td>
<td>$\tilde{O}\left(p \kappa + \frac{1}{p}\right)$ (c)</td>
<td>✓ (for $p &gt; \frac{1}{\kappa}$)</td>
</tr>
<tr>
<td>Scaffnew (g) (this work) for optimal $p = \frac{1}{\sqrt{\kappa}}$</td>
<td>$\sqrt{\kappa}$ (h)</td>
<td>$d$</td>
<td>$\frac{1}{L}$</td>
<td>✓</td>
<td>$\tilde{O}(\sqrt{\kappa})$ (c)</td>
<td>✓</td>
</tr>
</tbody>
</table>

(a) This is a special case of S-Local-SVRG, which is a more general method presented in (Gorbunov et al., 2021). S-Local-GD arises as a special case when full gradient is computed on each client.
(b) FedLin is a variant with a fixed but different number of local steps for each client. Earlier method S-Local-GD has the same update but random loop length.
(c) The $\tilde{O}$ notation hides logarithmic factors.
(d) $G$ is the level of dissimilarity from the assumption $\frac{1}{n} \sum_{i=1}^{n} \| \nabla f_i(x) \|^2 \leq G^2 + 2LB^2 (f(x) - f_*)$, $\forall x$.
(e) We use Scaffold’s cumulative local-global stepsize $\eta_i \eta_0$ for a fair comparison.
(f) The number of sent vectors depends on hyper-parameters, and it is randomized.
(g) Scaffnew (Algorithm 2) = ProxSkip (Algorithm 1) applied to the consensus formulation (6) + (7) of the finite-sum problem (5).
Federated Learning: ProxSkip vs Nesterov
Part 2

Consensus Reformulation
Consensus Reformulation

**Original problem:**
Optimization in $\mathbb{R}^d$

$$\min_{x \in \mathbb{R}^d} \left\{ f(x) \overset{\text{def}}{=} \frac{1}{n} \sum_{i=1}^{n} f_i(x) \right\}$$

**Consensus reformulation:**
Optimization in $\mathbb{R}^{nd}$

$$\min_{x_1, \ldots, x_n \in \mathbb{R}^d} \left\{ \frac{1}{n} \sum_{i=1}^{n} f_i(x_i) + \psi(x_1, \ldots, x_n) \right\}$$

$$\psi(x_1, \ldots, x_n) \overset{\text{def}}{=} \begin{cases} 0, & \text{if } x_1 = \cdots = x_n, \\ +\infty, & \text{otherwise.} \end{cases}$$

**Bad:** Non-differentiable function

**Good:** Indicator function of a nonempty closed convex set
Generalization 1: Constrained Optimization

Consensus reformulation: optimization in $\mathbb{R}^{nd}$

$$\min_{x_1, \ldots, x_n \in \mathbb{R}^d} \left\{ \frac{1}{n} \sum_{i=1}^{n} f_i(x_i) + \psi(x_1, \ldots, x_n) \right\}$$

$$\psi(x_1, \ldots, x_n) \overset{\text{def}}{=} \begin{cases} 0, & \text{if } x_1 = \cdots = x_n, \\ +\infty, & \text{otherwise}. \end{cases}$$

Generalization 1

$$\psi(x_1, \ldots, x_n) \overset{\text{def}}{=} \begin{cases} 0, & \text{if } (x_1, \cdots, x_n) \in C, \\ +\infty, & \text{otherwise}. \end{cases}$$

Arbitrary closed convex set (constraint)
Generalization 2: Composite Optimization

Consensus reformulation:

Optimization in $\mathbb{R}^{nd}$

$$\min_{x_1, \ldots, x_n \in \mathbb{R}^d} \left\{ \frac{1}{n} \sum_{i=1}^n f_i(x_i) + \psi(x_1, \ldots, x_n) \right\}$$

$$\psi(x_1, \ldots, x_n) \overset{\text{def}}{=} \begin{cases} 0, & \text{if } x_1 = \cdots = x_n, \\ +\infty, & \text{otherwise.} \end{cases}$$

Generalization 2

$\psi(x_1, \ldots, x_n): \mathbb{R}^{nd} \to \mathbb{R} \cup \{+\infty\}$ is a proper closed convex function

The epigraph of $\psi$ is a closed and convex set

$$\text{epi}(\psi) \overset{\text{def}}{=} \{(x, t) \mid \psi(x) \leq t\}$$
Conceptual Simplification: from $nd$ to $d'$

Composite optimization:

Optimization in $\mathbb{R}^{nd}$

$$\min_{x_1, \ldots, x_n \in \mathbb{R}^d} \left\{ \frac{1}{n} \sum_{i=1}^{n} f_i(x_i) + \psi(x_1, \ldots, x_n) \right\}$$

Composite optimization:

Optimization in $\mathbb{R}^{d'}$

$$\min_{x \in \mathbb{R}^{d'}} \left\{ f(x) + \psi(x) \right\}$$

\[
\begin{align*}
    d' &= nd \\
    x &= (x_1, \ldots, x_n) \\
    f(x) &= \frac{1}{n} \sum_{i=1}^{n} f_i(x_i) \\
    \psi(x) &= \psi(x_1, \ldots, x_n)
\end{align*}
\]
Part 3
Proximal Gradient Descent
Three Assumptions

\[
\min_{x \in \mathbb{R}^d} \ f(x) + \psi(x)
\]

**A1** \( f \) is \( \mu \)-convex and \( L \)-smooth:
\[
\frac{\mu}{2} \| x - y \|^2 \leq D_f(x, y) \leq \frac{L}{2} \| x - y \|^2
\]

Bregman divergence of \( f \):
\[
D_f(x, y) \overset{\text{def}}{=} f(x) - f(y) - \langle \nabla f(y), x - y \rangle
\]

**A2** \( \psi : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\} \) is proper, closed, and convex

**A3** \( \psi \) is proximable

The proximal operator \( \operatorname{prox}_\psi : \mathbb{R}^d \to \mathbb{R}^d \) defined by
\[
\operatorname{prox}_\psi(x) \overset{\text{def}}{=} \arg \min_{u \in \mathbb{R}^d} \left( \psi(u) + \frac{1}{2} \| u - x \|^2 \right)
\]

can be evaluated exactly (e.g., in closed form)

The epigraph of \( \psi \) is a closed and convex set
\[
\operatorname{epi}(\psi) \overset{\text{def}}{=} \{(x, t) \in \mathbb{R}^d \times \mathbb{R} \mid \psi(x) \leq t \}
\]

The epigraph of \( \psi \) is a closed and convex set
\[
\operatorname{epi}(\psi) \overset{\text{def}}{=} \{(x, t) \in \mathbb{R}^d \times \mathbb{R} \mid \psi(x) \leq t \}
\]
Key Method: Proximal Gradient Descent

proximal operator:

\[
\text{prox}_\psi(x) \overset{\text{def}}{=} \arg \min_{u \in \mathbb{R}^d} \left( \psi(u) + \frac{1}{2} \| u - x \|^2 \right)
\]

\[
x_t \leftarrow \gamma \nabla f(x_t)
\]

gradient operator

\[
x \mapsto x - \gamma \nabla f(x)
\]
Proximal Gradient Descent: Theory

**Theorem:**

\[ t \geq \frac{L}{\mu} \log \frac{1}{\varepsilon} \]

\[ \Rightarrow \quad \| x_t - x_* \|^2 \leq \varepsilon \| x_0 - x_* \|^2 \]

(for stepsize \( \gamma = \frac{1}{L} \))

\( f \) is \( \mu \)-convex and \( L \)-smooth:

\[ \frac{\mu}{2} \| x - y \|^2 \leq D_f(x, y) \leq \frac{L}{2} \| x - y \|^2 \]

\( \frac{L}{\mu} \) is the condition number of \( f \)

\( x_* \) defined as:

\[ x_* = \arg \min_{x \in \mathbb{R}^d} f(x) + \psi(x) \]
Part 4

The ProxSkip Algorithm
What to do When the Prox is Expensive?

Can we somehow get away with fewer evaluations of the proximity operator in the Proximal GD method?

<table>
<thead>
<tr>
<th>Approach 1</th>
<th>Approach 2 (ProxSkip)</th>
</tr>
</thead>
<tbody>
<tr>
<td>✅ We’ll skipp ALL prox evaluations!</td>
<td>✅ We’ll skip MANY prox evaluations!</td>
</tr>
<tr>
<td>❌ The method is NOT implementable!</td>
<td>✅ The method is implementable!</td>
</tr>
<tr>
<td>✅ Serves as an inspiration for Approach 2</td>
<td></td>
</tr>
</tbody>
</table>
Approach 1:
Simple, Extreme but Practically Useless Variant
Removing $\psi$ via a Reformulation

By the 1st order optimality conditions, the solution satisfies $\nabla f(x) - \nabla f(x_*) = 0$

$x_*$ is a solution of the above problem!

We do not know $h_* = \nabla f(x_*)$!
Apply Gradient Descent to the Reformulation

\[ x_{t+1} = x_t - \gamma (\nabla f(x_t) - h_*) \]

We do not need to evaluate the prox of \( \psi \) at all!

We do not know \( h_* \) and hence can’t implement the method!
Idea: Try to “Learn” the Optimal Gradient Shift

\[ x_{t+1} = x_t - \gamma (\nabla f(x_t) - h_t) \]

Desire: \( h_t \rightarrow h_\star \)

Perhaps we can learn \( h_\star \) with only occasional access to \( \psi \)?
Approach 2: The ProxSkip Method
ProxSkip: The Algorithm (Bird’s Eye View)

1. \[ \hat{x}_{t+1} = x_t - \gamma (\nabla f(x_t) - h_t) \]

2a. with probability \(1 - p\)
   
   \[ x_{t+1} = \hat{x}_{t+1} \]
   
   \[ h_{t+1} = h_t \]

2b. with probability \(p\)
   
   evaluate prox\(\psi_{\gamma/p}(?)\)

   \[ x_{t+1} = ? \]

   \[ h_{t+1} = ? \]
Algorithm 1 ProxSkip

1: stepsize $\gamma > 0$, probability $p > 0$, initial iterate $x_0 \in \mathbb{R}^d$, initial control variate $h_0 \in \mathbb{R}^d$, number of iterations $T \geq 1$
2: for $t = 0, 1, \ldots, T - 1$ do
3:   $\hat{x}_{t+1} = x_t - \gamma(\nabla f(x_t) - h_t)$  \hspace{1cm} \textcircled{Take a gradient-type step adjusted via the control variate $h_t$}
4:   Flip a coin $\theta_t \in \{0, 1\}$ where $\text{Prob}(\theta_t = 1) = p$ \hspace{1cm} \textcircled{Flip a coin that decides whether to skip the prox or not}
5:   if $\theta_t = 1$ then
6:     $x_{t+1} = \text{prox}_{\frac{p}{\gamma} \psi}(\hat{x}_{t+1} - \frac{p}{\gamma} h_t)$ \hspace{1cm} \textcircled{Apply prox, but only very rarely! (with small probability $p$)}
7:   else
8:     $x_{t+1} = \hat{x}_{t+1}$ \hspace{1cm} \textcircled{Skip the prox!}
9:   end if
10:  $h_{t+1} = h_t + \frac{p}{\gamma}(x_{t+1} - \hat{x}_{t+1})$ \hspace{1cm} \textcircled{Update the control variate $h_t$}
11: end for
Part 5
ProxSkip Theory
ProxSkip: Bounding the # of Iterations

Theorem:

\[ t \geq \max \left\{ \frac{L}{\mu}, \frac{1}{p^2} \right\} \log \frac{1}{\varepsilon} \implies \mathbb{E} [\Psi_t] \leq \varepsilon \Psi_0 \]

*Lyapunov function:*

\[ \Psi_t \overset{\text{def}}{=} \left\| x_t - x_* \right\|^2 + \frac{1}{L^2 p^2} \left\| h_t - h_* \right\|^2 \]

\( f \) is \( \mu \)-convex and \( L \)-smooth:

\[ \frac{\mu}{2} \| x - y \|^2 \leq D_f(x, y) \leq \frac{L}{2} \| x - y \|^2 \]

\( \frac{L}{\mu} \) is the condition number of \( f \)

\( p \) = probability of evaluating the prox
ProxSkip: Optimal Prox-Evaluation Probability

Since in each iteration we evaluate the prox with probability $p$, the expected number of prox evaluations after $t$ iterations is:

$$p \cdot t = p \cdot \max \left\{ \frac{L}{\mu}, \frac{1}{p^2} \right\} \cdot \log \frac{1}{\varepsilon} = \max \left\{ p \cdot \frac{L}{\mu}, \frac{1}{p} \right\} \cdot \log \frac{1}{\varepsilon}$$

Minimized for $p$ satisfying $p \cdot \frac{L}{\mu} = \frac{1}{p}$

$$\Rightarrow \quad p_\star = \frac{1}{\sqrt{L/\mu}}$$

$\frac{L}{\mu}$ is the condition number of $f$

Computation of optimal $p_\star$ for $\frac{L}{\mu} = 2$
**ProxSkip: # of Gradient and Prox Evaluations**

\[ p_* = \frac{1}{\sqrt{L/\mu}} \quad \Rightarrow \]

<table>
<thead>
<tr>
<th></th>
<th># of iterations</th>
<th># of gradient evaluations</th>
<th>Expected # of prox evaluations</th>
<th>Expected # of gradient evaluations between 2 prox evaluations</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>( \max \left{ \frac{L}{\mu}, \frac{1}{p^2} \right} \cdot \log \frac{1}{\varepsilon} )</td>
<td>( \max \left{ \frac{L}{\mu}, \frac{1}{p^2} \right} \cdot \log \frac{1}{\varepsilon} )</td>
<td>( \frac{L}{\mu} \cdot \log \frac{1}{\varepsilon} )</td>
<td>( \frac{1}{p} )</td>
</tr>
</tbody>
</table>
Part 6
ProxSkip: Application to Consensus Reformulation of Federated Learning
# Federated Learning: ProxSkip vs Baselines

Table 1. The performance of federated learning methods employing multiple local gradient steps in the strongly convex regime.

<table>
<thead>
<tr>
<th>method</th>
<th># local steps per round</th>
<th># floats sent per round</th>
<th>stepsize on client $i$</th>
<th>linear rate?</th>
<th># rounds</th>
<th>rate better than GD?</th>
</tr>
</thead>
<tbody>
<tr>
<td>GD (Nesterov, 2004)</td>
<td>1</td>
<td>$d$</td>
<td>$\frac{1}{L}$</td>
<td>✓</td>
<td>$\tilde{O}(\kappa)$ (c)</td>
<td>x</td>
</tr>
<tr>
<td>LocalGD (Khaled et al., 2019; 2020)</td>
<td>$\tau$</td>
<td>$d$</td>
<td>$\frac{1}{\tau L}$</td>
<td>x</td>
<td>$\mathcal{O}\left(\frac{G^2}{\mu n \tau \varepsilon}\right)$ (d)</td>
<td>x</td>
</tr>
<tr>
<td>Scaffold (Karimireddy et al., 2020)</td>
<td>$\tau$</td>
<td>$2d$</td>
<td>$\frac{1}{\tau L}$</td>
<td>✓</td>
<td>$\tilde{O}(\kappa)$ (c)</td>
<td>x</td>
</tr>
<tr>
<td>S-Local-GD (a) (Gorbunov et al., 2021)</td>
<td>$\tau$</td>
<td>$d &lt; # &lt; 2d$ (f)</td>
<td>$\frac{1}{\tau L}$</td>
<td>✓</td>
<td>$\tilde{O}(\kappa)$</td>
<td>x</td>
</tr>
<tr>
<td>FedLin (b) (Mitra et al., 2021)</td>
<td>$\tau_i$</td>
<td>$2d$</td>
<td>$\frac{1}{\tau_i L}$</td>
<td>✓</td>
<td>$\tilde{O}(\kappa)$ (c)</td>
<td>x</td>
</tr>
<tr>
<td>Scaffold (g) (this work)</td>
<td>for any $p \in (0, 1]$</td>
<td>$\frac{1}{p}$ (h)</td>
<td>$d$</td>
<td>✓</td>
<td>$\tilde{O}\left(\frac{p \kappa + \frac{1}{p}}{\kappa}\right)$ (c)</td>
<td>(for $p &gt; \frac{1}{\kappa}$)</td>
</tr>
<tr>
<td>Scaffold (g) (this work)</td>
<td>for optimal $p = \frac{1}{\sqrt{\kappa}}$</td>
<td>$\sqrt{\kappa}$ (h)</td>
<td>$d$</td>
<td>✓</td>
<td>$\tilde{O}(\sqrt{\kappa})$ (c)</td>
<td>✓</td>
</tr>
</tbody>
</table>

(a) This is a special case of S-Local-SVRG, which is a more general method presented in (Gorbunov et al., 2021). S-Local-GD arises as a special case when full gradient is computed on each client.
(b) FedLin is a variant with a fixed but different number of local steps for each client. Earlier method S-Local-GD has the same update but random loop length.
(c) The $\tilde{O}$ notation hides logarithmic factors.
(d) $G$ is the level of dissimilarity from the assumption $\frac{1}{n} \sum_{i=1}^{n} \| \nabla f_i(x) \|^2 \leq G^2 + 2LB^2 (f(x) - f_*)$ , $\forall x$.
(e) We use Scaffold’s cumulative local-global stepsize $\eta_l \eta_g$ for a fair comparison.
(f) The number of sent vectors depends on hyper-parameters, and it is randomized.
(g) Scaffold (Algorithm 2) = ProxSkip (Algorithm 1) applied to the consensus formulation (6) + (7) of the finite-sum problem (5).

ProxSkip (resp. Scaffold) takes a random number of gradient (resp. local) steps before prox (resp. communication) is computed (resp. performed). What is shown in the table is the expected number of gradient (resp. local) steps.
Part 7
Experiments
Scaffnew (=ProxSkip applied to FL) vs Nesterov

![Graph comparing Scaffnew and Nesterov with respect to communication rounds and function value]

- **Scaffnew** and **Nesterov** are compared in terms of their function value $f(x) - f^*$ as the number of communication rounds increases.
- The graph shows the decrease in $f(x) - f^*$ with an increasing number of communication rounds, indicating the performance of both algorithms.

Legend:
- Blue line: Nesterov
- Red line: ProxSkip
Scaffnew (=ProxSkip applied to FL) vs Baselines

Figure 1. Deterministic Problem. Comparison of Scaffnew to other local update methods that tackle data-heterogeneity and to LocalGD. In (a) we compare communication rounds with optimally tuned hyper-parameters. In (b) we compare communicated vectors (Scaffold, FedLin and S-Local-GD require transmission of additional variables). In (c), we compare communication rounds with the algorithm parameters set to the best theoretical stepsizes used in the convergence proofs.

L2-regularized logistic regression:

\[ f(x) = \frac{1}{n} \sum_{i=1}^{n} \log \left( 1 + \exp \left( -b_i a_i^T x \right) \right) + \frac{\lambda}{2} \|x\|^2 \]

\[ a_i \in \mathbb{R}^d, \quad b_i \in \{-1, +1\}, \quad \lambda = L/10^4 \]

A dataset from LIBSVM library (Chang & Lin, 2011)
Figure 2. Stochastic Problem. Comparison of Scaffnew to other local update methods that tackle data-heterogeneity and to LocalSGD. In (a) we compare communication rounds with optimally tuned hyper-parameters. In (b) we compare communicated vectors and in (c), we compare communication rounds with the algorithm parameters set to the best theoretical stepsizes used in the convergence proofs.

L2-regularized logistic regression:

\[ f(x) = \frac{1}{n} \sum_{i=1}^{n} \log \left( 1 + \exp \left( -b_i a_i^T x \right) \right) + \frac{\lambda}{2} \|x\|^2 \]

\[ a_i \in \mathbb{R}^d, \quad b_i \in \{-1, +1\}, \quad \lambda = L/10^4 \]

w8a dataset from LIBSVM library (Chang & Lin, 2011)
Extension 1: From Gradients to Stochastic Gradients

- As described, in ProxSkip each worker computes the full gradient of its local function.
- It’s often better to consider a cheap stochastic approximation of the gradient instead.
  - We consider this extension in the paper.
  - We provide theoretical convergence rates.

\[ \nabla f_i(x_t) \quad \Rightarrow \quad g_i(x_t) \]

**Full gradient** \quad **Stochastic gradient**

**Assumptions:**

- **(unbiasedness)** \( \mathbb{E}[g_{i,t}(x_t) \mid x_t] = \nabla f_i(x_t) \)
- **(expected smoothness)** \( \mathbb{E} [\|g_{i,t}(x_t) - \nabla f(x_*)\|^2 \mid x_t] \leq 2AD_f(x_t, x_*) + C \)

(Gower et al, 2019)
In each communication round of ProxSkip, each worker sends messages to all other workers (e.g., through a server).
- We can think of ProxSkip workers as the nodes of a fully-connected network.
- In each communication round, all workers communicate with their neighbors.
- In the paper, we provide extension to arbitrary connected networks.

Extension 2: From Fully Connected Network to Arbitrary Connected Network

Fully connected network

Arbitrary connected network
The End