Federated Learning and optimization: from a gentle introduction to recent results

LPSM, 25 March 2022

Based on several papers, including work with Constantin Philippenko

Aymeric DIEULEVEUT
Assistant Professor, École Polytechnique, Institut Polytechnique de Paris.
Federated Learning: a collaborative learning framework

**Objective:**
- building better models in Machine Learning

Central server

Individual Agents
Federated Learning: a collaborative learning framework

Objective:
- building better models in Machine Learning
- by enabling multiple participants to participate in training process
Federated Learning: a collaborative learning framework

**Objective:**
- building better models in Machine Learning
- by enabling multiple participants to participate in training process

**Constraints**
- Heterogeneity and Adaptation
**Federated Learning: a collaborative learning framework**

**Objective:**
- building better models in Machine Learning
- by enabling multiple participants to participate in training process

**Central server**

**Individual Agents**

**Constraints**
- Heterogeneity and Adaptation
Federated Learning: a collaborative learning framework

Objective:
- building better models in Machine Learning
- by enabling multiple participants to participate in training process

Constraints
- Heterogeneity and Adaptation
- Privacy and trust
Federated Learning: a collaborative learning framework

Objective:
- building better models in Machine Learning
- by enabling multiple participants to participate in training process

Constraints
- Heterogeneity and Adaptation
- Privacy and trust
- Communication, device availability
Federated Learning: a collaborative learning framework

**Objective:**
- building better models in Machine Learning
- by enabling multiple participants to participate in training process

**Constraints**
- Heterogeneity and Adaptation
- Privacy and trust
- Communication, device availability

**Some challenges**
- Statistics
- Optimization and Algorithms
- Privacy guarantees
Federated Learning: a collaborative learning framework

Mathematical framework:

\[ w_* = \arg \min_{w \in \mathbb{R}^d} \left\{ F(w) := \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}_{z \sim D_i} [\ell(z, w)] \right\} . \]

- \( F \): global cost function
- \( F_i \): local loss
- \( N \): workers
- \( d \): dimension
- \( w \): model
- \( D_i \): local data distribution
Federated Learning: a collaborative learning framework

Mathematical framework:

\[ w_* = \arg \min_{w \in \mathbb{R}^d} \left\{ F(w) := \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}_{z \sim D_i} [\ell(z, w)] \right\}. \]

- \( F \): global cost function
- \( F_i \): local loss
- \( N \): workers
- \( d \): dimension
- \( w \): model
- \( D_i \): local data distribution

\[ F(w) := \frac{1}{N} \sum_{i=1}^{N} F_i(w) \]

Optimization based on Stochastic Approximation, SGD
Federated Learning: a collaborative learning framework

Mathematical framework:

$$w_\ast = \arg \min_{w \in \mathbb{R}^d} \left\{ F(w) := \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}_{z \sim D_i} [\ell(z, w)] \right\}.$$  

- \(F\): global cost function
- \(F_i\): local loss
- \(N\): workers
- \(d\): dimension
- \(w\): model
- \(D_i\): local data distribution

→ Optimization based on Stochastic Approximation, SGD

Simplest case:

$$w_k = w_{k-1} - \gamma \left( \frac{1}{N} \sum_{i=1}^{N} g^i_k(w_{k-1}) \right).$$
Federated Learning: a collaborative learning framework

Mathematical framework:

\[ w_* = \arg \min_{w \in \mathbb{R}^d} \left\{ F(w) := \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}_{z \sim D_i} [\ell(z, w)] \right\}. \]

**F**: global cost function

**\( F_i \)**: local loss

**N**: workers

**d**: dimension

**w**: model

**D_i**: local data distribution

Optimization based on Stochastic Approximation, SGD

Simplest case:

\[ w_k = w_{k-1} - \gamma \left( \frac{1}{N} \sum_{i=1}^{N} g_i^k(w_{k-1}) \right) \]

Equivalent to (stratified) mini-batch SGD.
What makes it difficult?

Reducing the communication cost:

1. Performing multiple local updates before exchanging information
2. Compressing the message sent.

How do compression and local updates influence convergence?
What makes it difficult?

**Reducing the communication cost:**

1. Performing multiple local updates before exchanging information
2. Compressing the message sent.

→ How do compression and local updates influence convergence?

**Heterogeneity**

1. \( w_* \neq N^{-1} \sum_{i=1}^{N} w_{*,i} \)
2. \( w_* \) is not a stable point for GD on any \( F_i \)

→ How to adapt algorithms? E.g., *control variates* and links with V.R.
What makes it difficult?

**Reducing the communication cost:**

1. Performing multiple local updates before exchanging information
2. Compressing the message sent.

→ How do compression and local updates influence convergence?

**Heterogeneity**

1. \( w_* \neq N^{-1} \sum_{i=1}^{N} w_{*,i} \)
2. \( w_* \) is not a stable point for GD on any \( F_i \)

→ How to adapt algorithms? E.g., *control variates* and links with V.R.

**Privacy**

1. DP can be achieved by adding noise on the exchanged gradients.

→ Composition rules? How can we obtain tight theoretical guarantees?
Some directions we have worked on

1. **Adapting algorithms with compression**
   - Artemis: tight convergence guarantees for bidirectional compression in Federated Learning [PD20]
   - *QLSD: Quantised Langevin stochastic dynamics for Bayesian federated learning,* [VPD+21], w. M. Vono, V. Plassier, A. Durmus, E. Moulines, Aistats22.

2. **Proposing compression operators & Impact of their properties on convergence**

3. **Handling privacy and heterogeneity with local updates**
   - *Differentially Private Federated Learning on Heterogeneous Data* [NBD21], w. M. Noble, A. Bellet, Aistats22.

→ Let us focus on compression!
Compression: a well identified problem in FL. [KMA+19].

\[ F(w) := \frac{1}{N} \sum_{i=1}^{N} F_i(w) \]
Compression: a well identified problem in FL. [KMA⁺19].

\[ F(w) := \frac{1}{N} \sum_{i=1}^{N} F_i(w) \]
Compression: a well identified problem in FL. [KMA+19].

\[ F(w) := \frac{1}{N} \sum_{i=1}^{N} F_i(w) \]

1. Proposing compression operators.
   - QSGD, Nu-QSGD,
   - Atomo, Power-SGD, HSQ etc.

2. Studying the impact of the properties of algorithms on convergence:
   - Biased vs Unbiased
   - Independent or not
   - Bounded variance, relatively bounded variance, adaptation

3. Adapting algorithms with compression.
   Compression can prevent the algorithm from converging.
   - Impact of Bias in the compression operator. Error-Feedback [SCJ18, SK19]
   - Impact of Heterogeneity. Memory [MGTR19].
   - bidirectional compression [LLTY20, PD20, TYL+19, ZHK19]
Compression Operators, examples

- To limit the number of bits exchanged, we compress each signal before transmitting it.

Example: quantization [AGL$^+$ 17, Eli75]

\[ V = \begin{bmatrix}
1 \\
5 \\
10 \\
-2 \\
-8 \\
4 
\end{bmatrix} \]
Compression Operators, examples

جملة: To limit the number of bits exchanged, we **compress** each signal before transmitting it.

Example: *quantization* [AGL\textsuperscript{+}17, Eli75]

\[
V = \begin{bmatrix}
1 \\
5 \\
10 \\
-2 \\
-8 \\
4
\end{bmatrix} \quad \Rightarrow (s = 1 \Leftrightarrow 2 \text{ levels}) \quad \hat{V}_{\text{2 levels}} = \frac{1}{s(=1)} \times \|V\|_2 \times \begin{bmatrix}
0 \\
0 \\
1 \\
0 \\
-1 \\
0
\end{bmatrix}
\]
Compression Operators, examples

To limit the number of bits exchanged, we compress each signal before transmitting it.

Example: quantization [AGL+17, Eli75]

\[
V = \begin{bmatrix}
1 \\
5 \\
10 \\
-2 \\
-8 \\
4
\end{bmatrix} \quad \Rightarrow (s = 1 \Leftrightarrow 2 \text{ levels}) \quad \hat{V}_{2 \text{ levels}} = \frac{1}{s(= 1)} \times \|V\|_2 \times \begin{bmatrix}
0 \\
0 \\
1 \\
0 \\
-1 \\
0
\end{bmatrix}
\]

\[
\Rightarrow (s = 2 \Leftrightarrow 3 \text{ levels}) \quad \hat{V}_{3 \text{ levels}} = \frac{1}{s(= 2)} \times \|V\|_2 \times \begin{bmatrix}
0 \\
1 \\
2 \\
0 \\
-2 \\
1
\end{bmatrix}
\]
Compression Operators, examples

To limit the number of bits exchanged, we compress each signal before transmitting it.

Example: quantization [AGL\textsuperscript{+}17, Eli75]

\[ V = \begin{bmatrix} 1 \\ 5 \\ 10 \\ -2 \\ -8 \\ 4 \end{bmatrix} \Rightarrow (s = 1 \Leftrightarrow 2 \text{ levels}) \]

\[ \hat{V}_{2 \text{ levels}} = \frac{1}{s(= 1)} \times \| V \|_2 \times \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ -1 \\ 0 \end{bmatrix} \]

\[ \Rightarrow (s = 2 \Leftrightarrow 3 \text{ levels}) \]

\[ \hat{V}_{3 \text{ levels}} = \frac{1}{s(= 2)} \times \| V \|_2 \times \begin{bmatrix} 0 \\ 1 \\ 2 \\ 0 \\ -2 \\ 1 \end{bmatrix} \]

How to quantify the loss resulting from compression? Which assumptions do we need in order to maintain convergence?
Key assumption: unbiasedness & relatively bounded variance.

Assumption 1 (Compression operators $C_{dwn}$ and $C_{up}$ are U-RBV.)

For $dir \in \{up, dwn\}$, there exists a constant $\omega_{dir} \in \mathbb{R}^*$ s.t. $C_{dir}$ satisfies, for all $\Delta$ in $\mathbb{R}^d$:

$$E[C_{dir}(\Delta)] = \Delta \quad and \quad E\left[\|C_{dir}(\Delta) - \Delta\|^2\right] \leq \omega_{dir} \|\Delta\|^2.$$  

$\Rightarrow$ Satisfied by quantization, random sparsification, etc.
Key assumption: unbiasedness & relatively bounded variance.

Assumption 1 (Compression operators $C_{\text{dwn}}$ and $C_{\text{up}}$ are U-RBV.)

For $\text{dir} \in \{\text{up, dwn}\}$, there exists a constant $\omega_{\text{dir}} \in \mathbb{R}^+$ s.t. $C_{\text{dir}}$ satisfies, for all $\Delta$ in $\mathbb{R}^d$:

$$
\mathbb{E}[C_{\text{dir}}(\Delta)] = \Delta \quad \text{and} \quad \mathbb{E}\left[\|C_{\text{dir}}(\Delta) - \Delta\|^2\right] \leq \omega_{\text{dir}} \|\Delta\|^2.
$$

$\mapsto$ Satisfied by quantization, random sparsification, etc.

$\Rightarrow$ SGD with double compression:

$$
C_{\text{dwn}}\left(\frac{1}{N} \sum_{i=1}^{N} C_{\text{up}}(g_k^i)\right).
$$

$\mapsto$ What do we need for convergence?
Key assumption: unbiasedness & relatively bounded variance.

Assumption 1 (Compression operators $C_{\text{dwn}}$ and $C_{\text{up}}$ are U-RBV.)

For $\text{dir} \in \{\text{up}, \text{dwn}\}$, there exists a constant $\omega_{\text{dir}} \in \mathbb{R}^+$ s.t. $C_{\text{dir}}$ satisfies, for all $\Delta$ in $\mathbb{R}^d$:

$$
\mathbb{E}[C_{\text{dir}}(\Delta)] = \Delta \quad \text{and} \quad \mathbb{E}\left[\|C_{\text{dir}}(\Delta) - \Delta\|^2\right] \leq \omega_{\text{dir}} \|\Delta\|^2 .
$$

\(\mapsto\) Satisfied by quantization, random sparsification, etc.

\(\Rightarrow\) SGD with double compression:

$$
w_k = w_{k-1} - \gamma C_{\text{dwn}} \left( \frac{1}{N} \sum_{i=1}^{N} C_{\text{up}}(g^i_k) \right) .
$$

\(\mapsto\) What do we need for convergence?

For SGD:
Expected results for Double compression

\[ E[C_{dir}(\Delta)] = \Delta \quad \text{and} \quad E \left[ \| C_{dir}(\Delta) - \Delta \|_2^2 \right] \leq \omega_{dir} \| \Delta \|_2^2. \]

1. The level of noise in the gradient increases,
2. Proportionally to \( \omega_{dwn} (1 + \omega_{up} / N) \)
3. In fact, we can prove that the limit Variance indeed provably increases [PD20].
Expected results for Double compression

\[ E[C_{\text{dir}}(\Delta)] = \Delta \quad \text{and} \quad E\left[\|C_{\text{dir}}(\Delta) - \Delta\|^2\right] \leq \omega_{\text{dir}} \|\Delta\|^2. \]

1. The level of noise in the gradient increases,
2. Proportionally to

\[ \omega_{\text{dwn}} (1 + \omega_{\text{up}}/N) \]

3. In fact, we can prove that the limit Variance indeed provably increases [PD20].

What if \( \sigma^2 = 0 \)?
Key idea number 1: *memory / control variates*

**Motivation:** The distribution of the observations on worker $i$ and $j$ are different.

**Assumption 2 (Device heterogeneity.)**

*For all $i \in [N]$:*

\[ \| \nabla F_i(w_\star) \|^2 \leq B^2 \]

**Objective:** Compression of a quantity that goes to 0!
Key idea number 1: memory / control variates

Motivation: The distribution of the observations on worker $i$ and $j$ are different.

Assumption 2 (Device heterogeneity.)

For all $i \in [N]$: $\|\nabla F_i(w_*)\|^2 \leq B^2$

Objective: Compression of a quantity that goes to 0 !

Solution: Compute (on the server and the worker independently) a “memory” $h^i_k$ s.t.

$h^i_k \to_{k \to \infty} \nabla F_i(w_*)$.

⇒ The update equation becomes:

$$w_k = w_{k-1} - \gamma C_{\text{dwn}} \left( \frac{1}{N} \sum_{i=1}^{N} C_{\text{up}} (g^i_k - h^i_k) + h^i_k \right)$$

$$h^i_{k+1} = h^i_k + \alpha C_{\text{up}} (g^i_k - h^i_k)$$

Crucial role of (uplink)-memory on heterogeneous data. [MGTR19, PD20].
The memory mechanism

Expected improvement with uplink memory in the heterogeneous framework.
Key idea number 2: “non-degraded” update

Classical double compression (e.g., Artemis)- compress the update sent back to the workers and use it to update the model.

\[ w_k = w_{k-1} - \gamma C_{\text{dwn}} \left( \frac{1}{N} \sum_{i=1}^{N} C_{\text{up}} (g^i_k(w_{k-1})) \right) \]

The gradient is taken at the point \( w_k \) held by the central server.
Key idea number 2: “non-degraded” update

Classical double compression (e.g., Artemis)- compress the update sent back to the workers and use it to update the model.

\[ w_k = w_{k-1} - \gamma C_{dwn} \left( \frac{1}{N} \sum_{i=1}^{N} C_{up} \left( g^i_k(w_{k-1}) \right) \right) \]

The gradient is taken at the point \( w_k \) held by the central server.

MCM - preserve the model on the central server.

\[ w_k = w_{k-1} - \gamma \left( \frac{1}{N} \sum_{i=1}^{N} C_{up} \left( g^i_k(\hat{w}_{k-1}) \right) \right) \]

\[ \hat{w}_k = w_{k-1} - \gamma C_{dwn} \left( \frac{1}{N} \sum_{i=1}^{N} C_{up} \left( g^i_k(\hat{w}_{k-1}) \right) \right) \] (1)

The gradient is taken at a random point \( \hat{w}_k \) s.t. \( \mathbb{E}[\hat{w}_k|w_k] = w_k \)
Key idea number 2: “non-degraded” update

Classical double compression (e.g., Artemis)- **compress the update sent back to the workers and use it to update the model.**

\[ w_k = w_{k-1} - \gamma C_{\text{dwn}} \left( \frac{1}{N} \sum_{i=1}^{N} C_{\text{up}} (g_i^k(w_{k-1})) \right) \]

The gradient is taken at the point \( w_k \) held by the central server.

**MCM - preserve the model on the central server.**

\[ w_k = w_{k-1} - \gamma \left( \frac{1}{N} \sum_{i=1}^{N} C_{\text{up}} (g_i^k(\hat{w}_{k-1})) \right) \]

\[ \hat{w}_k = w_{k-1} - \gamma C_{\text{dwn}} \left( \frac{1}{N} \sum_{i=1}^{N} C_{\text{up}} (g_i^k(\hat{w}_{k-1})) \right) \] (1)

The gradient is taken at a random point \( \hat{w}_k \) s.t. \( \mathbb{E}[\hat{w}_k|w_k] = w_k \)

**Update (1) is not feasible in practice.** We refer to this algorithm as a Ghost algorithm.
Idea: link with randomized smoothing.
Ghost algorithm

What do we hope for?
Ghost algorithm

What do we hope for?

Summary:
1. Motivation for compression
2. Assumptions: U-RLB and heterogeneity
3. KI1: Memory
4. KI2: Preserved iterate
Ghost algorithm

What do we hope for?

Summary:
1. Motivation for compression
2. Assumptions: U-RLB and heterogeneity
3. KI1: Memory
4. KI2: Preserved iterate

Outline towards proof of convergence:
1. Assumptions
2. Convergence of Ghost
3. Sketch of proof
4. Adaptation into a practical algorithm
5. Extensions
Doubly compressed SGD

Classical approach

1. Compute local gradient $g_i^k$
2. Quantize local gradient: $C_{up}(g_i^k)$
3. Server gathers and aggregates: $\frac{1}{N} \sum_{i=1}^{N} C_{up}(g_i^k)$
4. Server compresses: $C_{down}(\frac{1}{N} \sum_{i=1}^{N} C_{up}(g_i^k))$
5. Server updates and broadcasts the new iterate: $w_k = w_k + \frac{1}{N} \sum_{i=1}^{N} C_{up}(g_i^k)$
Classical approach

1. Compute local gradient $g^j_k$
Classical approach

1. Compute local gradient $g_k^i$

2. Quantize local gradient:

$C_{up}(g_k^i)$
Classical approach

1. Compute local gradient $g^i_k$

2. Quantize local gradient:

$$C_{up}(g^i_k)$$

3. Server gathers and aggregates:

$$\frac{1}{N} \sum_{i=1}^{N} C_{up}(g^i_k)$$
Classical approach

1. Compute local gradient $g_k^i$

2. Quantize local gradient:
   
   $C_{up}(g_k^i)$

3. Server gathers and aggregates:
   
   $\frac{1}{N} \sum_{i=1}^{N} C_{up}(g_k^i)$

4. Server compresses:
   
   $C_{dwn}\left(\frac{1}{N} \sum_{i=1}^{N} C_{up}(g_k^i)\right)$
Classical approach

1. Compute local gradient $\mathbf{g}_k^i$

2. Quantize local gradient:
   
   $$C_{\text{up}}(\mathbf{g}_k^i)$$

3. Server gathers and aggregates:
   
   $$\frac{1}{N} \sum_{i=1}^{N} C_{\text{up}}(\mathbf{g}_k^i)$$

4. Server compresses:
   
   $$C_{\text{dwn}} \left( \frac{1}{N} \sum_{i=1}^{N} C_{\text{up}}(\mathbf{g}_k^i) \right)$$

5. Server updates and broadcasts the new iterate:
   
   $$\mathbf{w}_k = \mathbf{w}_{k-1} - \gamma C_{\text{dwn}} \left( \frac{1}{N} \sum_{i=1}^{N} C_{\text{up}}(\mathbf{g}_k^i) \right)$$
**Classical approach**

1. Compute local gradient $g^i_k$
2. Quantize local gradient: $C_{up}(g^i_k)$
3. Server gathers and aggregates:
   \[
   \frac{1}{N} \sum_{i=1}^{N} C_{up}(g^i_k)
   \]
4. Server compresses:
   \[
   C_{dwn}\left(\frac{1}{N} \sum_{i=1}^{N} C_{up}(g^i_k)\right)
   \]
5. Server updates and broadcasts the new iterate:
   \[
   w_k = w_{k-1} - \gamma C_{dwn}\left(\frac{1}{N} \sum_{i=1}^{N} C_{up}(g^i_k)\right)
   \]

**New Approach**
Classical approach

1. Compute local gradient $g_k^i$
2. Quantize local gradient: $C_{up}(g_k^i)$
3. Server gathers and aggregates:
   $$\frac{1}{N} \sum_{i=1}^{N} C_{up}(g_k^i)$$
4. Server compresses:
   $$C_{dwn}\left(\frac{1}{N} \sum_{i=1}^{N} C_{up}(g_k^i)\right)$$
5. Server updates and broadcasts the new iterate:
   $$w_k = w_{k-1} - \gamma C_{dwn}\left(\frac{1}{N} \sum_{i=1}^{N} C_{up}(g_k^i)\right)$$

New Approach

1. Compute local gradient $g_k^i(\hat{w}_k)$ at $\hat{w}_{k-1}$
$s.t \ E[\hat{w}_{k-1} | w_{k-1}] = w_{k-1}$
Classical approach

1. Compute local gradient $g^i_k$
2. Quantize local gradient: $C_{up}(g^i_k)$
3. Server gathers and aggregates:
   $$\frac{1}{N} \sum_{i=1}^{N} C_{up}(g^i_k)$$
4. Server compresses:
   $$C_{dwn} \left( \frac{1}{N} \sum_{i=1}^{N} C_{up}(g^i_k) \right)$$
5. Server updates and broadcasts the new iterate:
   $$w_k = w_{k-1} - \gamma C_{dwn} \left( \frac{1}{N} \sum_{i=1}^{N} C_{up}(g^i_k) \right)$$

New Approach

1. Compute local gradient $g^i_k(\hat{w}_k)$ at $\hat{w}_{k-1}$ s.t $\mathbb{E}[\hat{w}_{k-1} \mid w_{k-1}] = w_{k-1}$
2. Quantize local gradient: $C_{up}(g^i_k(\hat{w}_{k-1}))$
Classical approach

1. Compute local gradient $g^i_k$
2. Quantize local gradient: $C_{up}(g^i_k)$
3. Server gathers and aggregates:
   $$\frac{1}{N} \sum_{i=1}^{N} C_{up}(g^i_k)$$
4. Server compresses:
   $$C_{dnw} \left( \frac{1}{N} \sum_{i=1}^{N} C_{up}(g^i_k) \right)$$
5. Server updates and broadcasts the new iterate:
   $$w_k = w_{k-1} - \gamma \frac{1}{N} \sum_{i=1}^{N} C_{up}(g^i_k)$$

New Approach

1. Compute local gradient $g^i_k(\hat{w}_k)$ at $\hat{w}_k-1$
s.t $\mathbb{E}[\hat{w}_k-1 | w_{k-1}] = w_{k-1}$
2. Quantize local gradient: $C_{up}(g^i_k(\hat{w}_k-1))$
3. Server gathers and aggregates:
   $$\frac{1}{N} \sum_{i=1}^{N} C_{up}(g^i_k(\hat{w}_k-1))$$
4. Server performs global update:
   $$w_k = w_{k-1} - \gamma \frac{1}{N} \sum_{i=1}^{N} C_{up}(g^i_k(\hat{w}_k-1))$$
5. Server sends compressed information to remote servers:
Doubly compressed SGD

### Classical approach

1. Compute local gradient $g_k^i$
2. Quantize local gradient: $C_{up}(g_k^i)$
3. Server gathers and aggregates: $\frac{1}{N} \sum_{i=1}^{N} C_{up}(g_k^i)$
4. Server compresses: $C_{dwn} \left( \frac{1}{N} \sum_{i=1}^{N} C_{up}(g_k^i) \right)$
5. Server updates and broadcasts the new iterate:
   \[ w_k = w_{k-1} - \gamma \frac{1}{N} \sum_{i=1}^{N} C_{up}(g_k^i) \]

### New Approach

1. Compute local gradient $g_k^i(\hat{w}_k)$ at $\hat{w}_{k-1}$ s.t $\mathbb{E}[\hat{w}_{k-1} | w_{k-1}] = w_{k-1}$
2. Quantize local gradient: $C_{up} \left( g_k^i(\hat{w}_{k-1}) \right)$
3. Server gathers and aggregates: $\frac{1}{N} \sum_{i=1}^{N} C_{up} \left( g_k^i(\hat{w}_{k-1}) \right)$
4. Server performs global update:
   \[ w_k = w_{k-1} - \gamma \frac{1}{N} \sum_{i=1}^{N} C_{up} \left( g_k^i(\hat{w}_{k-1}) \right) \]
### Classical approach

1. Compute local gradient $g^i_k$
2. Quantize local gradient: $C_{up}(g^i_k)$
3. Server gathers and aggregates:
   $$\frac{1}{N} \sum_{i=1}^{N} C_{up}(g^i_k)$$
4. Server compresses:
   $$C_{dwn} \left( \frac{1}{N} \sum_{i=1}^{N} C_{up}(g^i_k) \right)$$
5. Server updates and broadcasts the new iterate:
   $$w_k = w_{k-1} - \gamma C_{dwn} \left( \frac{1}{N} \sum_{i=1}^{N} C_{up}(g^i_k) \right)$$

### New Approach

1. Compute local gradient $g^i_k(\hat{w}_k)$ at $\hat{w}_{k-1}$ s.t $\mathbb{E}[\hat{w}_{k-1} | w_{k-1}] = w_{k-1}$
2. Quantize local gradient: $C_{up}(g^i_k(\hat{w}_{k-1}))$
3. Server gathers and aggregates:
   $$\frac{1}{N} \sum_{i=1}^{N} C_{up}(g^i_k(\hat{w}_{k-1}))$$
4. Server performs global update:
   $$w_k = w_{k-1} - \gamma \frac{1}{N} \sum_{i=1}^{N} C_{up}(g^i_k(\hat{w}_{k-1}))$$
5. Server sends compressed information to remote servers:
   $$\hat{w}_k = \hat{w}_{k-1} - \gamma C_{dwn} \left( \frac{1}{N} \sum_{i=1}^{N} C_{up}(g^i_k(\hat{w}_{k-1})) \right)$$
Assumptions

We make standard assumptions on $F : \mathbb{R}^d \rightarrow \mathbb{R}$.

**Assumption 3 (Smoothness)**

*F is twice continuously differentiable, and is $L$-smooth, that is for all vectors $w_1, w_2$ in $\mathbb{R}^d$: \[\|\nabla F(w_1) - \nabla F(w_2)\| \leq L\|w_1 - w_2\|.*
Assumptions

We make standard assumptions on $F : \mathbb{R}^d \to \mathbb{R}$.

Assumption 3 (Smoothness)

$F$ is twice continuously differentiable, and is $L$-smooth, that is for all vectors $w_1, w_2$ in $\mathbb{R}^d$:

$$\|\nabla F(w_1) - \nabla F(w_2)\| \leq L\|w_1 - w_2\|.$$

Assumption 4 (Convexity)

$F$ is convex, that is for all vectors $w_1, w_2$ in $\mathbb{R}^d$:

$$F(w_2) \geq F(w_1) + (w_2 - w_1)^T \nabla F(w_1).$$
Assumptions

We make standard assumptions on $F : \mathbb{R}^d \to \mathbb{R}$.

**Assumption 3 (Smoothness)**

*F is twice continuously differentiable, and is L-smooth, that is for all vectors $w_1, w_2$ in $\mathbb{R}^d$: $\|\nabla F(w_1) - \nabla F(w_2)\| \leq L\|w_1 - w_2\|.*

**Assumption 4 (Convexity)**

*F is convex, that is for all vectors $w_1, w_2$ in $\mathbb{R}^d$: $F(w_2) \geq F(w_1) + (w_2 - w_1)^T \nabla F(w_1)$.*

**Assumption 5 (Noise over stochastic gradients computation)**

*The noise over stochastic gradients for a mini-batch of size $b$, is uniformly bounded: there exists a constant $\sigma \in \mathbb{R}_+$, such that for all $k$ in $\mathbb{N}$, for all $i$ in $[1, N]$ and for all $w$ in $\mathbb{R}^d$ we have: $E[\|g^i_k(w) - \nabla F(w)\|^2] \leq \sigma^2/b.$*
Assumptions

We make standard assumptions on $F : \mathbb{R}^d \rightarrow \mathbb{R}$.

Assumption 3 (Smoothness)

$F$ is twice continuously differentiable, and is $L$-smooth, that is for all vectors $w_1, w_2$ in $\mathbb{R}^d$:

$$\|\nabla F(w_1) - \nabla F(w_2)\| \leq L\|w_1 - w_2\|.$$

Assumption 4 (Convexity)

$F$ is convex, that is for all vectors $w_1, w_2$ in $\mathbb{R}^d$:

$$F(w_2) \geq F(w_1) + (w_2 - w_1)^T \nabla F(w_1).$$

Assumption 5 (Noise over stochastic gradients computation)

The noise over stochastic gradients for a mini-batch of size $b$, is uniformly bounded: there exists a constant $\sigma \in \mathbb{R}_+$, such that for all $k$ in $\mathbb{N}$, for all $i$ in $\{1, N\}$ and for all $w$ in $\mathbb{R}^d$ we have:

$$E[\|g_k^i(w) - \nabla F(w)\|^2] \leq \sigma^2 / b.$$

No heterogeneity in the next slides: $B^2 = 0$, to focus on KI2.
Theoretical results - Ghost

Convergence of Ghost

Definition 1 (Ghost algorithm)

Recall that the Ghost algorithm is defined as follows, for $k \in \mathbb{N}$, for all $i \in [1, N]$ we have:

$$w_k = w_{k-1} - \gamma \left( \frac{1}{N} \sum_{i=1}^{N} C_{up} \left( g^i_k(\hat{w}_{k-1}) \right) \right)$$

$$\hat{w}_k = w_{k-1} - \gamma C_{dwn} \left( \frac{1}{N} \sum_{i=1}^{N} C_{up} \left( g^i_k(\hat{w}_{k-1}) \right) \right)$$

Proposition 1

Consider the Ghost update in eq. (1), under Assumptions 1, 3 and 5, for all $k$ in $\mathbb{N}$ with the convention $\nabla F(w_{-1}) = 0$:

$$\mathbb{E} \left[ \|w_k - \hat{w}_k\|^2 \left| \hat{w}_{k-1} \right\| \right] \leq \gamma^2 \omega_{dwn} \left( 1 + \frac{\omega_{up}}{N} \right) \|\nabla F(\hat{w}_{k-1})\|^2 + \frac{\gamma^2 \omega_{dwn}(1 + \omega_{up}) \sigma^2}{N b}.$$
Sketch of Proof

Proof.

The proof of Proposition 1 is straightforward using 1. Let \( k \) in \( \mathbb{N} \), by 1 we have:

\[
\| \hat{w}_k - w_k \|^2 = \left\| \left( w_{k-1} - \gamma C_{\text{dwn}} \left( \frac{1}{N} \sum_{i=1}^{N} \hat{g}_k^i(\hat{w}_{k-1}) \right) \right) - \left( w_{k-1} - \gamma \frac{1}{N} \sum_{i=1}^{N} \hat{g}_k^i(\hat{w}_{k-1}) \right) \right\|^2
\]

\[
= \gamma^2 \left\| C_{\text{dwn}} \left( \frac{1}{N} \sum_{i=1}^{N} \hat{g}_k^i(\hat{w}_{k-1}) \right) - \frac{1}{N} \sum_{i=1}^{N} \hat{g}_k^i(\hat{w}_{k-1}) \right\|^2
\]

Taking expectation w.r.t. down compression, as \( \frac{1}{N} \sum_{i=1}^{N} \hat{g}_k^i(\hat{w}_{k-1}) \) is \( w_k \)-measurable:

\[
\mathbb{E} \left[ \| w_k - \hat{w}_k \|^2 \mid w_k \right] = \gamma^2 \omega_{\text{dwn}} \mathbb{E} \left[ \left\| \frac{1}{N} \sum_{i=1}^{N} \hat{g}_k^i(\hat{w}_{k-1}) \right\|^2 \mid w_k \right] = \gamma^2 \omega_{\text{dwn}} \left\| \hat{g}_k \right\|^2,
\]

then we do a Bias Variance decomposition.

\( \Downarrow \) the variance of the local model is bounded by an affine function of the squared norm of the previous stochastic gradients \( \nabla F(\hat{w}_{k-1}) \).
Sketch of proof, 2

Then, classical perturbed iterate approach \([MPP^{+}16]\),

\[
\mathbb{E} \| w_k - w_* \|^2 = \mathbb{E} \| w_{k-1} - w_* \|^2 - 2\gamma \mathbb{E} \langle \nabla F(\hat{w}_{k-1}) \mid w_{k-1} - w_* \rangle + \gamma^2 \mathbb{E} \left[ \| g_k(\hat{w}_{k-1}) \|^2 \right].
\]

Moreover,

\[
-2\gamma \mathbb{E} \langle \nabla F(\hat{w}_{k-1}) \mid w_{k-1} - w_* \rangle = -2\gamma \mathbb{E} \langle \nabla F(\hat{w}_{k-1}) \mid \hat{w}_{k-1} - w_* \rangle
\]

\[
+ 2\gamma \mathbb{E} \langle \nabla F(\hat{w}_{k-1}) - \nabla F(w_{k-1}) \mid w_{k-1} - \hat{w}_{k-1} \rangle.
\]

as \( \mathbb{E} [\hat{w}_{k-1} \mid w_{k-1}] = w_{k-1} \).
Sketch of proof, 2

Then, classical perturbed iterate approach [MPP⁺16],

\[ \mathbb{E} \| w_k - w_* \|^2 = \mathbb{E} \| w_{k-1} - w_* \|^2 - 2\gamma \mathbb{E} \langle \nabla F(\hat{w}_{k-1}) \mid w_{k-1} - w_* \rangle + \gamma^2 \mathbb{E} \left[ \| \hat{g}_k(\hat{w}_{k-1}) \|^2 \right]. \]

Moreover,

\[ -2\gamma \mathbb{E} \langle \nabla F(\hat{w}_{k-1}) \mid w_{k-1} - w_* \rangle = -2\gamma \mathbb{E} \langle \nabla F(\hat{w}_{k-1}) \mid \hat{w}_{k-1} - w_* \rangle + 2\gamma \mathbb{E} \langle \nabla F(\hat{w}_{k-1}) - \nabla F(w_{k-1}) \mid w_{k-1} - \hat{w}_{k-1} \rangle. \]

as \( \mathbb{E} [\hat{w}_{k-1} \mid w_{k-1}] = w_{k-1} \).

1. \(-2\gamma \mathbb{E} \langle \nabla F(\hat{w}_{k-1}) \mid \hat{w}_{k-1} - w_* \rangle \) “strong contraction”, upper bounded by
   - \(-2\gamma (\mu \| \hat{w}_{k-1} - w_* \|^2 + F(\hat{w}_{k-1}) - F_* ) \)
   - \(-2\gamma \| \nabla F(\hat{w}_{k-1}) \|^2 / L \)

2. \( 2\gamma \mathbb{E} \langle \nabla F(\hat{w}_{k-1}) - \nabla F(w_{k-1}) \mid w_{k-1} - \hat{w}_{k-1} \rangle \) positive residual term.

Theorem 2 (Contraction for Ghost, convex case)

\[ \mathbb{E} \| w_k - w_* \|^2 \leq \mathbb{E} \| w_{k-1} - w_* \|^2 - \gamma \mathbb{E} \left( F(w_{k-1}) - F_* \right) - \frac{\gamma}{2L} \mathbb{E} \left[ \| \nabla F(\hat{w}_{k-1}) \|^2 \right] \]

\[ + 2\gamma^3 \omega_{\text{dwn}} L \left( 1 + \frac{\omega_{\text{up}}}{N} \right) \mathbb{E} \| \nabla F(\hat{w}_{k-2}) \|^2 + \gamma^2 \frac{(1 + \omega_{\text{up}}) \sigma^2}{Nb} \left( 1 + 2\gamma L \omega_{\text{dwn}} \right). \]
Theoretical results - Ghost

Contraction for Ghost

Theorem 3 (Contraction for Ghost, convex case)

Under Assumptions 1 and 3 to 5, with $\mu = 0$, if $\gamma L(1 + \omega_{up}/N) \leq \frac{1}{2}$,

$$\mathbb{E}\|w_k - w_*\|^2 \leq \mathbb{E}\|w_{k-1} - w_*\|^2 - \gamma \mathbb{E}(F(w_{k-1}) - F_*) - \frac{\gamma}{2L} \mathbb{E}\|\nabla F(\hat{w}_{k-1})\|^2$$

$$+ 2\gamma^3 \omega_{dwn} L \left(1 + \frac{\omega_{up}}{N}\right) \mathbb{E}\|\nabla F(\hat{w}_{k-2})\|^2 + \gamma^2 \frac{(1 + \omega_{up})\sigma^2}{Nb} (1 + 2\gamma L \omega_{dwn}).$$

We can make the following observations:

1. At step $k$, the residual can be upper bounded by a constant times squared norm of the gradient at point $\hat{w}_{k-2}$.
2. If $2\gamma^3 \omega_{dwn} L(1 + \omega_{up}/N) \leq \gamma/(2L)$, then these terms eventually cancel out.
3. This is equivalent to $2\gamma L \sqrt{\omega_{dwn}} (1 + \omega_{up}/N) \leq 1$. It is natural to chose $\gamma \leq 1/(2L \max(1 + \omega_{up}/N, 1 + \omega_{dwn}))$.

Line of proof is the same for strongly convex, but different for non-convex.
Noise level, Ghost

Theorem 4 (Contraction for Ghost, convex case)

Under Assumptions 1 and 3 to 5, with \( \mu = 0 \), if \( \gamma L(1 + \omega_{up}/N) \leq \frac{1}{2} \),

\[
\mathbb{E}\|w_k - w_*\|^2 \leq \mathbb{E}\|w_{k-1} - w_*\|^2 - \gamma \mathbb{E}(F(w_{k-1}) - F_*) - \frac{\gamma}{2L} \mathbb{E}\left[\|\nabla F(\hat{w}_{k-1})\|^2\right] \\
+ 2\gamma^3 \omega_{dwn} L \left(1 + \frac{\omega_{up}}{N}\right) \mathbb{E}\|\nabla F(\hat{w}_{k-2})\|^2 + \gamma^2 \frac{(1 + \omega_{up})\sigma^2}{Nb} (1 + 2\gamma L\omega_{dwn}).
\]

For Ghost algorithm

\[
\gamma^2 \frac{(1 + \omega_{up})\sigma^2}{Nb} (1 + 2\gamma L\omega_{dwn}).
\]

For classical double compression

\[
\gamma^2 \omega_{dwn} \frac{(1 + \omega_{up})\sigma^2}{Nb}.
\]

For unidirectional-compression

\[
\gamma^2 \frac{(1 + \omega_{up})\sigma^2}{Nb}.
\]
A practical algorithm?

Summary:

1. For a hypothetical iterate, we can obtain convergence in the “preserved central iterate” framework.
2. The limit Variance is nearly of the same order as with simple compression.
3. This algorithm cannot be implemented in practice!
A practical algorithm?

Summary:

1. For a hypothetical iterate, we can obtain convergence in the “preserved central iterate” framework
2. The limit Variance is nearly of the same order as with simple compression.
3. This algorithm cannot be implemented in practice!

New attempts:

Ghost

\[
w_k = w_{k-1} - \gamma \left( \frac{1}{N} \sum_{i=1}^{N} C_{up} (g^i_k(\hat{w}_{k-1})) \right)
\]

\[
\hat{w}_k = w_{k-1} - \gamma C_{dwn} \left( \frac{1}{N} \sum_{i=1}^{N} C_{up} (g^i_k(\hat{w}_{k-1})) \right)
\]

Update compression

\[
w_k = w_{k-1} - \gamma \left( \frac{1}{N} \sum_{i=1}^{N} C_{up} (g^i_k(\hat{w}_{k-1})) \right)
\]

\[
\hat{w}_k = \hat{w}_{k-1} - \gamma C_{dwn} \left( \frac{1}{N} \sum_{i=1}^{N} C_{up} (g^i_k(\hat{w}_{k-1})) \right)
\]

Model compression ($\alpha_{dwn} = 0$)

\[
w_k = w_{k-1} - \gamma \left( \frac{1}{N} \sum_{i=1}^{N} C_{up} (g^i_k(\hat{w}_{k-1})) \right)
\]

\[
\hat{w}_k = C_{dwn} (w_k)
\]

Model difference compression ($\alpha_{dwn} = 1$)

\[
w_k = w_{k-1} - \gamma \left( \frac{1}{N} \sum_{i=1}^{N} C_{up} (g^i_k(\hat{w}_{k-1})) \right)
\]

\[
\hat{w}_k = \hat{w}_{k-1} - \gamma C_{dwn} (w_k - \hat{w}_{k-1})
\]
First attempts - Variance of the local iterate is too high.

- **Update compression**
- **Model difference compression** ($\alpha_{dwn} = 1$)
- **Model compression** ($\alpha_{dwn} = 0$)
- **MCM**

**Figure:** Comparing MCM on two datasets with three other algorithms using a non-degraded update, $\gamma = 1/L$. 
The downlink memory mechanism for MCM

We introduce a *downlink memory term* \((H_k)_{k \in \mathbb{N}}\):

1. available on both workers and central server
2. the difference \(\Omega_{k+1}\) between the model and this memory is compressed and exchanged
3. the local model is reconstructed from this information

\[
\begin{align*}
    w_k & = w_{k-1} - \gamma \left( \frac{1}{N} \sum_{i=1}^{N} C_{\text{up}}(g^i_k(\hat{w}_{k-1})) \right) \\
    \Omega_{k+1} & = w_{k+1} - H_k \\
    \hat{w}_{k+1} & = H_k + C_{\text{dwn}}(\Omega_{k+1}) \\
    H_{k+1} & = H_k + \alpha_{\text{dwn}} C_{\text{dwn}}(\Omega_{k+1}).
\end{align*}
\]

Introducing this memory mechanism is crucial to control the variance of the local model \(\hat{w}_{k+1}\).
Convergence theorem

Theorem 5 (Convergence of MCM, convex case)

Under all previous assumptions, for $K$ in $\mathbb{N}$, with a large enough step-size $\gamma = \sqrt{\frac{\delta_0^2 N_b}{(1+\omega_{up})\sigma^2 K}}$, denoting $\bar{w}_K = \frac{1}{K} \sum_{i=0}^{K-1} w_i$, we have:

$$
\mathbb{E} [F(\bar{w}_K) - F_\ast] \leq 2 \sqrt{\frac{\delta_0^2 (1 + \omega_{up}) \sigma^2}{N_b K}} \ + O\left(\frac{\omega_{up} \omega_{dwn}}{K}\right).
$$
Convergence theorem

**Theorem 5 (Convergence of MCM, convex case)**

*Under all previous assumptions, for $K$ in $\mathbb{N}$, with a large enough step-size*

\[
\gamma = \sqrt{\frac{\delta_0^2 Nb}{(1+\omega_{up})\sigma^2 K}}, \text{ denoting } \bar{w}_K = \frac{1}{K} \sum_{i=0}^{K-1} w_i, \text{ we have:}
\]

\[
\mathbb{E} [F(\bar{w}_K) - F_\star] \leq 2 \sqrt{\frac{\delta_0^2 (1 + \omega_{up})\sigma^2}{NbK}} + O\left(\frac{\omega_{up}\omega_{dwn}}{K}\right) .
\]

*dominant term*  \quad *lower order term*
Convergence theorem

Theorem 5 (Convergence of MCM, convex case)

Under all previous assumptions, for $K$ in $\mathbb{N}$, with a large enough step-size

$\gamma = \sqrt{\frac{\delta_0^2 N b}{(1+\omega_{up})\sigma^2 K}}$, denoting $\bar{w}_K = \frac{1}{K} \sum_{i=0}^{K-1} w_i$, we have:

$$
\mathbb{E} [F(\bar{w}_K) - F_*] \leq 2 \sqrt{\frac{\delta_0^2 (1 + \omega_{up})\sigma^2}{N b K}} + O\left(\frac{\omega_{up}\omega_{dwn}}{K}\right).
$$

- independent of $\omega_{dwn}$
- identical to Diana (uni-compression)
- depends on $\omega_{dwn}$
- asymptotically negligible
Convergence theorem

Theorem 5 (Convergence of MCM, convex case)

Under all previous assumptions, for $K$ in $\mathbb{N}$, with a large enough step-size

$$\gamma = \sqrt{\frac{\delta_0^2 N_b}{(1 + \omega_{up})\sigma^2 K}},$$

denoting $\bar{w}_K = \frac{1}{K} \sum_{i=0}^{K-1} w_i$, we have:

$$\mathbb{E}[F(\bar{w}_K) - F_*] \leq 2\sqrt{\frac{\delta_0^2 (1 + \omega_{up})\sigma^2}{N_b K}} + O\left(\frac{\omega_{up}\omega_{up}}{K}\right).$$

Moreover if $\sigma^2 = 0$, we recover a faster convergence:

$$\mathbb{E}[F(\bar{w}_K) - F_*] = O(K^{-1}).$$
Convergence theorem

Theorem 5 (Convergence of MCM, convex case)

Under all previous assumptions, for $K$ in $\mathbb{N}$, with a large enough step-size

$$\gamma = \sqrt{\frac{\delta_0^2 Nb}{(1+\omega_{up})\sigma^2 K}},$$
denoting $\bar{w}_K = \frac{1}{K} \sum_{i=0}^{K-1} w_i$, we have:

$$\mathbb{E} [F(\bar{w}_K) - F_*] \leq 2 \sqrt{\frac{\delta_0^2 (1 + \omega_{up}) \sigma^2}{NbK}} + O\left(\frac{\omega_{up} \omega_{up}}{K}\right). \quad (4)$$

Moreover if $\sigma^2 = 0$, we recover a faster convergence:

$$\mathbb{E}[F(\bar{w}_K) - F_*] = O(K^{-1}).$$

Remark: Theorem 7 is also extended to both strongly-convex and non-convex cases.

More details in the following slides for those who are interested.
Control of the local Variance

Let $\gamma_k := \|w_k - H_{k-1}\|^2$.

**Theorem 6**

*Consider the MCM update. Under Assumptions 1, 3 and 5 with $\mu = 0$, if $\gamma \leq (8\omega_{dwn}L)^{-1}$ and $\alpha_{dwn} \leq (4\omega_{dwn})^{-1}$, then for all $k$ in $\mathbb{N}$:

$$
\mathbb{E}[\gamma_k] \leq \left(1 - \frac{\alpha_{dwn}}{2}\right) \mathbb{E}[\gamma_{k-1}] + 2\gamma^2 \left(\frac{1}{\alpha_{dwn}} + \frac{\omega_{up}}{N}\right) \mathbb{E}[\|\nabla F(\hat{w}_{k-1})\|^2] + \frac{2\gamma^2 \sigma^2 (1 + \omega_{up})}{Nb}.
$$

*
Convergence of MCM - Convex

Let

1. \( V_k = \mathbb{E}[\|w_k - w_*\|^2] + 32\gamma L \omega_{dwn}^2 \mathbb{E}[\gamma_k] \).
2. \( \Phi(\gamma) := (1 + \omega_{up})(1 + 64\gamma L \omega_{dwn}^2) \).

Consequently, for \( K \) in \( \mathbb{N} \) large enough, a step-size \( \eta = \sqrt{\mathbb{E}[\|w_0 - w_*\|^2] + \Phi(\gamma)} \) we have:

\[ \mathbb{E}[\|w_K - w_*\|^2] \leq O(K^{-1}). \]

Moreover if \( \gamma = 0 \), we recover a faster convergence:

\[ \mathbb{E}[\|w_K - w_*\|^2] = O(K^{-1}). \]
**Convergence of MCM - Convex**

Let

1. \( V_k = \mathbb{E}[\|w_k - w_*\|^2] + 32\gamma L\omega_{dwn}^2 \mathbb{E}[\gamma_k] \).
2. \( \Phi(\gamma) := (1 + \omega_{up})(1 + 64\gamma L\omega_{dwn}^2) \).

**Theorem 7 (Convergence of MCM, convex case)**

*Under Assumptions 1 and 3 to 5 with \( \mu = 0 \). For all \( k > 0 \), for any \( \gamma \leq \gamma_{\text{max}} \), we have, for \( \bar{w}_k = \frac{1}{k} \sum_{i=0}^{k-1} w_i \), *

\[
\gamma \mathbb{E} [F(w_{k-1}) - F(w_*)] \leq V_{k-1} - V_k + \frac{\gamma^2 \sigma^2 \Phi(\gamma)}{Nb} \implies \mathbb{E}[F(\bar{w}_k) - F_*] \leq \frac{V_0}{\gamma k} + \frac{\gamma \sigma^2 \Phi(\gamma)}{Nb}.
\]
Convergence of MCM - Convex

Let

\[ V_k = \mathbb{E}[\|w_k - w_*\|^2] + 32\gamma L\omega_{dwn}^2 \mathbb{E}[\gamma_k]. \]

\[ \Phi(\gamma) := (1 + \omega_{up}) (1 + 64\gamma L\omega_{dwn}^2). \]

Theorem 7 (Convergence of MCM, convex case)

Under Assumptions 1 and 3 to 5 with \( \mu = 0 \). For all \( k > 0 \), for any \( \gamma \leq \gamma_{\text{max}} \), we have, for \( \tilde{w}_k = \frac{1}{k} \sum_{i=0}^{k-1} w_i \),

\[ \gamma \mathbb{E}[F(w_{k-1}) - F(w_*)] \leq V_{k-1} - V_k + \frac{\gamma^2 \sigma^2 \Phi(\gamma)}{Nb} \implies \mathbb{E}[F(\tilde{w}_k) - F_*] \leq \frac{V_0}{\gamma k} + \frac{\gamma \sigma^2 \Phi(\gamma)}{Nb}. \]

Consequently, for \( K \) in \( \mathbb{N} \) large enough, a step-size \( \gamma = \sqrt{\frac{\|w_0 - w_*\|^2 Nb}{(1+\omega_{up})\sigma^2 K}} \), we have,

\[ \mathbb{E}[F(\tilde{w}_K) - F_*] \leq 2\sqrt{\frac{\|w_0 - w_*\|^2 (1 + \omega_{up})\sigma^2}{NbK}} + O(K^{-1}). \]

Moreover if \( \sigma^2 = 0 \), we recover a faster convergence: \( \mathbb{E}[F(\tilde{w}_K) - F_*] = O(K^{-1}). \)
Comparison to previous results: Limit Variance

Better limit variance $\Rightarrow$ better rate.

For a constant $\gamma$,

1. the variance term is upper bounded by

$$\frac{\gamma^2\sigma^2}{Nb}(1 + \omega_{up})(1 + 64\gamma L\omega_{dwn}^2).$$

2. impact of the downlink compression is attenuated by a factor $\gamma$. As $\gamma \to 0$ we get close to Diana, i.e., without downlink compression [MGTR19, Eq. 16 in Th. 2]

$$\frac{\gamma^2\sigma^2}{Nb}(1 + \omega_{up}).$$

3. This is much lower than the variance for previous algorithms using double compression for

$$\gamma^2\sigma^2(1 + \omega_{up})(1 + \omega_{dwn})/N$$

- for Dore, see Corollary 1 in Liu et al. [LLTY20] (who indicate $(1 - \rho)^{-1} \geq (1 + \omega_{up}/N)(1 + \omega_{dwn}))$,
- for Artemis see Table 2 and Th. 3 point 2 in [PD20],
- for Gorbunov et al. [GKMR20], see Theorem I.1. (with $\gamma D_1 \propto \gamma^2\sigma^2(1 + \omega_{up})(1 + \omega_{dwn})/N$).
Comparison to previous results: **Limit learning rate**

**Limit learning rate:** Maximal learning rate to ensure convergence.

\[ \gamma_{\text{max}} := \min(\gamma_{\text{up}}, \gamma_{\text{max}}^{\text{dwn}}, \gamma_{\text{max}}^{\uparrow}), \]

- \( \gamma_{\text{max}}^{\text{up}} := (2L (1 + \omega_{\text{up}}/N))^{-1} \) corresponds to the classical constraint on the learning rate in the unidirectional regime [see MGTR19, PD20],
- \( \gamma_{\text{max}}^{\text{dwn}} := (8L\omega_{\text{dwn}})^{-1} \) is a similar constraint coming from the downlink compression,
- \( \gamma_{\text{max}}^{\uparrow} := (8\sqrt{2}L\omega_{\text{dwn}}\sqrt{8\omega_{\text{dwn}} + \omega_{\text{up}}/N})^{-1} \) is a combined constraint that arises when controlling the variance term \( \gamma \).

Remarks

- weaker constraints than in the “degraded” framework [LLTY20, PD20], in which \( \gamma_{\text{max}}^{\text{Dore}} \leq (8L(1 + \omega_{\text{dwn}})(1 + \omega_{\text{up}}/N))^{-1} \).
- e.g., if \( \omega_{\text{up,dwn}} \to \infty \) and \( \omega_{\text{dwn}} \simeq \omega_{\text{up}} \simeq \omega \), the maximal learning rate for MCM is \( (L\omega^{3/2})^{-1} \), while it is \( (L\omega^{2})^{-1} \) in [LLTY20, PD20]. Our \( \gamma_{\text{max}} \) is thus larger by a factor \( \sqrt{\omega} \).

---

\[ ^1 \]The dependency in \( \omega^{3/2} \) is similar to the one obtained by Horváth [HKM\dagger 19] in unidirectional compression in the non-convex case (Theorem 4).
Convergence of MCM - Strongly Convex

We define $\tilde{L}$ such that $\gamma_{\max} = (2\tilde{L})^{-1}$.

Theorem 8 (Convergence of MCM in the homogeneous and strongly-convex case)

Under Assumptions 1 and 3 to 5 with $\mu > 0$, for $k$ in $\mathbb{N}$, for any sequence $(\gamma_k)_{k \geq 0} \leq \gamma_{\max}$:

$$V_k \leq (1 - \gamma_k \mu) V_{k-1} - \gamma_k \mathbb{E} [F(\hat{w}_{k-1}) - F(w_*)] + \frac{\gamma_k^2 \sigma^2 \Phi(\gamma_k)}{Nb},$$

where $\Phi(\gamma_k) = (1 + \omega_{\text{up}}) (1 + 64\gamma_k L \omega_{\text{dwn}}^2)$. Consequently,

1. if $\sigma^2 = 0$ (noiseless case), for $\gamma_k \equiv \gamma_{\max}$ we recover a linear convergence rate:
   $$\mathbb{E} [\|w_k - w_*\|^2] \leq (1 - \gamma_{\max} \mu)^k V_0;$$

2. if $\sigma^2 > 0$, taking for all $K$ in $\mathbb{N}$, $\gamma_K = 2/((\mu (K + 1) + \tilde{L})$, for the weighted Polyak-Ruppert average $\bar{w}_K = \sum_{k=1}^{K} \lambda_k w_{k-1}/ \sum_{k=1}^{K} \lambda_k$, with $\lambda_k : = (\gamma_{k-1})^{-1}$,

$$\mathbb{E} [F(\bar{w}_K) - F(w_*)] \leq \frac{\mu + 2\tilde{L}}{4\mu K^2} \|w_0 - w_*\|^2 + \frac{4\sigma^2 (1 + \omega_{\text{up}})}{\mu KNb} \left(1 + \frac{64L \omega_{\text{dwn}}^2}{\mu K} \ln(\mu K + \tilde{L}) \right).$$
Summary of rates and complexities

Summary of rates. In this Table, we summarize the rates and complexities, and maximal learning rate for Diana, Artemis, Dore and MCM. For simplicity, we ignore absolute constants, and provide asymptotic values for large $\omega_{\text{up}}$, $\omega_{\text{dwn}}$, and complexities for $\epsilon \to 0$.

Table: Summary of rates on the initial condition, limit variance, asympt. complexities and $\gamma_{\text{max}}$.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Diana</th>
<th>Artemis, Dore</th>
<th>MCM, Rand-MCM</th>
</tr>
</thead>
<tbody>
<tr>
<td>$L \gamma_{\text{max}} \propto$</td>
<td>$1/(1 + \omega_{\text{up}})$</td>
<td>$1/(1 + \omega_{\text{up}})(1 + \omega_{\text{dwn}})$</td>
<td>$1/(1 + \omega_{\text{dwn}}) \sqrt{1 + \omega_{\text{up}}} \wedge 1/(1 + \omega_{\text{up}})$</td>
</tr>
<tr>
<td>Lim. var. $\propto \gamma^2 \sigma^2 / n \times$</td>
<td>$(1 + \omega_{\text{up}})$</td>
<td>$(1 + \omega_{\text{up}})(1 + \omega_{\text{dwn}})$</td>
<td>$(1 + \omega_{\text{up}})(1 + \gamma L \omega_{\text{dwn}}^2)$</td>
</tr>
<tr>
<td>Str.-convex Rate on init. cond. (SC)</td>
<td>$(1 - \gamma \mu)^k$</td>
<td>$(1 - \gamma \mu)^k$</td>
<td>$(1 - \gamma \mu)^k$</td>
</tr>
<tr>
<td>Complexity</td>
<td>$(1 + \omega_{\text{up}}) / \mu \epsilon N$</td>
<td>$(1 + \omega_{\text{dwn}})(1 + \omega_{\text{up}}) / \mu \epsilon N$</td>
<td>$(1 + \omega_{\text{up}}) / \mu \epsilon N$</td>
</tr>
<tr>
<td>Convex Complexity</td>
<td>$(\omega_{\text{up}} + 1) / \epsilon^2$</td>
<td>$(1 + \omega_{\text{up}})(1 + \omega_{\text{dwn}}) / \epsilon^2$</td>
<td>$(\omega_{\text{up}} + 1) / \epsilon^2$</td>
</tr>
</tbody>
</table>
### Double compression: summary of related work

**Table:** Features of the main existing algorithms performing compression. $e^i_k$ (resp. $E_k$) denotes the use of error-feedback at uplink (resp. downlink). $h^i_k$ (resp. $H_k$) denotes the use of a memory at uplink (resp. downlink). Note that Dist-EF-SGD is identical to Double-Squeeze but has been developed simultaneously and independently.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Compr.</th>
<th>$e^i_k$</th>
<th>$h^i_k$</th>
<th>$E_k$</th>
<th>$H_k$</th>
<th>Rand.</th>
<th>update point</th>
</tr>
</thead>
<tbody>
<tr>
<td>Qsgd [AGL+17]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>one-way</td>
</tr>
<tr>
<td>ECQ-sgd [WHHZ18]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td>one-way</td>
</tr>
<tr>
<td>Diana [MGTR19]</td>
<td></td>
<td></td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
<td>one-way</td>
</tr>
<tr>
<td>Dore [LLTY20]</td>
<td>two-way</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>degraded</td>
</tr>
<tr>
<td>Double-Squeeze [TYL+19], Dist-EF-SGD [ZHK19]</td>
<td>two-way</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>degraded</td>
</tr>
<tr>
<td>Artemis [PD20]</td>
<td>two-way</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td>degraded</td>
</tr>
<tr>
<td>Doubly compressed SGD [GKMR20]</td>
<td>two-way</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>degraded</td>
</tr>
<tr>
<td>MCM</td>
<td>two-way</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>non-degraded</td>
</tr>
<tr>
<td>Rand-MCM</td>
<td>two-way</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>non-degraded</td>
</tr>
</tbody>
</table>

Precise comparison of convergence results will be given afterwards.
Extensions - and partial take away

1. Heterogeneous framework: previous theorems are valid in the heterogeneous framework (at the cost of a constant 2), under Assumption 2.

2. Another theorem is provided in the non-convex regime, with similar take-away.

Take away:

1. **MCM** = Model Compression with memory

2. Uses a **memory** on the downlink direction, as introduced by Mishchenko [MGTR19] for the uplink.

3. Leverages the unbiased-ness of $\hat{w}_k$ around $w_k$. 
Next step: worker dependent downlink compression: Rand-MCM!

No (or few) reasons to use the same compression for all workers!

\[
\begin{aligned}
  w_k &= w_{k-1} - \gamma \left( \frac{1}{N} \sum_{i=1}^{N} C_{up}(g^i_k(\hat{w}^i_{k-1})) \right) \\
  \Omega_{k+1} &= w_{k+1} - H_k \\
  \hat{w}^i_{k+1} &= H^i_k + C_{dwn}^i(\Omega_{k+1}) \\
  H^i_{k+1} &= H^i_k + \alpha_{dwn} C_{dwn}^i(\Omega_{k+1}).
\end{aligned}
\]  

(5)

Advantages:

1. Independence could help reduce the variance
2. Workers can be allowed to choose the size (or equivalently the compression level) of their updates.
3. Helps in case of Partial Participation
4. Could be leveraged to tackle honest-but-curious clients.
Next step: worker dependent downlink compression: Rand-MCM!

No (or few) reasons to use the same compression for all workers!

\[
\begin{align*}
    w_k &= w_{k-1} - \gamma \left( \frac{1}{N} \sum_{i=1}^{N} C_{\text{up}}(g^i_k(\hat{w}^i_{k-1})) \right) \\
    \Omega_{k+1} &= w_{k+1} - H_k \\
    \hat{w}^i_{k+1} &= H^i_k + C_{\text{dwn},i}(\Omega_{k+1}) \\
    H^i_{k+1} &= H^i_k + \alpha_{\text{dwn}} C_{\text{dwn},i}(\Omega_{k+1}).
\end{align*}
\] (5)

Advantages:

1. Independence could help reduce the variance
2. Workers can be allowed to choose the size (or equivalently the compression level) of their updates.
3. Helps in case of Partial Participation
4. Could be leveraged to tackle honest-but-curious clients.

Drawbacks

1. Storing the \( N \) memories \( (H^i_k)_{i \in [N]} \) instead of one
Next step: worker dependent downlink compression: Rand-MCM!

No (or few) reasons to use the same compression for all workers!

\[
\begin{align*}
    w_k &= w_{k-1} - \gamma \left( \frac{1}{N} \sum_{i=1}^{N} C_{up}(g_k^i(\hat{w}_{k-1}^i)) \right) \\
    \Omega_{k+1} &= w_{k+1} - H_k \\
    \hat{w}_{k+1}^j &= H_k^j + C_{dwn,i}(\Omega_{k+1}) \\
    H_{k+1}^j &= H_k^j + \alpha_{dwn} C_{dwn,i}(\Omega_{k+1}).
\end{align*}
\]

Advantages:

1. Independence could help reduce the variance
2. Workers can be allowed to choose the size (or equivalently the compression level) of their updates.
3. Helps in case of Partial Participation
4. Could be leveraged to tackle honest-but-curious clients.

Drawbacks

1. Storing the \( N \) memories \((H_k^i)_{i \in [N]}\) instead of one

Solutions:

1. Keep and use a single memory \( \bar{H}_k = N^{-1} \sum_{i=1}^{N} H_k^i \).
   - It is then necessary to periodically reset the local memories \( H_k^i \) on all workers to the averaged value \( \bar{H}_k \) (rarely enough not to impact the communication budget)
2. Use Rand-MCM with an arbitrary number of groups \( G \ll N \) of workers. In each group \( g \in [G] \), all workers share the same memory \( (H_k^g) \) and receive the same update \( C_{dwn,g}(w_{k+1} - H_k^g) \). We call this algorithm Rand-MCM-G.
Convergence of Rand-MCM

1. At least as good:

Theorem 9

Theorems 6 to 8 are valid for Rand-MCM and Rand-MCM-G.
Convergence of Rand-MCM

1. At least as good:

Theorem 9

Theorems 6 to 8 are valid for Rand-MCM and Rand-MCM-G.

2. Better on residual term:

Theorem 10 (Convergence in the quadratic case)

Under Assumptions 1 and 3 to 5 with $\mu = 0$, if the function is quadratic, after running $K > 0$ iterations, for any $\gamma \leq \gamma_{\text{max}}$, and we have

$$
\mathbb{E}[F(\tilde{w}_K) - F_*] \leq \frac{V_0}{\gamma K} + \frac{\gamma \sigma^2 \Phi^{Rd}(\gamma)}{Nb},
$$

with $\Phi^{Rd}(\gamma) = (1 + \omega_{\text{up}}) \left(1 + \frac{4\gamma^2 L^2 \omega_{\text{dwn}}}{K} \left(\frac{1}{C} + \frac{\omega_{\text{up}}}{N}\right)\right)$ and $C = N$ for Rand-MCM, $C = G$ Rand-MCM-G, and $C = 1$ for MCM.

Extending the proof beyond quadratic functions is possible, though it requires an assumption on third or higher order derivatives of $F$ (e.g., using self-concordance [Bac10]) to control of $\mathbb{E} \left[\|\nabla F(\hat{w}_{k-1}) - \mathbb{E}[\nabla F(\hat{w}_{k-1})]\|^2 \mid w_{k-1}\right]$. 
Figure: Quantum with $b = 400, \gamma = 1/L$ (LSR).
More experiments

Figure: Convergence on toy dataset on LSR (a,b) and on neural networks (c, d).

(a) $\sigma^2 \neq 0$, $\gamma = (L\sqrt{k})^{-1}$

(b) $\sigma^2 = 0$, $\gamma = L^{-1}$

(c) MNIST with a CNN

(d) CIFAR10 with LeNet

(a) Quantum in #iter. (b) Quantum in #bits (c) Superconduct in #iter. (d) Superconduct in #bits
More experiments (convex)

<table>
<thead>
<tr>
<th></th>
<th>Excess loss after 450 epochs</th>
<th>SGD</th>
<th>DIANA</th>
<th>MCM</th>
<th>DORE</th>
</tr>
</thead>
<tbody>
<tr>
<td>a9a b=50</td>
<td></td>
<td>−3.5</td>
<td>−2.7</td>
<td>−2.7</td>
<td>−1.8</td>
</tr>
<tr>
<td>Phishing b=50</td>
<td></td>
<td>−3.7</td>
<td>−3.5</td>
<td>−3.4</td>
<td>−2.7</td>
</tr>
<tr>
<td>w8a b=8</td>
<td></td>
<td>−3.5</td>
<td>−3.0</td>
<td>−2.5</td>
<td>−1.75</td>
</tr>
<tr>
<td>Compression</td>
<td></td>
<td>no</td>
<td>uni-dir</td>
<td>bi-dir</td>
<td>bi-dir</td>
</tr>
</tbody>
</table>

Aymeric DIEULEVEUT

MCM & DP-Scaffold
### More experiments, non convex

<table>
<thead>
<tr>
<th>Nonconvex framework</th>
<th>MNIST (CNN, $d=2e4$, 2 bits-quantization with norm 2)</th>
<th>Fashion MNIST (FashionSimpleNet, $d=4e5$, 2 bits-quantization with norm 2)</th>
<th>Heterogeneous EMNIST (CNN, $d=2e4$, 2 bits-quantization with norm 2)</th>
<th>CIFAR-10 (LeNet, $d=62e3$, 2-bits-quantization)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline accuracy for the selected network [Ref]</td>
<td></td>
<td>92.3% [Link]</td>
<td></td>
<td>67.52% [Link]</td>
</tr>
<tr>
<td>Accuracy after 300 epochs</td>
<td>SGD: 99.0%</td>
<td>SGD: 92.4%</td>
<td>SGD: 99.0%</td>
<td>SGD: 69.1%</td>
</tr>
<tr>
<td>Diana: 98.9%</td>
<td>Diana: 92.4%</td>
<td>Diana: 98.9%</td>
<td>Diana: 64.0%</td>
<td></td>
</tr>
<tr>
<td>MCM: 98.8%</td>
<td>MCM: 90.6%</td>
<td>MCM: 98.9%</td>
<td>MCM: 63.5%</td>
<td></td>
</tr>
<tr>
<td>Artemis: 97.9%</td>
<td>Artemis: 86.7%</td>
<td>Artemis: 98.3%</td>
<td>Artemis: 54.8%</td>
<td></td>
</tr>
<tr>
<td>Dore: 97.9%</td>
<td>Dore: 87.9%</td>
<td>Dore: 98.5%</td>
<td>Dore: 56.3%</td>
<td></td>
</tr>
<tr>
<td>Train loss after 300 epochs</td>
<td>SGD: 0.025</td>
<td>SGD: 0.093</td>
<td>SGD: 0.026</td>
<td>SGD: 0.909</td>
</tr>
<tr>
<td>Diana: 0.034</td>
<td>Diana: 0.141</td>
<td>Diana: 0.031</td>
<td>Diana: 1.047</td>
<td></td>
</tr>
<tr>
<td>MCM: 0.033</td>
<td>MCM: 0.209</td>
<td>MCM: 0.030</td>
<td>MCM: 1.096</td>
<td></td>
</tr>
<tr>
<td>Artemis: 0.075</td>
<td>Artemis: 0.332</td>
<td>Artemis: 0.052</td>
<td>Artemis: 1.342</td>
<td></td>
</tr>
<tr>
<td>Dore: 0.072</td>
<td>Dore: 0.300</td>
<td>Dore: 0.048</td>
<td>Dore: 1.292</td>
<td></td>
</tr>
</tbody>
</table>
Experiments: Randomization + single memory.

Figure: Rand-MCM (PP) on quantum with a single memory ($s = 2$).
Conclusion and open directions

**MCM** underlines the importance to not degrade the global model.

**Summary:**
1. New algorithm for bi-directional compression with a **preserved** central model
2. Reduces (nearly cancels) impact of downlink compression
3. Achieves the same asymptotic rate of convergence as unidirectional compression.
Conclusion and open directions

**MCM underlines the importance to not degrade the global model.**

**Summary:**
- New algorithm for bi-directional compression with a preserved central model
- Reduces (nearly cancels) impact of downlink compression
- Achieves the same asymptotic rate of convergence as unidirectional compression.

**Open directions:**
- Can we provably benefit from the smoothing effect?
- Extending proofs of Rand-MCM to the self-concordant framework
- Leveraging the randomization effect in applications
- Even better double compression:
  - combination with better techniques on the up-link direction
  - unaffected $\gamma_{\text{max}}$
  - biased compression operators.
Conclusion and open directions

MCM underlines the importance to not degrade the global model.

Summary:
1. New algorithm for bi-directional compression with a preserved central model
2. Reduces (nearly cancels) impact of downlink compression
3. Achieves the same asymptotic rate of convergence as unidirectional compression.

Open directions:
1. Can we provably benefit from the smoothing effect?
2. Extending proofs of Rand-MCM to the self-concordant framework
3. Leveraging the randomization effect in applications
4. Even better double compression:
   - combination with better techniques on the up-link direction
   - unaffected $\gamma_{max}$
   - biased compression operators.

Thank you for your attention :)
Outline

1. Introduction
2. Doubly compressed SGD
3. Theoretical results - Ghost
4. Theoretical results - MCM
5. Theoretical results - Rand-MCM
6. Experiments
7. Conclusion
8. DP-Scaffold
Federated Learning: large-scale distributed learning very used today, which faces two challenges:

- training efficiently from highly heterogeneous user data (difficulties encountered with FedAvg),
- protecting the privacy of participating users, which is ensured by Differential privacy (DP).

Our work: while a lot of effort has gone into addressing these two challenges separately, we propose a novel approach (DP-SCAFFOLD) which combines them and demonstrate its superiority over the state-of-the-art algorithm DP-FedAvg, both theoretically and numerically.
Federated Learning Framework

Global Setting:
- $M$ users, each one holding a private local dataset $D_i = \{d_1^i, ..., d_R^i\}$ made of $R$ observations.
- one central server aggregating the local updates shared by the users.

Goal: solve the empirical risk minimization problem:

$$\min_{x \in \mathbb{R}^d} F(x) := \frac{1}{M} \sum_{i=1}^{M} F_i(x),$$

where $F_i(x) := \frac{1}{R} \sum_{j=1}^{R} f_i(x, d_j^i)$ is the empirical risk on user $i$, and for all $x \in \mathbb{R}^d$, $f_i(x, d)$ is the loss of the model $x$ on observation $d$.

Computational Setting:
- the server is active for $T$ communication rounds.
- at each round $t \in [T]$, the server randomly subsamples a set $C^t \subset [M]$ of $lM$ users, where $l \in (0, 1)$, and transmits the model to them.
- each user $i \in C^t$ performs $K$ steps of local SGD before sharing their update to the server.
- at each step $k \in [K]$, the user randomly subsamples a set $S_i^t \subset D_i$ of $sR$ data records, where $s \in (0, 1)$. 
Privacy model

**Goal:** control the information leakage from datasets $D_i$ in the updates shared by the users to the server and in the final model, by ensuring privacy:

1. towards a **third party** observing the final model,
2. towards an **honest-but-curious server**.

**Privacy scale:** *Record-level* privacy with respect to the joint dataset $D := D_1 \sqcup \ldots \sqcup D_M$. The DP budget is set in advance in the algorithm, denoted by $(\varepsilon, \delta)$ and corresponds to the desired level of privacy towards a third party observing the final model.
Description of DP–SCAFFOLD

Algorithm 1: DP-SCAFFOLD($T, K, l, s, \sigma_y, C$)

Server Input / i-th User Input: initial $x^0$, initial $c^0_i$ / initial $c^0_i$

Output: $x^T$

for $t = 1, \ldots, T$ do

User subsampling by the server:

Sample $C^t \subset [M]$ of size $[lM]$

Server sends $(x^{t-1}, c^{t-1})$ to users $i \in C^t$

for user $i \in C^t$ do

Initialize model: $y_i^0 \leftarrow x^{t-1}$

for $k = 1, \ldots, K$ do

Data subsampling by user $i$:

Sample $S_i^k \subset D_i$ of size $[sR]$

for sample $j \in S_i^k$ do

Compute gradient: $g_{ij} \leftarrow \nabla f_i(y_i^{k-1}, d_j^i)$

Clip gradient: $\tilde{g}_{ij} \leftarrow g_{ij}/\max(1, ||g_{ij}||_2/C)$

Add DP noise to local gradients:

$\tilde{H}_i^k \leftarrow \frac{1}{sR} \sum_{j \in S_i^k} \tilde{g}_{ij} + \frac{2C}{sR} \mathcal{N}(0, \sigma_y^2)$

$y_i^k \leftarrow y_i^{k-1} - \eta_l(\tilde{H}_i^k - c_i^{k-1} + c^l-1)$

$c_i^l \leftarrow c_i^{k-1} - c^{l-1} + \frac{1}{K_R}(x^{t-1} - y_i^K)$

$(\Delta y_i^l, \Delta c_i^l) \leftarrow (y_i^K - x^{t-1}, c_i^l - c_i^{l-1})$

User $i$ sends to server $(\Delta y_i^l, \Delta c_i^l)$

Server aggregates:

$(\Delta x^l, \Delta c^l) \leftarrow \frac{1}{1M} \sum_{i \in C^t} (\Delta y_i^l, \Delta c_i^l)$

$x^t \leftarrow x^{t-1} + \eta_l \Delta x^l$, $c^t \leftarrow c^{t-1} + l \Delta c^t$
Framework

**Local Gaussian mechanism:** centered Gaussian noise independently added to local stochastic gradients and calibrated with the $\ell_2$-sensitivity $2C/sR$ and the scale $\sigma_g$ (see Step 14). This requires to previously clip the *per-example* gradients with threshold $C$.

**Heterogeneity issue:** tackled in the SGD steps with a drift correction term $(c_i - c)$ where:

- $c_i$ stands for the *local* direction of the gradients, updated by user $i \in C^t$ at the end of each inner loop (see Step 16), and transmitted to the server,

- $c$ stands for the *averaged* direction of the gradients, updated by the server at the end of each round as the mean of the new values of $c_i$ (see Step 22).

Removing these control variates (which are noisy by definition) recovers DP-FedAvg.

**Warm-start version:** we keep aside the first few rounds of communication to set $c^0 = \frac{1}{M} \sum_{i=1}^{M} c_i^0$, while ensuring DP, which gives DP-SCAFFOLD-warm.
Theoretical results: Privacy

**High privacy Assumptions** We consider a noise level $\sigma_g$, a privacy budget $\epsilon > 0$ and a data-subsampling ratio $s$ s.t.: (H1) $s = o(1)$, (H2) $\epsilon < 1$, (H3) $\sigma_g = \Omega(s\sqrt{K/\log(2TL/\delta)})$. These assumptions allow to obtain closed forms for the privacy guarantee but are not used in practice.

**Privacy guarantee** Under Assumptions H1, H2 and H3, for DP-SCAFFOLD(-warm) and DP-FedAvg with calibration $\sigma_g = \Omega(s\sqrt{ITK\log(2TL/\delta)\log(2/\delta)/\epsilon\sqrt{M}})$, $x^T$ is:

1. $(O(\epsilon), \delta)$-DP towards a third-party,
2. $(O(\epsilon_s), \delta_s)$-DP towards the server, where $\epsilon_s = \epsilon \sqrt{\frac{M}{I}}$ and $\delta_s = \frac{\delta}{2}(\frac{1}{I} + 1)$. 
Theoretical results: Utility

**A1: Smoothness and regularity** For all $i \in [M]$, $F_i$ is differentiable and $\nu$-smooth.

**A2: Stochastic gradients and data sampling** For any iteration $t \in [T]$, $k \in [K]$,

- the stochastic gradient $\nabla f_i(y_{i}^{k-1}, d_j^k)$ is conditionally unbiased and has variance bounded by $\varsigma^2$,
- there exists a clipping constant $C$ independent of $i, j$ such that $\|\nabla f_i(y_{i}^{k-1}, d_j^k)\| \leq C$.

**A3: Bounded Gradient dissimilarity** There exist constants $G \geq 0$ and $B \geq 1$ such that:

$$\forall x \in \mathbb{R}^d, \frac{1}{M} \sum_{i=1}^{M} \|\nabla F_i(x)\|^2 \leq G^2 + B^2 \|\nabla F(x)\|^2.$$

We consider $\sigma_g^* := s\sqrt{TK \log(2TI/\delta) \log(2/\delta)} / \epsilon \sqrt{M}$, which is the order of magnitude of noise scale to approximately ensure end-to-end $(\epsilon, \delta)$-DP w.r.t. $D$ in DP-SCAFFOLD(-warm) and DP-FedAvg.
Theoretical results : Utility

**Utility result (convex case)** Assume that $F$ is bounded by $F^*$ and for all $i \in [M]$, $F_i$ is convex.

Under Assumptions $A1$ and $A2$, we consider the iterates $(x^t)_{t \geq 0}$ for DP-SCAFFOLD-warm and DP-FedAvg, starting from $x^0 \in \mathbb{R}^d$ (with $D_0 := ||x^0 - x^*||$), calibrated with DP noise $\sigma_g := \sigma^*_g$. Then there exist step-sizes $(\eta_g, \eta_l)$ and weights $(w_t)_{t \in [T]}$ such that

$$\mathbb{E}[F(\sum_{t=1}^T w_t x^t)] - F^*$$

is bounded,

- for DP-FedAvg, under Assumption $A3$, by:

$$\mathcal{O}\left(\frac{D_0 C \sqrt{d \log(T/l/\delta) \log(1/\delta)}}{\epsilon MR} + \frac{\zeta D_0}{\sqrt{sRIMKT}} + \frac{B^2 \nu D_0^2}{T} + \frac{GD_0 \sqrt{1 - l}}{\sqrt{IMT}} + \frac{D_0^{4/3} \nu^{1/3} G^{2/3}}{T^{2/3}}\right)$$

  privacy bound  

  optimization bound

- for DP-SCAFFOLD-warm by:

$$\mathcal{O}\left(\frac{D_0 C \sqrt{d \log(T/l/\delta) \log(1/\delta)}}{\epsilon MR} + \frac{\zeta D_0}{\sqrt{sRIMKT}} + \frac{\nu D_0^2}{\rho/3 T}\right)$$

  privacy bound  

  optimization bound
Experiments (heterogeneity increasing from left to right)

**Figure:** Test Accuracy on FEMNIST data under \((11.4, 10^{-5})\)-DP: 
\[ M = 40, \ R = 2500, \ s = l = 0.2, \ K = 50 \] (Logistic Regression).

**Figure:** Test Accuracy on MNIST data under \((7.2, 1.7 \times 10^{-5})\)-DP: 
\[ M = 60, \ R = 1000, \ s = l = 0.2, \ K = 50 \] (Neural Network).


Do we need double compression?

Objectives of compression:

1. Accelerate the learning process,
2. Limit the number of communicated bits

In terms of speed, double compression depends on how the exchange is performed:

- If broadcast (1 to $N$) is much faster than upload ($N$ to 1) then no need for double compression.
- If we consider mobile devices (using for example fast Internet connexion), only a small difference between upload and download speed.

In terms of communicated bits: upload and download are symmetric. Example: no one wants to download a large update everyday on its phone!