Distributed Optimization and Error Feedback

Mher Safaryan

Assistant Professor (Lecturer)

Lancaster University

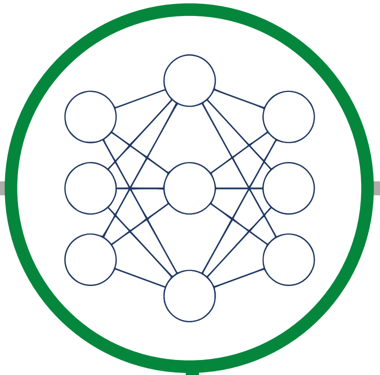
MARS

Mathematics for AI in Real-world Systems

Supervised Machine Learning

hypothesis $h^x, x \in \mathbb{R}^d$

model size



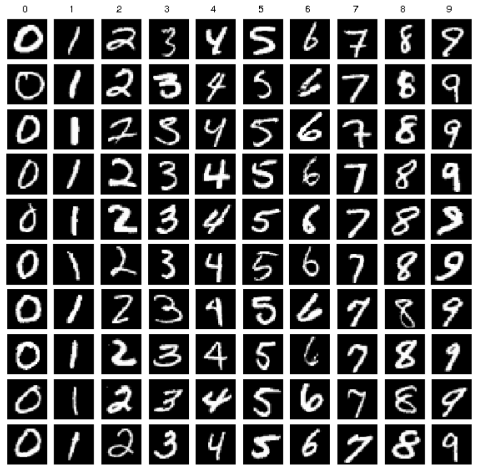
9 Empirical Risk Minimization (ERM)

Predicted label

Correct label

$$(\boxed{h^x(a_j)}, \boxed{b_j})$$

loss function to gauge how different are two labels b' and b.
For example, $\text{loss}(b', b) = \frac{1}{2}(b' - b)^2$



train data

data size

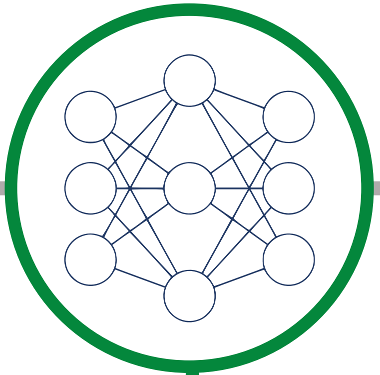
$$\mathcal{D} = \{(a_j, b_j)\}_{j=1}^{N}$$

$= \{($  $, 9), \dots\}$

Supervised Machine Learning

hypothesis $h^x, x \in \mathbb{R}^d$

model size



9

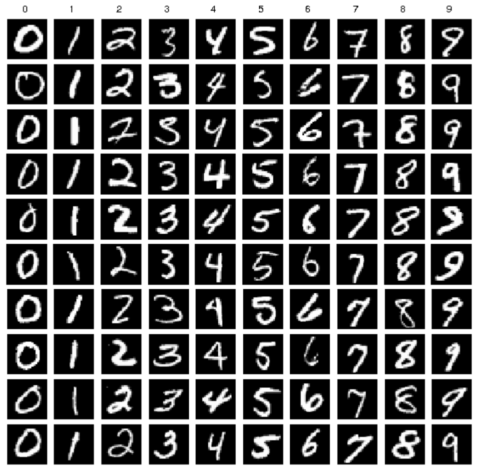
Empirical Risk Minimization (ERM)

Predicted label

Correct label

$$\min_{x \in \mathbb{R}^d} \frac{1}{N} \sum_{j=1}^{N} \text{loss}(\boxed{h^x(a_j)}, \boxed{b_j})$$

loss function to gauge how different are two labels b' and b. For example, $\text{loss}(b', b) = \frac{1}{2}(b' - b)^2$



train data

data size

$$\mathcal{D} = \{(a_j, b_j)\}_{j=1}^{N}$$

$= \{($  $, 9), \ldots\}$

Empirical Risk Minimization

$$\min_{x \in \mathbb{R}^d} \frac{1}{N} \sum_{j=1}^{N} \text{loss}(h^x(a_j), b_j)$$

$$\mathcal{D} = \{(a_j, b_j)\}_{j=1}^{N}$$

$$\min_{x \in \mathbb{R}^d} \mathbb{E}_{(a,b)\sim\mathcal{D}} \left[\text{loss}(h^x(a), b)\right]$$

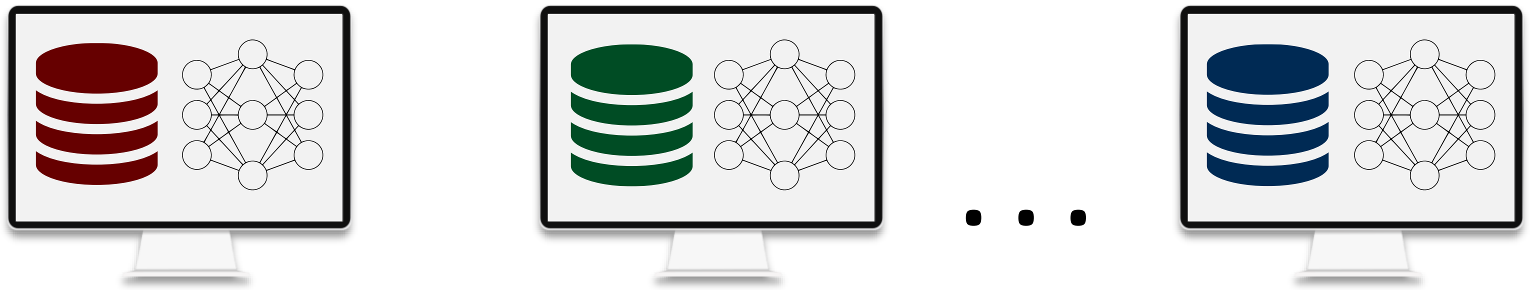
$$\xi \stackrel{def}{=} (a, b)$$

$$f_\xi(x) \stackrel{def}{=} \text{loss}(h^x(a), b)$$

$$\min_{x \in \mathbb{R}^d} \mathbb{E}_{\xi\sim\mathcal{D}} \left[f_\xi(x)\right]$$

Distributed Machine Learning

Why Distributed?



Reason 1: BIG Data does not fit into a single device

Reason 2: Data Privacy in Federated Learning

Reason 3: Parallel Computation

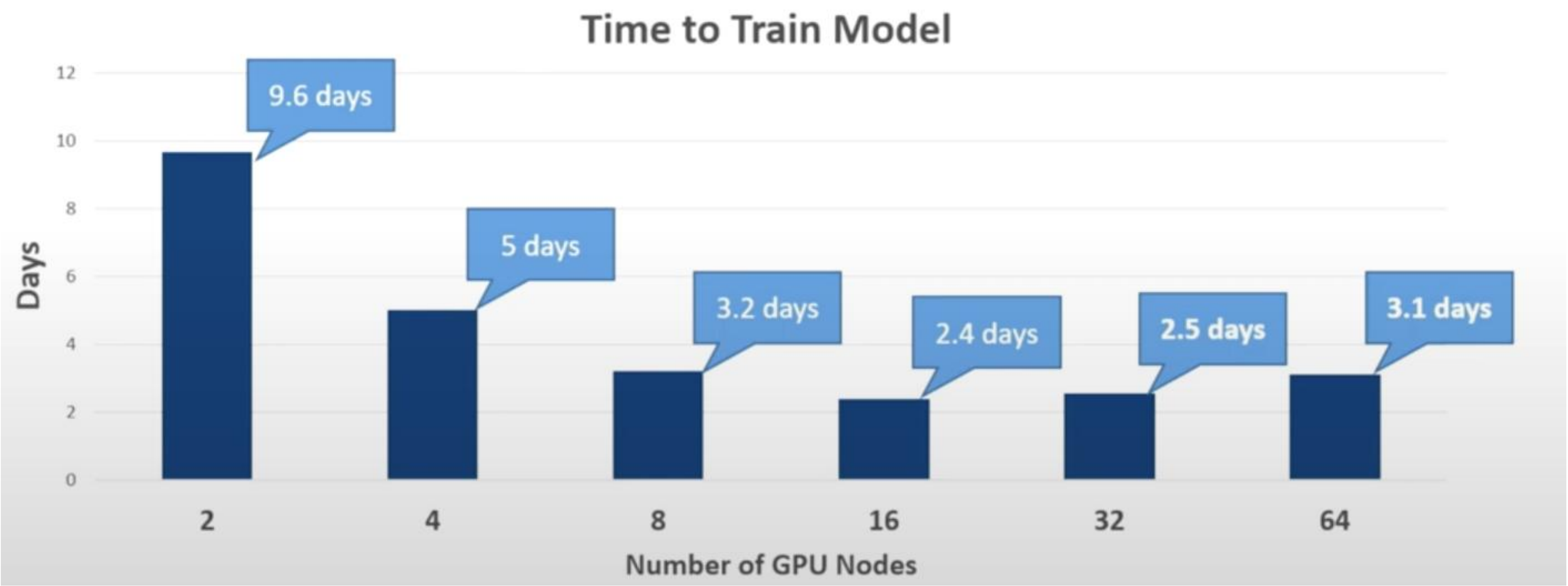
Distributed Learning in Practice

CSCS: Europe's Top Supercomputer (World 4^{th})

- 4500+ GPU Nodes, state-of-the-art interconnect

Task:

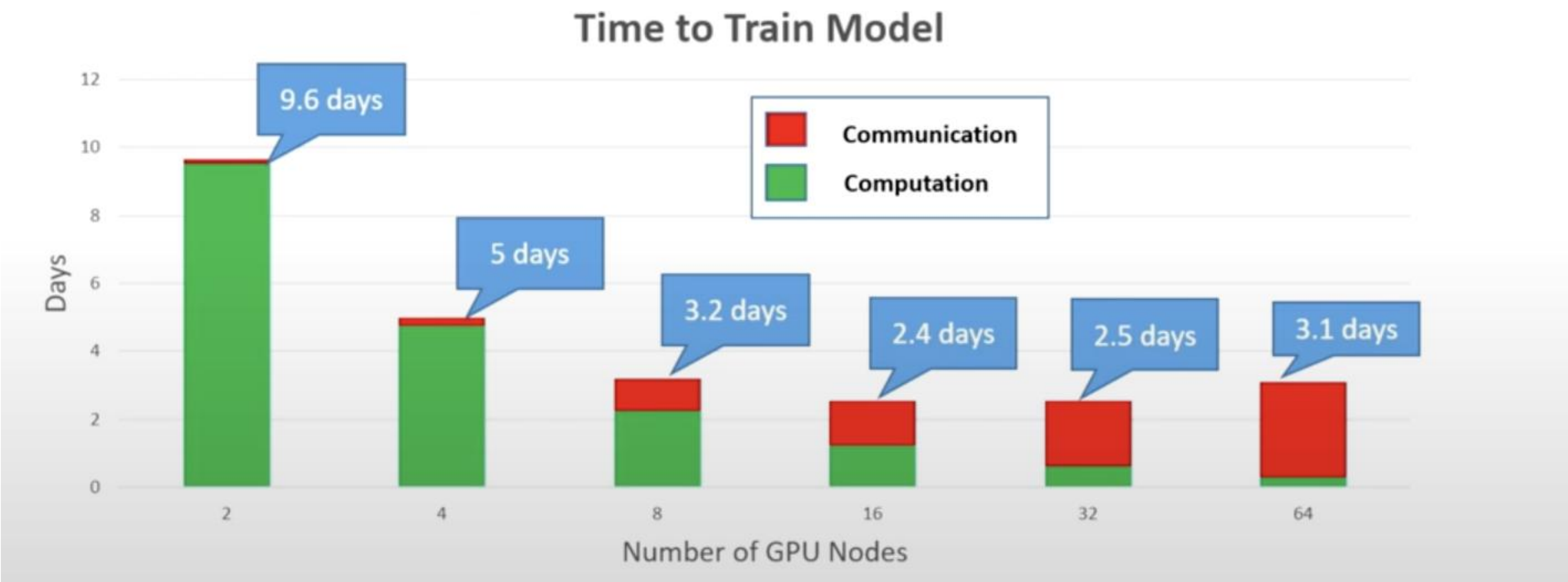
- Image Classification (ResNet-152 on ImageNet)
- Single Node time (TensorFlow): **19 days**
- 1024 Nodes: **25 minutes** (***in theory***)



Distributed Learning in Practice

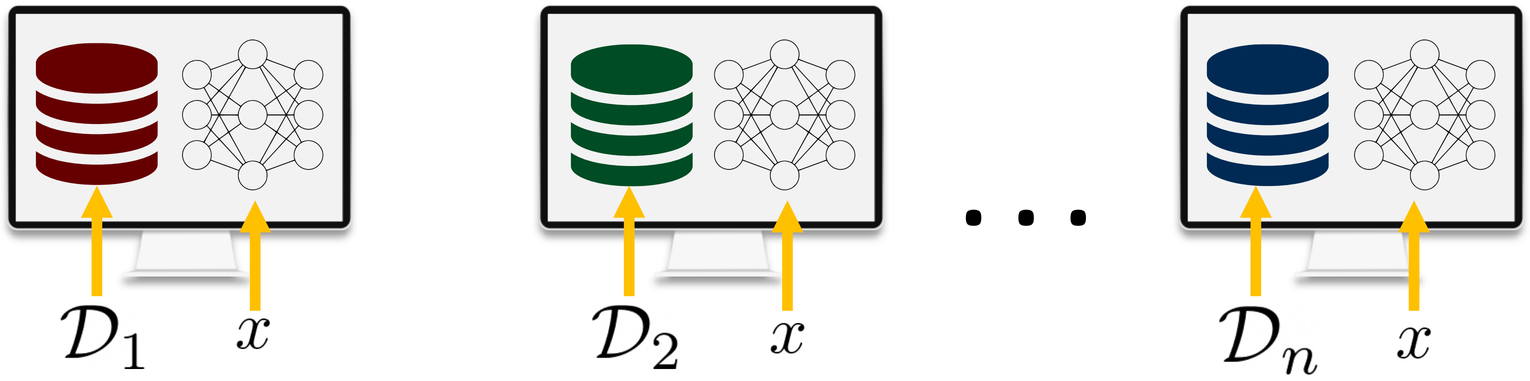
CSCS: Europe's Top Supercomputer (World 4th)
- 4500+ GPU Nodes, state-of-the-art interconnect

Task:
- Image Classification (ResNet-152 on ImageNet)
- Single Node time (TensorFlow): **19 days**
- 1024 Nodes: **25 minutes** (***in theory***)

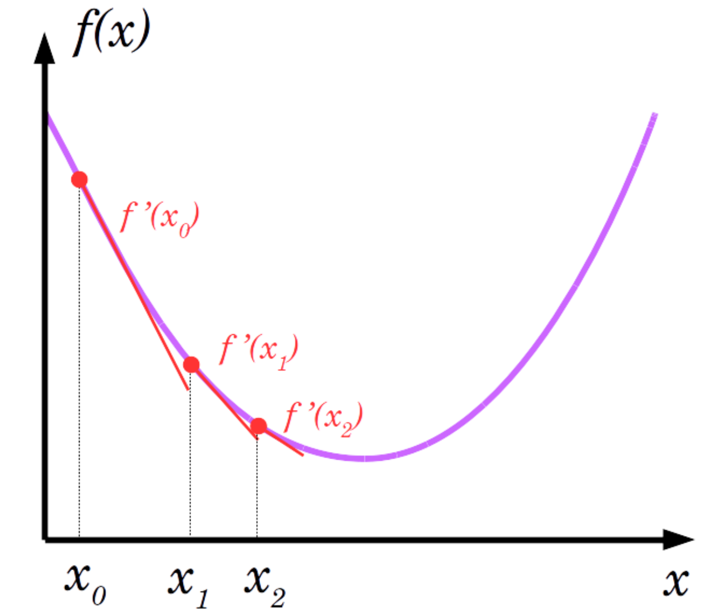


Distributed Optimization Problem

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



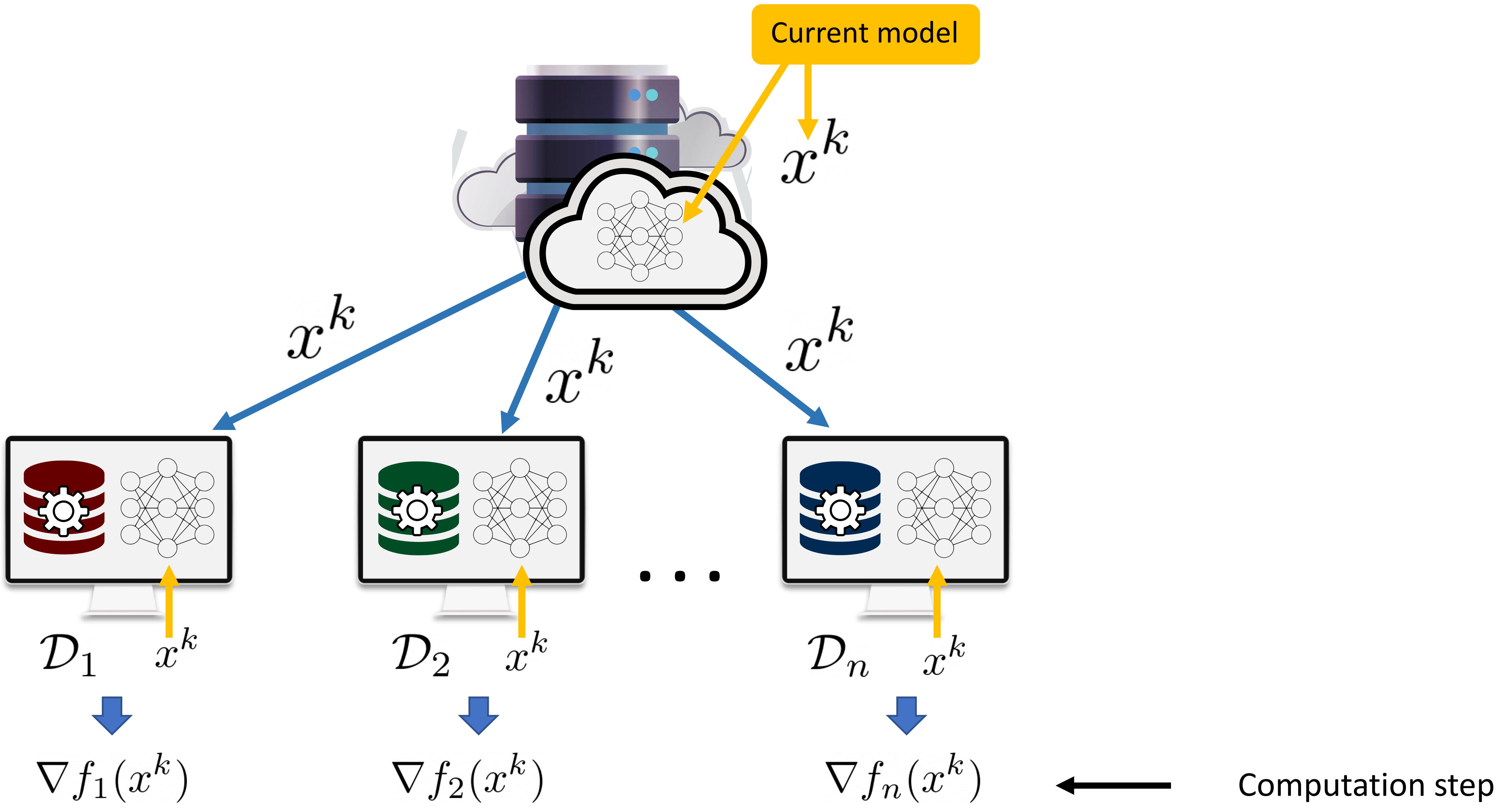
$f_1(x) = \mathbb{E}_{\xi\sim\mathcal{D}_1}[f_\xi(x)]$ $\quad f_2(x) = \mathbb{E}_{\xi\sim\mathcal{D}_2}[f_\xi(x)]$ $\quad f_n(x) = \mathbb{E}_{\xi\sim\mathcal{D}_n}[f_\xi(x)]$



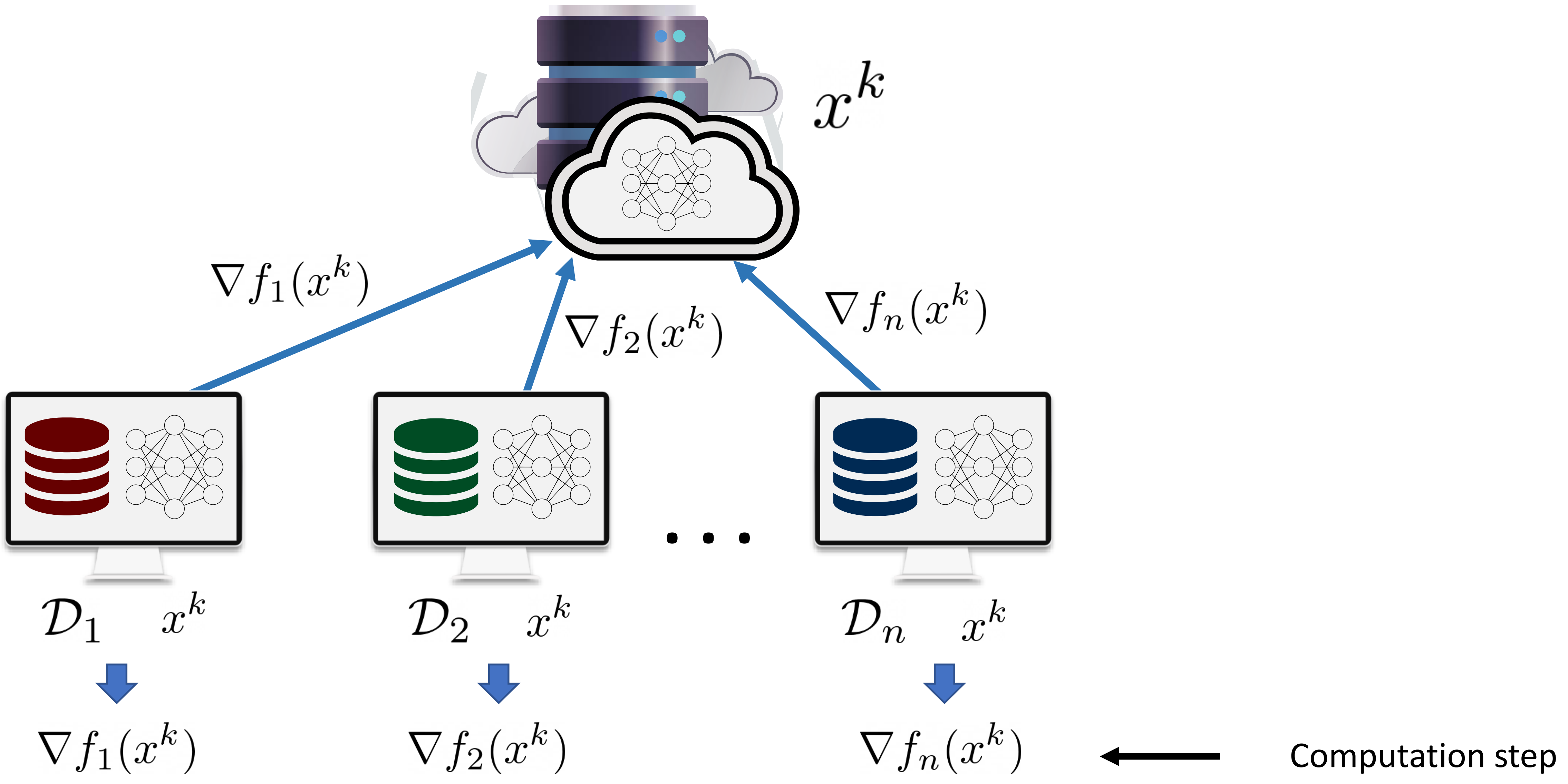
Gradient Descent

$$x^{k+1} = x^k - \gamma \boxed{\nabla f(x^k)}$$

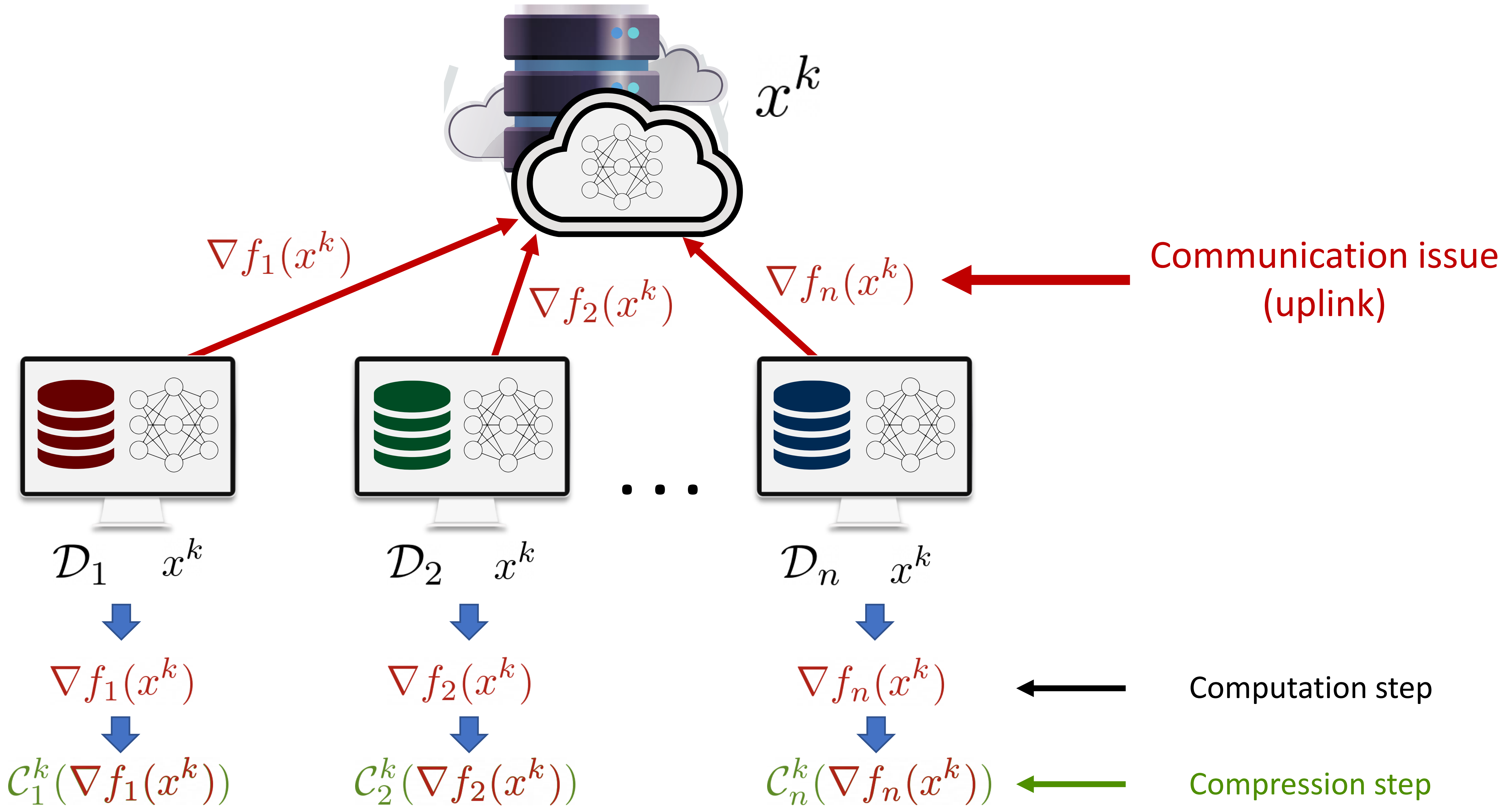
Distributed Gradient Descent



Distributed Gradient Descent



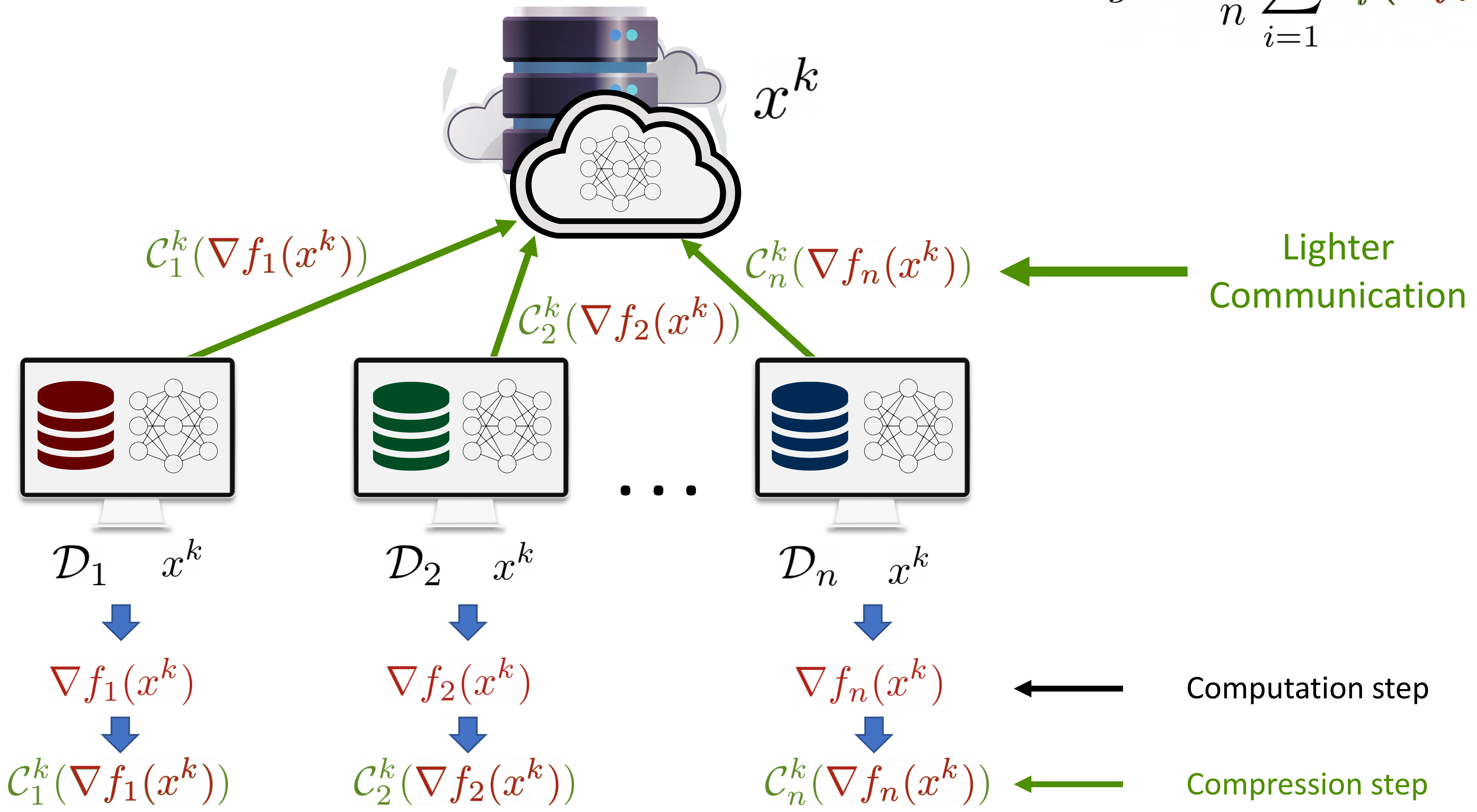
Distributed Gradient Descent



Distributed **Compressed** Gradient Descent

Aggregated gradient

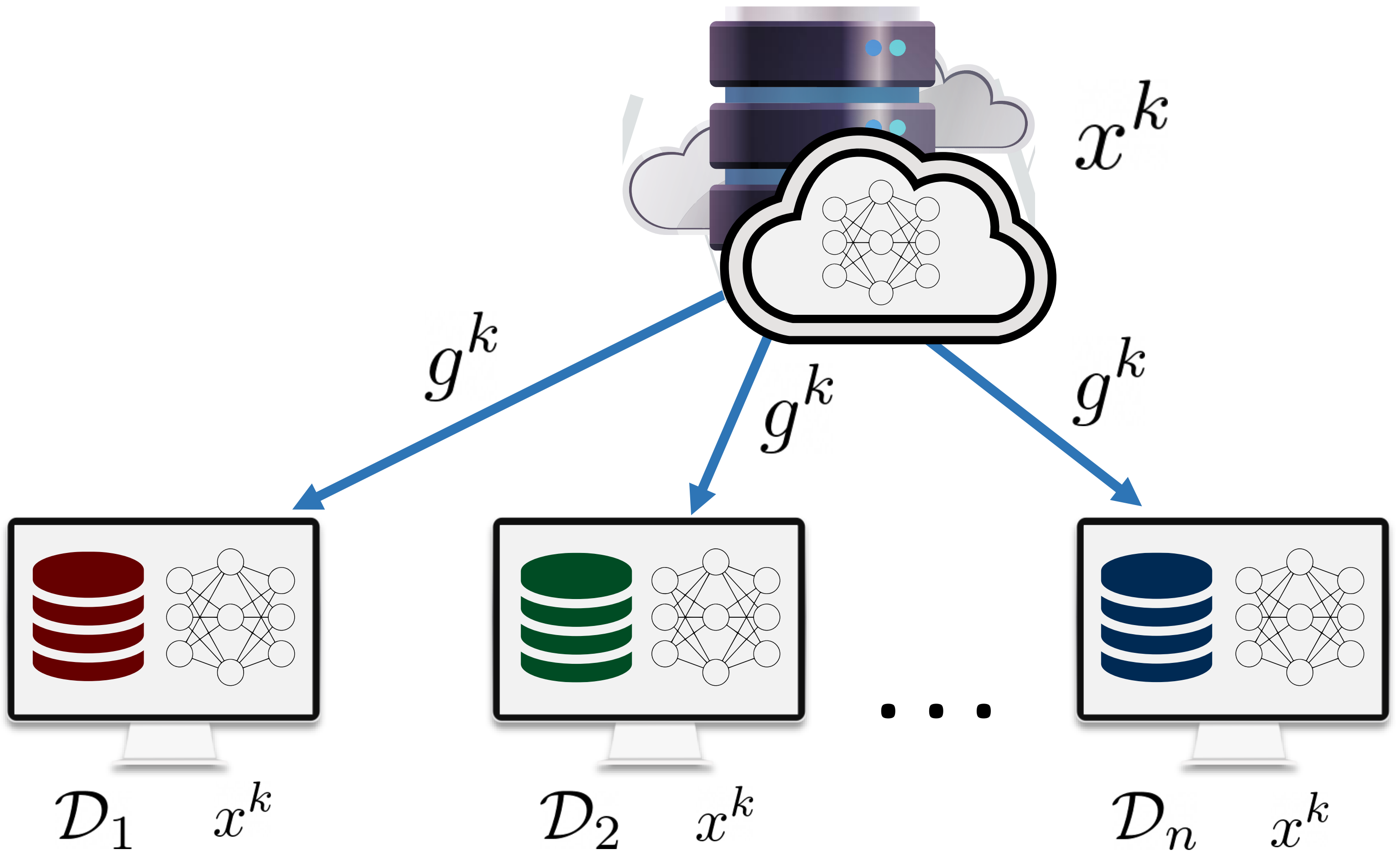
$$g^k = \frac{1}{n}\sum_{i=1}^{n} \mathcal{C}_i^k(\nabla f_i(x^k))$$



Distributed **Compressed** Gradient Descent

Aggregated gradient

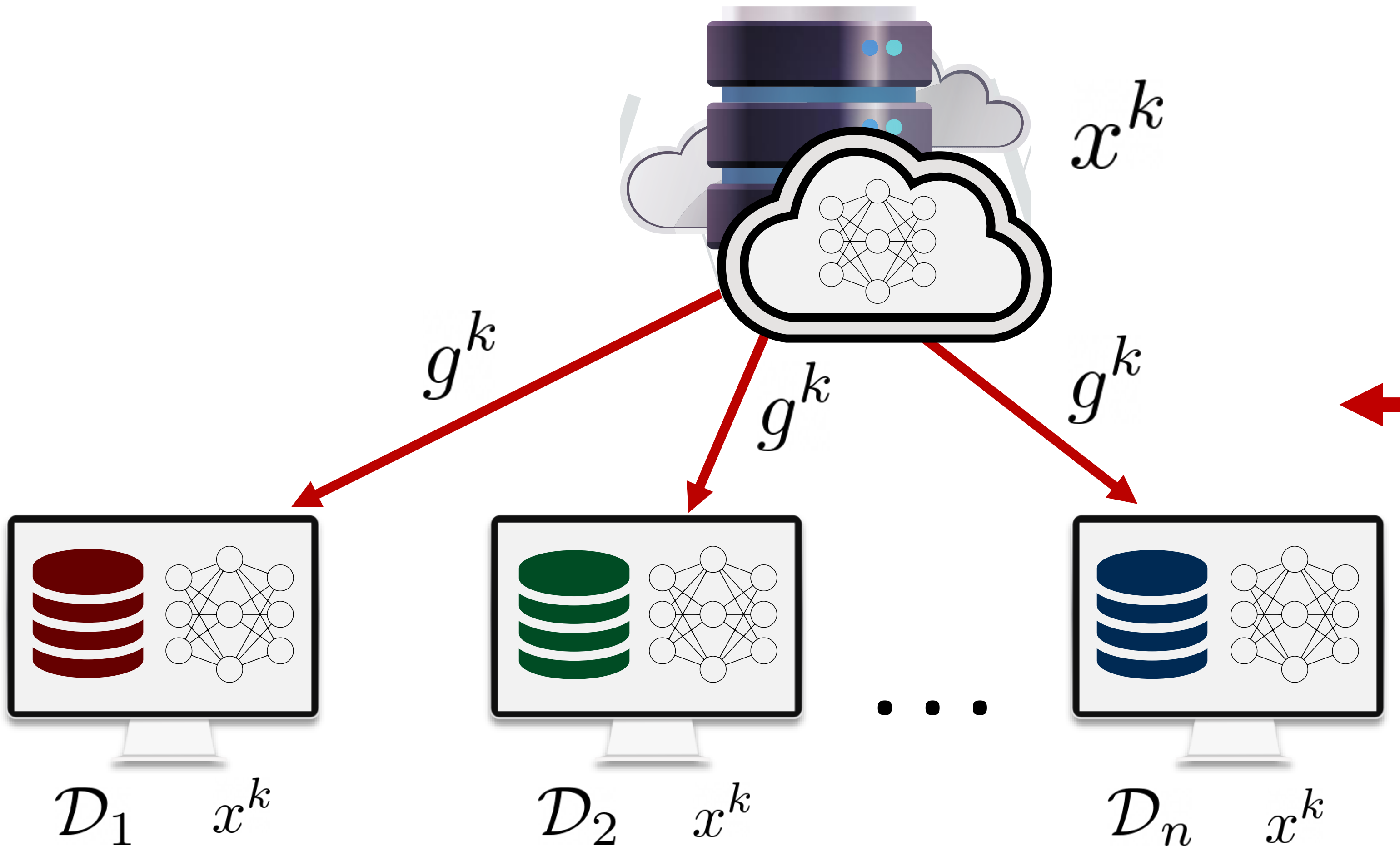
$$g^k = \frac{1}{n}\sum_{i=1}^{n} \mathcal{C}_i^k(\nabla f_i(x^k))$$



Distributed **Compressed** Gradient Descent

Aggregated gradient

$$g^k = \frac{1}{n}\sum_{i=1}^{n} \mathcal{C}_i^k(\nabla f_i(x^k))$$



Communication issue (downlink)

Distributed **Compressed** Gradient Descent

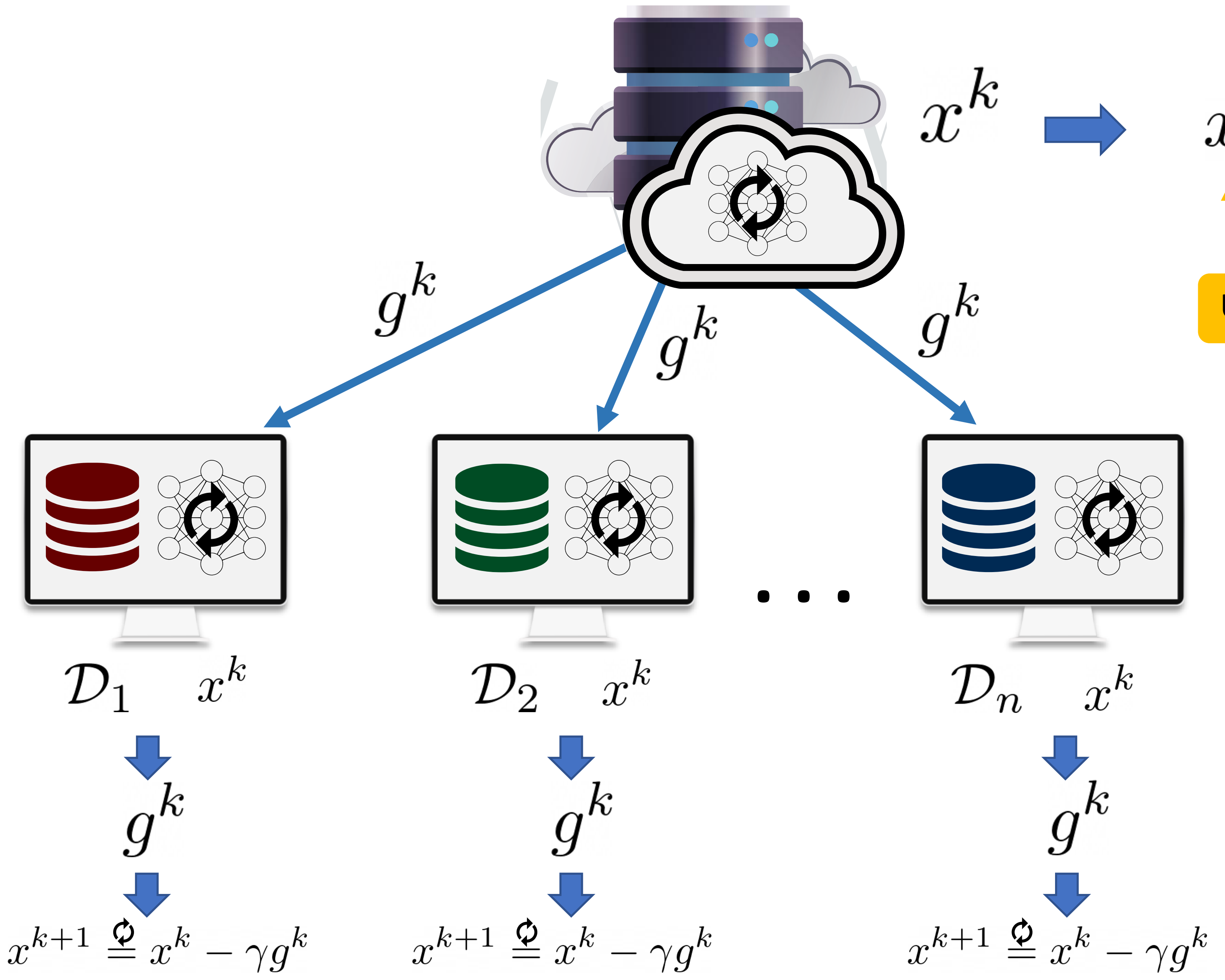
Aggregated gradient

$$g^k = \frac{1}{n}\sum_{i=1}^{n} \mathcal{C}_i^k(\nabla f_i(x^k))$$

$$x^{k+1} \overset{\circlearrowleft}{=} x^k - \gamma g^k$$

Updated model

stepsize



Aggregated gradient

Update the model

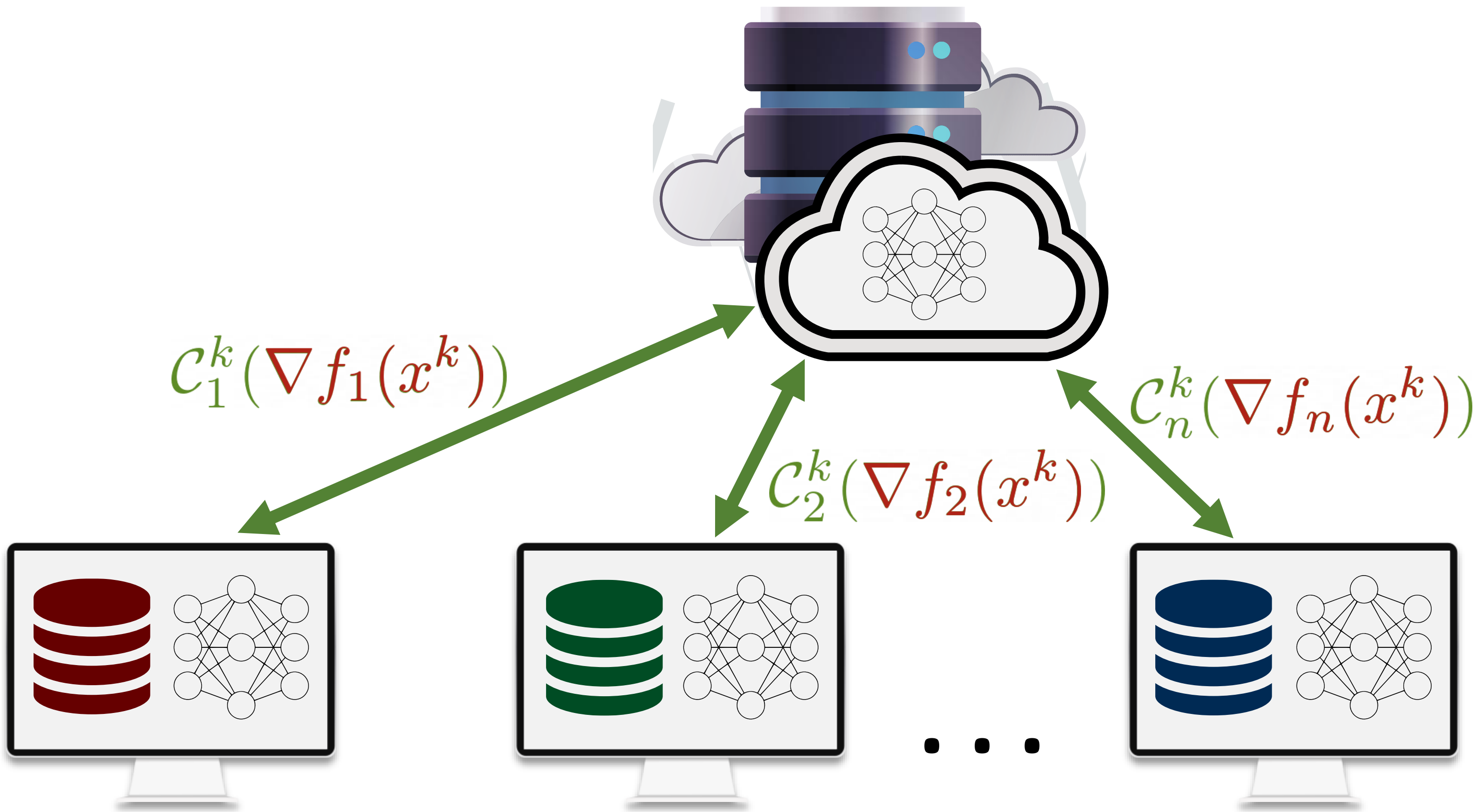
Distributed **Compressed** Gradient Descent

Compression operator

$\mathcal{C}_i^k : \mathbb{R}^d \to \mathbb{R}^d$

$$x^{k+1} = x^k - \frac{\gamma}{n} \sum_{i=1}^{n} \mathcal{C}_i^k(\nabla f_i(x^k))$$

Contributions



➢ **Compressed communication**

- **Sign (1-bit) compression**

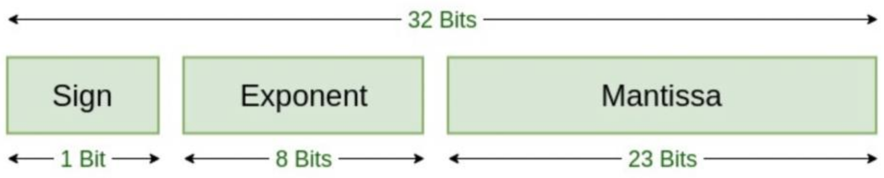
Mher Safaryan, Peter Richtárik
Stochastic Sign Descent Methods: New Algorithms and Better Theory, ***ICML 2021***

vector $g \in \mathbb{R}^d$

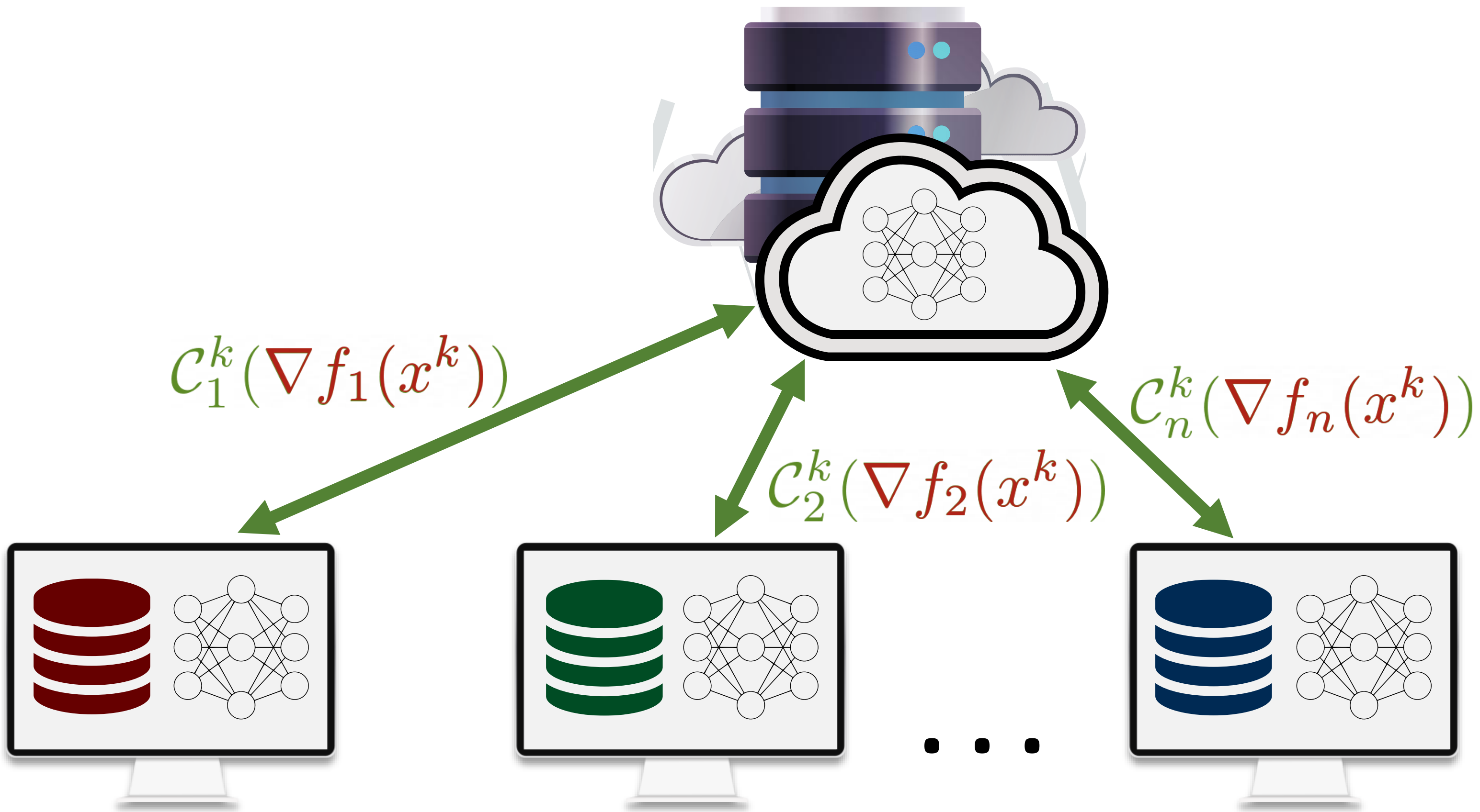
$$(\mathrm{sign}\, g)_j = \begin{cases} +1 & \text{if} \quad g_j > 0 \\ -1 & \text{if} \quad g_j < 0 \\ 0 & \text{if} \quad g_j = 0 \end{cases}$$

entry $j \in \{1, 2, \ldots, d\}$

$$\mathrm{sign} \begin{bmatrix} 0.4 \\ 0 \\ -0.3 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}$$



Contributions



- **Compressed communication**
  - **Sign (1-bit) compression**

Mher Safaryan, Peter Richtárik
Stochastic Sign Descent Methods: New Algorithms and Better Theory, ***ICML 2021***

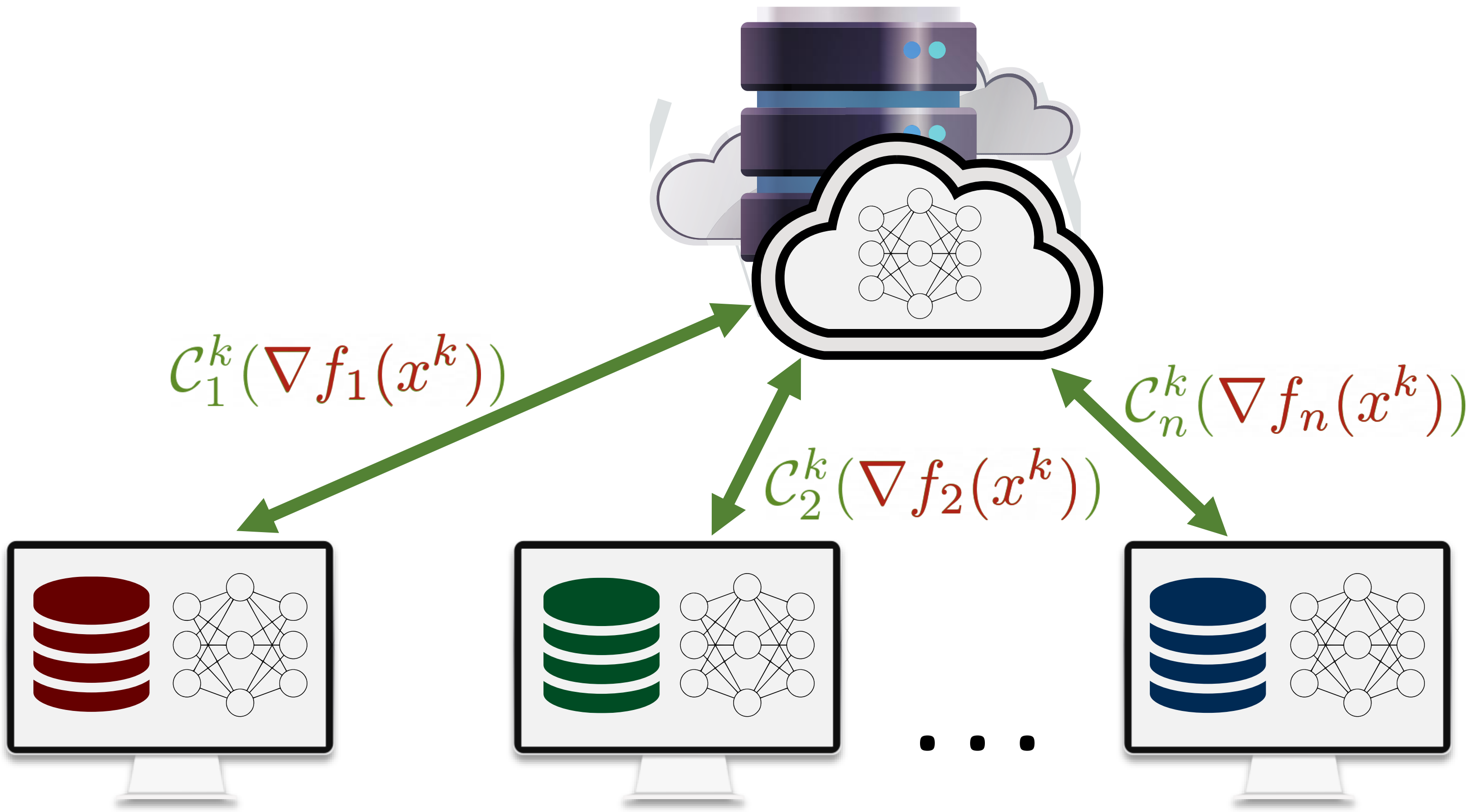
$$\left(\widetilde{\mathrm{sign}}\, g\right)_j = \boxed{\begin{cases} +1 & \text{with prob. } \quad \frac{1}{2} + \frac{1}{2}\frac{g_j}{\|g\|} \\ -1 & \text{with prob. } \quad \frac{1}{2} - \frac{1}{2}\frac{g_j}{\|g\|} \end{cases}}$$

$$\widetilde{\mathrm{sign}} \begin{bmatrix} 0.4 \\ 0 \\ -0.3 \end{bmatrix} = \begin{bmatrix} \mathrm{B}(0.9) \\ \mathrm{B}(0.5) \\ \mathrm{B}(0.2) \end{bmatrix}$$

Stochastic sign

Bernoulli random variable $\mathrm{B}\left(\frac{1}{2} + \frac{1}{2}\frac{g_j}{\|g\|}\right)$

Contributions



$$\mathrm{Top}_k: \quad \begin{bmatrix} -0.4 \\ 12.1 \\ 0.76 \\ 2.8 \\ -9.7 \end{bmatrix} \xrightarrow{k=2} \begin{bmatrix} 0 \\ 12.1 \\ 0 \\ 0 \\ -9.7 \end{bmatrix}$$

$$\mathrm{Rand}_k: \quad \begin{bmatrix} -0.4 \\ 12.1 \\ 0.76 \\ 2.8 \\ -9.7 \end{bmatrix} \xrightarrow{k=2} \frac{5}{2} \cdot \begin{bmatrix} -0.4 \\ 0 \\ 0 \\ 2.8 \\ 0 \end{bmatrix}$$

➢ **Compressed communication**

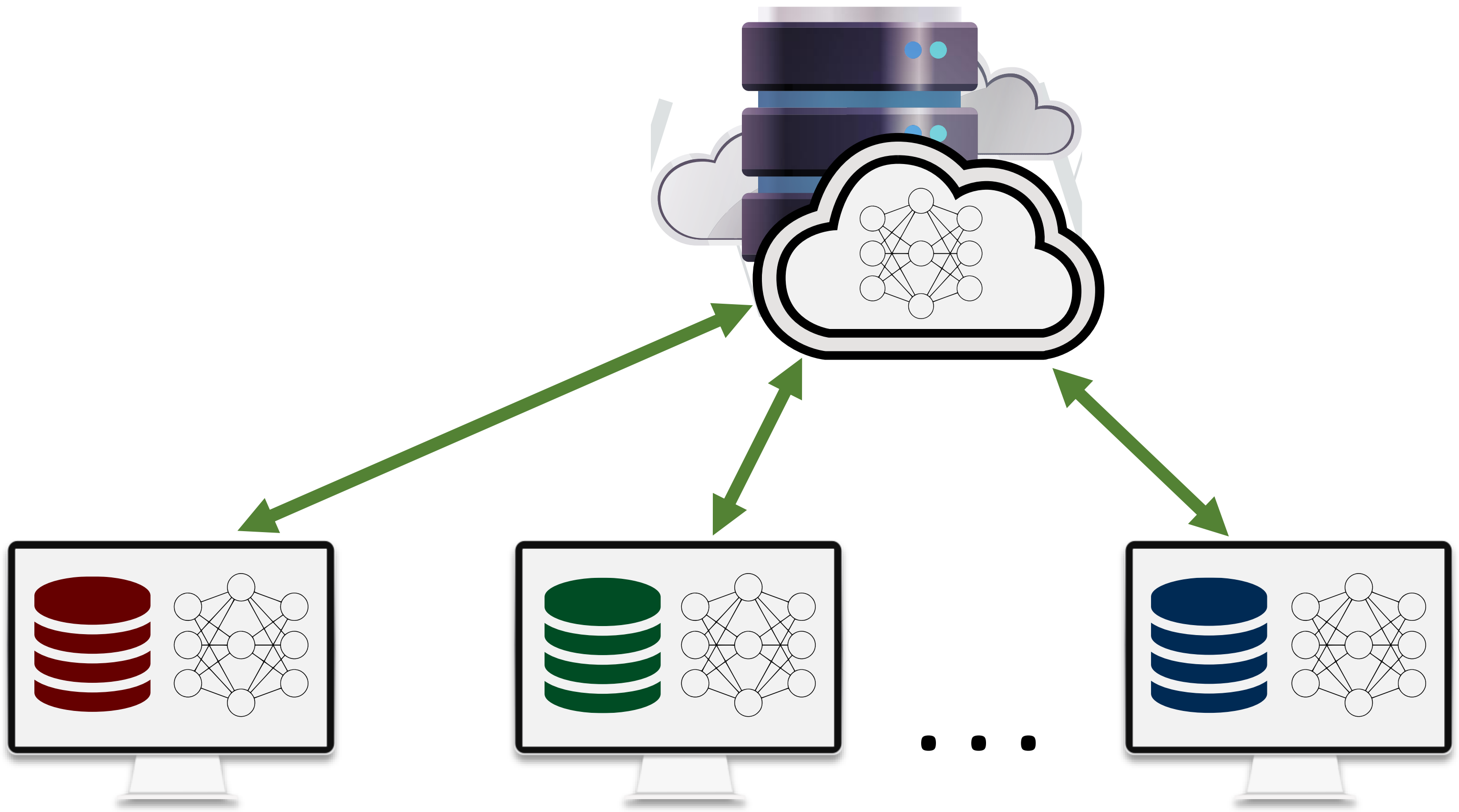
- **Sign (1-bit) compression**
- **Contractive compression**

Aleksandr Beznosikov, Samuel Horváth, Peter Richtárik, Mher Safaryan
On Biased Compression for Distributed Learning, ***JMLR 2023***

Mher Safaryan, Filip Hanzely, Peter Richtárik
Smoothness Matrices Beat Smoothness Constants: Better Communication Compression Techniques for Distributed Optimization, ***NeurIPS 2021***

Bokun Wang, Mher Safaryan, Peter Richtárik
Theoretically Better and Numerically Faster Distributed Optimization with Smoothness-Aware Quantization Techniques, ***NeurIPS 2022***

Contributions



$$x^{k+1} = x^k - \left(\frac{1}{n}\sum_{i=1}^{n} \boxed{\nabla^2 f_i(x^k)}\right)^{-1}\left(\frac{1}{n}\sum_{i=1}^{n} \boxed{\nabla f_i(x^k)}\right)$$

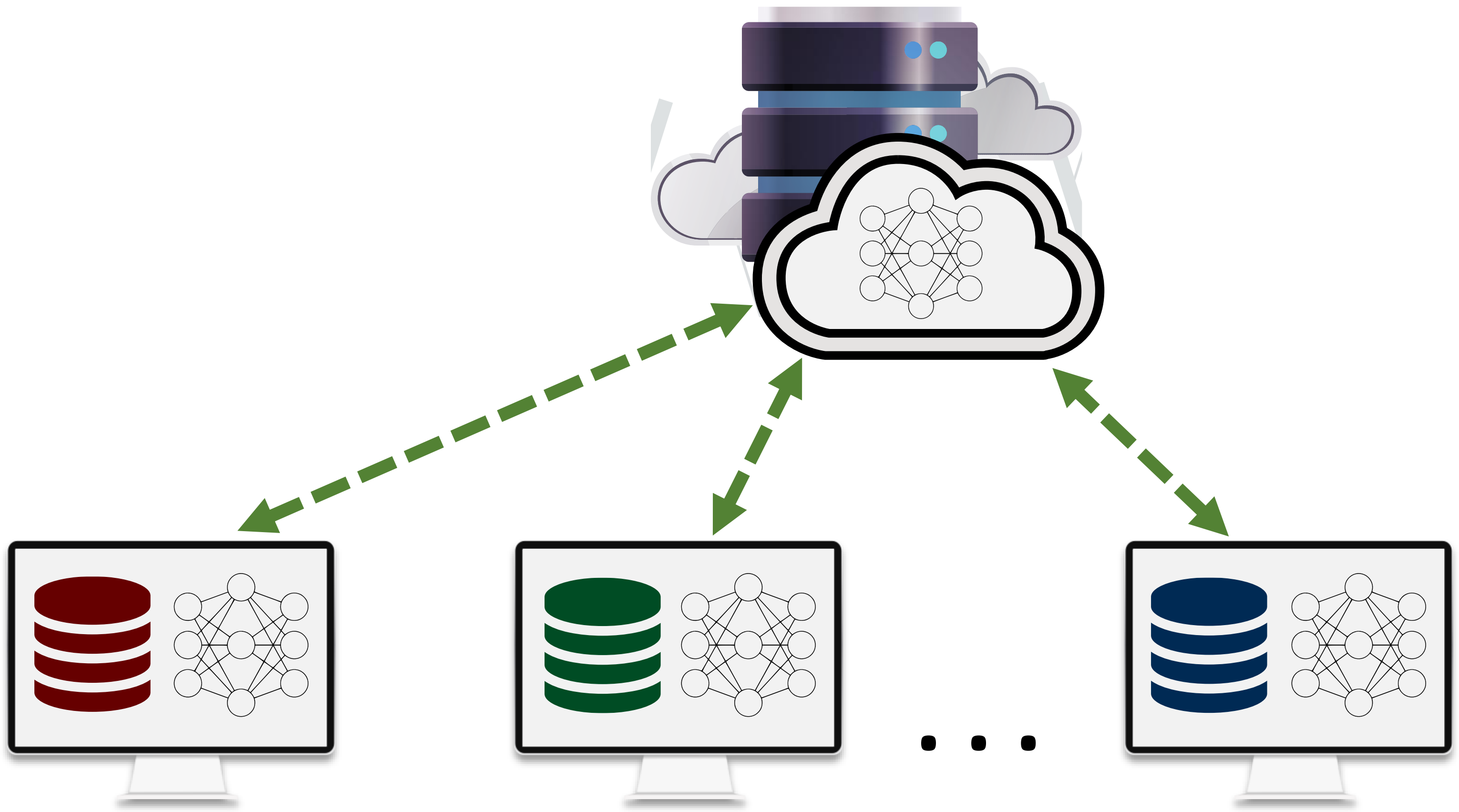
- **Compressed communication**
  - **Sign (1-bit) compression**
  - **Contractive compression**
  - **Second-order optimization**

Mher Safaryan, Rustem Islamov,
Xun Qian, Peter Richtárik
FedNL: Making Newton-type methods applicable to federated learning, ***ICML 2022***

Xun Qian, Rustem Islamov,
Mher Safaryan, Peter Richtárik
Basis Matters: Better Communication-Efficient Second Order Methods for Federated Learning, ***AISTATS 2022***

Rustem Islamov, Xun Qian, Slavomír Hanzely,
Mher Safaryan, Peter Richtárik
Distributed Newton-type methods with communication compression and Bernoulli aggregation, ***TMLR 2023***

Contributions



- **Compressed communication**
  - **Sign (1-bit) compression**
  - **Contractive compression**
  - **Second-order optimization**
- **Infrequent communication**

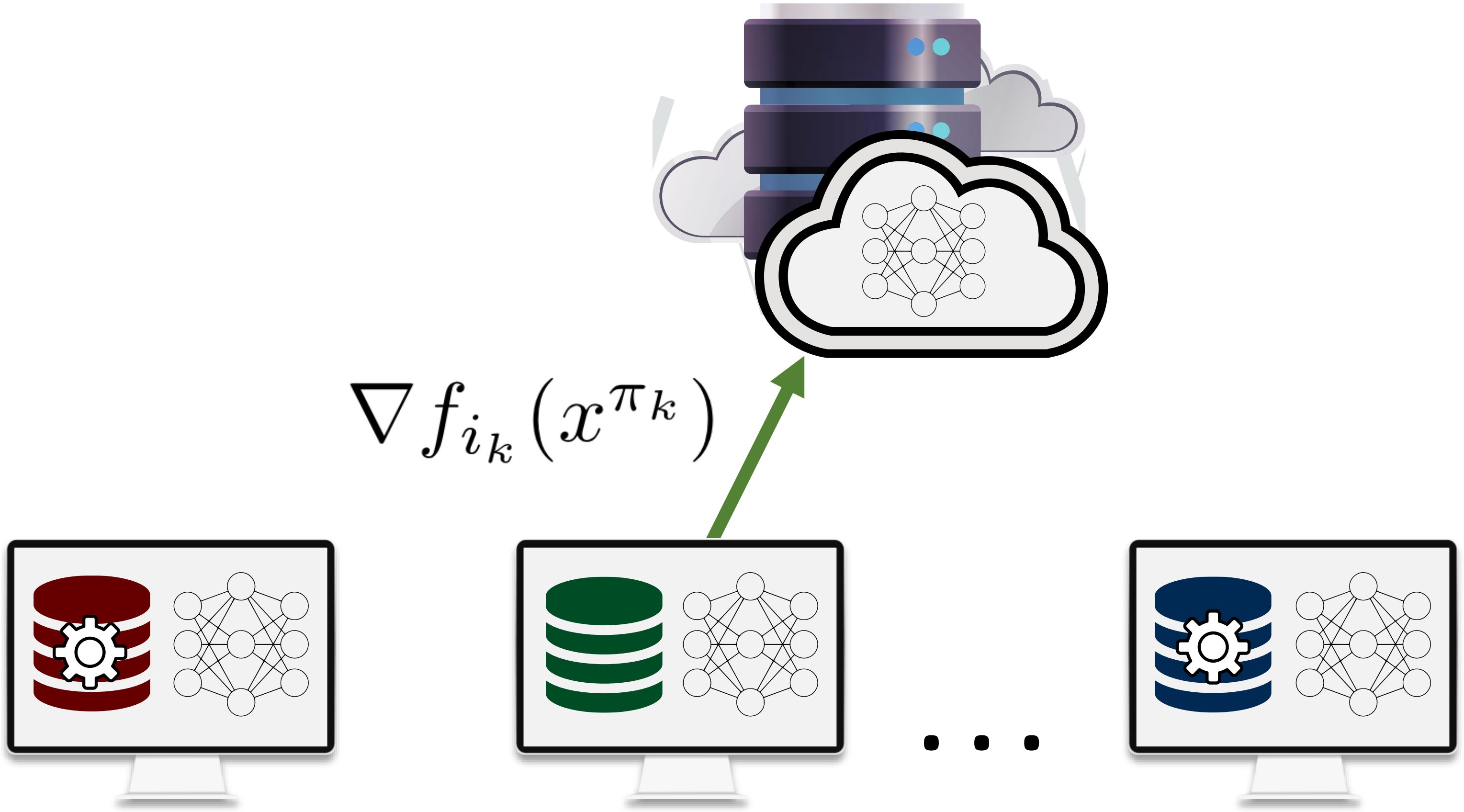
Artavazd Maranjyan, Mher Safaryan, Peter Richtárik
Gradskip: Communication-accelerated local gradient methods with better computational complexity, ***Master's thesis, YSU, 2022***

$$x_i^{k+1} = \begin{cases} x_i^k - \gamma g_i^k, & \boxed{\text{if } c_{k+1} = 0,} \\ \boxed{\frac{1}{n}\sum_{i=1}^{n}}\left(x_i^k - \gamma g_i^k\right), & \boxed{\text{if } c_{k+1} = 1,} \end{cases}$$

No communication!

Communication step

Contributions



$$x^{k+1} = x^k - \gamma \nabla f_{i_k}(x^{\pi_k})$$

Index of a machine

Potentially outdated model

- **Compressed communication**
  - **Sign (1-bit) compression**
  - **Contractive compression**
  - **Second-order optimization**
- **Infrequent communication**
- **Asynchronous communication**

Rustem Islamov, Mher Safaryan, Dan Alistarh
A Sharp Unified Analysis of Asynchronous-SGD Algorithms, *Master's thesis, IP Paris, 2023, AISTATS 2024*

Distributed **Compressed** Gradient Descent

Compression operator

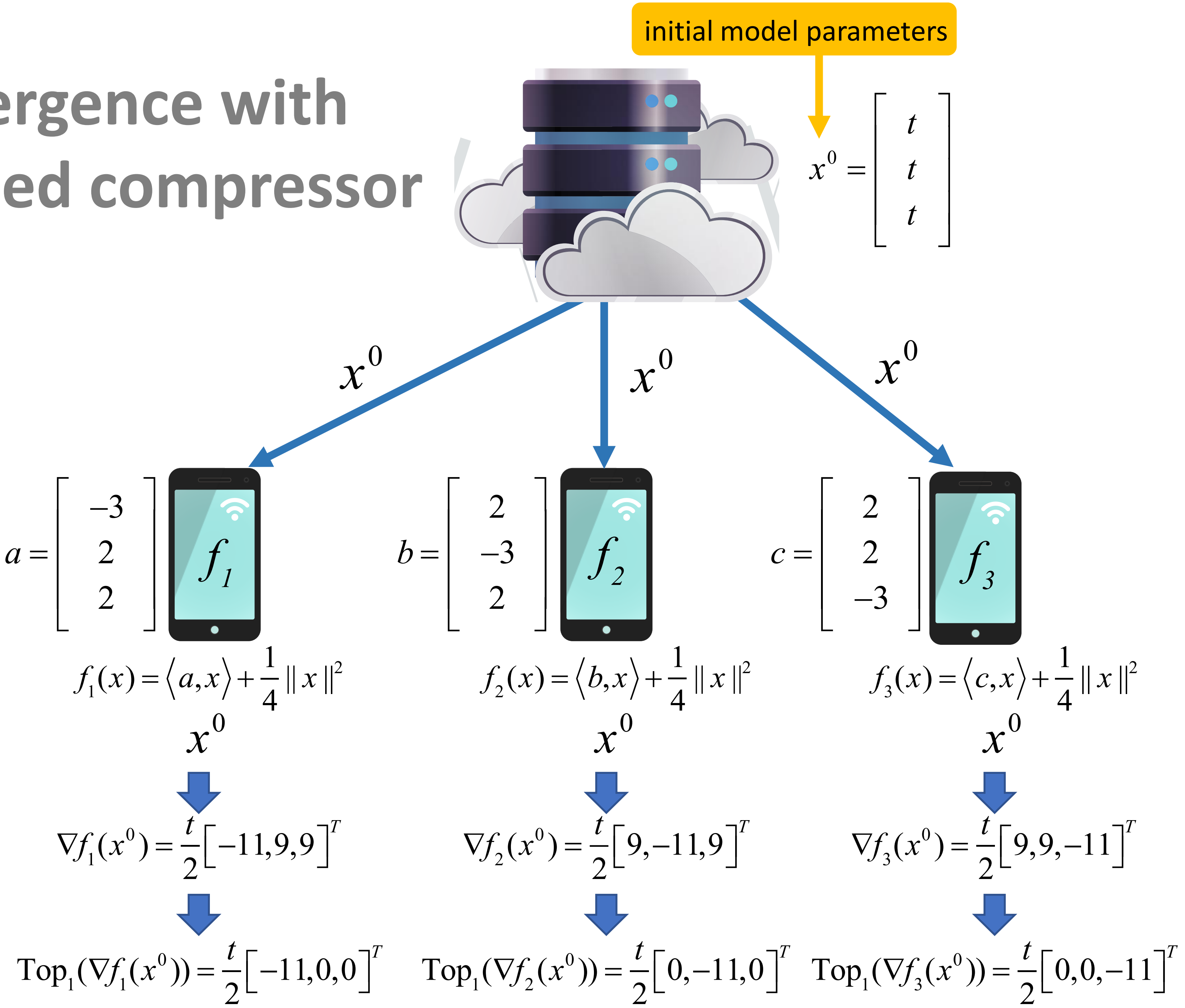
$\mathcal{C}_i^k : \mathbb{R}^d \to \mathbb{R}^d$

$$x^{k+1} = x^k - \frac{\gamma}{n} \sum_{i=1}^{n} \mathcal{C}_i^k(\nabla f_i(x^k))$$

?

Do we have convergence ?

Divergence with biased compressor



Divergence with biased compressor

initial model parameters

stepsize

$$x^0 = \begin{bmatrix} t \\ t \\ t \end{bmatrix} \Rightarrow x^1 = x^0 - \eta \frac{1}{3}\sum_{i=1}^{3} \frac{t}{2} \begin{bmatrix} -11 \\ -11 \\ -11 \end{bmatrix} = \left(1 + \frac{11\eta}{6}\right) x^0$$

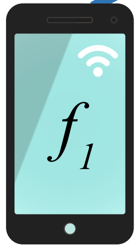
k-th iterations

$$x^k = \left(1 + \frac{11\eta}{6}\right)^k x^0$$

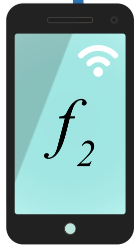
Diverges at exponential rate

$$\frac{t}{2}[-11,0,0]^T \qquad \frac{t}{2}[0,-11,0]^T \qquad \frac{t}{2}[0,0,-11]^T$$

$$a = \begin{bmatrix} -3 \\ 2 \\ 2 \end{bmatrix}$$



$$b = \begin{bmatrix} 2 \\ -3 \\ 2 \end{bmatrix}$$



$$c = \begin{bmatrix} 2 \\ 2 \\ -3 \end{bmatrix}$$

$$f_1(x) = \langle a, x \rangle + \frac{1}{4}\|x\|^2 \qquad f_2(x) = \langle b, x \rangle + \frac{1}{4}\|x\|^2 \qquad f_3(x) = \langle c, x \rangle + \frac{1}{4}\|x\|^2$$

$$x^0 \qquad x^0 \qquad x^0$$

$$\nabla f_1(x^0) = \frac{t}{2}[-11,9,9]^T \qquad \nabla f_2(x^0) = \frac{t}{2}[9,-11,9]^T \qquad \nabla f_3(x^0) = \frac{t}{2}[9,9,-11]^T$$

$$\mathrm{Top}_1(\nabla f_1(x^0)) = \frac{t}{2}[-11,0,0]^T \qquad \mathrm{Top}_1(\nabla f_2(x^0)) = \frac{t}{2}[0,-11,0]^T \qquad \mathrm{Top}_1(\nabla f_3(x^0)) = \frac{t}{2}[0,0,-11]^T$$

Error Feedback

$m_1 \qquad m_2 \qquad m_3 \qquad \ldots$

compression operator

$\mathcal{C}(m_1 + \boxed{e_1})$

initial error

$\boxed{e_1} = 0$

before compression

after compression

updated error

$\boxed{e_2} = \boxed{m_1 + e_1} - \boxed{\mathcal{C}(m_1 + e_1)}$

before compression

after compression

updated error

$\boxed{e_3} = \boxed{m_2 + e_2} - \boxed{\mathcal{C}(m_2 + e_2)}$

$\ldots\ldots$

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

1: **Input:** learning rate γ, compressor $\mathcal{C}(\cdot)$, $\mathbf{x}_0 \in \mathbb{R}^d$
2: **Initialize:** $\mathbf{e}_0 = \mathbf{0} \in \mathbb{R}^d$
3: **for** $t = 0, \ldots, T-1$ **do**
4: $\quad \mathbf{g}_t := \text{stochasticGradient}(\mathbf{x}_t)$
5: $\quad \mathbf{p}_t := \gamma \mathbf{g}_t + \mathbf{e}_t$ ▷ *error correction*
6: $\quad \Delta_t := \mathcal{C}(\mathbf{p}_t)$ ▷ *compression*
7: $\quad \mathbf{x}_{t+1} := \mathbf{x}_t - \Delta_t$ ▷ *update iterate*
8: $\quad \mathbf{e}_{t+1} := \mathbf{p}_t - \Delta_t$ ▷ *update residual error*
9: **end for**

Sebastian U. Stich and Sai Praneeth Karimireddy
The error-feedback framework: Better rates for SGD with delayed gradients and compressed communication.
arXiv preprint arXiv:1909.05350, 2019.

Frank Seide, Hao Fu, Jasha Droppo, Gang Li, and Dong Yu
1-bit stochastic gradient descent and application to data-parallel distributed training of speech DNNs.
In Interspeech 2014, September 2014.

Distributed SGD with Biased Compression and Error Feedback

Parameters: Compressors $\mathcal{C}_i^k \in \mathbb{B}^3(\delta)$; Stepsizes $\{\eta^k\}_{k\geq 0}$; Iteration count K
Initialization: Choose $x^0 \in \mathbb{R}^d$ and $e_i^0 = 0$ for all i
for $k = 0, 1, 2, \ldots, K$ **do**
Server sends x^k to all n machines
All machines in parallel perform these updates:

$$\tilde{g}_i^k = \mathcal{C}_i^k(e_i^k + \eta^k g_i^k)$$
$$e_i^{k+1} = e_i^k + \eta^k g_i^k - \tilde{g}_i^k$$

Each machine i sends $\tilde{g}_i^k$ to the server
Server performs aggregation:

$$x^{k+1} = x^k - \frac{1}{n}\sum_{i=1}^{n} \tilde{g}_i^k$$

end for

Thank you