Byzantine Robustness and Partial Participation Can Be Achieved at Once: Just Clip Gradient Differences

Grigory Malinovsky
KAUST

Peter Richtárik
KAUST

Samuel Horváth
MBZUAI

Eduard Gorbunov
MBZUAI

3rd Workshop on Principles of Distributed Learning

MOHAMED BIN ZAYED UNIVERSITY OF ARTIFICIAL INTELLIGENCE

KAUST

June 21, 2024

G. Malinovsky, P. Richtárik, S. Horváth, E. Gorbunov. *Byzantine Robustness and Partial Participation Can Be Achieved at Once: Just Clip Gradient Differences* (arXiv:2311.14127)

Grigory Malinovsky
PhD student at KAUST

Peter Richtárik
Professor at KAUST

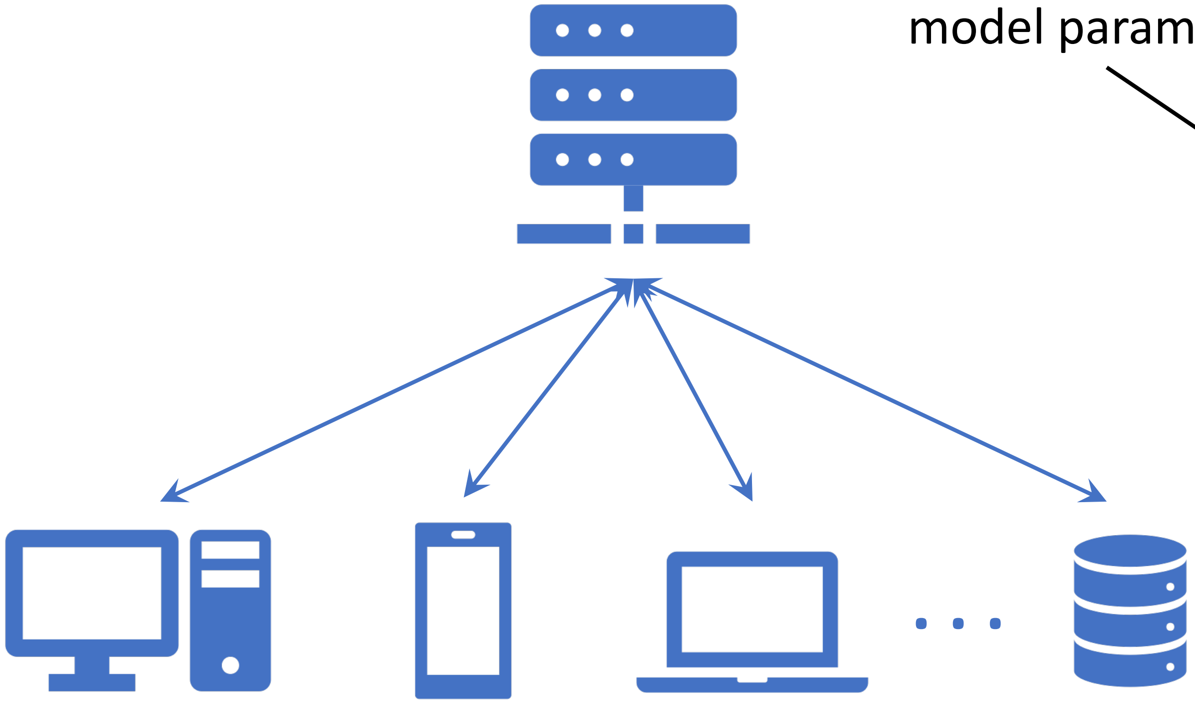
Samuel Horváth
Assistant professor at MBZUAI

Outline

1. Byzantine-Robust Training
2. Robust Aggregation
3. Partial Participation of Clients
4. Ingredient 1: Clipping
5. Ingredient 2: Variance Reduction
6. New Method

Byzantine-Robust Training

The Problem

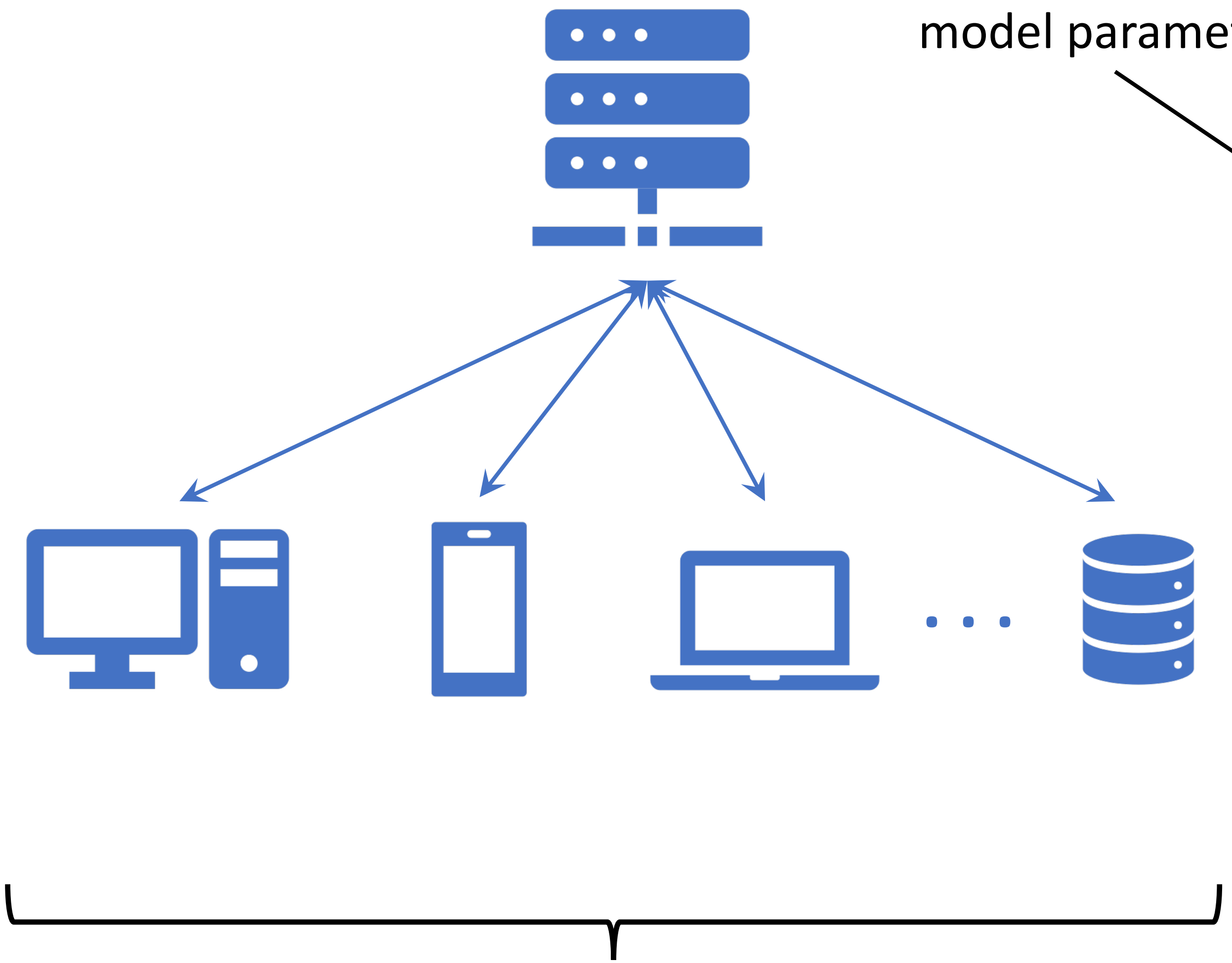


model parameters

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

# of parameters

The Problem



n workers/clients

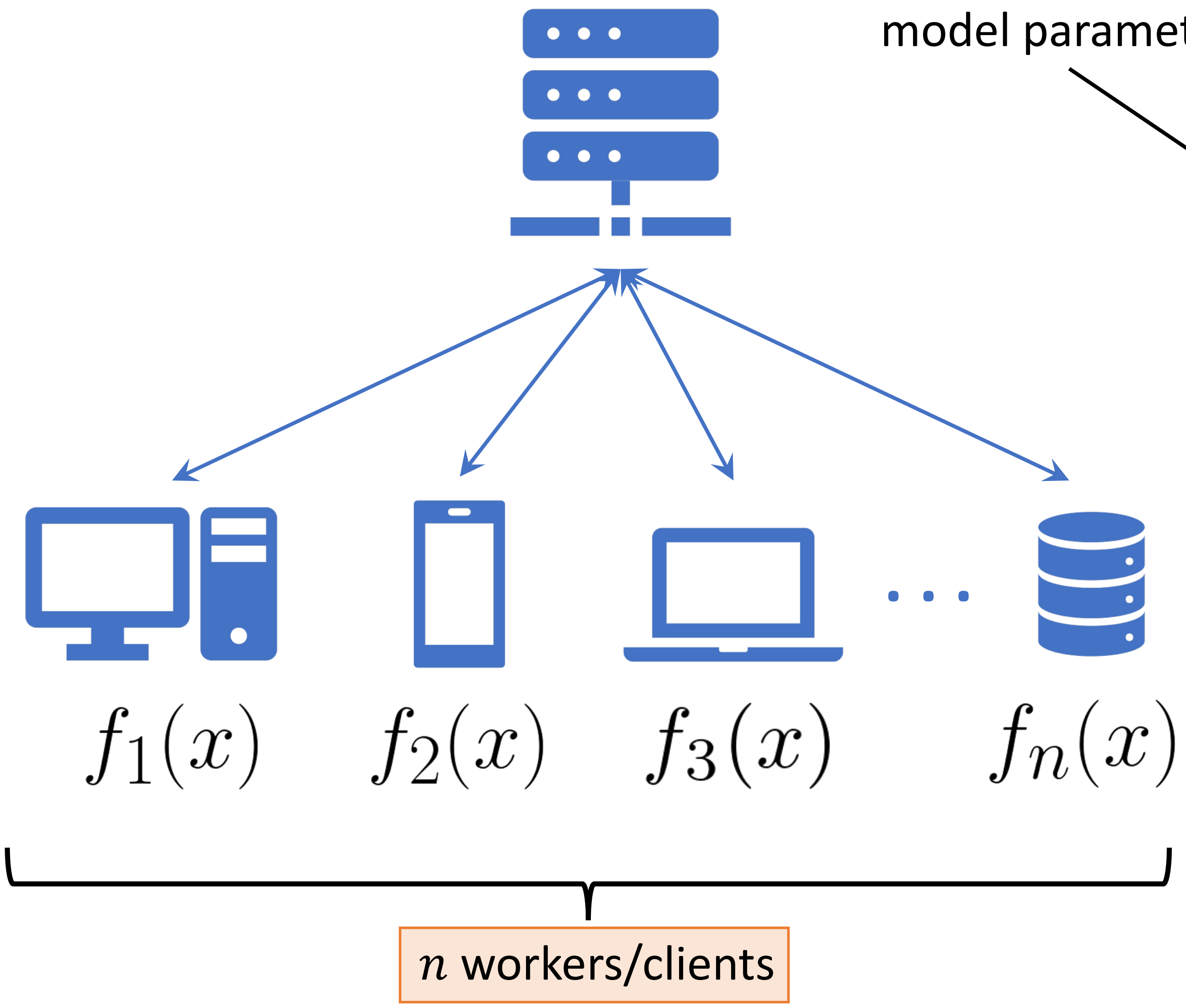
model parameters

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

# of parameters

# of workers/clients

The Problem



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

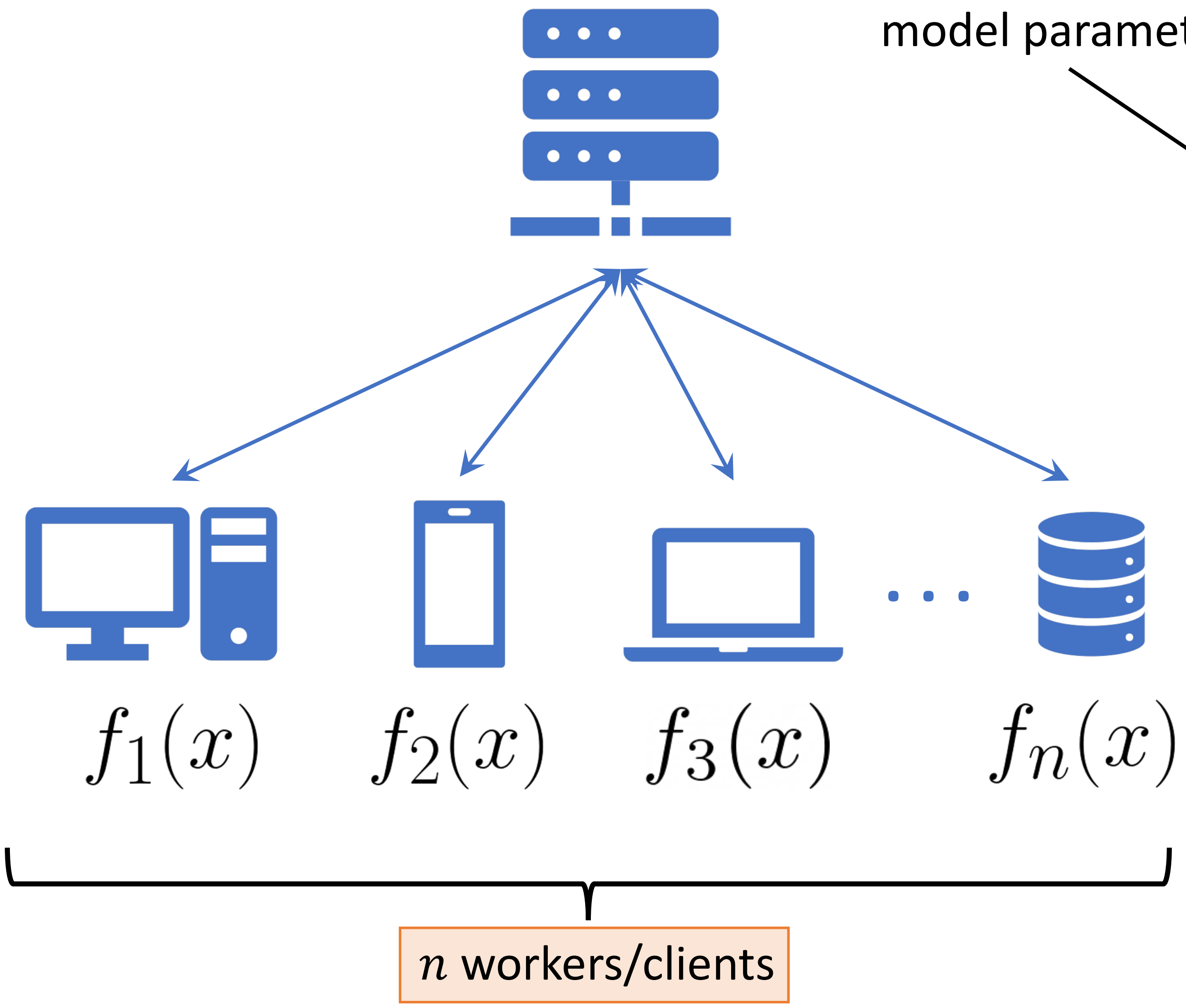
model parameters

\# of parameters

\# of workers/clients

loss on the data accessible on worker i

The Problem



model parameters

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

# of parameters

# of workers/clients

loss on the data accessible on worker i

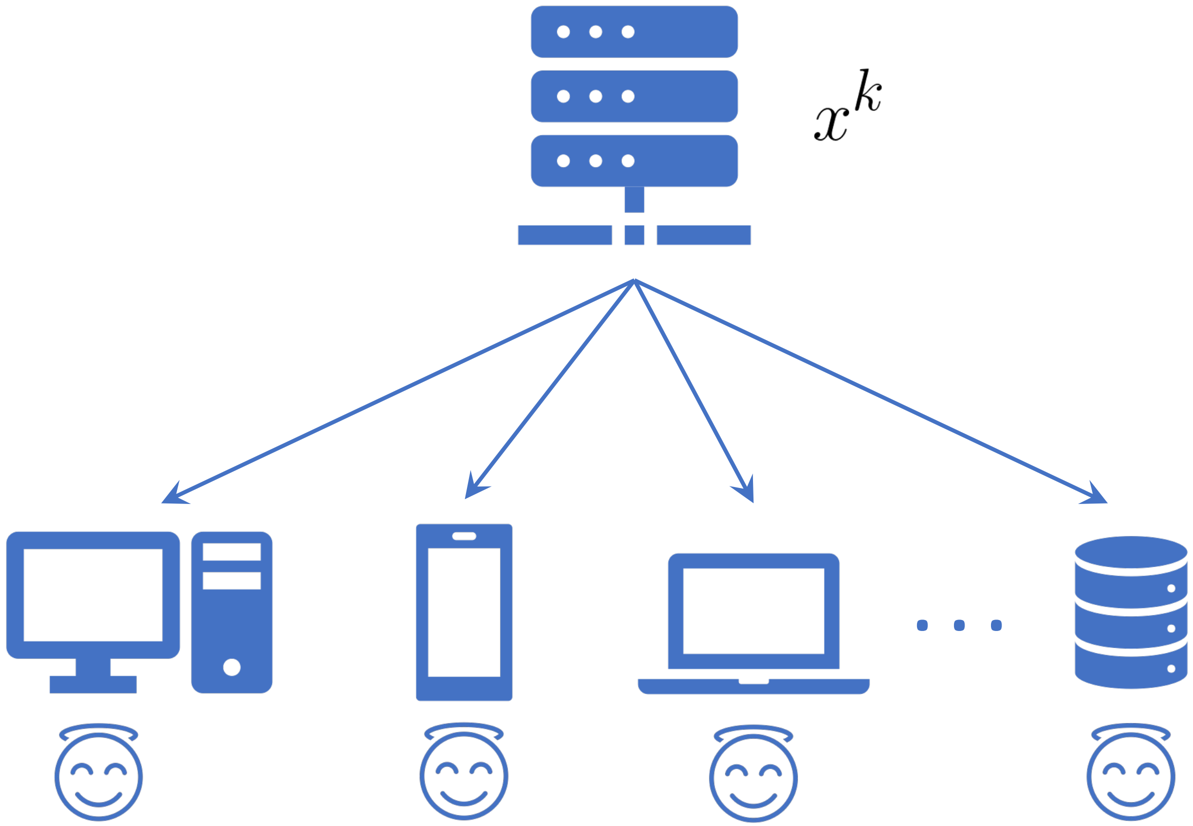
Key features:

- The problem is hard to solve for one client
- Clients do not know each other

Parallel SGD

Iteration k:

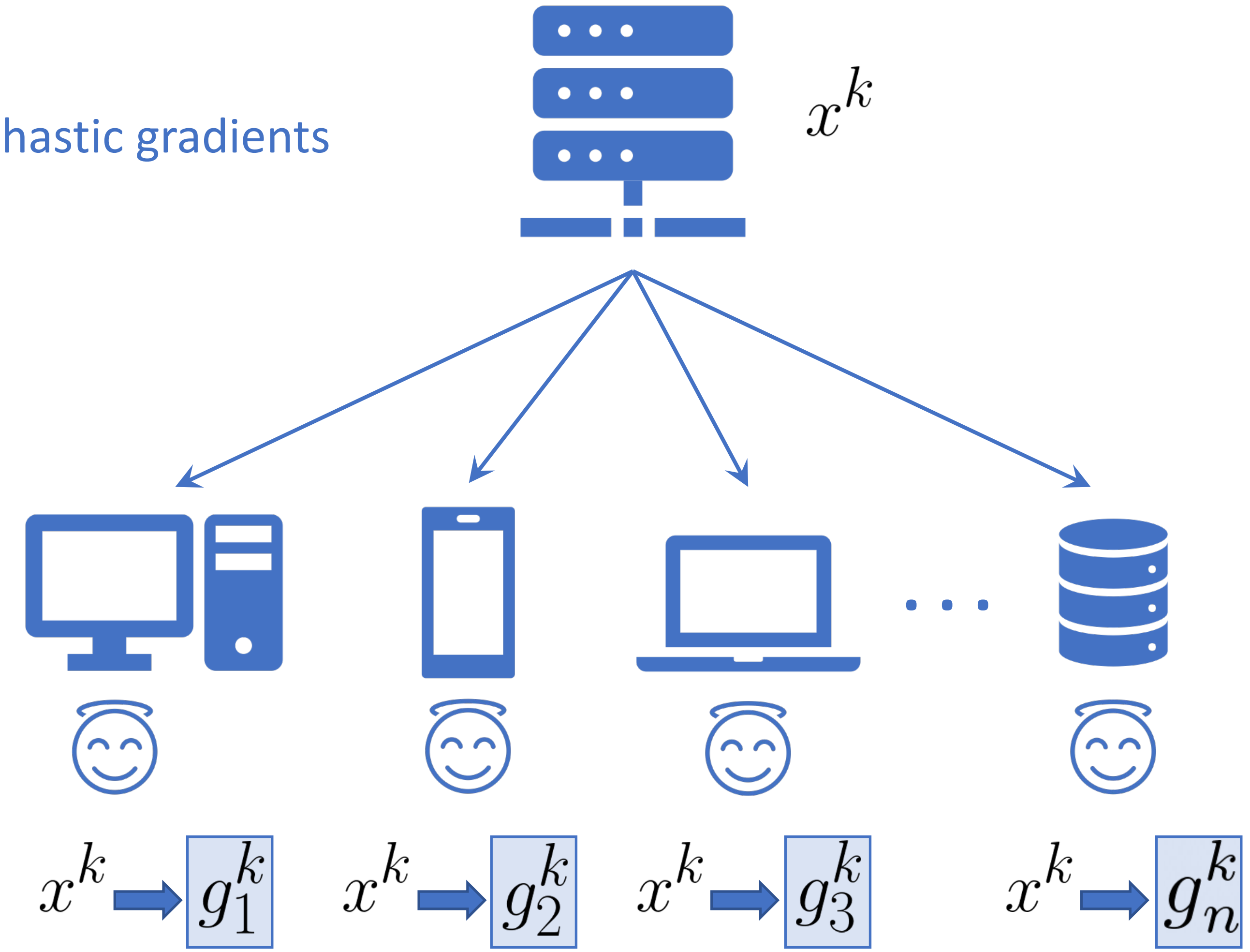
1. Server broadcasts x^k



Parallel SGD

Iteration k:

1. Server broadcasts x^k
2. Workers compute stochastic gradients

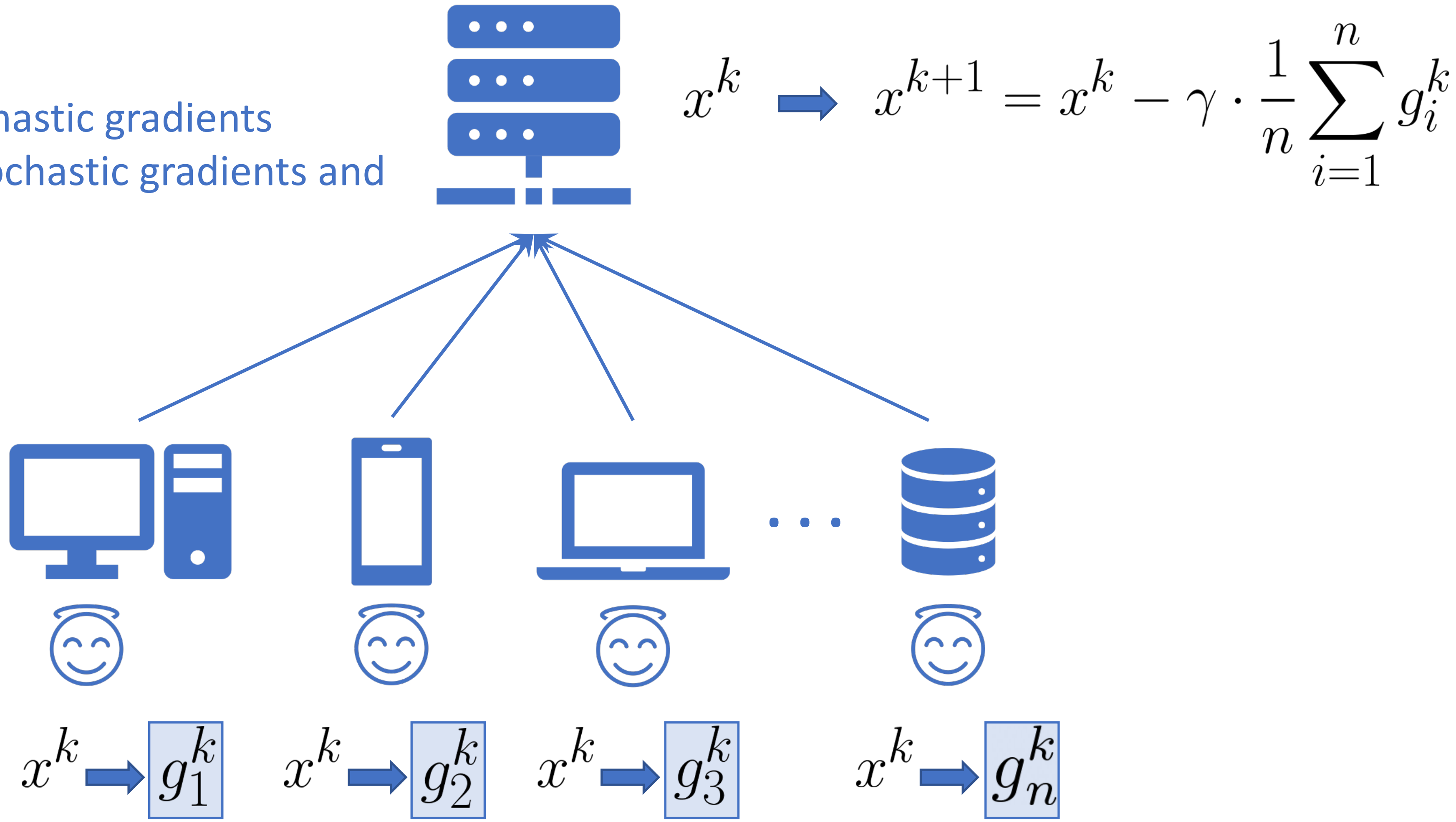


$$\mathbb{E}_k[g_i^k] = \nabla f_i(x^k)$$

Parallel SGD

Iteration k:

1. Server broadcasts x^k
2. Workers compute stochastic gradients
3. Server averages the stochastic gradients and makes an SGD step

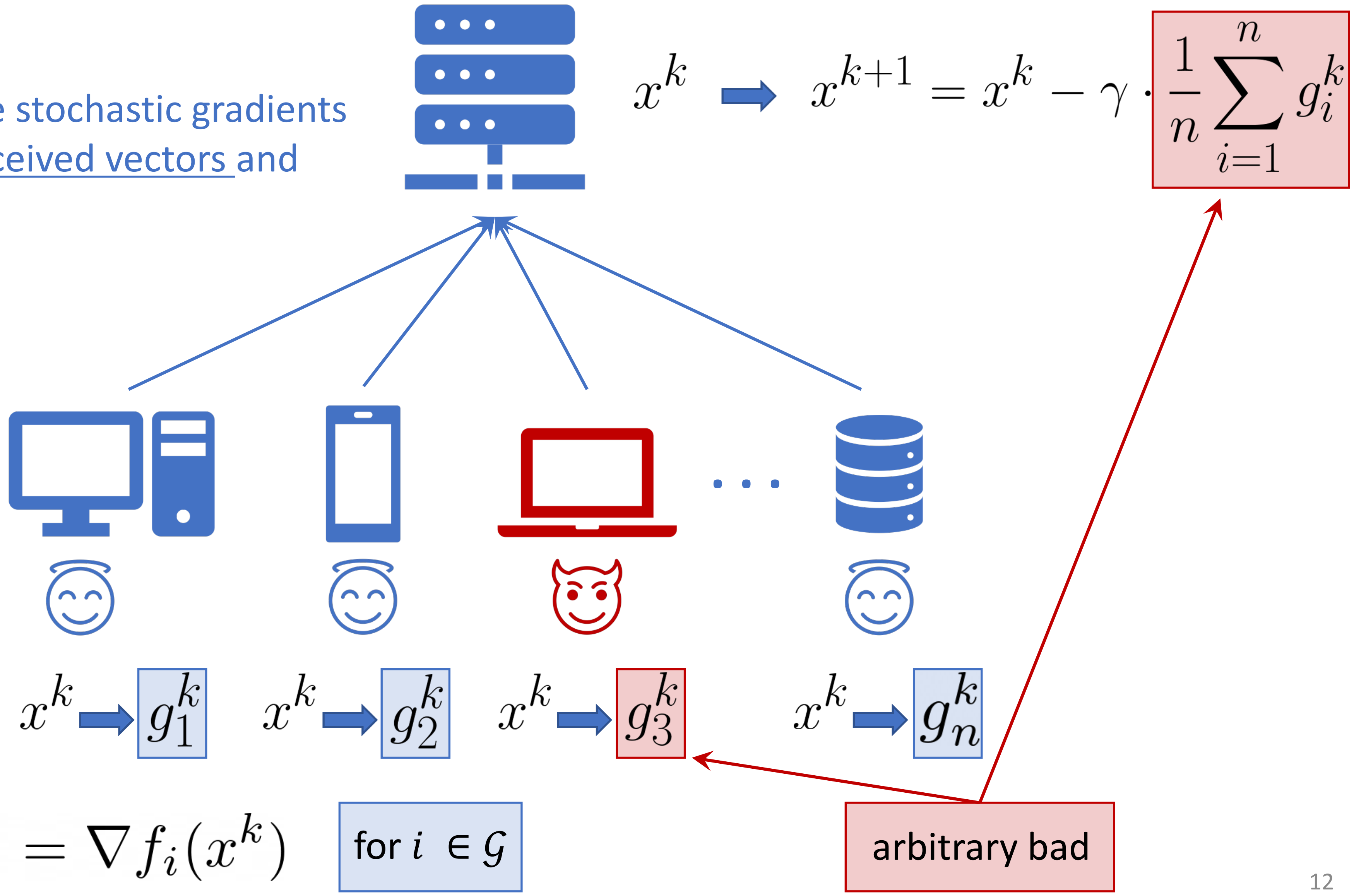


$$\mathbb{E}_k[g_i^k] = \nabla f_i(x^k)$$

Parallel SGD Is Fragile

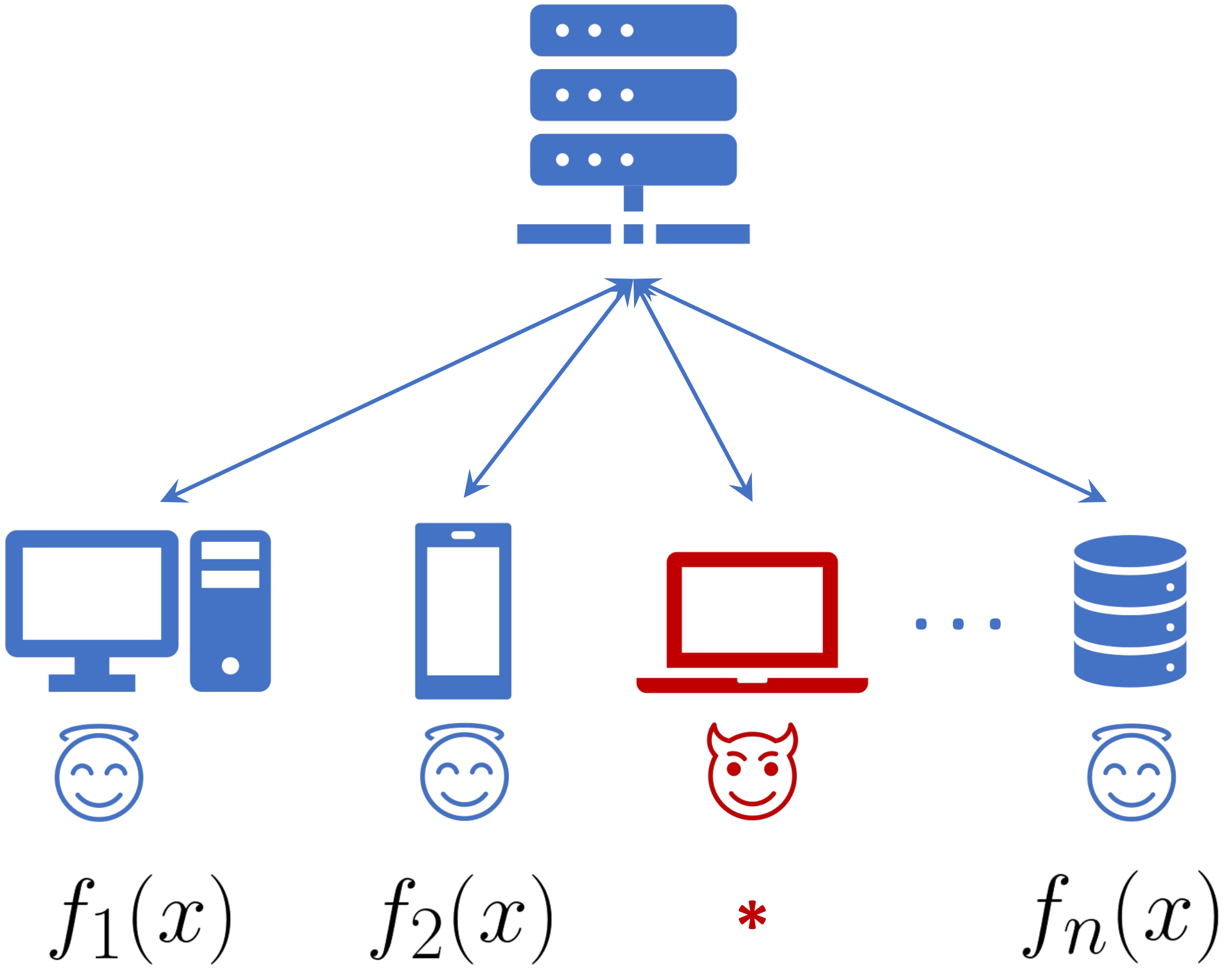
Iteration k:

1. Server broadcasts x^k
2. Good workers compute stochastic gradients
3. Server averages the received vectors and makes an SGD step



$$\mathbb{E}_k[g_i^k] = \nabla f_i(x^k)$$ for $i \in \mathcal{G}$

The Refined Problem Formulation

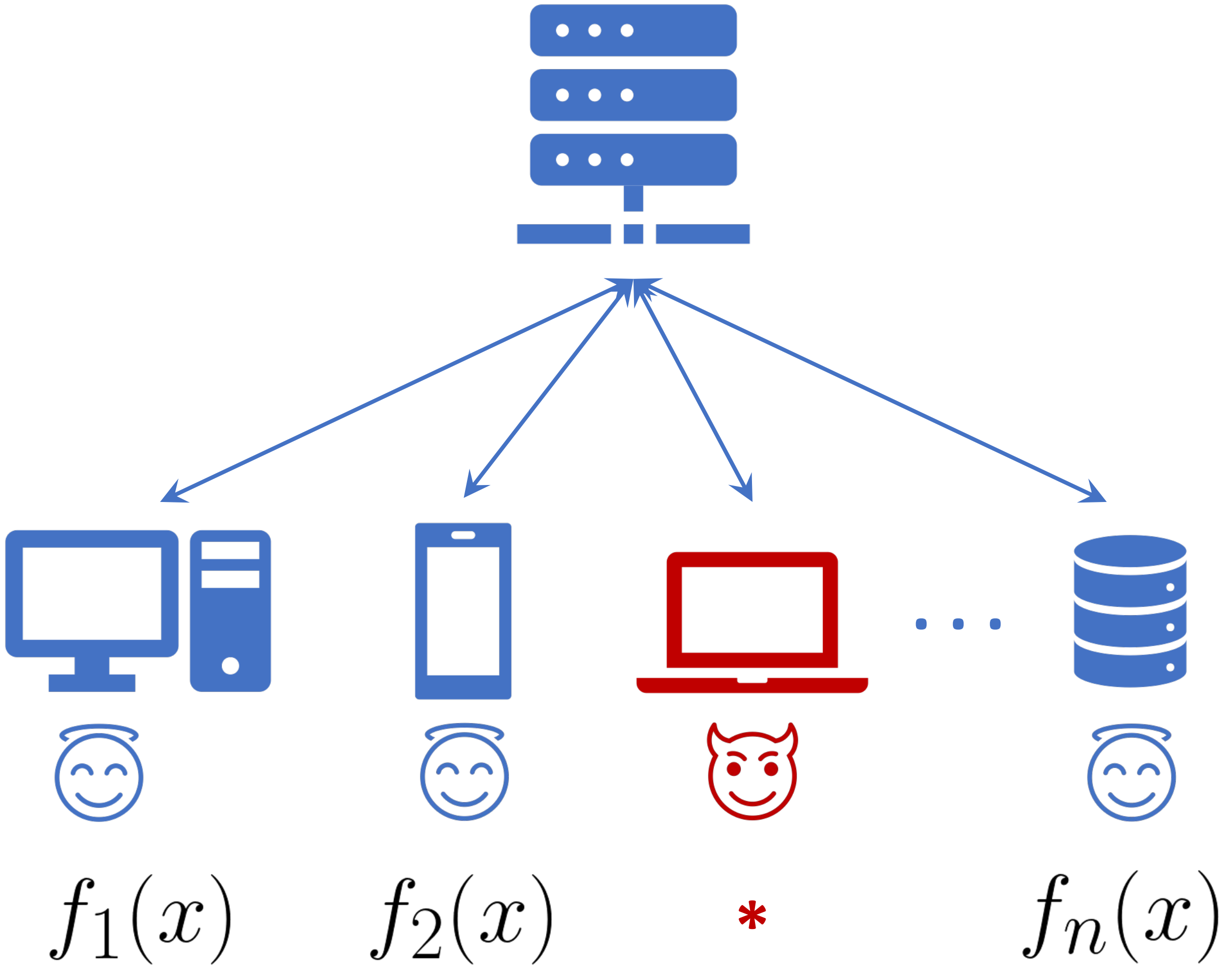


$$\min_{x\in\mathbb{R}^d}\left\{f(x) := \frac{1}{G}\sum_{i\in\mathcal{G}} f_i(x)\right\}$$

Good workers form the majority:

- $\mathcal{G}$ – good workers
- $\mathcal{B}$ – Byzantines (see the page “Byzantine fault” in Wikipedia)
- $\mathcal{G} \sqcup \mathcal{B} = [n],\ \ |\mathcal{G}| = G,\ \ |\mathcal{B}| = B$
- $B \le \delta n,\ \ \delta < {}^1/_2$
- Byzantines are omniscient

The Refined Problem Formulation



$$\min_{x \in \mathbb{R}^d} \left\{ f(x) := \frac{1}{G} \sum_{i \in \mathcal{G}} f_i(x) \right\}$$

Good workers form the majority:

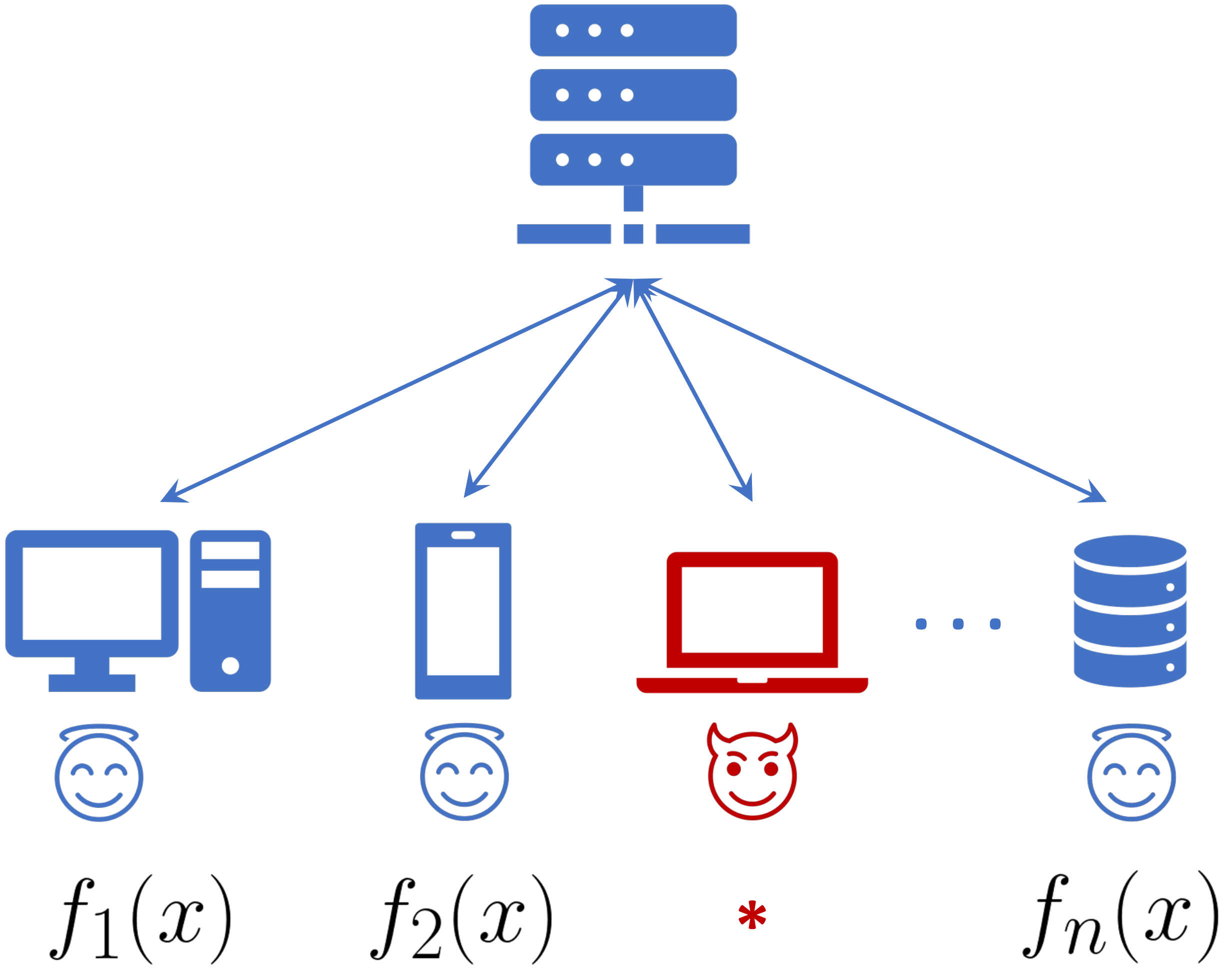
- $\mathcal{G}$ – good workers
- $\mathcal{B}$ – Byzantines (see the page "Byzantine fault" in Wikipedia)
- $\mathcal{G} \sqcup \mathcal{B} = [n], \quad |\mathcal{G}| = G, \quad |\mathcal{B}| = B$
- $B \leq \delta n, \quad \delta < 1/2$
- Byzantines are omniscient

On the heterogeneity:

- Loss functions on good peers cannot be arbitrary heterogeneous
- In this talk, we will assume that

$$\forall\, i \in \mathcal{G} \;\rightarrow\; f_i = f$$

The Refined Problem Formulation



Question: how to solve such problems?

$$\min_{x\in\mathbb{R}^d}\left\{f(x) := \frac{1}{G}\sum_{i\in\mathcal{G}} f_i(x)\right\}$$

Good workers form the majority:

- $\mathcal{G}$ – good workers
- $\mathcal{B}$ – Byzantines (see the page "Byzantine fault" in Wikipedia)
- $\mathcal{G} \sqcup \mathcal{B} = [n], \quad |\mathcal{G}| = G, \quad |\mathcal{B}| = B$
- $B \leq \delta n, \quad \delta < {}^1/_2$
- Byzantines are omniscient

On the heterogeneity:

- Loss functions on good peers cannot be arbitrary heterogeneous
- In this talk, we will assume that

$$\forall\, i \in \mathcal{G} \;\rightarrow\; f_i = f$$

Robust Aggregation

“Middle-Seekers” Aggregators

Natural idea: replace the averaging with more robust aggregation rule!

$$x^{k+1} = x^k - \gamma g^k \quad \Rightarrow \quad x^{k+1} = x^k - \gamma \widehat{g}^k$$

$$g^k = \frac{1}{n}\sum_{i=1}^{n} g_i^k \quad \Rightarrow \quad \widehat{g}^k = \texttt{RAgg}\left(g_1^k, g_2^k, \ldots, g_n^k\right)$$

Question: how to choose aggregator?

“Middle-Seekers” Aggregators

- Geometric median (RFA):

Pillutla, K., Kakade, S. M., & Harchaoui, Z. (2019). Robust aggregation for federated learning. arXiv preprint arXiv:1912.13445.

$$\widehat{g}^k = \arg\min_{g \in \mathbb{R}^d} \sum_{i=1}^{n} \|g - g_i^k\|_2$$

"Middle-Seekers" Aggregators

- Geometric median (RFA):

  Pillutla, K., Kakade, S. M., & Harchaoui, Z. (2019). Robust aggregation for federated learning. arXiv preprint arXiv:1912.13445.

$$\widehat{g}^k = \arg\min_{g \in \mathbb{R}^d} \sum_{i=1}^{n} \|g - g_i^k\|_2$$

- Coordinate-wise median (CM):

  Yin, D., Chen, Y., Kannan, R., & Bartlett, P. (2018, July). Byzantine-robust distributed learning: Towards optimal statistical rates. *In International Conference on Machine Learning* (pp. 5650-5659). PMLR.

$$\widehat{g}^k = \arg\min_{g \in \mathbb{R}^d} \sum_{i=1}^{n} \|g - g_i^k\|_1$$

"Middle-Seekers" Aggregators

- Geometric median (RFA):

Pillutla, K., Kakade, S. M., & Harchaoui, Z. (2019). Robust aggregation for federated learning. arXiv preprint arXiv:1912.13445.

$$\widehat{g}^k = \arg\min_{g \in \mathbb{R}^d} \sum_{i=1}^{n} \|g - g_i^k\|_2$$

- Coordinate-wise median (CM):

Yin, D., Chen, Y., Kannan, R., & Bartlett, P. (2018, July). Byzantine-robust distributed learning: Towards optimal statistical rates. *In International Conference on Machine Learning* (pp. 5650-5659). PMLR.

$$\widehat{g}^k = \arg\min_{g \in \mathbb{R}^d} \sum_{i=1}^{n} \|g - g_i^k\|_1$$

- Krum estimator:

Blanchard, P., El Mhamdi, E. M., Guerraoui, R., & Stainer, J. (2017, December). Machine learning with adversaries: Byzantine tolerant gradient descent. *In Proceedings of the 31st International Conference on Neural Information Processing Systems* (pp. 118-128).

$$\widehat{g}^k = \arg\min_{g \in \{g_1^k, \ldots, g_n^k\}} \sum_{i \in \mathcal{N}_{n-B-2}(g)} \|g - g_i^k\|_2^2$$

indices of the closest $n - B - 2$ workers to g

Simple Example When “Middle-Seekers” Are Good

Let $d = 1, \mathcal{G} = \{1, 2, 3, 4\}, \mathcal{B} = \{5, 6\}, g_1^k = 1.5, g_2^k = 2, g_3^k = 2.5, g_4^k = 3$, and Byzantines are trying to shift the estimator via sending $g_5^k = g_6^k = 1000$. In this case,

Simple Example When "Middle-Seekers" Are Good

Let $d = 1, \mathcal{G} = \{1,2,3,4\}, \mathcal{B} = \{5,6\}, g_1^k = 1.5, g_2^k = 2, g_3^k = 2.5, g_4^k = 3$, and Byzantines are trying to shift the estimator via sending $g_5^k = g_6^k = 1000$. In this case,

- Average of the good workers: $\bar{g}^k = \frac{1}{4}\sum_{i=1}^{4} g_4^k = 2.25$
- Average estimator: $g^k = \frac{1}{6}\sum_{i=1}^{6} g_i^k = 335$
- Median: $\hat{g}^k$ – any number from $[2.5, 3]$
- Krum estimator: $\hat{g}^k = 2$ or 2.5

Simple Example When “Middle-Seekers” Are Good

Let $d = 1, \mathcal{G} = \{1, 2, 3, 4\}, \mathcal{B} = \{5, 6\}, g_1^k = 1.5, g_2^k = 2, g_3^k = 2.5, g_4^k = 3$, and Byzantines are trying to shift the estimator via sending $g_5^k = g_6^k = 1000$. In this case,

- Average of the good workers: $\bar{g}^k = \frac{1}{4}\sum_{i=1}^{4} g_4^k = 2.25$
- Average estimator: $g^k = \frac{1}{6}\sum_{i=1}^{6} g_i^k = 335$
- Median: $\hat{g}^k$ – any number from $[2.5, 3]$
- Krum estimator: $\hat{g}^k = 2$ or 2.5

“Middle-seekers” can be good for reducing the effect of outliers

When “Middle-Seekers” Can Be Bad

Karimireddy, S. P., He, L., & Jaggi, M. (2021, July). Learning from history for byzantine robust optimization. *In International Conference on Machine Learning* (pp. 5311-5319). PMLR.

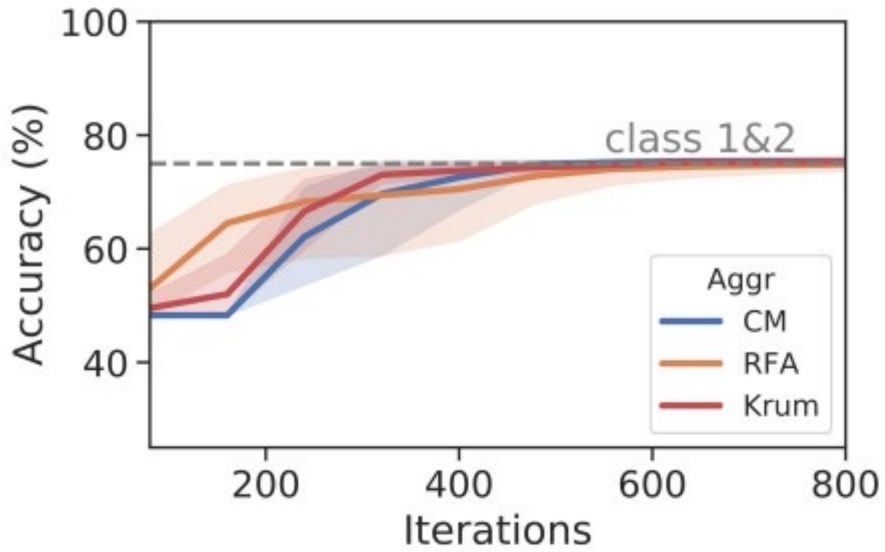


Figure 1: Failure of existing methods on imbalanced MNIST dataset. Only the head classes (class 1 and 2 here) are learnt, and the rest 8 classes are ignored. See Sec. 7.1.

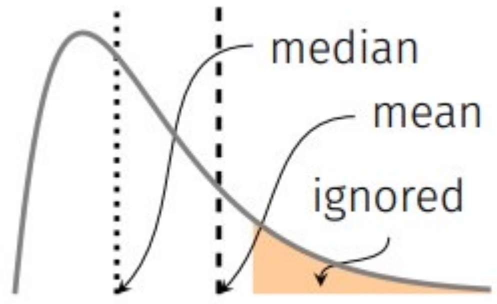
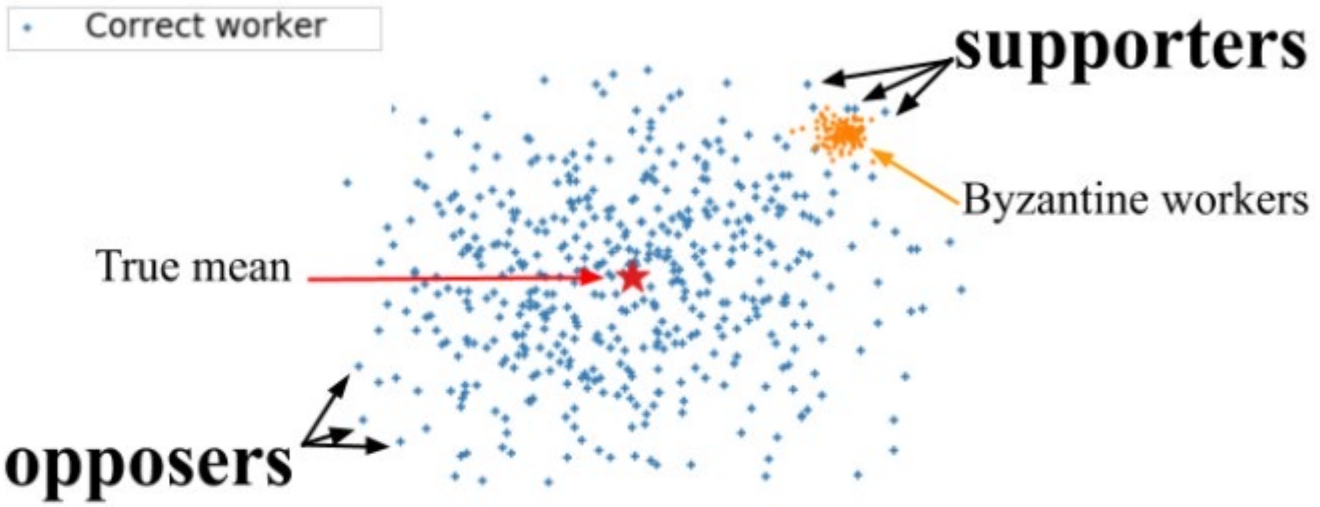


Figure 2: For fat-tailed distributions, median based aggregators ignore the tail. This bias remains even if we have infinite samples.

A Little Is Enough (ALIE) Attack

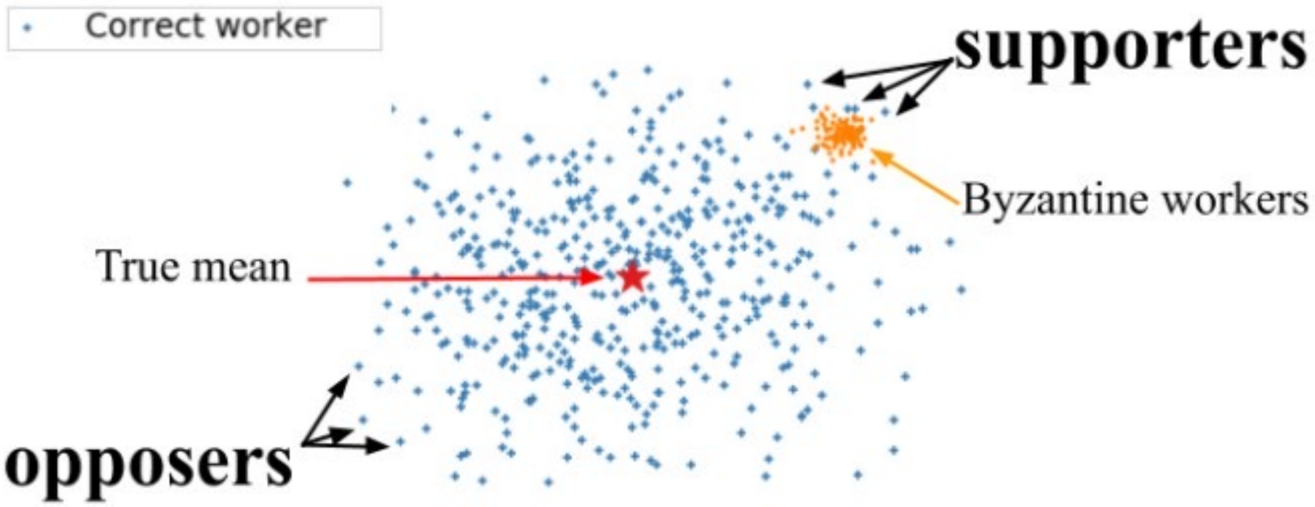
Baruch, G., Baruch, M., & Goldberg, Y. (2019). A little is enough: Circumventing defenses for distributed learning. *Advances in Neural Information Processing Systems*, 32.



Byzantines send the following vectors: $$g_i^k = \mu_{\mathcal{G}} - z\sigma_{\mathcal{G}}$$

A Little Is Enough (ALIE) Attack

Baruch, G., Baruch, M., & Goldberg, Y. (2019). A little is enough: Circumventing defenses for distributed learning. *Advances in Neural Information Processing Systems*, 32.



Byzantines send the following vectors: $$g_i^k = \mu_{\mathcal{G}} - z\sigma_{\mathcal{G}}$$

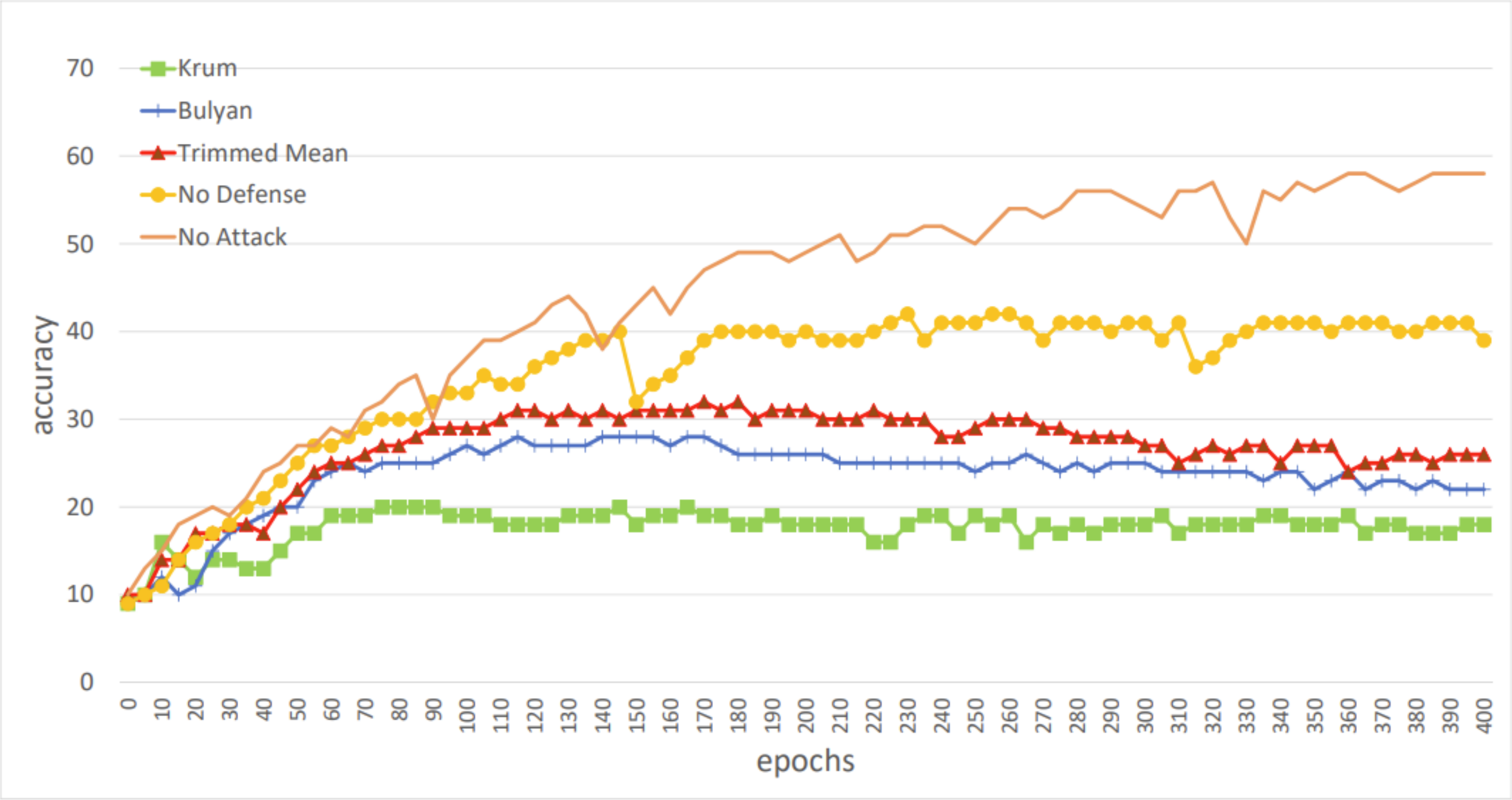
mean of the good workers

coordinate-wise standard deviation of good workers

- Byzantines choose z such that they are close to the "boundary of the cloud"
- Since Byzantines are closer to the mean, "middle-seekers" will treat opposers as outliers

The Result of ALIE Attack on the Training @ CIFAR10

Baruch, G., Baruch, M., & Goldberg, Y. (2019). A little is enough: Circumventing defenses for distributed learning. *Advances in Neural Information Processing Systems*, 32.



"No defense" strategy is more robust! Formal definition of robust aggregation is required!

Robust Aggregation Formalism

Karimireddy, S. P., He, L., & Jaggi, M. (2021, July). Learning from history for byzantine robust optimization. *In International Conference on Machine Learning* (pp. 5311-5319). PMLR.

Definition of (δ, c)–robust aggregator

Let $g_1 \ldots, g_n$ be random variables such that there exist a good subset $\mathcal{G} \subseteq [n]$ of size $G \geq (1-\delta)n > n/2$ such that $\{g_i\}_{\{i \in \mathcal{G}\}}$ are independent and for all fixed pairs of good workers $i, j \in \mathcal{G}$ we have

$$\mathbb{E}\left[\|g_i - g_j\|^2\right] \leq \sigma^2.$$

Let $\bar{g} = \frac{1}{G}\sum_{i \in \mathcal{G}} g_i$. Then $\hat{g} = \mathrm{RAgg}(g_1, \ldots, g_n)$ is called (δ, c)–robust aggregator if for some $c > 0$

$$\mathbb{E}\left[\|\hat{g} - \bar{g}\|^2\right] \leq c\delta\sigma^2$$

Robust Aggregation Formalism

Karimireddy, S. P., He, L., & Jaggi, M. (2021, July). Learning from history for byzantine robust optimization. *In International Conference on Machine Learning* (pp. 5311-5319). PMLR.

Definition of (δ, c)–robust aggregator

Let $g_1 \dots, g_n$ be random variables such that there exist a good subset $\mathcal{G} \subseteq [n]$ of size $G \geq (1-\delta)n > n/2$ such that $\{g_i\}_{\{i \in \mathcal{G}\}}$ are independent and for all fixed pairs of good workers $i, j \in \mathcal{G}$ we have

$$\mathbb{E}\left[\|g_i - g_j\|^2\right] \leq \sigma^2.$$

Let $\bar{g} = \frac{1}{G}\sum_{i \in \mathcal{G}} g_i$. Then $\hat{g} = \mathrm{RAgg}(g_1, \dots, g_n)$ is called (δ, c)–robust aggregator if for some $c > 0$

$$\mathbb{E}\left[\|\hat{g} - \bar{g}\|^2\right] \leq c\delta\sigma^2$$

- Medians and Krum estimators do not satisfy this definition
- **Question:** do such aggregators exist?

Bucketing Fixes “Middle-Seekers”

Karimireddy, S. P., He, L., & Jaggi, M. (2022). Byzantine-Robust Learning on Heterogeneous Datasets via Bucketing. *In International Conference on Learning Representations*.

Bucketing takes $\{g_1, \ldots, g_n\}$, positive integer s, and aggregator Aggr as an input and returns

$$\hat{g} = \texttt{Aggr}(y_1, \ldots, y_{\lceil n/s \rceil})$$

Bucketing Fixes "Middle-Seekers"

Karimireddy, S. P., He, L., & Jaggi, M. (2022). Byzantine-Robust Learning on Heterogeneous Datasets via Bucketing. *In International Conference on Learning Representations*.

Bucketing takes $\{g_1, \ldots, g_n\}$, positive integer s, and aggregator Aggr as an input and returns

$$\widehat{g} = \mathtt{Aggr}(y_1, \ldots, y_{\lceil n/s \rceil})$$

where $y_i = \frac{1}{s} \sum_{k=s(i-1)+1}^{\min\{si,n\}} x_{\pi(k)}$ and $\pi = (\pi(1), \ldots, \pi(n))$ is a random permutation of $[n]$

Bucketing Fixes “Middle-Seekers”

Karimireddy, S. P., He, L., & Jaggi, M. (2022). Byzantine-Robust Learning on Heterogeneous Datasets via Bucketing. *In International Conference on Learning Representations*.

Bucketing takes $\{g_1, \dots, g_n\}$, positive integer s, and aggregator Aggr as an input and returns

$$\widehat{g} = \texttt{Aggr}(y_1, \dots, y_{\lceil n/s \rceil})$$

where $y_i = \frac{1}{s} \sum_{k=s(i-1)+1}^{\min\{si,n\}} x_{\pi(k)}$ and $\pi = (\pi(1), \dots, \pi(n))$ is a random permutation of $[n]$

For any $\delta \leq \delta_{\max}$ and $s = \lfloor \delta_{\max}/\delta \rfloor$

- Krum ∘ Bucketing is (δ, c)–robust aggregator with $c = \mathcal{O}(1)$ and $\delta_{\max} < 1/4$

Bucketing Fixes "Middle-Seekers"

Karimireddy, S. P., He, L., & Jaggi, M. (2022). Byzantine-Robust Learning on Heterogeneous Datasets via Bucketing. *In International Conference on Learning Representations*.

Bucketing takes $\{g_1, \ldots, g_n\}$, positive integer s, and aggregator Aggr as an input and returns

$$\widehat{g} = \texttt{Aggr}(y_1, \ldots, y_{\lceil n/s \rceil})$$

where $y_i = \frac{1}{s} \sum_{k=s(i-1)+1}^{\min\{si,n\}} x_{\pi(k)}$ and $\pi = (\pi(1), \ldots, \pi(n))$ is a random permutation of $[n]$

For any $\delta \leq \delta_{\max}$ and $s = \lfloor {}^{\delta_{\max}}/_{\delta} \rfloor$

- Krum ∘ Bucketing is (δ, c)–robust aggregator with $c = \mathcal{O}(1)$ and $\delta_{\max} < {}^1/_4$
- RFA ∘ Bucketing is (δ, c)–robust aggregator with $c = \mathcal{O}(1)$ and $\delta_{\max} < {}^1/_2$

Bucketing Fixes "Middle-Seekers"

Karimireddy, S. P., He, L., & Jaggi, M. (2022). Byzantine-Robust Learning on Heterogeneous Datasets via Bucketing. *In International Conference on Learning Representations*.

Bucketing takes $\{g_1, \ldots, g_n\}$, positive integer s, and aggregator Aggr as an input and returns

$$\widehat{g} = \texttt{Aggr}(y_1, \ldots, y_{\lceil n/s \rceil})$$

where $y_i = \frac{1}{s} \sum_{k=s(i-1)+1}^{\min\{si,n\}} x_{\pi(k)}$ and $\pi = (\pi(1), \ldots, \pi(n))$ is a random permutation of $[n]$

For any $\delta \leq \delta_{\max}$ and $s = \lfloor {}^{\delta_{\max}}/_{\delta} \rfloor$

- Krum ∘ Bucketing is (δ, c)–robust aggregator with $c = \mathcal{O}(1)$ and $\delta_{\max} < {}^1/_4$
- RFA ∘ Bucketing is (δ, c)–robust aggregator with $c = \mathcal{O}(1)$ and $\delta_{\max} < {}^1/_2$
- CM ∘ Bucketing is (δ, c)–robust aggregator with $c = \mathcal{O}(d)$ and $\delta_{\max} < {}^1/_2$

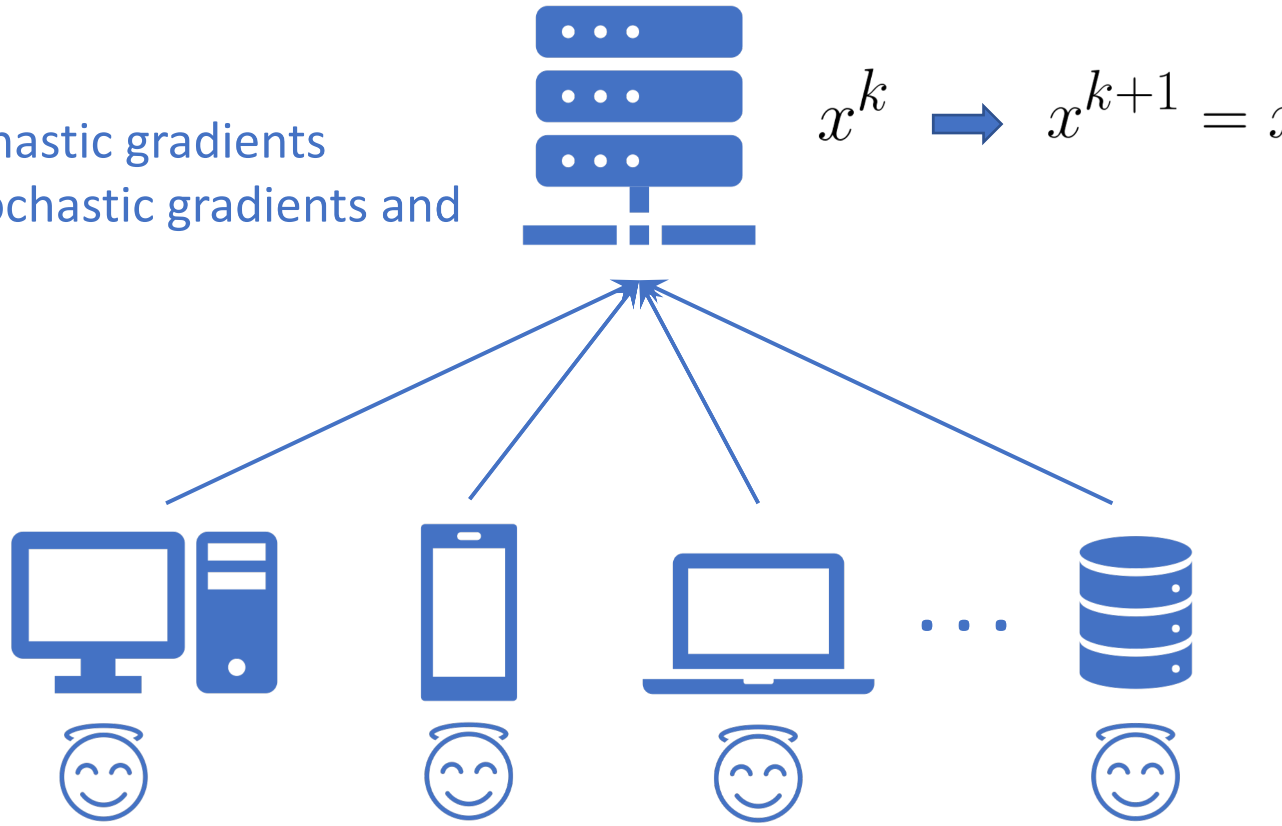
Moreover, these estimators are agnostic to σ^2!

Partial Participation

Parallel SGD

Iteration k:

1. Server broadcasts x^k
2. Workers compute stochastic gradients
3. Server averages the stochastic gradients and makes an SGD step



$$x^k \rightarrow x^{k+1} = x^k - \gamma \cdot \frac{1}{n}\sum_{i=1}^{n} g_i^k$$

$$x^k \rightarrow g_1^k \quad x^k \rightarrow g_2^k \quad x^k \rightarrow g_3^k \quad \ldots \quad x^k \rightarrow g_n^k$$

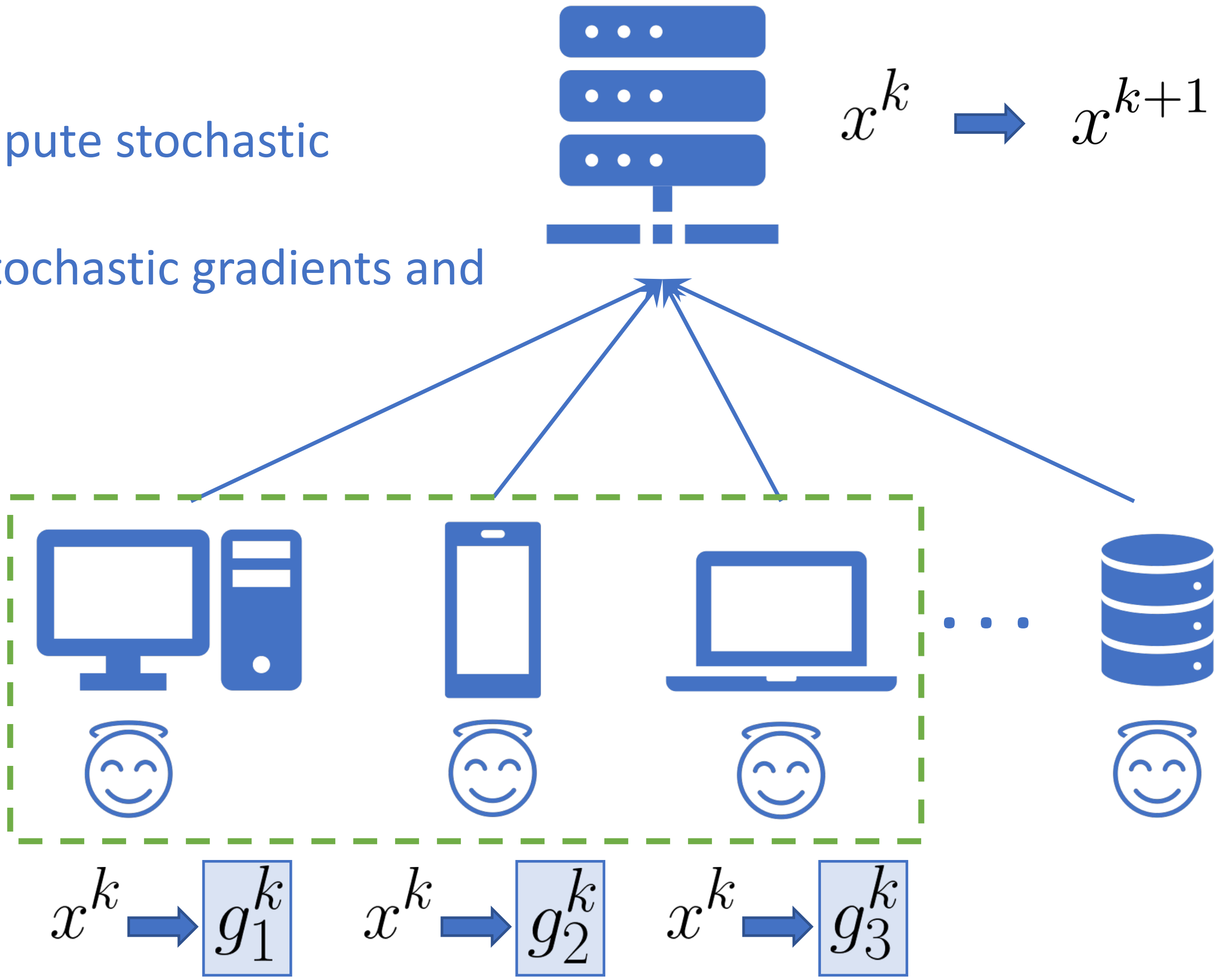
$$\mathbb{E}_k\left[g_i^k\right] = \nabla f_i(x^k)$$

Parallel SGD with Partial Participation of Clients

Iteration k:

1. Server broadcasts x^k
2. <u>Sampled workers</u> compute stochastic gradients
3. Server averages the stochastic gradients and makes an SGD step

$$x^k \rightarrow x^{k+1} = x^k - \gamma \cdot \frac{1}{3}\sum_{i=1}^{3} g_i^k$$



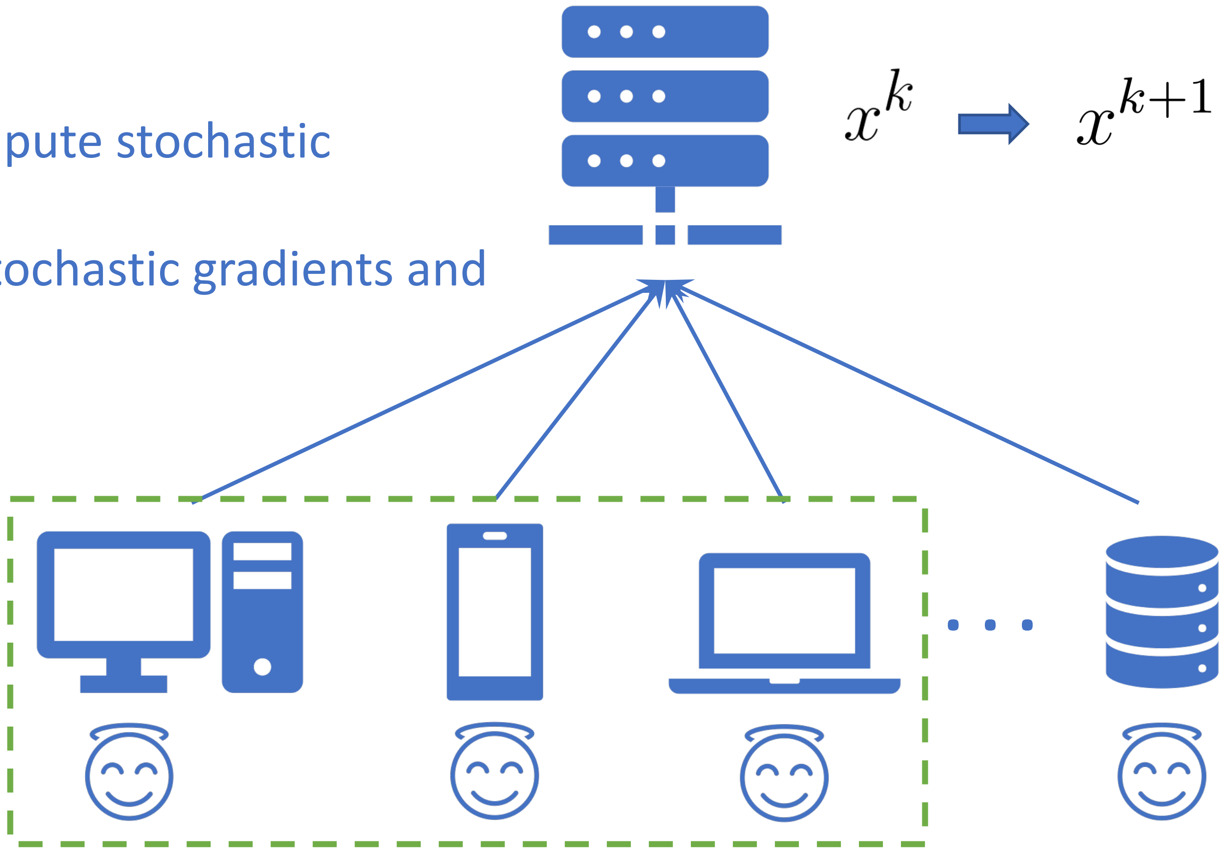
$$\mathbb{E}_k[g_i^k] = \nabla f_i(x^k)$$

Parallel SGD with Partial Participation of Clients

Iteration k:

1. Server broadcasts x^k
2. Sampled workers compute stochastic gradients
3. Server averages the stochastic gradients and makes an SGD step

$$x^k \rightarrow x^{k+1} = x^k - \gamma \cdot \frac{1}{3}\sum_{i=1}^{3} g_i^k$$



$x^k \rightarrow g_1^k$ $\quad$ $x^k \rightarrow g_2^k$ $\quad$ $x^k \rightarrow g_3^k$

$$\mathbb{E}_k[g_i^k] = \nabla f_i(x^k)$$

Why is it used?

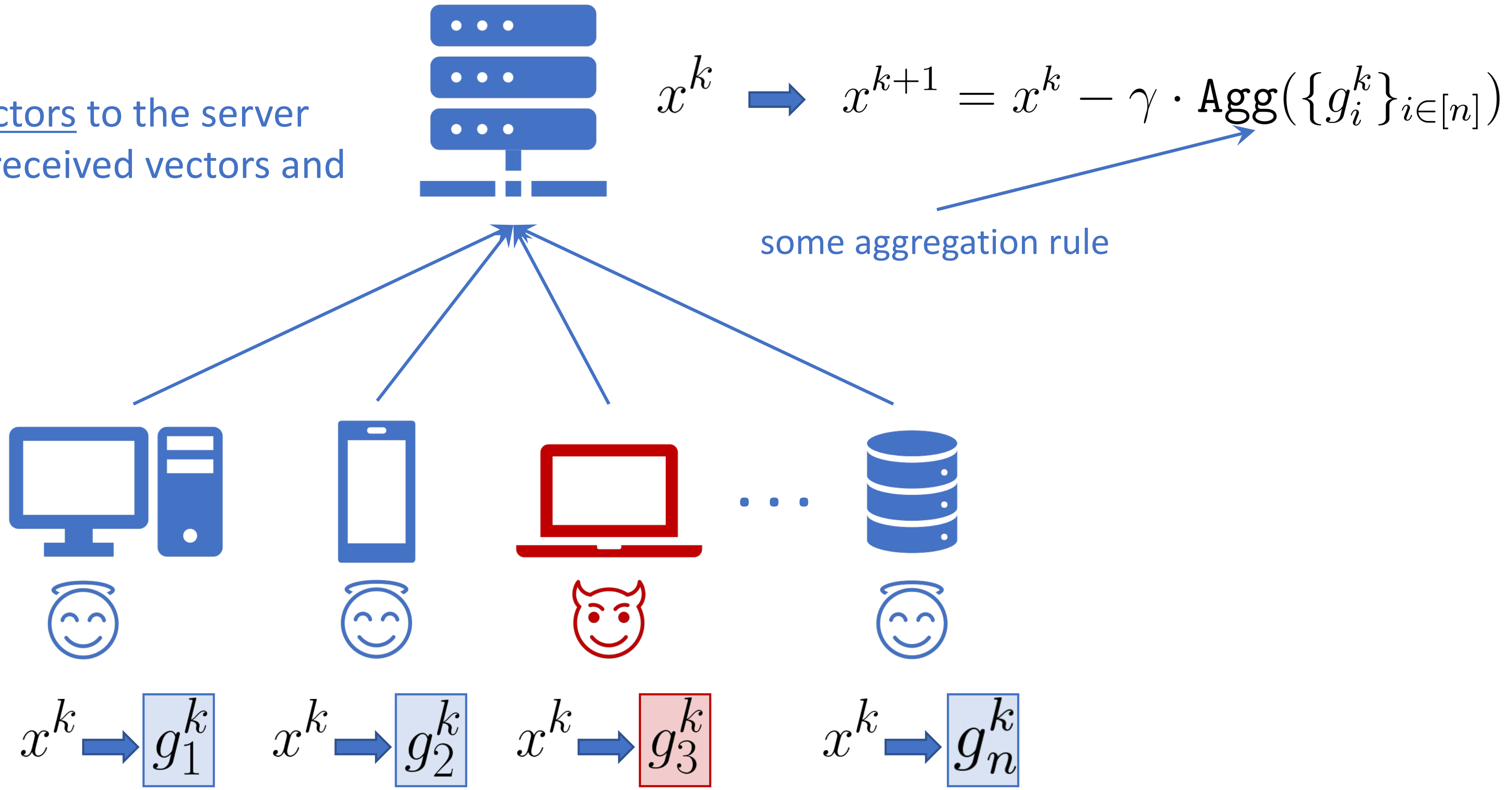
Clients sampling may speed up the training

Some clients may be unavailable at certain moments (poor connection, low battery, no free compute power)

Byzantine-Robust Method

Iteration k:

1. Server broadcasts x^k
2. Workers send some vectors to the server
3. Server aggregates the received vectors and makes an SGD step



Byzantine-Robust Method with Partial Participation

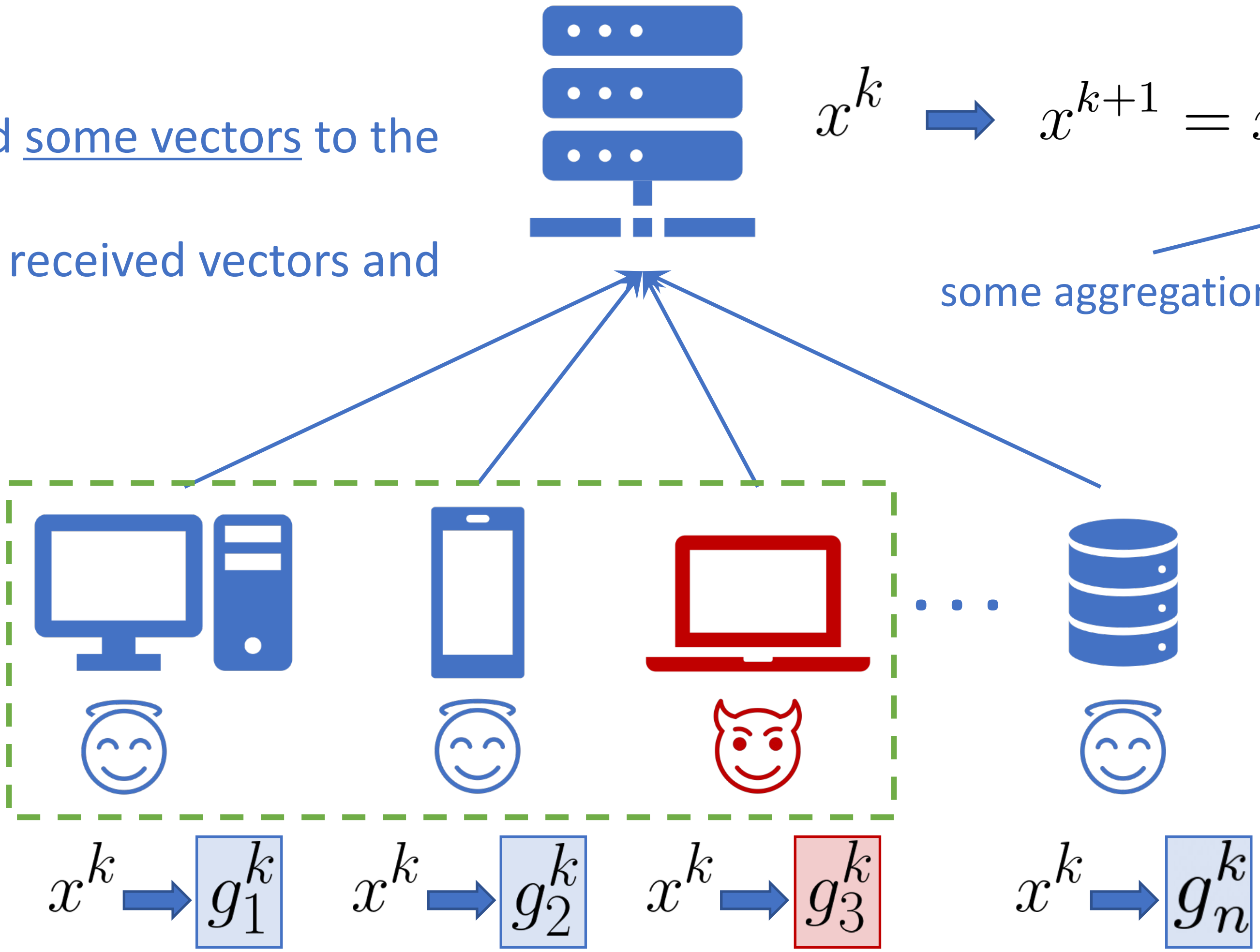
Iteration k:

1. Server broadcasts x^k
2. Sampled workers send some vectors to the server
3. Server aggregates the received vectors and makes an SGD step

$$x^k \rightarrow x^{k+1} = x^k - \gamma \cdot \texttt{Agg}(\{g_i^k\}_{S_k})$$

some aggregation rule

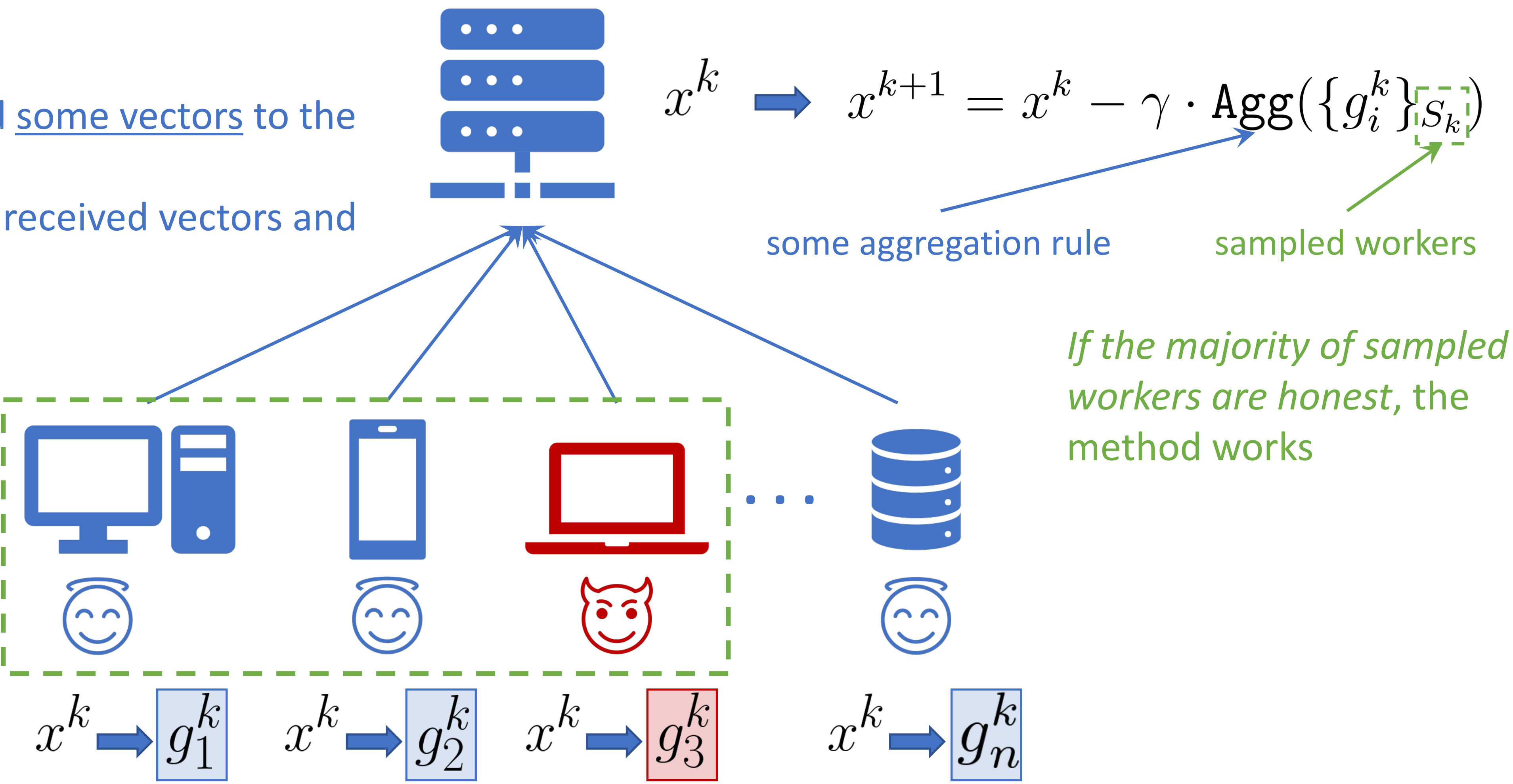
sampled workers



Byzantine-Robust Method with Partial Participation

Iteration k:

1. Server broadcasts x^k
2. Sampled workers send some vectors to the server
3. Server aggregates the received vectors and makes an SGD step



If the majority of sampled workers are honest, the method works

Byzantine-Robust Method with Partial Participation

Iteration k:

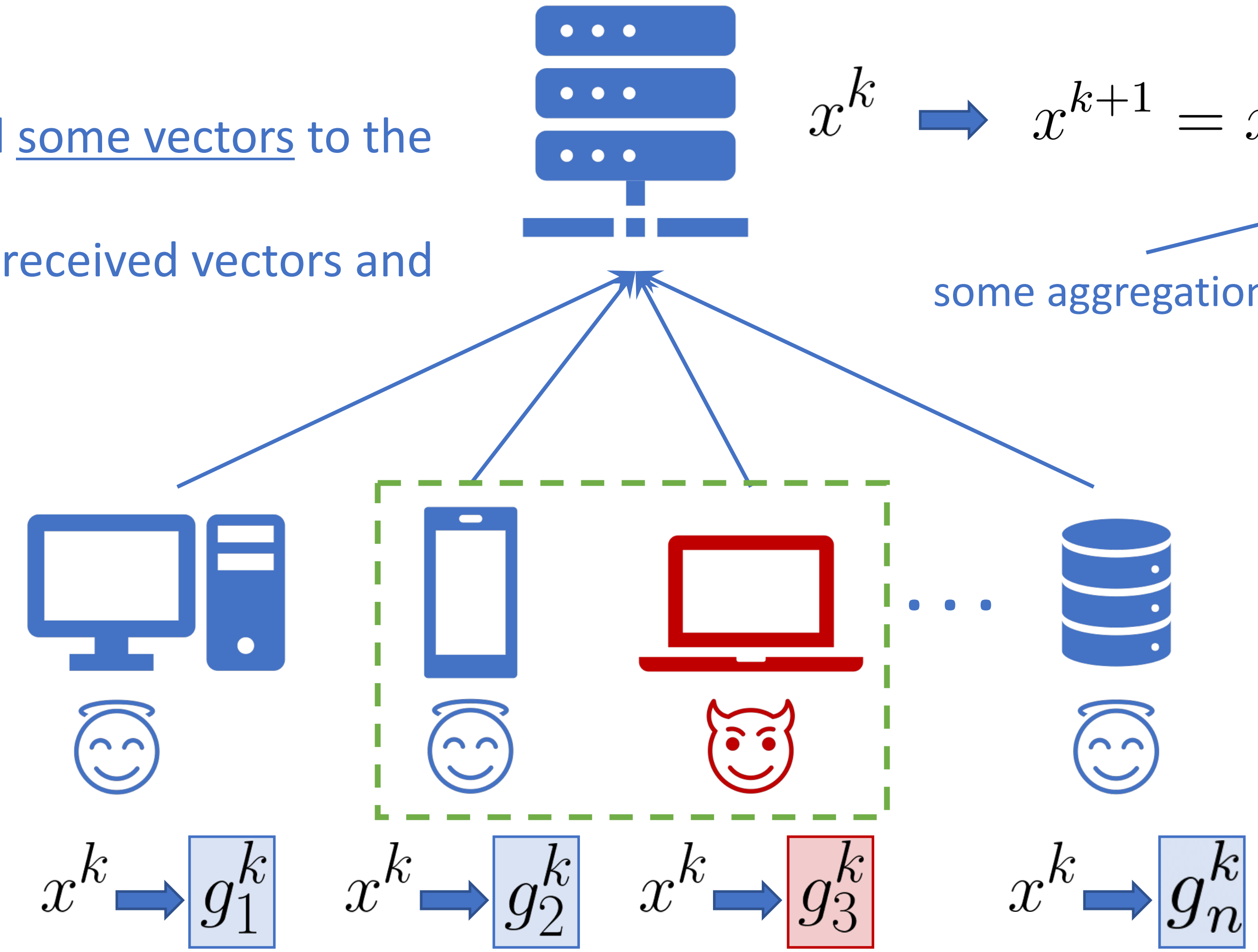
1. Server broadcasts x^k
2. Sampled workers send some vectors to the server
3. Server aggregates the received vectors and makes an SGD step

No robustness when honest workers are not in majority!

$$x^k \Rightarrow x^{k+1} = x^k - \gamma \cdot \mathrm{Agg}(\{g_i^k\}_{S_k})$$

some aggregation rule

sampled workers



If the majority of sampled workers are honest, the method works

If honest workers are not in majorty, the method can fail

Byzantine-Robust Method with Partial Participation

Iteration k:

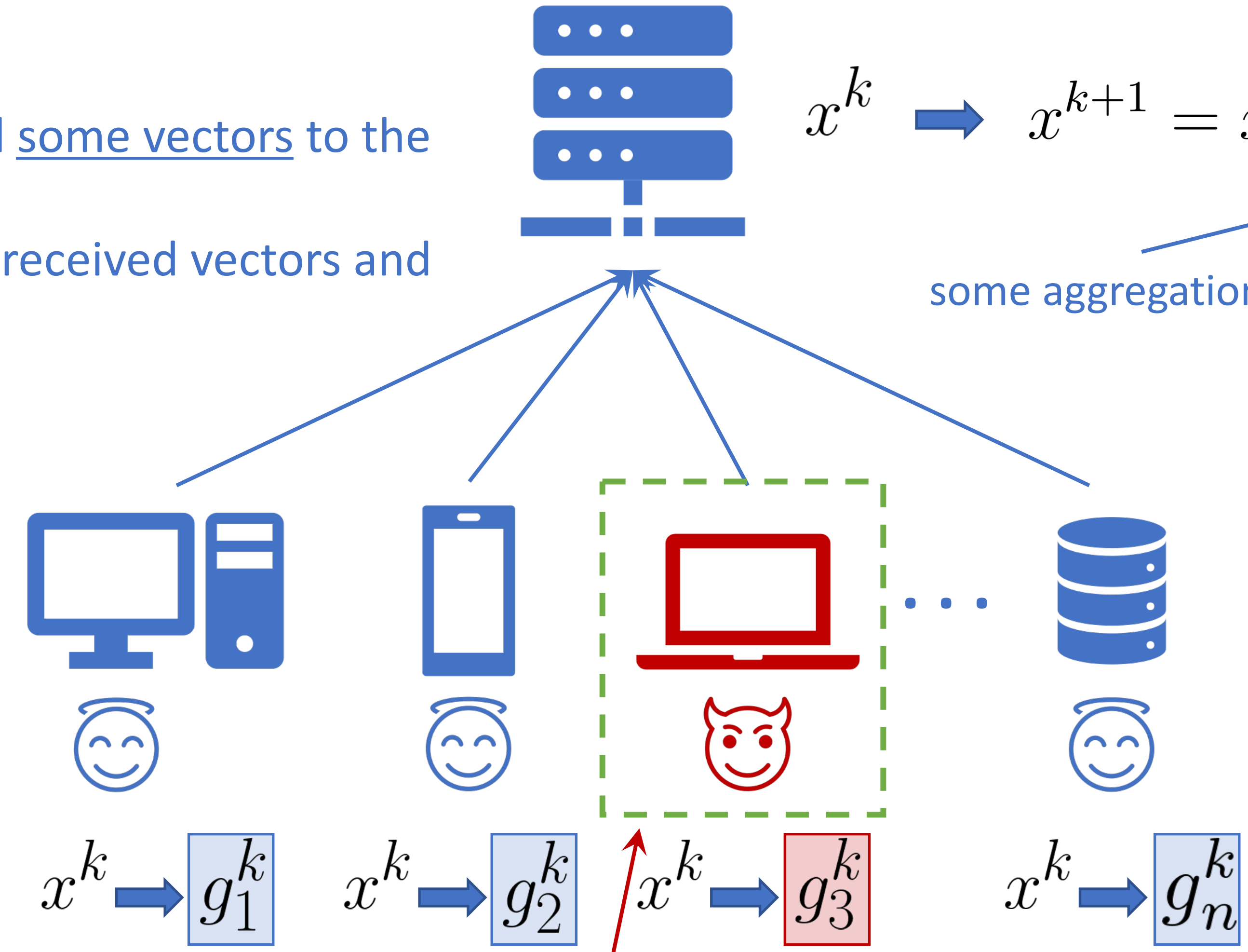
1. Server broadcasts x^k
2. Sampled workers send some vectors to the server
3. Server aggregates the received vectors and makes an SGD step

No robustness when honest workers are not in majority!

$$x^k \Rightarrow x^{k+1} = x^k - \gamma \cdot \mathrm{Agg}(\{g_i^k\}_{S_k})$$

some aggregation rule

sampled workers



If the majority of sampled workers are honest, the method works

If honest workers are not in majorty, the method can fail

The worst situation: all sampled workers are Byzantines

Ingredient 1: Clipping

Clipping Operator

💡 **Natural idea:** make all updates bounded via clipping

$$\mathrm{clip}(x, \lambda) = \begin{cases} \min\left\{1, \frac{\lambda}{\|x\|}\right\} x, & \text{if } x \neq 0 \\ 0, & \text{otherwise} \end{cases}$$

Useful properties:

Boundeness $\|\mathrm{clip}(x, \lambda)\| \leq \lambda$

Clipping Operator

💡 **Natural idea:** make all updates bounded via clipping

$$\mathrm{clip}(x,\lambda) = \begin{cases} \min\left\{1, \frac{\lambda}{\|x\|}\right\} x, & \text{if } x \neq 0 \\ 0, & \text{otherwise} \end{cases}$$

Useful properties:

Boundeness

$$\|\mathrm{clip}(x,\lambda)\| \leq \lambda$$

Controlled bias

$$\|\mathrm{clip}(x,\lambda) - x\| \leq \left(1 - \min\left\{1, \frac{\lambda}{\|x\|}\right\}\right)\|x\|$$

Clipping Operator

💡 **Natural idea:** make all updates bounded via clipping

$$\mathrm{clip}(x, \lambda) = \begin{cases} \min\left\{1, \frac{\lambda}{\|x\|}\right\} x, & \text{if } x \neq 0 \\ 0, & \text{otherwise} \end{cases}$$

Useful properties:

Boundeness $\quad \|\mathrm{clip}(x, \lambda)\| \leq \lambda$

Controlled bias $\quad \|\mathrm{clip}(x, \lambda) - x\| \leq \left(1 - \min\left\{1, \frac{\lambda}{\|x\|}\right\}\right) \|x\|$

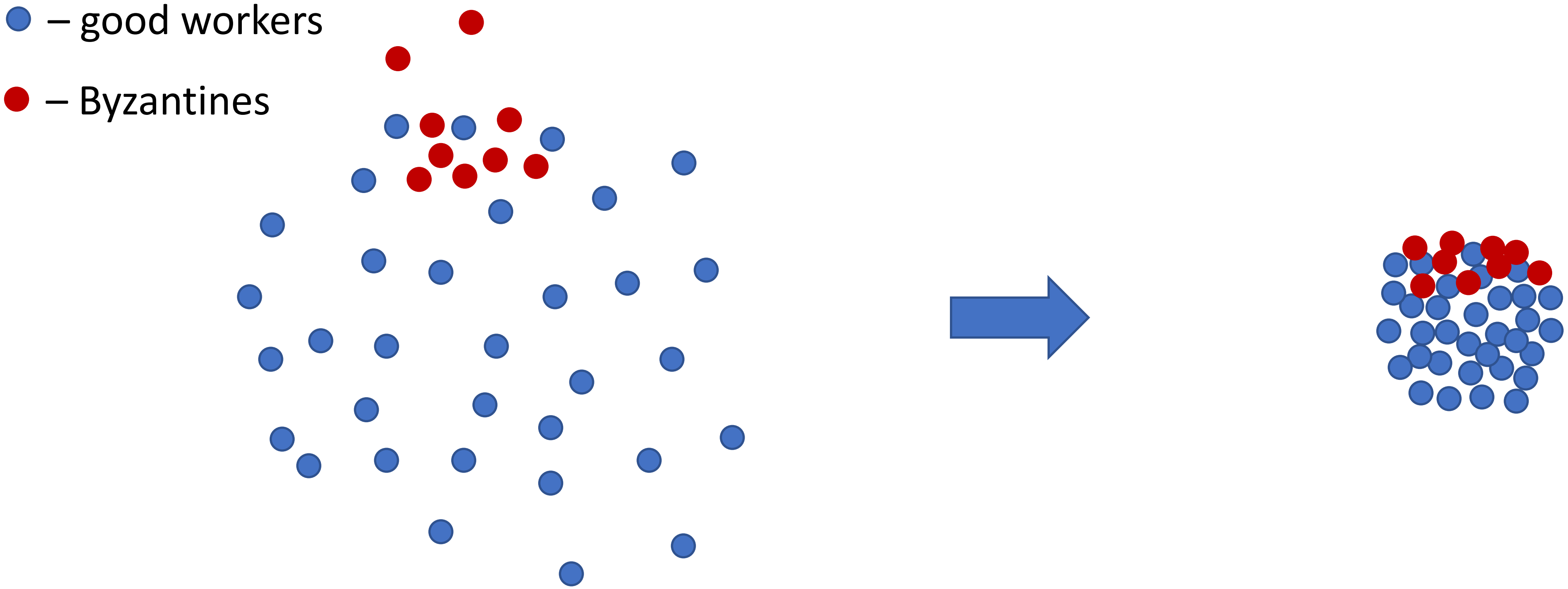
Direction is preserved

Ingredient 2: Variance Reduction

Why Variance Reduction?

Wu, Z., Ling, Q., Chen, T., & Giannakis, G. B. (2020). Federated variance-reduced stochastic gradient descent with robustness to byzantine attacks. IEEE Transactions on Signal Processing, 68, 4583-4596.

Natural idea: if the variance of good vectors gets smaller, it becomes progressively harder for Byzantines to shift the result of the aggregation from the true mean



- **Large variance** allows Byzantines to hide in noise and still create large bias
- Hard to detect outliers

- **Small variance** does not allow Byzantines to create large bias easily
- Easy to detect outliers

Byrd-SAGA: Byzantine-Robust SAGA

Wu, Z., Ling, Q., Chen, T., & Giannakis, G. B. (2020). Federated variance-reduced stochastic gradient descent with robustness to byzantine attacks. IEEE Transactions on Signal Processing, 68, 4583-4596.

Finite-sum optimization:

$$\min_{x \in \mathbb{R}^d} \left\{ f(x) := \frac{1}{m} \sum_{j=1}^{m} f_j(x) \right\}$$

# of samples in the dataset

loss on j-th sample

Byrd-SAGA: Byzantine-Robust SAGA

Wu, Z., Ling, Q., Chen, T., & Giannakis, G. B. (2020). Federated variance-reduced stochastic gradient descent with robustness to byzantine attacks. IEEE Transactions on Signal Processing, 68, 4583-4596.

Finite-sum optimization:

$$\min_{x \in \mathbb{R}^d} \left\{ f(x) := \frac{1}{m} \sum_{j=1}^{m} f_j(x) \right\}$$

# of samples in the dataset

loss on j-th sample

Byrd-SAGA:

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

- Good workers compute SAGA-estimators
- Server uses geometric median aggregator

$$\hat{g}^k = \texttt{RFA}(g_1^k, \ldots, g_n^k)$$

$$g_i^k = \begin{cases} \nabla f_{j_{i_k}}(x^k) - \nabla f_{j_{i_k}}(\phi_{i,j_{i_k}}^k) + \frac{1}{m} \sum\limits_{j=1}^{m} \nabla f_j(\phi_{i,j}^k), & \text{if } i \in \mathcal{G}, \\ *, & \text{if } i \in \mathcal{B} \end{cases}$$

$$\phi_{i,j}^{k+1} = \begin{cases} \phi_{i,j}^k, & \text{if } j \neq j_{i_k}, \\ x^k, & \text{if } j = j_{i_k} \end{cases} \quad \forall i \in \mathcal{G}$$

Complexity of Byrd-SAGA

Wu, Z., Ling, Q., Chen, T., & Giannakis, G. B. (2020). Federated variance-reduced stochastic gradient descent with robustness to byzantine attacks. IEEE Transactions on Signal Processing, 68, 4583-4596.

Assumptions:

- μ–strong convexity of f: $$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{\mu}{2}\|y - x\|^2 \quad \forall x, y \in \mathbb{R}^d$$
- L–smoothness of $f_1, \ldots, f_m$: $$\|\nabla f_j(y) - \nabla f_j(x)\| \leq L\|y - x\| \quad \forall x, y \in \mathbb{R}^d, j \in [m]$$

Complexity of Byrd-SAGA

Wu, Z., Ling, Q., Chen, T., & Giannakis, G. B. (2020). Federated variance-reduced stochastic gradient descent with robustness to byzantine attacks. IEEE Transactions on Signal Processing, 68, 4583-4596.

Assumptions:

- μ–strong convexity of f: $\quad f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{\mu}{2}\|y - x\|^2 \quad \forall x, y \in \mathbb{R}^d$
- L–smoothness of $f_1, \dots, f_m$: $\quad \|\nabla f_j(y) - \nabla f_j(x)\| \leq L\|y - x\| \quad \forall x, y \in \mathbb{R}^d, j \in [m]$

Theorem:

Let $\delta < {}^1/_2$ and the above assumptions hold. Then, there exists a choice of the stepsize γ such that the mini-batched version of Byrd-SAGA (with batchsize b) produces x^k satisfying $\mathbb{E}\left[\|x^k - x^*\|^2\right] \leq \varepsilon$ after

$$\mathcal{O}\left(\frac{m^2L^2}{b^2(1-2\delta)\mu^2}\log\frac{1}{\varepsilon}\right)$$ iterations

Reflecting on the Complexities

- Complexity of Byrd-SAGA ($b = 1,\ \delta > 0$): $\mathcal{O}\left(\frac{m^2 L^2}{(1-2\delta)\mu^2} \log \frac{1}{\varepsilon}\right)$
- Complexity of Byrd-SAGA ($b = 1,\ \delta = 0$): $\mathcal{O}\left(\frac{m^2 L^2}{\mu^2} \log \frac{1}{\varepsilon}\right)$
- Complexity of SAGA ($b = 1,\ \delta = 0$): $\mathcal{O}\left(\left(m + \frac{L}{\mu}\right) \log \frac{1}{\varepsilon}\right)$

Reflecting on the Complexities

- Complexity of Byrd-SAGA ($b = 1,\ \delta > 0$): $\mathcal{O}\left(\frac{m^2L^2}{(1-2\delta)\mu^2}\log\frac{1}{\varepsilon}\right)$
- Complexity of Byrd-SAGA ($b = 1,\ \delta = 0$): $\mathcal{O}\left(\frac{m^2L^2}{\mu^2}\log\frac{1}{\varepsilon}\right)$
- Complexity of SAGA ($b = 1,\ \delta = 0$): $\mathcal{O}\left(\left(m + \frac{L}{\mu}\right)\log\frac{1}{\varepsilon}\right)$

The reason for such a dramatic deterioration in the complexity of Byrd-SAGA in comparison to SAGA:

$$\mathbb{E}_k[\widehat{g}^k] \neq \nabla f(x^k)$$

Analysis of SAGA/SVRG-based methods is very sensitive to unbiasedness!

Biased VR: You Cannot “Break” What Is Already “Broken”!

SARAH/Geom-SARAH/PAGE (1 node case):

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

Nguyen, L. M., Liu, J., Scheinberg, K., & Takáč, M. (2017, July). SARAH: A novel method for machine learning problems using stochastic recursive gradient. In International Conference on Machine Learning (pp. 2613-2621). PMLR.

Horváth, S., Lei, L., Richtárik, P., & Jordan, M. I. (2022). Adaptivity of stochastic gradient methods for nonconvex optimization. SIAM Journal on Mathematics of Data Science, 4(2), 634-648.

Li, Z., Bao, H., Zhang, X., & Richtárik, P. (2021, July). PAGE: A simple and optimal probabilistic gradient estimator for nonconvex optimization. In International Conference on Machine Learning (pp. 6286-6295). PMLR.

Biased VR: You Cannot "Break" What Is Already "Broken"!

SARAH/Geom-SARAH/PAGE (1 node case):

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

$$g^k = \begin{cases} \nabla f(x^k), & \text{with prob. } p \\ g^{k-1} + \frac{1}{b} \sum_{j \in J_k} \left(\nabla f_j(x^k) - \nabla f_j(x^{k-1})\right), & \text{with prob. } 1-p \end{cases}$$

Nguyen, L. M., Liu, J., Scheinberg, K., & Takáč, M. (2017, July). SARAH: A novel method for machine learning problems using stochastic recursive gradient. In International Conference on Machine Learning (pp. 2613-2621). PMLR.

Horváth, S., Lei, L., Richtárik, P., & Jordan, M. I. (2022). Adaptivity of stochastic gradient methods for nonconvex optimization. SIAM Journal on Mathematics of Data Science, 4(2), 634-648.

Li, Z., Bao, H., Zhang, X., & Richtárik, P. (2021, July). PAGE: A simple and optimal probabilistic gradient estimator for nonconvex optimization. In International Conference on Machine Learning (pp. 6286-6295). PMLR.

Biased VR: You Cannot “Break” What Is Already “Broken”!

SARAH/Geom-SARAH/PAGE (1 node case):

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

$$g^k = \begin{cases} \nabla f(x^k), & \text{with prob. } p \\ g^{k-1} + \frac{1}{b} \sum_{j \in J_k} \left(\nabla f_j(x^k) - \nabla f_j(x^{k-1})\right), & \text{with prob. } 1-p \end{cases}$$

J_k– indices in the mini-batch, $|J_k| = b$

Nguyen, L. M., Liu, J., Scheinberg, K., & Takáč, M. (2017, July). SARAH: A novel method for machine learning problems using stochastic recursive gradient. In International Conference on Machine Learning (pp. 2613-2621). PMLR.

Horváth, S., Lei, L., Richtárik, P., & Jordan, M. I. (2022). Adaptivity of stochastic gradient methods for nonconvex optimization. SIAM Journal on Mathematics of Data Science, 4(2), 634-648.

Li, Z., Bao, H., Zhang, X., & Richtárik, P. (2021, July). PAGE: A simple and optimal probabilistic gradient estimator for nonconvex optimization. In International Conference on Machine Learning (pp. 6286-6295). PMLR.

Biased VR: You Cannot "Break" What Is Already "Broken"!

SARAH/Geom-SARAH/PAGE (1 node case):

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

$p \sim {}^{b}/_{m}$ – probability of computing the full gradient

$$g^k = \begin{cases} \nabla f(x^k), & \text{with prob. } p \\ g^{k-1} + \frac{1}{b} \sum_{j \in J_k} \left(\nabla f_j(x^k) - \nabla f_j(x^{k-1})\right), & \text{with prob. } 1 - p \end{cases}$$

J_k – indices in the mini-batch, $|J_k| = b$

Nguyen, L. M., Liu, J., Scheinberg, K., & Takáč, M. (2017, July). SARAH: A novel method for machine learning problems using stochastic recursive gradient. In International Conference on Machine Learning (pp. 2613-2621). PMLR.

Horváth, S., Lei, L., Richtárik, P., & Jordan, M. I. (2022). Adaptivity of stochastic gradient methods for nonconvex optimization. SIAM Journal on Mathematics of Data Science, 4(2), 634-648.

Li, Z., Bao, H., Zhang, X., & Richtárik, P. (2021, July). PAGE: A simple and optimal probabilistic gradient estimator for nonconvex optimization. In International Conference on Machine Learning (pp. 6286-6295). PMLR.

Biased VR: You Cannot "Break" What Is Already "Broken"!

SARAH/Geom-SARAH/PAGE (1 node case):

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

$p \sim {}^b/_m$ – probability of computing the full gradient

$$g^k = \begin{cases} \nabla f(x^k), & \text{with prob. } p \\ g^{k-1} + \frac{1}{b}\sum_{j \in J_k} \left(\nabla f_j(x^k) - \nabla f_j(x^{k-1})\right), & \text{with prob. } 1-p \end{cases}$$

J_k– indices in the mini-batch, $|J_k| = b$

$$\mathbb{E}_k[g^k] \neq \nabla f(x^k)$$

Estimator is biased from the beginning!

Nguyen, L. M., Liu, J., Scheinberg, K., & Takáč, M. (2017, July). SARAH: A novel method for machine learning problems using stochastic recursive gradient. In International Conference on Machine Learning (pp. 2613-2621). PMLR.

Horváth, S., Lei, L., Richtárik, P., & Jordan, M. I. (2022). Adaptivity of stochastic gradient methods for nonconvex optimization. SIAM Journal on Mathematics of Data Science, 4(2), 634-648.

Li, Z., Bao, H., Zhang, X., & Richtárik, P. (2021, July). PAGE: A simple and optimal probabilistic gradient estimator for nonconvex optimization. In International Conference on Machine Learning (pp. 6286-6295). PMLR.

Byz-PAGE

$$x^{k+1} = x^k - \gamma \widehat{g}^k \qquad \widehat{g}^k = \texttt{ARAggr}(g_1^k, \ldots, g_n^k)$$

E. Gorbunov, S. Horváth, P. Richtárik, G. Gidel. *Variance Reduction is an Antidote to Byzantines: Better Rates, Weaker Assumptions and Communication Compression as a Cherry on the Top* (ICLR 2023)

Byz-PAGE

(δ, c)–robust aggregator agnostic to the variance, e.g., Krum/RFA/CM ∘ Bucketing

$$x^{k+1} = x^k - \gamma \widehat{g}^k \qquad \widehat{g}^k = \boxed{\mathtt{ARAggr}(g_1^k, \ldots, g_n^k)}$$

E. Gorbunov, S. Horváth, P. Richtárik, G. Gidel. *Variance Reduction is an Antidote to Byzantines: Better Rates, Weaker Assumptions and Communication Compression as a Cherry on the Top* (ICLR 2023)

Byz-PAGE

(δ, c)–robust aggregator agnostic to the variance, e.g., Krum/RFA/CM ∘ Bucketing

$$x^{k+1} = x^k - \gamma \widehat{g}^k \qquad \widehat{g}^k = \texttt{ARAggr}(g_1^k, \ldots, g_n^k)$$

$$g_i^k = \begin{cases} \nabla f(x^k), & \text{with prob. } p \\ g^{k-1} + \frac{1}{b} \sum\limits_{j \in J_k} \left(\nabla f_j(x^k) - \nabla f_j(x^{k-1})\right), & \text{with prob. } 1-p \end{cases} \quad \forall i \in \mathcal{G}$$

Geom-SARAH/PAGE–estimator

The method achieves theoretical SOTA rates but uses full participation of clients

E. Gorbunov, S. Horváth, P. Richtárik, G. Gidel. *Variance Reduction is an Antidote to Byzantines: Better Rates, Weaker Assumptions and Communication Compression as a Cherry on the Top* (ICLR 2023)

New Method

New Method: Byz-PAGE-PP

💡 **Key idea:** clip gradient differences with $\lambda_k \sim \|x^k - x^{k-1}\|$

$$g_i^{k+1} = \begin{cases} \nabla f_i(x^{k+1}), & \text{with prob. } p \\ g^k + \text{clip}\left(\frac{1}{b}\sum_{j\in J_k}(\nabla f_j(x^k) - \nabla f_j(x^{k-1})), \lambda_k\right), & \text{with prob. } 1-p \end{cases} \quad \forall i \in \mathcal{G}$$

New Method: Byz-PAGE-PP

💡 **Key idea:** clip gradient differences with $\lambda_k \sim \|x^k - x^{k-1}\|$

$$g_i^{k+1} = \begin{cases} \nabla f_i(x^{k+1}), & \text{with prob. } p \\ g^k + \text{clip}\left(\frac{1}{b}\sum_{j\in J_k}(\nabla f_j(x^k) - \nabla f_j(x^{k-1})), \lambda_k\right), & \text{with prob. } 1-p \end{cases} \quad \forall i \in \mathcal{G}$$

$$g^{k+1} = \begin{cases} \texttt{ARAgg}\left(\{g_i^{k+1}\}_{i\in S_k}\right), & \text{with prob. } p, \\ g^k + \texttt{ARAgg}\left(\left\{\text{clip}\left(\frac{1}{b}\sum_{j\in J_k}(\nabla f_j(x^k) - \nabla f_j(x^{k-1})), \lambda_k\right)\right\}_{i\in S_k}\right), & \text{with prob. } 1-p \end{cases}$$

S_k - subset of sampled clients

New Method: Byz-PAGE-PP

💡 **Key idea:** clip gradient differences with $\lambda_k \sim \|x^k - x^{k-1}\|$

$$g_i^{k+1} = \begin{cases} \nabla f_i(x^{k+1}), & \text{with prob. } p \\ g^k + \text{clip}\left(\frac{1}{b}\sum_{j\in J_k}(\nabla f_j(x^k) - \nabla f_j(x^{k-1})), \lambda_k\right), & \text{with prob. } 1-p \end{cases} \quad \forall i \in \mathcal{G}$$

$$g^{k+1} = \begin{cases} \texttt{ARAgg}\left(\{g_i^{k+1}\}_{i\in S_k}\right), & \text{with prob. } p, \\ g^k + \texttt{ARAgg}\left(\left\{\text{clip}\left(\frac{1}{b}\sum_{j\in J_k}(\nabla f_j(x^k) - \nabla f_j(x^{k-1})), \lambda_k\right)\right\}_{i\in S_k}\right), & \text{with prob. } 1-p \end{cases}$$

S_k - subset of sampled clients

$$|S_k| = \begin{cases} \widehat{C}, & \text{with prob. } p, \\ C, & \text{with prob. } 1-p \end{cases}$$

$$\max\left\{1, \frac{\delta_{\text{real}} n}{\delta}\right\} \leq \widehat{C} \leq n$$

$$1 \leq C \leq n$$

New Method: Byz-PAGE-PP

💡 **Key idea:** clip gradient differences with $\lambda_k \sim \|x^k - x^{k-1}\|$

$$g_i^{k+1} = \begin{cases} \nabla f_i(x^{k+1}), & \text{with prob. } p \\ g^k + \text{clip}\left(\frac{1}{b}\sum_{j\in J_k}(\nabla f_j(x^k) - \nabla f_j(x^{k-1})), \lambda_k\right), & \text{with prob. } 1-p \end{cases} \quad \forall i \in \mathcal{G}$$

$$g^{k+1} = \begin{cases} \texttt{ARAgg}\left(\{g_i^{k+1}\}_{i\in S_k}\right), & \text{with prob. } p, \\ g^k + \texttt{ARAgg}\left(\left\{\text{clip}\left(\frac{1}{b}\sum_{j\in J_k}(\nabla f_j(x^k) - \nabla f_j(x^{k-1})), \lambda_k\right)\right\}_{i\in S_k}\right), & \text{with prob. } 1-p \end{cases}$$

S_k - subset of sampled clients

$$|S_k| = \begin{cases} \widehat{C}, & \text{with prob. } p, \\ C, & \text{with prob. } 1-p \end{cases}$$

$$\max\left\{1, \frac{\delta_{\text{real}} n}{\delta}\right\} \le \widehat{C} \le n$$

$$1 \le C \le n$$

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

Complexity of Byz-PAGE-PP (Simplified)

Assumptions:

- f is lower-bounded: $$f_* = \inf_{x \in \mathbb{R}^d} f(x) > -\infty$$
- L–smoothness of $f_1, \ldots, f_m$: $$\|\nabla f_j(y) - \nabla f_j(x)\| \leq L\|y - x\| \quad \forall x, y \in \mathbb{R}^d, j \in [m]$$

Complexity of Byz-PAGE-PP (Simplified)

Assumptions:

- f is lower-bounded: $f_* = \inf_{x \in \mathbb{R}^d} f(x) > -\infty$
- L–smoothness of $f_1, \ldots, f_m$: $\|\nabla f_j(y) - \nabla f_j(x)\| \leq L\|y - x\| \quad \forall x, y \in \mathbb{R}^d, j \in [m]$

Theorem 1:

Let the above assumptions hold and ARAggr be (δ, c)–robust aggregator. Then, there exists a choice of the stepsize γ such that Byz-PAGE produces $\hat{x}^k$ satisfying $\mathbb{E}\left[\left\|\nabla f(\hat{x}^k)\right\|^2\right] \leq \varepsilon^2$ after

Complexity of Byz-PAGE-PP (Simplified)

Assumptions:

- f is lower-bounded: $f_* = \inf_{x \in \mathbb{R}^d} f(x) > -\infty$
- L–smoothness of $f_1, \ldots, f_m$: $\|\nabla f_j(y) - \nabla f_j(x)\| \leq L\|y - x\| \quad \forall x, y \in \mathbb{R}^d, j \in [m]$

Theorem 1:
Let the above assumptions hold and ARAggr be (δ, c)–robust aggregator. Then, there exists a choice of the stepsize γ such that Byz-PAGE produces $\hat{x}^k$ satisfying $\mathbb{E}\left[\left\|\nabla f(\hat{x}^k)\right\|^2\right] \leq \varepsilon^2$ after

$$\mathcal{O}\left(\frac{\left(1 + \sqrt{\frac{p_G G \mathcal{P}_{\mathcal{G}_C^k}}{pC}\left(\frac{1}{C} + \frac{c\delta}{p}\right) + \frac{(1-p_G)(1+F_{\mathcal{A}}^2)}{p^2}}\right) L\left(f(x^0) - f_*\right)}{\varepsilon^2}\right)$$ iterations

Complexity of Byz-PAGE-PP (Simplified)

Assumptions:

- f is lower-bounded: $$f_* = \inf_{x \in \mathbb{R}^d} f(x) > -\infty$$
- L–smoothness of $f_1, \dots, f_m$: $$\|\nabla f_j(y) - \nabla f_j(x)\| \leq L\|y - x\| \quad \forall x, y \in \mathbb{R}^d, j \in [m]$$

Theorem 1:
Let the above assumptions hold and ARAggr be (δ, c)–robust aggregator. Then, there exists a choice of the stepsize γ such that Byz-PAGE produces $\hat{x}^k$ satisfying $\mathbb{E}\left[\left\|\nabla f(\hat{x}^k)\right\|^2\right] \leq \varepsilon^2$ after

$$\mathcal{O}\left(\frac{\left(1 + \sqrt{\frac{p_G G \mathcal{P}_{\mathcal{G}_C^k}}{pC}\left(\frac{1}{C} + \frac{c\delta}{p}\right) + \frac{(1-p_G)(1+F_{\mathcal{A}}^2)}{p^2}}\right) L\left(f(x^0) - f_*\right)}{\varepsilon^2}\right)$$ iterations

$$p_G = \mathrm{Prob}\{G_C^k \geq (1-\delta)C\}$$
$$\mathcal{P}_{\mathcal{G}_C^k} = \mathrm{Prob}\left\{i \in \mathcal{G}_C^k \mid G_C^k \geq (1-\delta)\,C\right\}$$

$F_{\mathcal{A}}$ - aggregation-dependent constant

Byz-PAGE vs Byz-PAGE-PP

Byz-PAGE-PP:

$$\mathcal{O}\left(\frac{\left(1+\sqrt{\frac{p_G G \mathcal{P}_{\mathcal{G}_C^k}}{pC}\left(\frac{1}{C}+\frac{c\delta}{p}\right)+\frac{(1-p_G)(1+F_{\mathcal{A}}^2)}{p^2}}\right)L\left(f(x^0)-f_*\right)}{\varepsilon^2}\right)$$

Byz-PAGE:

$$\mathcal{O}\left(\frac{\left(1+\sqrt{\frac{1}{p}\left(\frac{1}{n}+\frac{c\delta}{p}\right)}\right)L\left(f(x^0)-f_*\right)}{\varepsilon^2}\right)$$

Byz-PAGE vs Byz-PAGE-PP

Byz-PAGE-PP:

$$\mathcal{O}\left(\frac{\left(1+\sqrt{\frac{p_G G \mathcal{P}_{\mathcal{G}_C^k}}{pC}\left(\frac{1}{C}+\frac{c\delta}{p}\right)+\frac{(1-p_G)(1+F_{\mathcal{A}}^2)}{p^2}}\right)L\left(f(x^0)-f_*\right)}{\varepsilon^2}\right)$$

Byz-PAGE:

$$\mathcal{O}\left(\frac{\left(1+\sqrt{\frac{1}{p}\left(\frac{1}{n}+\frac{c\delta}{p}\right)}\right)L\left(f(x^0)-f_*\right)}{\varepsilon^2}\right)$$

Matching results when all clients participate

Byz-PAGE vs Byz-PAGE-PP

Byz-PAGE-PP: $$\mathcal{O}\left(\frac{\left(1+\sqrt{\frac{p_G G \mathcal{P}_{\mathcal{G}_C^k}}{pC}\left(\frac{1}{C}+\frac{c\delta}{p}\right)+\frac{(1-p_G)(1+F_{\mathcal{A}}^2)}{p^2}}\right)L\left(f(x^0)-f_*\right)}{\varepsilon^2}\right)$$

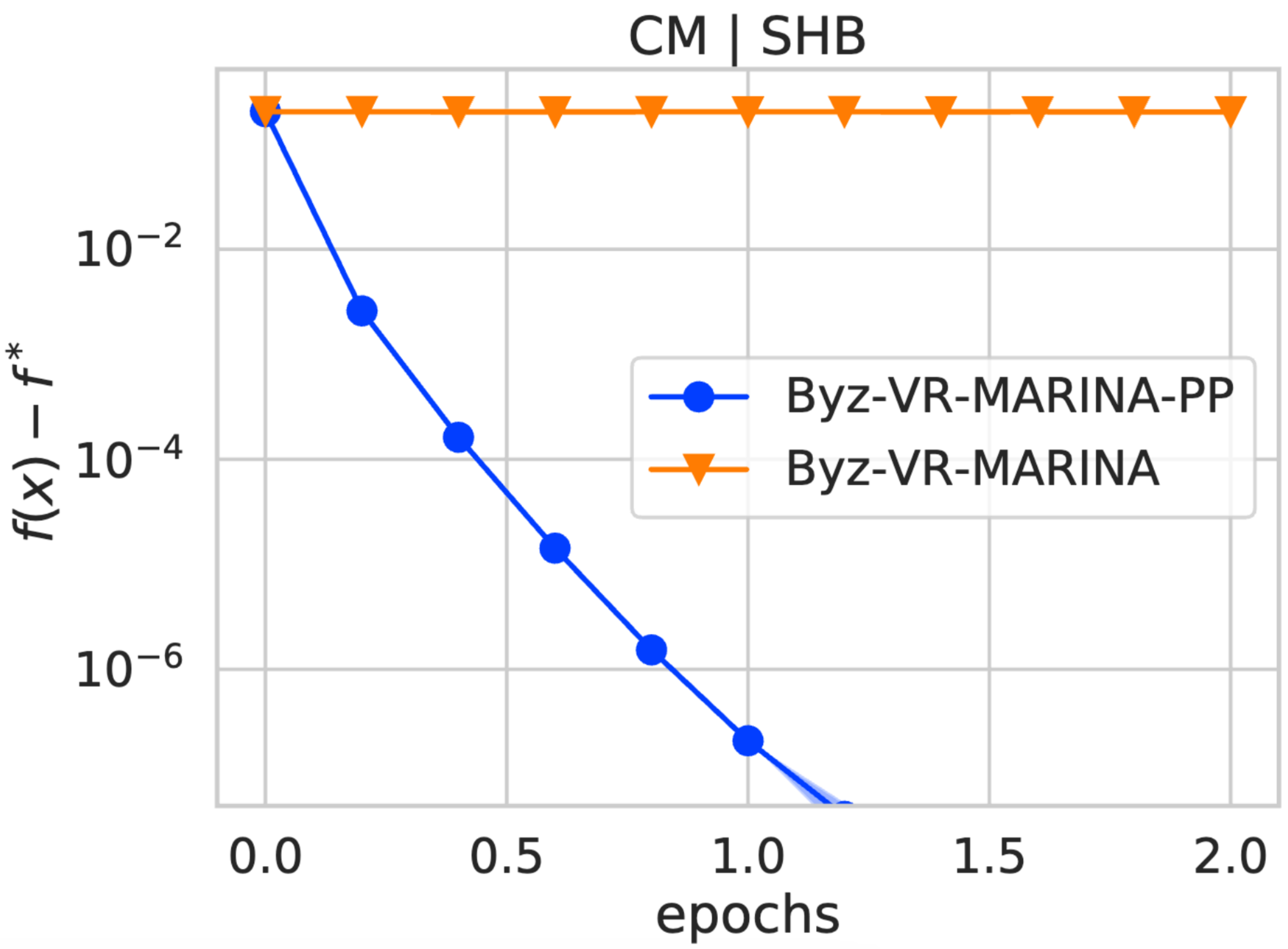
Byz-PAGE: $$\mathcal{O}\left(\frac{\left(1+\sqrt{\frac{1}{p}\left(\frac{1}{n}+\frac{c\delta}{p}\right)}\right)L\left(f(x^0)-f_*\right)}{\varepsilon^2}\right)$$

Matching results when all clients participate

When $p_G = 1$ (C is large enough) and $c\delta \geq p/C$, complexities are the same,
while Byz-PAGE-PP uses only $C \leq n$ workers at each step (on average) → provable benefits of PP!

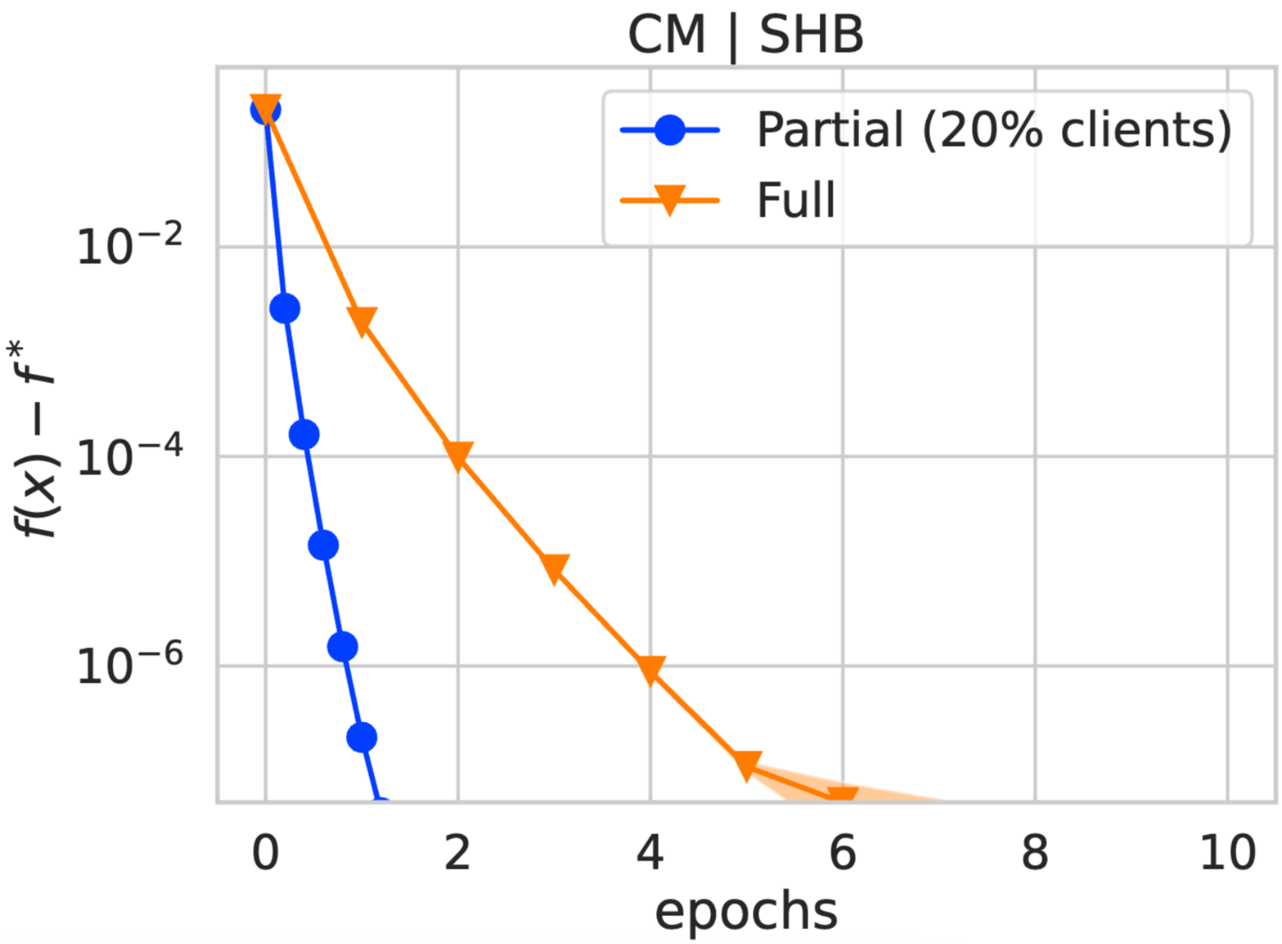
Numerical Results: Logistic Regression

- We tested the proposed method on the logistic regression tasks
- In this experiment, we have 15 good workers and 5 Byzantines
- Shift-back attack (SHB): when Byzantines form a majority they send $x^0 - x^k$
- Aggregation rule: coordinate-wise median (CM) with Bucketing
- Each round we sample 4 clients



Numerical Results: Benefits of PP

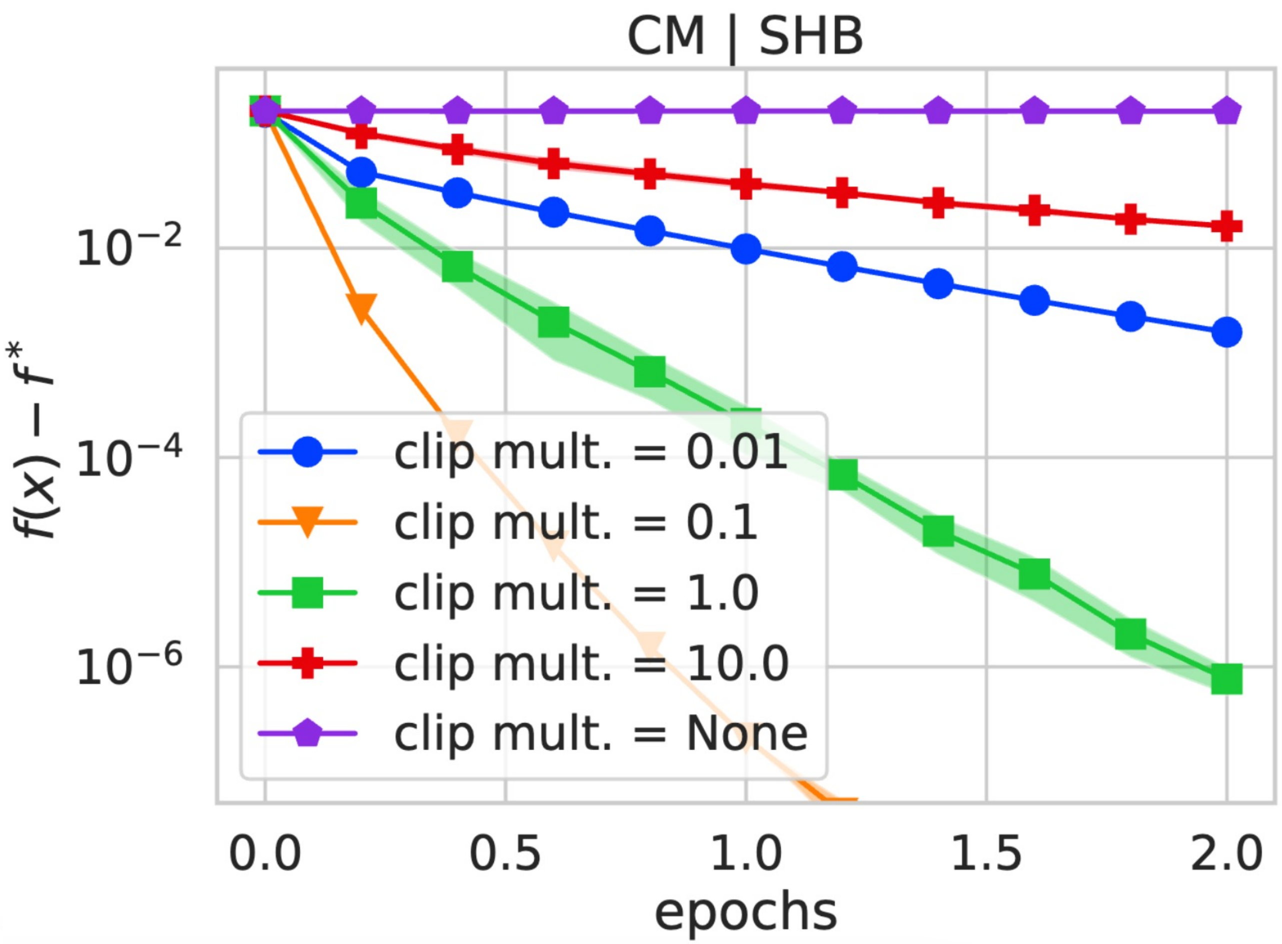
- The method benefits from partial participation



Numerical Results: Sensivity to Clipping Level

- We also tested our method with different clipping multipliers λ: $\lambda_k = \lambda \| x^k - x^{k-1} \|$

- The method converges for different clipping values, though the speed depends on λ



Heuristic Extension

🤔 How to adjust any Byzantine-robust method to the case of Partial Participation?

$$x^{k+1} = x^k - \gamma \cdot \texttt{Agg}(\{g_i^k\}_{i \in [n]})$$

Heuristic Extension

🤔 How to adjust any Byzantine-robust method to the case of Partial Participation?

$$x^{k+1} = x^k - \gamma \cdot \texttt{Agg}(\{g_i^k\}_{i\in[n]})$$

💡 Clip differences!

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

$$g^k = g^{k-1} + \texttt{Agg}\left(\left\{\text{clip}(g_i^k - g^{k-1}, \lambda_k)\right\}_{i\in S_k}\right)$$

Heuristic Extension

🤔 How to adjust any Byzantine-robust method to the case of Partial Participation?

$$x^{k+1} = x^k - \gamma \cdot \texttt{Agg}(\{g_i^k\}_{i\in[n]})$$

💡 Clip differences!

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

$$g^k = g^{k-1} + \texttt{Agg}\left(\{\text{clip}(g_i^k - g^{k-1}, \lambda_k)\}_{i\in S_k}\right)$$

Heuristic Extension

🤔 How to adjust any Byzantine-robust method to the case of Partial Participation?

$$x^{k+1} = x^k - \gamma \cdot \texttt{Agg}(\{g_i^k\}_{i \in [n]})$$

💡 Clip differences!

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

$$g^k = g^{k-1} + \texttt{Agg}\left(\{\mathrm{clip}(g_i^k - g^{k-1}, \lambda_k)\}_{i \in S_k}\right)$$

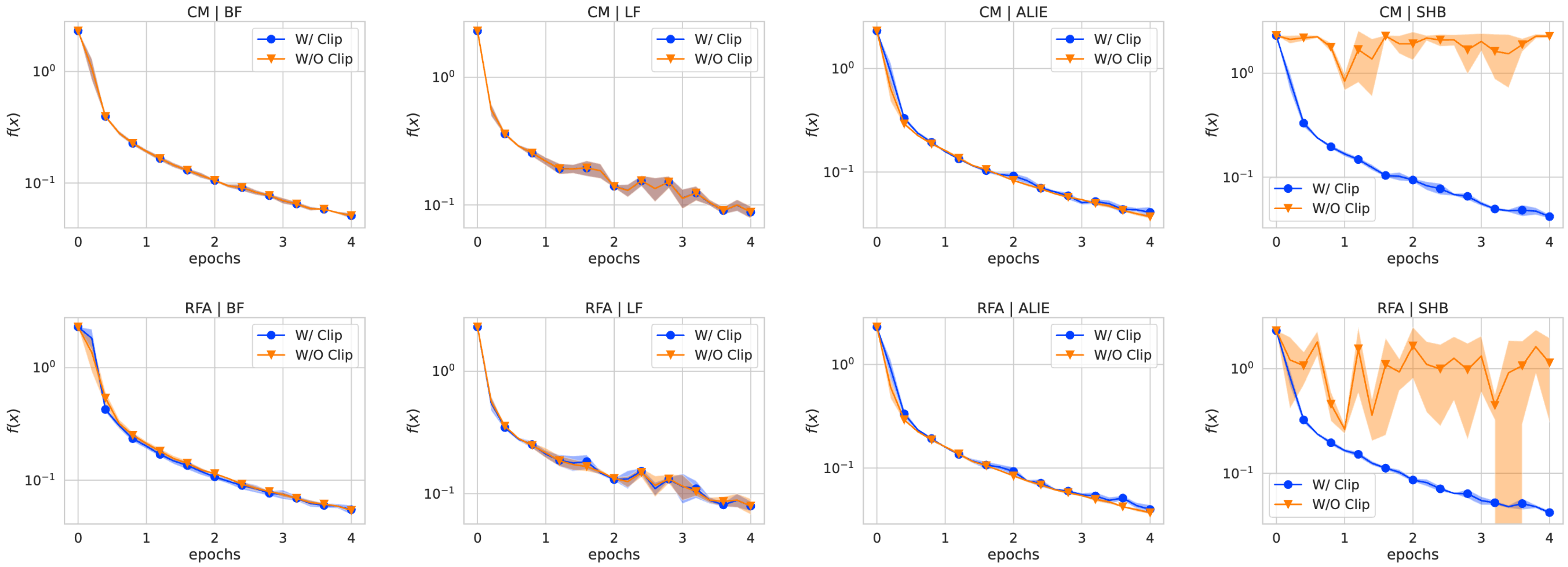
✓ We recommend to use $\lambda_k = \lambda \|x^k - x^{k-1}\|$ and tune λ in practice

Numerical Results: Neural Network Training

- We follow the setup from (Karimireddy et al., 2021) and train a certain NN on MNIST (LeCun and Cortes, 1998)
- In this experiment, we have 15 good workers and 5 Byzantines
- Attacks: A Little is Enough (ALIE) (Baruch et al., 2019), Bit Flipping (BF), Label Flipping (LF), Shift-Back (SHB)
- Aggregation rules: coordinate-wise median (CM), geometric median (RFA) with bucketing
- Each round we sample 4 clients
- Optimization method: Robust Momentum SGD (Karimireddy et al., 2021)

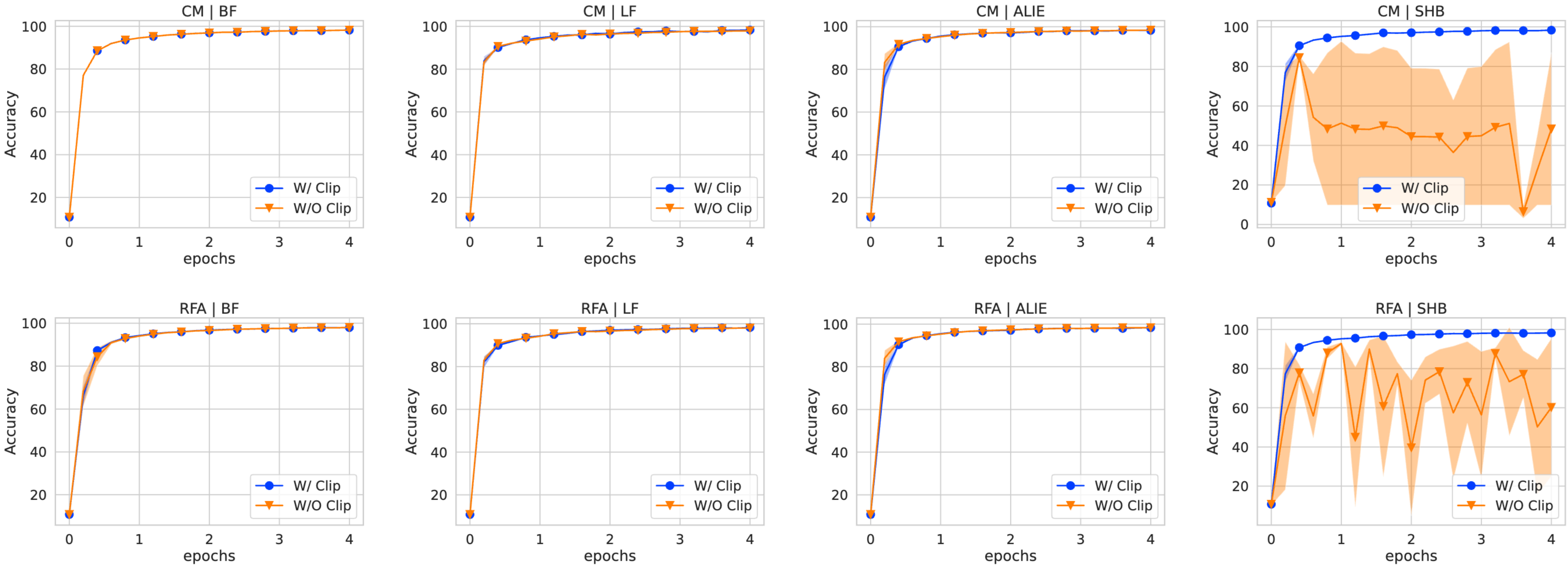
Karimireddy, S. P., He, L., Jaggi, M. *Learning from history for byzantine robust optimization* (ICML 2021)
LeCun, Y. and Cortes, C. *The MNIST database of handwritten digits* (http://yann.lecun.com/exdb/mnist/, 1998)
Baruch, G., Baruch, M., Goldberg, Y. *A Little is Enough: Circumventing defenses for distributed learning* (NeurIPS 2019)

Numerical Results: Neural Network Training



- Clipping does not spoil the convergence
- Clipping helps when Byzantine workers form majority (see SHB attack)

Numerical Results: Neural Network Training



- Clipping does not spoil the convergence
- Clipping helps when Byzantine workers form majority (see SHB attack)

Concluding Remarks

In the Paper We Also Have

- Analysis of the version with compression (Byz-VR-MARINA-PP)
- Analysis under bounded heterogeneity
- Non-uniform sampling of stochastic gradients
- Analysis taking into account data-similarity

Thank you!