Présentation MAP578

SignSGD with Majority Vote is Communication Efficient and Fault Tolerant

Jeremy Bernstein, Jiawei Zhao, Kamyar Azizzadenesheli, Anima Anandkumar

Jason Akoun, Sébastien Meyer

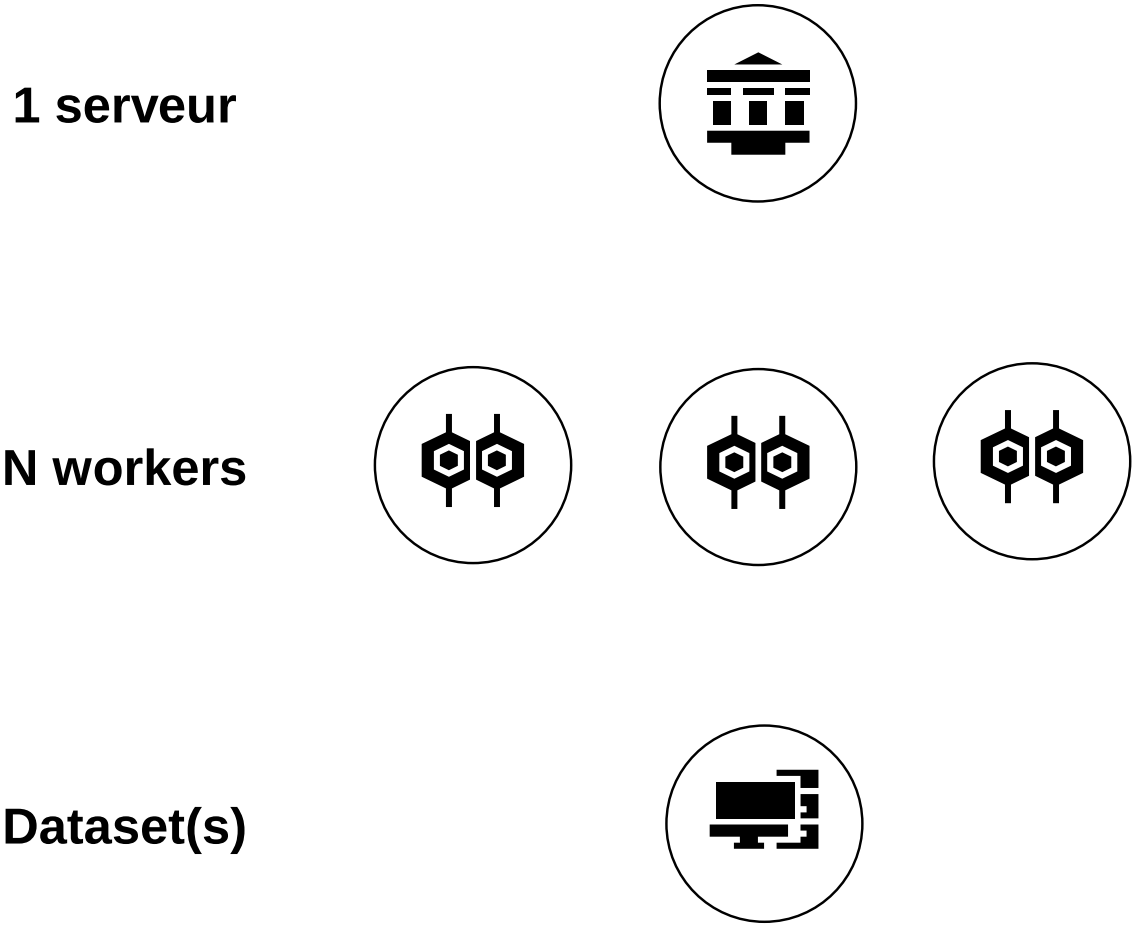
14 décembre 2021

Sommaire

1. **Mise en situation : Distributed Learning**
2. **Présentation des résultats du papier**
3. **Simulations**
4. **Quelle robustesse face aux byzantins ?**

1. Mise en situation : Distributed Learning

- **Situation classique distribuée**
  - Les **données** peuvent être partagées ou non entre les workers
  - Le serveur récupère les **gradients** des workers
  - Le serveur applique une **règle d'agrégation**
  - Fonctionnement **synchrone**



1. Mise en situation : Distributed Learning

- **Quatre points essentiels sont recherchés**

D1 fast algorithmic convergence	**D3** communication efficiency
D2 good generalisation performance	**D4** robustness to network faults

Conditions proposées par les auteurs de l'article

- **Les algorithmes usuels comme SGD ne remplissent pas toutes ces conditions**
  - SGD est largement utilisé car vérifie **D1** et **D2.** Néanmoins il ne vérifie **D3** que dans une moindre mesure et certainement pas **D4.**
  - Quel est l'objet le plus léger que vous puissiez envoyer en tant que worker ?

2. L'algorithme: signSGD with Majority Vote

Présentation de l'algorithme

Algorithm 1 Stochastic-Sign SGD with majority vote

Input: learning rate η, current hypothesis vector $w^{(t)}$, M workers each with an independent gradient $\boldsymbol{g}_m^{(t)}$, the 1-bit compressor $q(\cdot)$.
on server:
pull $q(\boldsymbol{g}_m^{(t)})$ **from** worker m.
push $\tilde{\boldsymbol{g}}^{(t)} = sign\left(\frac{1}{M}\sum_{m=1}^{M} q(\boldsymbol{g}_m^{(t)})\right)$ **to** all the workers.
on each worker:
update $w^{(t+1)} = w^{(t)} - \eta\tilde{\boldsymbol{g}}^{(t)}$.

Fig. *: Algorithme signSGD with Majority Vote*

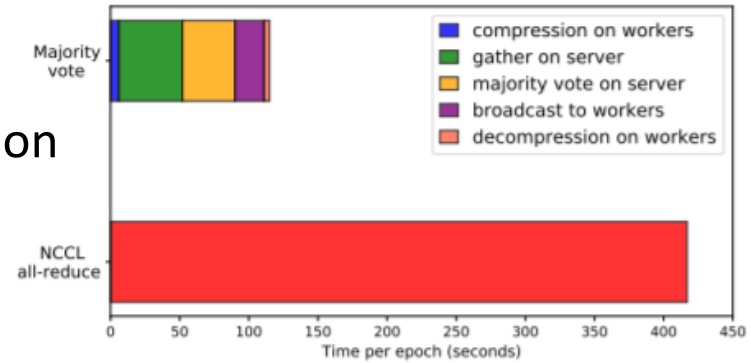
	Simple	Communication Efficace	Convergence Rapide	Tolérant aux fautes
signSGD Majority Vote	✓	✓	?	?

Caractéristiques de l'algorithme

2. L'algorithme: signSGD with Majority Vote

Performances

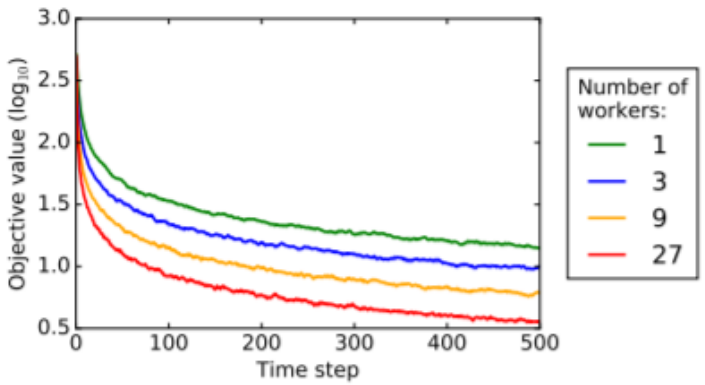
D1 Communication

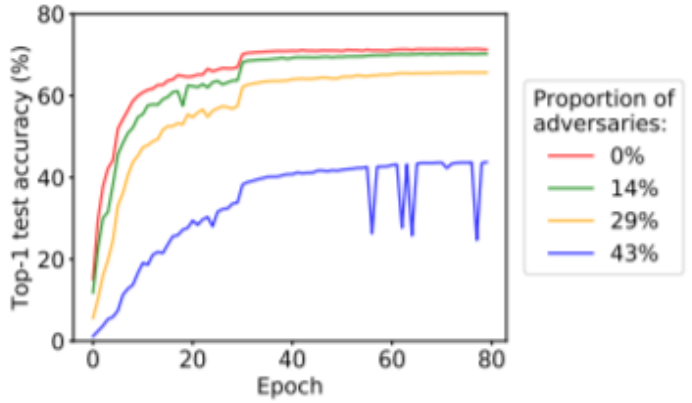


?

D2 Géneralisation

D3 Convergence





D4 Resistance aux fautes

Fig. *: Performances de signSGD with Majority Vote quant aux 4 conditions de départ*

2. L'algorithme: signSGD with Majority Vote

Vitesse de convergence théorique

Hypothèse 1. La fonction objectif est bornée.
Hypothèse 2. La fonction objectif est L-smooth.
Hypothèse 3. L'estimateur du gradient stochiastique est non biaisé et de variance borné.
Hypothèse 3. L'estimateur du gradient stochiastique suit une distribution unimodale autour de la moyenne.

Sur les hypothèses…

- Les hypothèses 1,2,3 sont **classiques**
- L'hypothèse 4 est plus **exigeante**
- L'hypothèse 4 est **vérifiée expérimentalement** sur CIFAR-10

Théorème 2. *Faire tourner signSGD durant K itérations sous les Hypothèses 1 à 4 (voir article). Pour chaque worker, fixer la learning rate $\eta = \sqrt{\frac{f_0 - f^*}{||L||_1 K}}$ et la taille du mini-batch $n = K$, alors si $\alpha < 1/2$ représente la proportion d'adversaires,*

$$\left[\frac{1}{K}\sum_{0}^{K-1}\mathbb{E}||g_k||_1\right]^2 \leq \frac{4}{\sqrt{N}}\left[\frac{1}{1-2\alpha}\cdot\frac{||\sigma||_1}{\sqrt{M}} + \sqrt{||L||_1(f_0 - f^*)}\right]^2$$

Fig. *: Majoration de la moyenne de la norme L1 du gradient*

Sur le Théorème 2…

- Convergence en $\frac{1}{\sqrt[4]{N}}$ pour signSGD vs $\frac{1}{\sqrt{N}}$ pour SGD *(N = nombre de calcul de gradients)*
- Borne décroissante en le **nombre de workers** et croissante en la **proportion d'adversaires**
- Résistant jusqu'à **50% d'adversaires blinds**

3. Simulations

Choix d'implémentation

- **Langage Python et bibliothèque PyTorch**
  - Accès à un support distribué qui permet de simuler des process
  - Intégration éventuelle sur des modèles complexes grâce à l'héritage de classes
  - Première version : https://github.com/sebastienmeyer2/signsgd-fault-tolerance

- **Estimation de l'effet d'adversaires sur l'algorithme**
  - Simulations de jeux de données en régressions linéaire et logistique
  - Simulations de process "blind" (voir Definition 1) ou Byzantins

Definition 1 (Blind multiplicative adversary)**.** *A blind multiplicative adversary may manipulate their stochastic gradient estimate $\tilde{g}_t$ at iteration t by element-wise multiplying $\tilde{g}_t$ with any vector v_t of their choice. The vector v_t must be chosen before observing $\tilde{g}_t$, so the adversary is 'blind'. Some interesting members of this class are:*

(i) adversaries that arbitrarily rescale their stochastic gradient estimate;

(ii) adversaries that randomise the sign of each coordinate of the stochastic gradient;

(iii) adversaries that invert their stochastic gradient estimate.

***Fig.** : Définition d'un adversaire blind dans l'article*

3. Simulations

Résultats (sans adversaires)

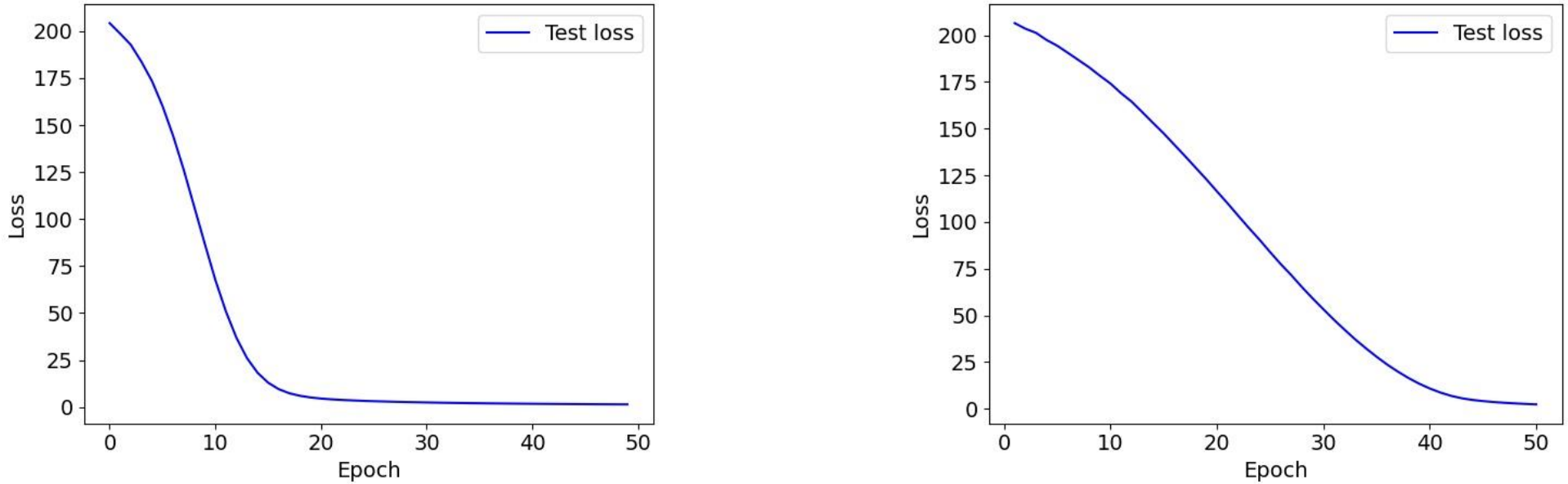


Fig. : *Évolution du loss pour un SGD distribué (à gauche) avec SignSGD et Signum (à droite).*

- Durée de convergence **plus longue** que pour le SGD classique ce qui est cohérent
- Convergence vers des **minima similaires** (ici dataset jouet)
- Si on ajoute des adversaires ?

3. Simulations

Résultats (avec adversaires "blind")

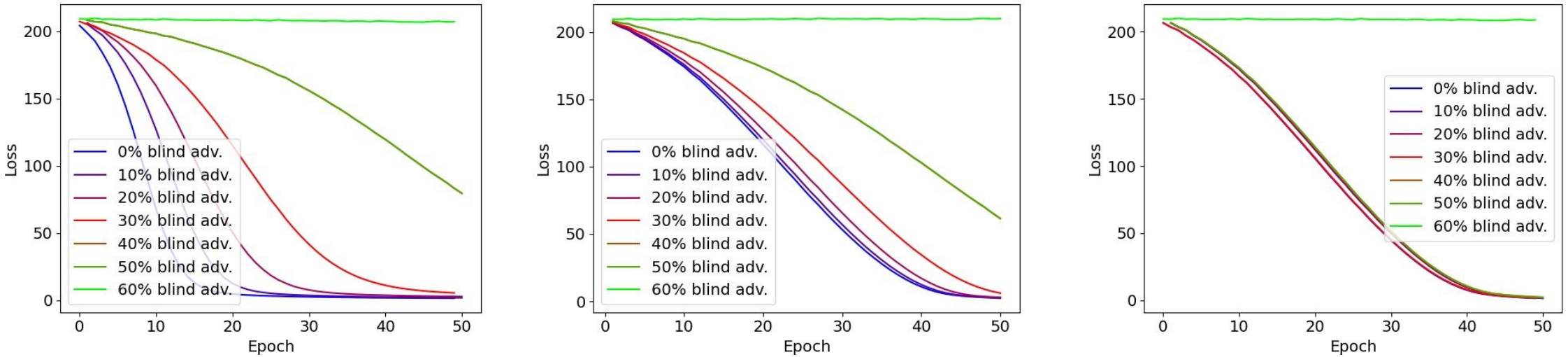


Fig. : *Évolution du loss pour un SGD distribué (à gauche) avec SignSGD (au milieu) et Signum (à droite) en présence d'adversaires qui inversent systématique leur gradient.*

- Comme annoncé par le Théorème 2 on converge jusqu'à **50% d'adversaires blinds**
- Le vote et la réduction de l'information sur 1 bit rend l'algorithme signSGD **très robuste**
- Si on ajoute des **adversaires Byzantins**?

4. Quelle robustesse face aux byzantins ?

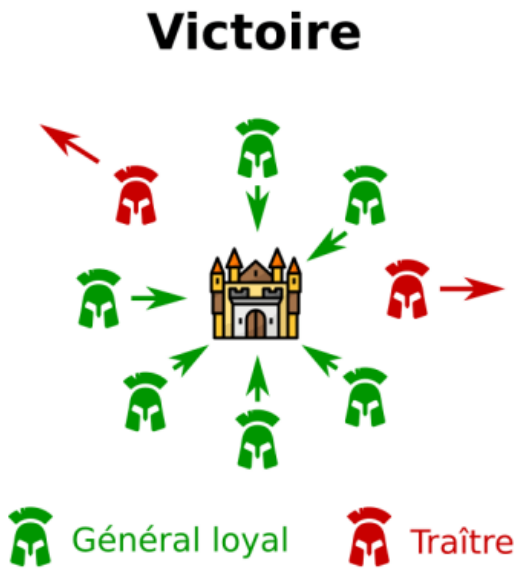
Notre démarche

Les adversaires blinds…

- SignSGD est très résistant jusqu'à 50% tout type de fautes blinds de par sa structure
- Résultat théorique prouvant la convergence
- Intuitivement même si les adversaires communiquent ils ne peuvent pas trop modifier le résultat du serveur car vote + 1 bit

Vers les Byzantins…

- Peut-on étendre le théorème de convergence au cas Byzantins ?
- Peut-on élaborer la « pire » stratégie Byzantine capable de casser l'algorithme ?
- Peut-on tester la convergence de signSGD face à cette stratégie ?



4. Quelle robustesse face aux byzantins ?

Notre théorème : vitesse de convergence face aux Byzantins

Théorème 2. bis *Faire tourner signSGD durant K itérations sous les Hypothèses 1 à 3 (voir article). Pour chaque worker, fixer la learning rate $\eta = \sqrt{\frac{f_0 - f^*}{\|L\|_1 K}}$ et la taille du mini-batch $n = K$, alors si $\alpha < 1 - \frac{1}{2p}$ représente la proportion d'adversaires Byzantins,*

$$\left[\frac{1}{K}\sum_{0}^{K-1}\mathbb{E}\|g_k\|_1\right]^2 \leq \frac{4}{\sqrt{N}}\left[\frac{1}{2\sqrt{2}}\frac{1}{p(1-\alpha)-\frac{1}{2}}\cdot\frac{\|\sigma\|_1}{\sqrt{M}} + \sqrt{\|L\|_1(f_0 - f^*)}\right]^2$$

avec $N = K^2$ et $p = \mathbb{P}\left(sgn(\widetilde{g_t}) = sgn(g_t)\right)$

Fig. *: Majoration de la moyenne de la norme L1 des gradients dans le cas Byzantins*

	Notre théorème	Stochastic-SignSGD	Election coding
Proportion maximale de Byzantins	$\alpha < 1 - \frac{1}{2p}$	$\bar{p}_i^{(t)} \leq \frac{M - k_i}{2M}$.	$1 - \alpha > \frac{1}{2u^*_{min}}$

Bornes du nombre de byzantins admissibles selon différents modèles

Sur le Théorème 2 bis…

- Similaire au cas blind avec une **constante modifiée**
- On a **relâché l'hypothèse 4**
- **Nouvelle condition** sur la proportion d'adversaires
- Si p = 1 on a la condition $\alpha < 0.5$ ce qui est **cohérent**
- Si $p < 0.5$ on a la condition $\alpha < 0$
- Notre borne est **similaire à celle dans la littérature**

4. Quelle robustesse face aux byzantins ?

Elaboration de notre stratégie

- **Quelle stratégie pour des adversaires byzantins ?**
  - Les byzantins, au nombre de **f** et **omniscients**, sont en mesure de connaître les valeurs envoyées par tous les autres workers, dont **n-f** sont honnêtes
  - Ils peuvent estimer à l'avance la règle d'agrégation

- **Des stratégies différentes en fonction de la règle d'agrégation**
  - Un seul byzantin suffit à casser SGD : envoyer l'opposé de la somme des gradients des autres workers
  - Dans le cas de SignSGD, il est impossible de faire exploser la valeur envoyée

- **Notre idée : les byzantins peuvent essayer de "tuer" le gradient**
  - La règle d'agrégation sur les workers honnêtes se situe entre -(n-f) et n-f
  - S'ils le peuvent, les byzantins peuvent l'amener à zéro puis osciller autour, sinon ils s'y opposent totalement

4. Quelle robustesse face aux byzantins ?

Simulation de notre stratégie

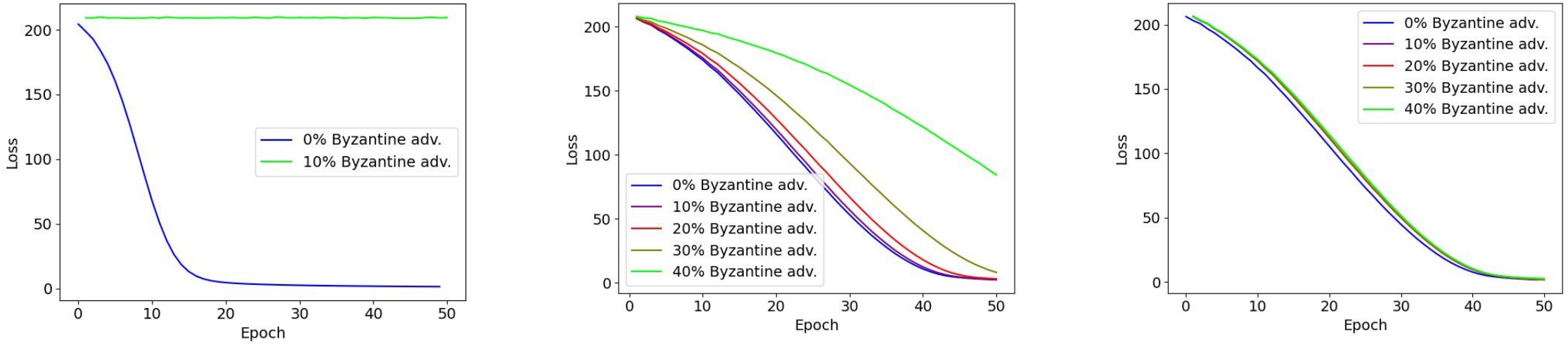


Fig. : *Évolution du loss pour un SGD distribué (à gauche) avec SignSGD (au milieu) et Signum (à droite) en présence d'adversaires byzantins.*

- Les résultats obtenus sont **similaires au cas des adversaires « blind »**
- **Cohérent avec notre borne** sur la proportion Byzantine
- Existe t-il une autre stratégie avec plus de dégâts ?

4. Quelle robustesse face aux byzantins ?

Pour aller plus loin

- ***Stochastic-Sign SGD for Federated Learning with Theoretical Guarantees***
  - Relâchement de l'hypothèse 4 prouvée que de manière **empirique** sur CIFAR-10
  - Preuve de convergence avec la même borne Byzantine que la nôtre (démonstration différente de la notre)

- **Election Coding for Distributed Learning: Protecting SignSGD against Byzantine Attacks**
  - Proposé par *Jy-yong Sohn, Dong-Jun Han, Beongjun Choi, Jaekyun Moon*
  - **Idée** : construire une couche intermédiaire de servers qui agrègent l'information et la retransmettent au server principal
  - Le théorème 2 prouve la convergence sous une proportion de byzantins plus fine que la notre (mais similaire asymptotiquement)

$$1-\alpha > \frac{(\sqrt{\log(\Delta)/n}+\sqrt{\log(\Delta)/n+4u^{\star}_{min}})^2}{8(u^{\star}_{min})^2}$$

***Fig.**: Proportion admissible de byzantins*

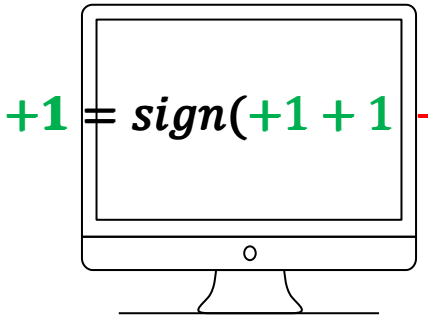
Theorem 2 (Convergence of the Bernoulli-coded SIGNSGD-MV)**.** *Suppose that Assumptions 1, 2, 3, 4 hold and the portion of Byzantine-free nodes satisfies* (3) *for* $\Delta > 2$*. Apply the random Bernoulli codes with connection probability* $p = \Theta(\sqrt{\log(n)/n})$ *on SignSGD-MV, and run SignSGD-MV for* T *steps with an initial model* $\boldsymbol{w}_0$*. Define the learning rate as* $\gamma(T) = \sqrt{\frac{f(\boldsymbol{w}_0)-f^*}{\|\boldsymbol{L}\|_1 T}}$*. Then, the suggested scheme converges as* T *increases in the sense*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\|g(\boldsymbol{w}_t)\|_1\right] \leq \frac{3\|\boldsymbol{L}\|_1}{2(1-2/\Delta)}\gamma(T) \to 0 \quad \text{as } T\to\infty.$$

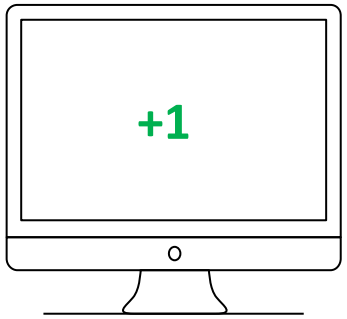
Annexes

M le nombre de workers
α la proportion de byzantins.
Z_t le nombre de bits corrects reçus par le serveur à l'itération t.
Z_t^g le nombre de bits corrects reçus par le serveur à l'itération t par les workers honnêtes.

$$
\begin{aligned}
P\left(Z_t \leq \frac{M}{2}\right) &\leq P\left(Z_t^g \leq \frac{M}{2}\right) && \textit{Pire des cas} \\
&= P\left(E\left(Z_t^g\right) - Z_t^g \geq E\left(Z_t^g\right) - \frac{M}{2}\right) && E\left(Z_t^g\right) > \frac{M}{2} \\
&\leq \frac{1}{1 + \frac{\left(E(Z_t^g) - \frac{M}{2}\right)^2}{Var(Z_t^g)}} && \textit{Inégalité de Cantelli} \\
&\leq \frac{1}{2} \frac{\sqrt{Var(Z_t^g)}}{E\left(Z_t^g\right) - \frac{M}{2}} && car\ 1 + x^2 \geq 2x \\
&= \frac{1}{2} \frac{\sqrt{p(1-p)(1-\alpha)}}{p(1-\alpha) - \frac{1}{2}} \frac{1}{\sqrt{M}} \\
&\leq \frac{1}{2} \frac{\sqrt{1-p}}{p(1-\alpha) - \frac{1}{2}} \frac{1}{\sqrt{M}} && p(1-\alpha) \leq 1
\end{aligned}
$$

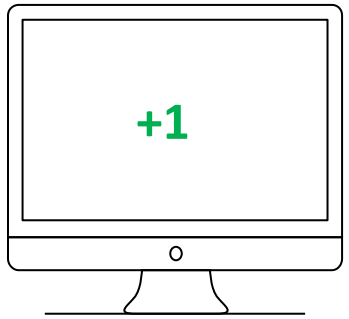


Serveur



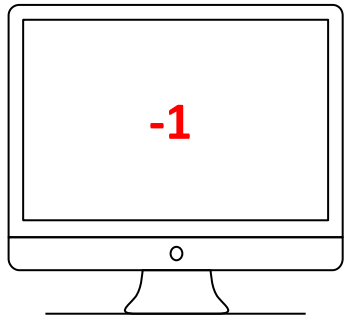
Worker 1

$data$



Worker 2

$data$



Worker 3

$data$