# Convergence guarantees for gradient descent methods in optimization for non-convex function approximation (distributed neural network training).

Hoang Trung Hieu

May, 2021.

## 1 Non-IID data analysis

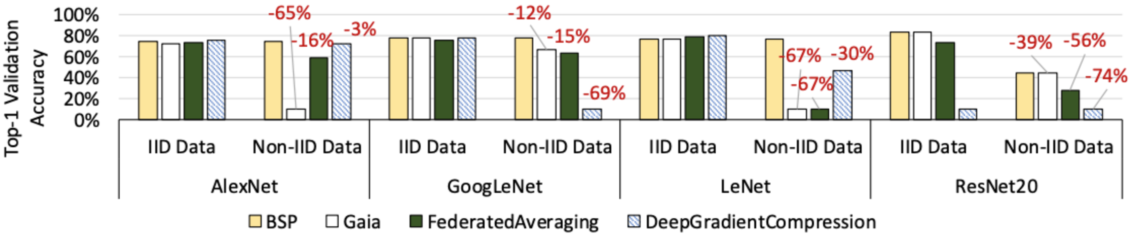
We formally introduce Federated Learning in the context of a $C-$class classification problem, which is defined over a compact feature space $X$ and a label space $Y = [C]$, where $[C] = \{1, \cdots, C.\}$ Let $(x, y)$ denote a particular labeled sample. Let $f : X \to S$ denote the prediction function, where $S = \{z| \sum_{i=1}^{C} z_i = 1, z_i \geq 0 \ \forall i \in [C]\}$. That is, the vector valued function $f$ yields a probability vector $z$ for each sample $x$, where $f_i$ predicts the probability that the sample belongs to the $i-$th class.
Let the vector $w$ denote model weights. For classification, the commonly used training loss is cross entropy, defined as

$$L(w) = \mathbb{E}_{x,y\sim p}[\sum_{i=1}^{C} \mathbb{1}_{y=i} \log f_i(x, w)] = \sum_{i=1}^{C} p(y = i)\mathbb{E}_{x|y=i}[\log f_i(x, w)]$$

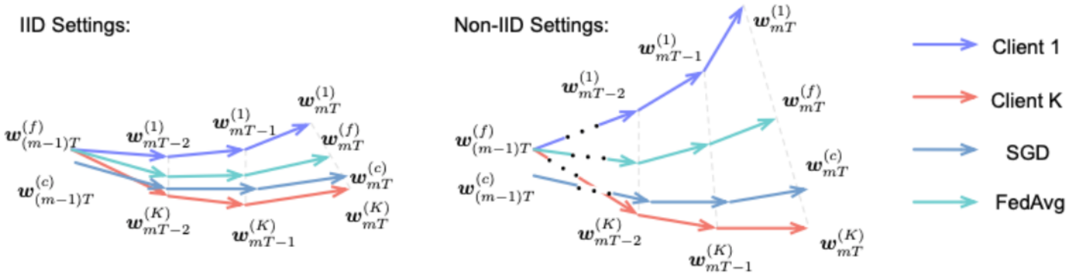
The learning problem is to solve the following optimization problem:

$$\min_{w} \sum_{i=1}^{C} p(y = i)\mathbb{E}_{x|y=i}[\log f_i(x, w)]$$

**Why Non-IID data is a trouble?** Some recent works ([3]) show that mose decentralized learning algorithms suffer from major model quality loss (or even divergence) when run on non-IID data partitions. However, it is interesting to note that BSP([5]) is robust to Non-IID data.



Figure[1]. Top-1 validation accuracy for IMAGE CLASSIFICATION over the CIFAR-10 dataset [3]
It is shown that the accuracy may be affected by the exact data distribution, i.e. the skewness of data distribution. More specifically, the skewness can be roughly interpreted as the distance between the data distribution on each client and the population distribution. In addition, such distance can be evaluated with the earth mover's distance(EMD) between distributions. Based on experiment on real-world dataset, the test accuracy falls sharply with respect to EMD beyond certain threshold.



Figure[2]. Ilustration of the weight divergence for federated learning with-IID and non-IID data [7]

## 2 Non Convex optimization on gradient-based methods

In practical, the objective functions arising in the training of neural networks which are expected to be non-convex and it shows rich sets of local minima (we denote by $x^*$) and saddle points. It is also has rich of critical points ($0 \in \nabla f(x)$) and first-order stationary points ($\|\nabla f(x)\| = 0$). We will see different gradient-based algorithms with convergence guarantee conditions when the objective are non-convex and even non-smooth in some cases.

### 2.1 Using local smoothness and maximal nondegeneracy

In general, we optimize the function $f : \mathbb{R} \to \mathbb{R}$ with respect to parameter $\theta$

$$f(\theta) = \mathbb{E}[F(\theta, X)]$$

where $F : \mathbb{R}^d \times \mathbb{R}^m \to \mathbb{R}$, $\theta$ can be considered as a parameter vector, while $d, m$ are the dimension of the parameter and training examples (resp.). In the above problem, $F = \sum_{i=1}^{C} \mathbb{1}_{y=i} \log f_i(x, w)$. We denote $X_{k,m,n} : \Omega \to S$ be iid random variables which satisfy $\mathbb{E}\left[F(\theta, X_{1,1,1})\right]^2 \leq \infty$ where $\Omega$ be a probability space and $S$ be a measurable space . And $f$ be the function such that $f(\theta) = \mathbb{E}\left[F(\theta, X_{1,1,1})\right]$ Let us denote $\mathcal{M} = \{\theta = \arg\inf f\}$ the set of minima of $f$.

**Assumption.**

- *$\mathcal{M}$ is locally smooth ( there exists an open set $U \subseteq \mathbb{R}^d$such that $M \cap U$ is a non-empty $\Psi-$dimensional $C^1-$submanifold of $\mathbb{R}^d$ ).*
- *$f$ is locally three times continous differentiable on the local set of $\mathcal{M}$. And the Hessian matrix of $f$ is maximally non degenerate. (rank Hess $(f)(\theta) = d - \Psi = codim(M \cap U)$)*

**Algorithm.** Initialization: The initial data was sample from the bounded open set $A \subseteq \mathbb{R}^d$ that contains at least one element in local set of $\mathcal{M}$. Denote by $\theta_0^{k,M,r} : \Omega \to \mathbb{R}^d$ indicates $k$-th sample in the sampling set of size $K$, mini-batch size $M$, and parameter $r > 0$ involving to learning parameter. The initial data is uniformly distributed on $A$. i.i.d and independent from $X_{k,m,n}$.
Weights update: We compute independent solutions to SGD in the way

$$\theta_n^{k,M,r} = \theta_{n-1}^{k,M,r} - \frac{r}{n^\rho M}\left[\sum_{i=1}^{m} \nabla_\theta F(\theta_{n-1}^{k,M,r}, X_{k,n,m})\right]$$

Mini-batch approximation: ([2]) $F^{K,\mathfrak{M},n} : \mathbb{R}^d \times \Omega \to \mathbb{R}$ is approximated as

$$F^{K,\mathfrak{M},n}(\theta, w) = \frac{1}{\mathfrak{M}} \sum_{i=1}^{\mathfrak{M}} F\left(\theta, X_{1,n+1,m}(w)\right)$$

After that, we identify the value that minimizes $F^{K,\mathfrak{M},n}$ in the sense that we compute a random variable $\theta_n^{K,M,\mathfrak{M},r} : \Omega \to \mathbb{R}^d$ which satisfies that

$$\sum_{m=1}^{\mathfrak{M}} F\left(\theta_n^{K,M,\mathfrak{M},r}, X_{1,n+1,m}\right) = \min_{k\in[K]} \left[\sum_{m=1}^{\mathfrak{M}} F\left(\theta_n^{k,M,r}, X_{1,n+1,m}\right)\right]$$

**Theorem 1.** *[1]After running the above algorithm with $p \in (2/3;1)$. There exist $\tau, c > 0$ , $\kappa \in (0,1)$ such that for every $n, k, M, \mathfrak{M}, r \in (0,\tau)$ , $\epsilon \in (0,1)$ we get*

$$\begin{aligned} &\mathbb{P}\big(\ \textit{Distance between} f(\theta_n^{k,M,\mathfrak{M},r}) \textit{ and minima bigger than } \epsilon\big) \\ &= \mathbb{P}\left(f(\theta_n^{k,M,\mathfrak{M},r}) - \inf_\theta f(\theta) \geq \epsilon\right) \leq \frac{cK}{\epsilon^2 \mathfrak{M}} + \left[\kappa + c\left(\frac{1}{\epsilon^2 n^\rho} + \frac{n^{1-p}}{\sqrt{M}}\right)\right]^K \end{aligned}$$

The conditions we used in this algorithm are satisfied by a four-parameter affine-linear network with a linear activation function and the case of a two-parameter network with the ReLU activation function. [1]

## 2.2 Using decomposition of objective function

The next problem is finding the optimum solution on the convex set $C$

$$\min_{x\in C} f(x) = \min_{x\in C}\{g(x) - h(x)\}$$

**Assumption.**

- *$f$ is bounded below $C$*
- *$h$ is continuous and convex*
- *$g$ is continuous differentiable and $M_g$ smooth*

**Algorithm 1:** Subgradient-type method.

```
Initialization. Choose $x_0 \in \int(C)$ , level set of $x_0$ is in $C$.
for each round $k = 1, 2, \cdots, T$ do
    Update: $x_{k+1} = x_k - \alpha(\nabla g(x_k) - u_k)$ where $u_k$ is chosen randomly from $\partial h(x_k)$ and learning
      rate $\alpha \in [0, 1/M_g]$.
end
```

**Theorem 2.** *[4] $f(x_k)$ is strictly decreasing and converges. The limit point of $x_k$ is also a critical point of $f$. Moreover, for all $k$,*

$$\operatorname{Avg}\left(\|\nabla f(x_k)\|_2^2\right) \leq \frac{2(f(x_0) - f(x*))}{(k+1)}$$

We now turn to a more general class of optimization problems of the form

$$\min_{x\in\mathbb{R}^d} f(x) = \min_{x\in\mathbb{R}^d} g(x) - h(x) + \varphi(x)$$

**Assumption.**

- *$f$ is bounded below $\mathbb{R}^d$.*
- *$h$ is continuous and convex.*
- *$g$ is continuous differentiable and $M_g-$smooth.*
- *$\varphi$ is proper, convex and lower semi-continuous.*

**Algorithm 2:** Proximal-type algorithm.

```
Initialization. Choose $x_0 \in dom(f)$ and learning rate $\alpha \in [0, 1/M_g]$.
for each round $k = 1, 2, \cdots, T$ do
    Update: $x_{k+1} = prox_{1/\alpha\varphi}(x_k - \alpha(\nabla g(x_k) - u_k))$ where $u_k$ is chosen randomly from $\partial h(x_k)$
end
```

Here the notion $prox_{\lambda f}(v) = \arg\min_x \left( f(x) + \frac{1}{2\lambda} \|x - v\|_2^2 \right)$

**Theorem 3.** *[4] $f(x_k)$ is strictly decreasing and converges. The limit point of $x_k$ is also a critical point of $f$. Moreover, for all $k$,*

$$\text{Avg}\left( \|\nabla x_k - x_{k-1}\|_2^2 \right) \leq \frac{2\alpha(f(x_0) - f(x^*))}{\alpha(k+1)}$$

In 2 algorithms we introduced above, the objective may be non convex or non smooth. But the problem is how we could decompose the objective function into components.

## 2.3 Using extrapolation

Problem

$$\min_{x \in \mathbb{R}^d} f(x) = \min_{x \in \mathbb{R}^d} \mathbb{E}\left[f\left(x, \xi\right)\right]$$

**Assumption.**

- *$f(x)$ is $L-$smooth, that means*

$$\|\nabla f(u) - \nabla f(v)\| \leq L \|u - v\|$$

- *There exist $\Delta$ such that $f(x) - f(x_*) \leq \Delta$ for all $x \in \mathbb{R}^d$ where $x_*$ be the global minimum of $f(x)$*
- *(Bounded variance)*

$$\mathbb{E}\left[\|\nabla F(x, \xi) - \nabla f(x)\|^2\right] \leq G^2, \ \forall x, \xi$$

**Algorithm 3:** Mini batch stochastic gradient descent with extrapolation (Mini-batch SGDE).

```
Initialization $z_0 = x_0$ and $g_0 = \dfrac{1}{m} \sum_{i=1}^m \nabla f(x_0, \xi_{i,0})$
for each round $t = 1, 2, \cdots, T$ do
    $x_t = z_{t-1} - \eta g_{t-1}$
    $g_t = \dfrac{1}{m} \sum_{i=1}^m \nabla f(x_t, \xi_{i,t})$
    $z_t = z_{t-1} - \eta g_t$
end
```

**Theorem 4.** *[6] Chosing the learning rate $\eta \leq \frac{1}{12L}$ and*

$$\min_{t \in \{1, \cdots, T\}} \mathbb{E}\left[\|\nabla f(x_t)\|^2\right] \leq \frac{3L\eta G^2}{2T} + \frac{8(f(x_0) - f(x_*))}{\eta T} + \frac{72G^2}{m} - \frac{1}{\eta^2 T} \sum_{0}^{T-1} \mathbb{E}\left[\|x_{t+1} - x_t\|^2\right]$$

**Assumption.**

- *$f(x)$ is $L-$smooth.*
- *There exist $\Delta$ such that $f(x) - f(x_*) \leq \Delta$ for all $x \in \mathbb{R}^d$ where $x_*$ be the global minimum of $f(x)$.*
- *$f(x, \xi)$ is differentiable.*
- *(Bounded variance) $\mathbb{E}\left[\|\nabla F(x, \xi) - \nabla f(x)\|^2\right] \leq G^2, \ \forall x, \xi$*

**Algorithm 4:** Stagewise SGDE

```
Algorithm StagewiseSGDE()
    Initialization. $x^0 = x_0$
    for $s = 1, \cdots, S$ do
        $f_s(x) = f(x) + \frac{1}{2\gamma}\left\|x - x^{s-1}\right\|^2$
        $x^s$= SGDE($x^{s-1}, f_s, \eta_s, T_s$)
    end
    return $x_\tau$ where $\tau$ is chosen from $1, \cdots, S$ with probability $\mathbb{P}(\tau = i) = \frac{w_i}{\sum_{j=1}^{s} w_s}$
Procedure SGDE($x_0, f, \eta, T$)
    Initialization. $z_0 = x_0; g_0 = \nabla f(x_0, \xi_0)$
    for each round $t = 1, 2, \cdots, T$ do
        $x_t = z_{t-1} - \eta g_{t-1}$
        $g_t = \nabla f(x_t, \xi_t)$
        $z_t = z_{t-1} - \eta g_t$
    end
    return $\hat{x_t} = \frac{1}{T}\sum_{t=1}^{T} x_t$
```

**Theorem 5.** *[6] After running Stagewise SGDE with $\gamma = 1/4L; w_s = s^\alpha$ $(\alpha > 1)$ and choosing the learning parameter and the number of iteration at $s-$stage as follow: $\eta_s = \frac{c\gamma}{3s} \le \frac{1}{2L}\frac{\gamma}{3}$ and $T_s = \frac{36s}{c}$, we got the estimation:*

$$\mathbb{E}\left[\left\|\nabla f(x_\tau)\right\|^2\right] \le \frac{20\Delta(\alpha+1)}{\gamma(S+1)} + \frac{480G^2c(\alpha+1)}{S+1} - \frac{60\sum_{s=1}^{S+1} w_s D_{T_s}}{\gamma\sum_{s=1}^{S+1} w_s}$$

*where $D_{T_s} = \dfrac{1}{16T_s\eta_s}\sum_{t=1}^{T_s}\left\|x_t - x_{t-1}\right\|^2$. If we expect $\mathbb{E}\left[\left\|\nabla f(x_\tau)\right\|^2\right] \le \epsilon^2$, the number of stage should be $O(1/\epsilon^2)$.*

# References

[1] Benjamin Fehrman, Benjamin Gess, and Arnulf Jentzen. Convergence rates for the stochastic gradient descent method for non-convex objective functions, 2019.

[2] Saeed Ghadimi, Guanghui Lan, and Hongchao Zhang. Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization, 2013.

[3] Kevin Hsieh, Amar Phanishayee, Onur Mutlu, and Phillip B. Gibbons. The non-iid data quagmire of decentralized machine learning. *CoRR*, abs/1910.00189, 2019.

[4] Koulik Khamaru and Martin Wainwright. Convergence guarantees for a class of non-convex and non-smooth optimization problems. In *International Conference on Machine Learning*, pages 2601–2610. PMLR, 2018.

[5] Leslie G. Valiant. A bridging model for parallel computation. *Commun. ACM*, 33(8):103–111, August 1990.

[6] Yi Xu, Zhuoning Yuan, Sen Yang, Rong Jin, and Tianbao Yang. On the convergence of (stochastic) gradient descent with extrapolation for non-convex optimization. *arXiv preprint arXiv:1901.10682*, 2019.

[7] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-iid data, 2018.