Lipschitz regularity of neural networks: analysis and efficient estimation

Kevin Scaman and Aladin Virmaux
Huawei Noah's Ark Lab

Background & Setting

1. Abstract

In this paper, we estimate and analyze the **Lipschitz constant of neural networks**, key metric for safe and robust deep learning. First, we show that even for two layer neural networks, **exact computation is NP-hard**. Then, we provide **two Lipschitz upper bounds**: *AutoLip*, the first generic algorithm for upper bounding the Lipschitz constant of any automatically differentiable function, and *SeqLip*, an improved algorithm for MLPs. Moreover, we provide an **AutoGrad compliant power method** algorithm, allowing efficient computations even on large convolutions. Our experiments show that *SeqLip* can significantly improve on the existing upper bounds.

2. Notations

1. A function $f : \mathbb{R}^n \to \mathbb{R}^m$ is *Lipschitz continuous* if
$$\forall x, y \in \mathbb{R}^n,\ \|f(x) - f(y)\|_2 \leq L \|x - y\|_2.$$
The smallest such L is the *Lipschitz constant* of f, denoted $L(f)$.
2. A K-layer *Multi-Layer-Perceptron* $f_{MLP} : \mathbb{R}^n \to \mathbb{R}^m$ is the function
$$f_{MLP}(x) = T_K \circ \rho_{K-1} \circ \cdots \circ \rho_1 \circ T_1(x),$$
where $T_k(x) = M_k x + b_k$ is affine and $\rho_k(x) = (g_k(x_i))_{i \in [\![1, n_k]\!]}$ non-linear.

3. Exact Lipschitz computation is NP-hard

Problem **LIP-CST**:

Input: Two matrices $M_1 \in \mathbb{R}^{l \times n}$ and $M_2 \in \mathbb{R}^{m \times l}$, and a constant $\ell \geq 0$.
Question: Let $f = M_2 \circ \rho \circ M_1$ where $\rho(x) = \max\{0, x\}$ is the ReLU activation function.
Is the Lipschitz constant $L(f) \leq \ell$?

Theorem LIP-CST is NP-hard.
Reduction to maximizing a convex quadratic function on the hypercube.

4. Lipschitz constant of affine layers

Algorithm 1 AutoGrad compliant power method

Input: affine function $f : \mathbb{R}^n \to \mathbb{R}^m$, number of iteration N
Output: approximation of the Lipschitz constant $L(f)$

for $k = 1$ to N **do**
  $v \leftarrow \nabla g(v)$ where $g(x) = \frac{1}{2}\|f(x) - f(0)\|_2^2$
  $\lambda \leftarrow \|v\|_2$
  $v \leftarrow v/\lambda$
end for
return $L(f) = \|f(v) - f(0)\|_2$

Other layers Most common other layers in neural networks including activation functions, pooling, batch normalization have simple and explicit Lipschitz constants.

Upper Bounds

5. AutoLip

For sequential networks, this upper bounds is simply the product of spectral norms. Otherwise, we propagate the Lipschitz upper bound with Alg. 2.

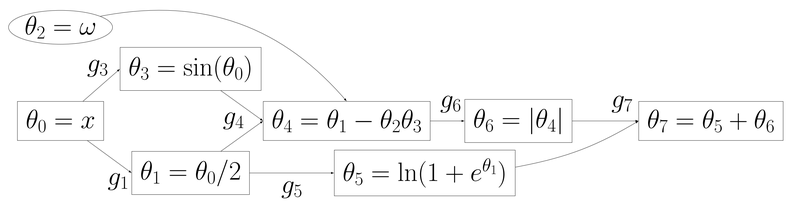

Figure: Example of a computation graph for $f_\omega(x) = \ln(1 + e^{x/2}) + |x/2 - \omega \sin(x)|$

Algorithm 2 AutoLip

Input: function $f : \mathbb{R}^n \to \mathbb{R}^m$ and its computation graph $(g_1, ..., g_K)$
Output: upper bound on the Lipschitz constant: $\hat{L}_{AL} \geq L(f)$

$\mathcal{Z} = \{(z_0, ..., z_K)\ :\ \forall k \in [\![0, K]\!], \theta_k \text{ is constant} \Rightarrow z_k = \theta_k(0)\}$
$L_0 \leftarrow 1$
for $k = 1$ to K **do**
  $L_k \leftarrow \sum_{i=1}^{k-1} \max_{z \in \mathcal{Z}} \|\partial_i g_k(z)\|_2 L_i$
end for
return $\hat{L}_{AL} = L_k$

6. SeqLip

Idea: Exploit the relationships between singular vectors of consecutive layers.

$$\begin{aligned} L(f_{MLP}) &\leq \max_{\forall i,\ \sigma_i \in [0,1]^{n_i}} \|M_K \operatorname{diag}(\sigma_{K-1}) \cdots \operatorname{diag}(\sigma_1) M_1\|_2\,, \\ &\leq \max_{\forall i,\ \sigma_i \in [0,1]^{n_i}} \|\Sigma_K V_K^\top \operatorname{diag}(\sigma_K) U_{K-1} \Sigma_{K-1} V_{K-1}^\top \operatorname{diag}(\sigma_{K-1}) \cdots\|_2 \\ &\leq \hat{L}_{SL} = \prod_{i=1}^{K-1} \max_{\sigma_i \in [0,1]^{n_i}} \left\| \widetilde{\Sigma}_{i+1} V_{i+1}^\top \operatorname{diag}(\sigma_{i+1}) U_i \widetilde{\Sigma}_i \right\|_2 \end{aligned}$$

where $M_i = U_i \Sigma_i V_i^\top$, $\widetilde{\Sigma}_i = \Sigma_i$ if $i \in \{1, K\}$ and $\widetilde{\Sigma}_i = \Sigma_i^{1/2}$ otherwise.

Theorem Let M_k be the matrix associated to the k-th linear layer, u_k (resp. v_k) its first left (resp. right) singular vector, and $r_k = s_{k,2}/s_{k,1}$ the ratio between its second and first singular values. Then we have

$$\hat{L}_{SL} \leq \hat{L}_{AL} \prod_{k=1}^{K-1} \sqrt{(1 - r_k - r_{k+1}) \max_{\sigma \in [0,1]^{n_k}} \langle \sigma \cdot v_{k+1}, u_k \rangle^2 + r_k + r_{k+1} + r_k r_{k+1}}\,.$$

If the ratios r_k are negligible, then

$$\hat{L}_{SL} \leq \hat{L}_{AL} \prod_{k=1}^{K-1} \max_{\sigma \in [0,1]^{n_k}} |\langle \sigma \cdot v_{k+1}, u_k \rangle| \quad \text{and} \quad \hat{L}_{SL} \approx \frac{\hat{L}_{AL}}{\pi^{K-1}}\,.$$

Experiments

7. Experiments

Theoretical setting

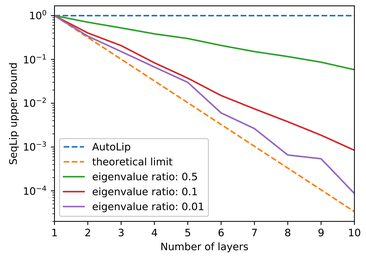

Figure: SeqLip in the ideal scenario

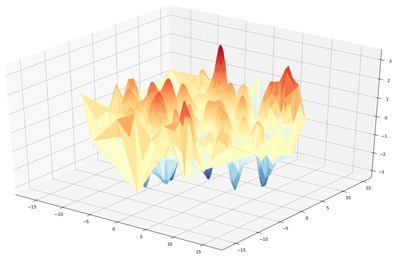

Figure: Synthetic function to train MLPs

MLP

# layers	Upper bounds Frob.	AutoLip	SeqLip	Greedy SL	Lower bounds Data	Annealing	Grid Search
4	648.2	33.04	21.47	21.47	4.36	4.55	6.56
5	4283.1	134.4	72.87	72.87	6.77	5.8	7.1
7	22341	294.6	130.2	130.2	5.4	5.27	6.51
10	7343800	19248.2	2463.44	2463.36	10.04	5.77	17.1

Figure: AutoLip and SeqLip for MLPs of various size

CNN

# layers	Upper bounds AutoLip	Greedy	SeqLip	Ratio	Lower bounds Dataset	Annealing
4	174	86	2		12.64	25.5
5	790.1	335	2.4		16.79	22.2
7	12141	3629	3.3		31.22	43.6
10	$4.5 \cdot 10^6$	$8.2 \cdot 10^5$	5.4		38.26	107.8

Figure: AutoLip and SeqLip for MNIST-CNNs of various size

AlexNet

AutoLip	Greedy SeqLip
3.62×10^7	5.45×10^6

Figure: AutoLip and SeqLip for AlexNet

8. Conclusion and perspectives

SeqLip can improve by a **large factor** the upper-bound given by AutoLip (up to a **factor 8** in real scenarios). However, these bounds remain **very large for vision networks**, and it is yet an open question to know how close these bounds are to the **real Lipschitz constant**.