

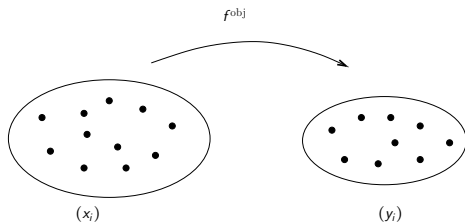
# Lipschitz stability of Deep Neural Networks in view of applications

B. Després (LJLL-SU)  
M. Pintore (Post-doc AeX-Inria and PEPR-IA)

## Introduction

Lipschitz  
stabilityExtension to  
CNN

- Take a large enough **dataset** :  $\mathcal{D} = \{(x_i, y_i), i = 1, \dots\} \subset \mathbb{R}^m \times \mathbb{R}^n$



Postulate : dataset corresponds to unknown objective/transfer function

$$H : \mathbb{R}^m \longrightarrow \mathbb{R}^n$$

with  $x_i \in \mathbb{R}^m$ ,  $y_i = H(x_i) + \varepsilon_i \in \mathbb{R}^n$ , and noise  $\varepsilon_i \in \mathbb{R}^n$ .

# Least square representation with recursivity

Introduction

- Take a linear function  $f$  with **weight**  $W \in \mathcal{M}_{mn}(\mathbb{R})$  and **bias**  $b \in \mathbb{R}^n$

$$\begin{aligned} f : \mathbb{R}^m &\longrightarrow \mathbb{R}^n, \\ x &\longmapsto f(x) = Wx + b. \end{aligned} \tag{1}$$

Lipschitz stability

Extension to CNN

- Notations

$a_0 = m$  is the **input layer**

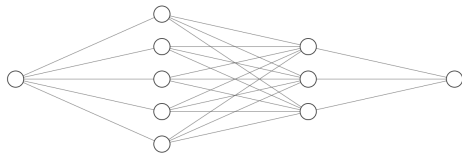
$a_{p+1} = n$  is the **output layer**

$(a_1, a_2, \dots, a_p) \in \mathbb{N}^p$  are the **(dense) hidden layers** with **neurons**

- Consider

$$\begin{aligned} f_r : \mathbb{R}^{a_r} &\longrightarrow \mathbb{R}^{a_{r+1}}, \\ X_r &\longmapsto f_r(X_r) = W_r X_r + b_r \end{aligned}$$

and the function  $f = f_p \circ f_{p-1} \dots \circ f_2 \circ f_1 \circ f_0$ .



Input Layer  $\in \mathbb{R}^1$

Hidden Layer  $\in \mathbb{R}^4$

Hidden Layer  $\in \mathbb{R}^4$

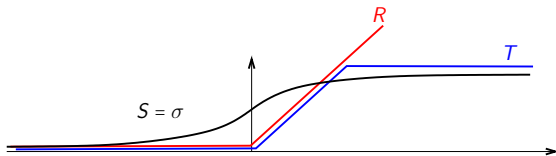
Output Layer  $\in \mathbb{R}^1$

- Non linearity is added with an **activation function**.

**Sigmoid**  $\in C^1(\mathbb{R})$ . A sigmoid  $\sigma$  is monotone,  $0 < \sigma' < 1$ , with limit value 0 at  $-\infty$  and limit value 1 at  $+\infty$ .

**ReLU**  $\in C^0(\mathbb{R})$ . It is defined by  $R(x) = \max(0, x)$ .  
Thresholding yields  $T(x) = \min(R(x), 1)$ .

Generalization component wise to activation functions  $\mathbb{R}^q \rightarrow \mathbb{R}^q$ .



A function  $f$  defined through a generic **feed-forward neural network** is :

$$f = f_{p+1} \circ S_{p+1} \circ f_p \circ \dots \circ f_1 \circ S_1 \circ f_0, \text{ where activation function is } S_r = \sigma \text{ or } R.$$

# Fit the coefficients with numerical optimisation

Introduction

Lipschitz  
stability

Extension to  
CNN

Lingo : deep learning= $p$  large, training=numerical optimisation, Machine Learning= optimization, epoch=global iteration, batches=data for local iterations, neural networks=special non linear functions, regression=Least square, ...

- Consider the cost function

$$J(W, b) = \frac{1}{\text{card}\mathcal{D}} \sum_{(x,y) \in \mathcal{D}} |f(x) - y|^2$$

An optimal value satisfies

$$J(W_*, b_*) \leq J(W, b) \quad \forall (W, b).$$

The training=*minimization session on the computer with ad-hoc stochastic gradient algorithms* can extremely difficult since the cost function  $J$  is **highly non convex for  $p \geq 2$**

- Important variation for classification with CNNs. Let  $y$  and  $p(z)$  be discrete probabilities :  $y_i \in [0, 1]$ ,  $\sum y_j = 1$  ;  $p_i = \frac{\exp z_i}{\sum_{j=1}^n \exp z_j} \in (0, 1)$ . Consider the Kullback-Leibler divergence (which is convex)

$$J(W, b) = - \sum_{(x,y) \in \mathcal{D}} (\log p(f(x)), y) \geq 0.$$

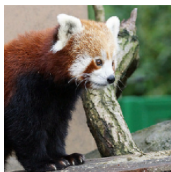
# What is the stability of the result ?

Introduction

- Biggio et al : 2013
- Example from Franceschi et al : 2021.

Lipschitz  
stability

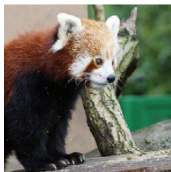
Extension to  
CNN



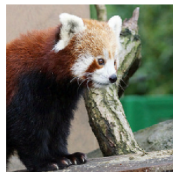
red panda  
(unperturbed)



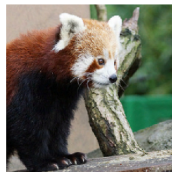
brown bear



teddy bear



polecat



polecat

- Madry et al : Towards deep learning models resistant to adversarial attacks, in 6th International Conference on Learning Representations, 2018.
- Nota Bene :  $l^1$  norm probably better to analyze tables of pixels  $\approx$  images.

A question is :

Introduction

Lipschitz  
stability

Extension to  
CNN

Can we **certify** with SciML= Scientific Machine Learning these functions/techniques/algorithms/... , with critical applications in mind ?



The emerging discipline needs much more on **mathematical stability** of NN functions.

Related ML key words : predictability, robust IA, control of generalization error, reproductibility, adversarial networks, explainibility, ...

## Introduction

### Lipschitz stability

### Extension to CNN

- Assume that  $f$  is produce by ML. Consider the dynamical system

$$\begin{cases} x'(t) = f(x(t)), \\ x(0) = x_0 \end{cases}$$

What can we say about the stability of solutions of the ODE ?

- Our model problem is then to :

⇒ **Evaluate the Lipschitz constant  $L$**

$$|f(x) - f(y)| \leq L|x - y|, \quad \forall x, y \in \mathbb{R}^n.$$

Immediate application to the control of the dynamics

$$|x(t) - y(t)| \leq e^{Lt}|x(0) - y(0)|.$$

Example : noisy initial condition  $x(0) = y(0) + \varepsilon$ .

# Setting of the problem

Introduction

- Take  $f : \mathbb{R}^{a_0} \rightarrow \mathbb{R}^{a_{\ell+1}}$

$$f = f_{\ell} \circ S_{\ell} \circ f_{\ell-1} \circ S_{\ell-1} \circ \dots \circ S_1 \circ f_0$$

Lipschitz stability

The regularity is

$$f \in \text{Lip}(\mathbb{R}^{a_0})^{a_{\ell+1}} \subset C^0(\mathbb{R}^{a_0})^{a_{\ell+1}}.$$

Extension to CNN

Rademacher's Theorem : the gradient is

$$\nabla f(x) = W_{\ell} D_{\ell}(x) W_{\ell-1} D_{\ell-1}(x) \dots D_1(x) W_0.$$

- Take a vectorial norm, for example  $\|x\|_{l^p(\mathbb{R}^a)} = (\sum_{i=1}^a |x_i|^p)^{1/p}$ .  
The induced norm for matrices is  $M \in \mathcal{M}_{ab}(\mathbb{R}) = \mathbb{R}^{a \times b}$  is

$$\|M\| = \|M\|_{l^p(\mathcal{M}_{ab}(\mathbb{R}))} = \max_{x \neq 0} \frac{\|Mx\|_{l^p(\mathbb{R}^a)}}{\|x\|_{l^p(\mathbb{R}^b)}}. \quad (2)$$

- Our goal is to obtain sharp upper bounds on

$$L = \sup_{x \in \mathbb{R}^{a_0}} \|\nabla f(x)\|_{l^p(\mathcal{M}_{a_{\ell+1}, a_0}(\mathbb{R}))} = \left( \|\nabla f\|_{l^p(\mathcal{M}_{a_{\ell+1}, a_0}(\mathbb{R}))} \right)_{L^{\infty}(\mathbb{R}^{a_0})}.$$

# Previous results (all in $L^2$ norms as far we understand)

Introduction

Lipschitz stability

Extension to CNN

- Everywhere. One has  $L \leq K_*$  where  $K_* = \prod_{r=0}^{\ell} \|W_r\|$ .
- Szegedy et al (2013). One has  $L \leq K \leq K_*$  where

$$K = \max_{D \in \mathcal{D}} \|W_{\ell} D_{\ell} W_{\ell-1} D_{\ell-1} \dots D_1 W_0\|,$$

Note immediately the complexity  $2^{a_0+a_1+\dots+a_{\ell}} = \prod_{r=0}^{\ell} 2^{a_r}$ .

- Virmaux-Scaman (2019). Find upper bound on  $K$  with SVD decomposition. The complexity reduces to  $\sum_{r=0}^{\ell} 2^{a_r}$ .
- Combettes-Pesquet (2020). Define  $W_{(t,s)} = W_{t-1} W_{t-2} \dots W_{s+1} W_s$ . Then  $K \leq K_1 \leq K_*$  where

$$K_1 = \frac{1}{2^{\ell}} \sum_{1 \leq r_1 < r_2 < \dots < r_n \leq \ell} \|W_{(\ell+1, r_n)}\| \|W_{(r_n, r_{n-1})}\| \dots \|W_{(r_2, r_1)}\| \|W_{(r_1, 0)}\|.$$

The complexity is  $2^{\ell} \ll \sum_{r=0}^{\ell} 2^{a_r} \ll \prod_{r=0}^{\ell} 2^{a_r}$ .

Introduction

Lipschitz  
stabilityExtension to  
CNN

Define  $Z_r = 2D_r - I_r = \text{diag}(\pm 1)$  so that  $D_r = \frac{1}{2}(I_r + Z_r)$ .

Define  $\mathcal{Z}_r = \{Z_r\}$  and  $\mathcal{Z} = \mathcal{Z}_\ell \times \dots \times \mathcal{Z}_1$ .

$$\begin{aligned}
 K &= \max_{D \in \mathcal{D}} \|W_\ell D_\ell W_{\ell-1} D_{\ell-1} \dots D_1 W_0\| \\
 &= \max_{Z \in \mathcal{Z}} \left\| W_\ell \left( \frac{1}{2}(I_\ell + Z_\ell) \right) W_{\ell-1} \left( \frac{1}{2}(I_{\ell-1} + Z_{\ell-1}) \right) \dots \left( \frac{1}{2}(I_1 + Z_1) \right) W_0 \right\| \\
 &\leq \frac{1}{2^\ell} \max_{Z \in \mathcal{Z}} \sum_{1 \leq r_1 < r_2 < \dots < r_n \leq \ell} \|W_{(\ell+1, r_n)} Z_{r_n} W_{(r_n, r_{n-1})} \dots W_{(r_2, r_1)} Z_{r_1} W_{(r_1, 0)}\| \\
 &\leq \frac{1}{2^\ell} \max_{Z \in \mathcal{Z}} \sum_{1 \leq r_1 < r_2 < \dots < r_n \leq \ell} \|W_{(\ell+1, r_n)}\| \|Z_{r_n}\| \|W_{(r_n, r_{n-1})}\| \dots \|W_{(r_2, r_1)}\| \|Z_{r_1}\| \|W_{(r_1, 0)}\| \\
 &\leq \frac{1}{2^\ell} \sum_{1 \leq r_1 < r_2 < \dots < r_n \leq \ell} \|W_{(\ell+1, r_n)}\| \|W_{(r_n, r_{n-1})}\| \dots \|W_{(r_2, r_1)}\| \|W_{(r_1, 0)}\| \\
 &= K_1 \leq K_*.
 \end{aligned}$$

"Trivially", one also  $K \leq K_*$ .

It is well known that

$$\|A\|_1 = \max_{1 \leq j \leq n} \sum_{i=1}^m |A_{ij}| \text{ and } \|A\|_\infty = \max_{1 \leq i \leq m} \sum_{j=1}^n |A_{ij}|.$$

## Definition

Given a matrix  $A \in \mathcal{M}_{m,n}(\mathbb{R})$ , denote  $A^{\text{abs}} \in \mathcal{M}_{m,n}(\mathbb{R})$  the matrix such that

$$(A^{\text{abs}})_{ij} = |A_{ij}|, \quad \forall i, j.$$

One obtains

$$\|A\|_1 = \|A^{\text{abs}}\|_1 \text{ and } \|A\|_\infty = \|A^{\text{abs}}\|_\infty \text{ (not true for } l^2\text{-based norms).}$$

## Theorem

One has the bound  $K \leq K_3$  where

$$K_3 = \|W_\ell^{\text{abs}} W_{\ell-1}^{\text{abs}} \dots W_0^{\text{abs}}\|_{1,\infty}.$$

Introduction

Lipschitz  
stability

Extension to  
CNN

Combine with the Combettes-Pesquet trick

$$K_4 = \frac{1}{2^\ell} \sum_{1 \leq r_1 < r_2 < \dots < r_n \leq \ell} \|W_{(\ell+1, r_n)}^{\text{abs}} \dots W_{(r_2, r_1)}^{\text{abs}} W_{(r_1, 0)}^{\text{abs}}\|_{1, \infty}.$$

## Theorem

One has the bounds  $K \leq K_4 \leq K_1$ .

---

# Summary of these bounds

Introduction

Lipschitz  
stability

Extension to  
CNN

Here we use  $l^1$  and  $l^\infty$  norms

$$L \leq K \leq K_4 \leq \min(K_1, K_3) \leq \max(K_1, K_3) \leq K_*$$

# Approximation of $x^2$

Introduction

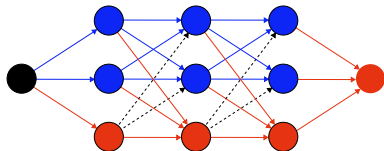
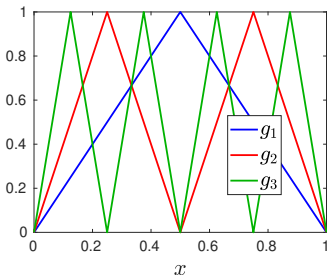
Lipschitz stability

Extension to CNN

$$g(x) = \begin{cases} 2x, & x \in [0, 0.5] \\ 2(1-x), & x \in [0.5, 1] \end{cases}, \quad g_r(x) = \underbrace{g \circ g \circ \dots \circ g}_{r \text{ times}}.$$

Converging series : Takagi (1902), Yarostky (2017), Devore et al, D., ...

$$x^2 = x - \sum_{r=0}^{\infty} \frac{g_r(x)}{4^r}.$$



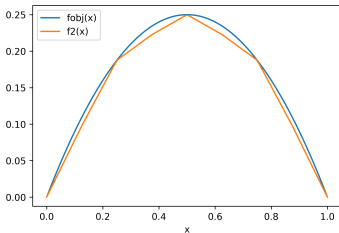
Graphical representation of the truncated network  $x - \sum_{r=0}^3 \frac{g_r(x)}{4^r}$ .

# Illustration : $x \mapsto x - x^2$

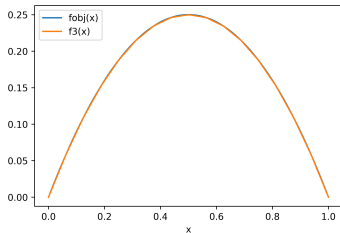
Introduction

Lipschitz stability

Extension to CNN



two layers



three layers

Clearly  $\|x^2 - (x - \sum_{r=1}^p \frac{1}{4^r} g_r(x))\|_{L^\infty(0,1)} \leq \sum_{p+1 \leq r} \frac{1}{4^r} = \frac{1}{3 \times 4^p}$ .

**Width  $N = 3$ , depth  $p$**  : accuracy is  $\varepsilon = O(4^{-p})$ , for a cost

Cost =  $O(3 \times p) \approx C |\log \varepsilon|$ .

## Theorem (Yarostky 2017)

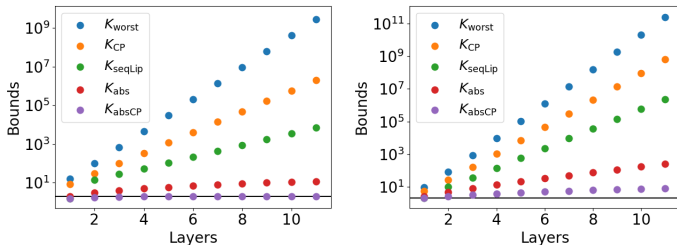
There exists a NN approximating all bounded functions  $\in W^{n,\infty}([0,1]^d)$  with uniform accuracy  $\varepsilon$ , uniform cost  $\approx \log 1/\varepsilon$  and at most  $\approx \varepsilon^{-d/n} \log 1/\varepsilon$  computational units.

# Results and optimality of $K_4$

Introduction

Lipschitz stability

Extension to CNN



Bounds depending on the ReLU representation of  $g$ .

$\ell$	$L$	$K_*$	$K_1$	$K_2$	$K_3$	$K_4$
1	1.5	3.0	2.0	1.5	2.0	1.5
2	1.75	9.0	3.53125	3.64531	3.0	1.75
3	1.875	33.0	6.92187	6.45925	4.0	1.875
4	1.93875	129.0	14.42877	12.45882	5.0	1.93875
5	1.96875	513.0	30.75637	24.68184	6.0	1.96875
6	1.984375	2049.0	66.00227	49.24501	7.0	1.984375

# A test for $T(x, y) = xy$

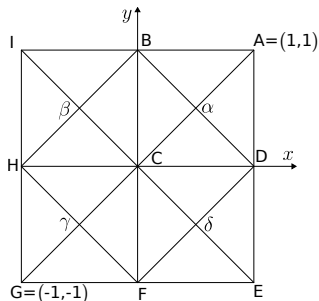
Introduction

Lipschitz  
stability

Extension to  
CNN

A new formula with strong connexion with FEM : consider the function

$$\Lambda = (\Lambda_1, \Lambda_2) : \mathbb{R}^2 \rightarrow \mathbb{R}^2$$



$$\Lambda_1 = \varphi_\alpha - \varphi_\beta + \varphi_\gamma - \varphi_\delta$$

$$\Lambda_2 = \varphi_D - \varphi_A + \varphi_B - \varphi_C + \varphi_F - \varphi_E - \varphi_G + \varphi_H - \varphi_I,$$

Introduction

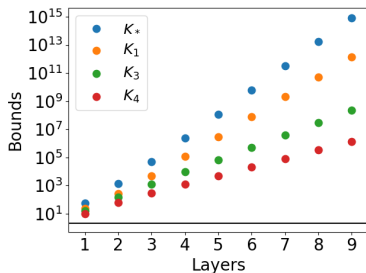
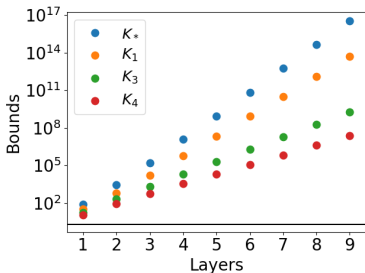
Lipschitz  
stabilityExtension to  
CNN

One can check the formula where  $e_0 \in P_{\text{FEM}}^1$

$$T = e_0 + \frac{1}{4} T \circ \Lambda \implies T = \sum_{n \geq 0} \frac{1}{4^n} e_0 \circ \underbrace{\Lambda \circ \dots \circ \Lambda}_{n \text{ times}}.$$

It reveals the power of composition.

Introduction

Lipschitz  
stabilityExtension to  
CNNLipschitz bounds for networks approximating the function  $xy$ .

Remark : the formula is an alternative to the multiplier function in the influential contribution Yarotsky 2017.

# Matrices with random weights (from C.P.)

Introduction

Lipschitz  
stability

Extension to  
CNN

Take  $W_3 \in \mathcal{M}_{3,6}(\mathbb{R})$ ,  $W_2 \in \mathcal{M}_{6,10}(\mathbb{R})$  and  $W_1 \in \mathcal{M}_{10,8}(\mathbb{R})$ . Their entries are i.i.d. realizations of the normal distribution  $\mathcal{N}(0, 1)$ .

$$f = f_3 \circ R \circ f_2 \circ R \circ f_1, \quad R = \text{ReLU}.$$

Statistic	$K/K_*$	$K_1/K_*$	$K_2/K_*$	$K_3/K_*$	$K_4/K_*$
Maximum	0.2772	0.5786	0.6789	0.8023	0.4608
Average	0.1422	0.4539	0.3256	0.5461	0.2875
Minimum	0.0595	0.3703	0.1597	0.2897	0.1604
Standard deviation	0.0343	0.0350	0.0685	0.0813	0.0483

Statistics over 1000 realizations.

Once again,  $K_4$  is the best one.

# Convolutional neural networks

Introduction

CNNs are central for classification, as exemplified by the MNIST dataset

Lipschitz  
stability

Extension to  
CNN



$$g = T_{\ell+1} \circ g_{\ell} \circ T_{\ell} \circ g_{\ell-1} \circ T_{\ell-1} \circ \cdots \circ T_1 \circ g_0.$$

For simplification we neutralize the last soft-max function  $T_{\ell+1} = I$ .

$$g = g_{\ell} \circ T_{\ell} \circ g_{\ell-1} \circ T_{\ell-1} \circ \cdots \circ T_1 \circ g_0.$$

Normalization is that  $g_r$  have ability to change the dimension and that the  $T_r$  do not change the dimension.

As before, a linear function  $g_i^{\text{lin}} : \mathbb{R}^{a_i} \rightarrow \mathbb{R}^{a_{i+1}}$  is written

$$g_i^{\text{lin}}(x) = W_i x + b_i.$$

# Convolutional layers

Introduction

- A convolutional layer  $g_i^{\text{conv}} : \mathbb{R}^{d_i} \rightarrow \mathbb{R}^{d_{i+1}}$

Lipschitz  
stability

$$g_i^{\text{conv}}(x) = K_i^{\text{conv}} * \bar{x} + \bar{b}_i.$$

Extension to  
CNN

Since convolution operators are linear operators, there exist a matrix  $W_i = W_i^{\text{conv}}$  such that

$$g_i^{\text{conv}}(x) = W_i x + b_i,$$

Example : take a vertical vector

$$x = [x_{11} \quad x_{12} \quad x_{13} \quad x_{21} \quad x_{22} \quad x_{23} \quad x_{31} \quad x_{32} \quad x_{33}]^T \in \mathbb{R}^9. \quad (3)$$

A re-indexation allows to write

$$\bar{x} = \begin{bmatrix} x_{11} & x_{12} & x_{13} \\ x_{21} & x_{22} & x_{23} \\ x_{31} & x_{32} & x_{33} \end{bmatrix} \quad (4)$$

- The normalization convention allows to represent the non-linear **max-pool function** as a  $g_i$

$$g_i(x) = (\max(x_{11}, x_{12}, x_{21}, x_{22}), \dots) \in \mathbb{R}^4.$$

Introduction

Lipschitz  
stability

Extension to  
CNN

$$g \in \text{Lip}(\mathbb{R}^{a_0})^{a_{\ell+1}} \subset C^0(\mathbb{R}^{a_0})^{a_{\ell+1}}.$$

All previous results are generalized mutatis mutandi.

# A test with CNN (MNIST, LeCun et al)

Introduction

Lipschitz  
stability

Extension to  
CNN

Training with regularization (0.01).

The accuracy on the test set is always between 95% and 97%.

Model	Approach	$l^1$				$l^\infty$			
		$K_*$	$K_1$	$K_3$	$K_4$	$K_*$	$K_1$	$K_3$	$K_4$
Model A	Explicit	1.058e5	1.107e4	9.377e3	2.222e3	1.565e6	1.734e5	2.993e5	4.756e4
	Implicit	1.562e2	4.721e1	5.191e1	<b>2.739e1</b>	3.678e3	9.021e2	1.859e3	<b>6.424e2</b>
Model B	Explicit	1.712e7	7.353e5	1.599e5	3.626e4	2.878e7	3.171e6	4.590e6	6.804e5
	Implicit	4.280e2	1.423e2	1.772e2	<b>1.016e2</b>	1.151e4	2.823e3	6.203e3	<b>2.294e3</b>
Model C	Explicit	4.950e8	1.910e7	3.120e6	7.327e5	8.241e8	7.749e7	9.068e7	1.170e7
	Implicit	7.735e2	2.337e2	1.853e2	<b>1.230e2</b>	2.060e4	4.570e3	6.574e3	<b>2.307e3</b>

Once again,  $K_4$  is the best one.

Introduction

Lipschitz  
stability

Extension to  
CNN

- Stability of NNs is a fundamental topic, at least for the development of SciML and potential applications to certification.
- Linear algebra has something to say about the Lipschitz stability of NNs. So far  $K_4$  is systematically the best estimate.
- **Open question 1** : find better bounds through spectral analysis.
- **Open question 2** : use such bounds to regularize training.
- **Open question 3** : use such bounds for applications such as "learning real problems" and "solving PDEs".

D.-Pintore : Certified computable Lipschitz bounds for Deep Neural Networks.