

Représentation neurale implicite avec fonction d'activation périodique

Marine VIEILLARD Romain HÛ

17 mars 2025

1. Pourquoi les SIREN ?

SIREN (SInusoidal REpresentation Networks)

Architectures actuelles :

- Incapables de modéliser des signaux avec des détails fins.
- Ne parviennent pas à représenter les dérivées spatiales et temporelles d'un signal.

Architecture SIREN :

- Capables de représenter avec précision des images, des champs d'ondes, des vidéos, du son et leurs dérivées.
- Résoudre des problèmes complexes de valeurs aux limites (équation de Poisson, équations de Helmholtz et des ondes).

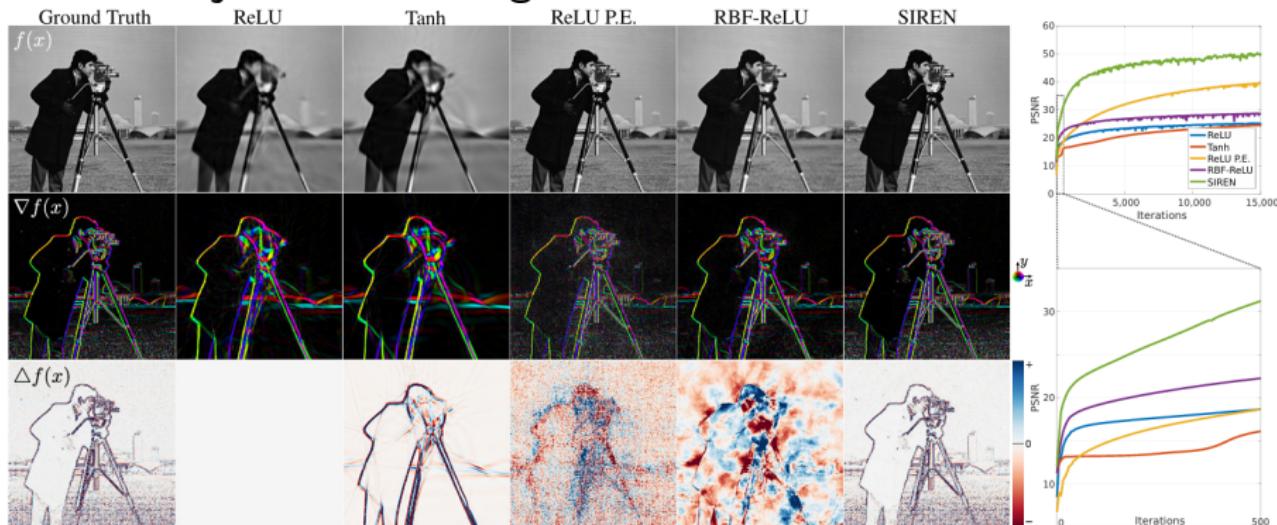
2. Pourquoi les SIREN ?

On s'intéresse à une classe de fonctions Φ qui satisfont des équations de la forme :

$$F(x, \Phi, \nabla_x \Phi, \nabla_x^2 \Phi, \dots) = 0, \quad \Phi : x \mapsto \Phi(x).$$

3. Pourquoi les SIREN ?

Comparaison de différentes architectures de réseaux implicites ajustant une image de référence et ses dérivées



1. Que sont les SIREN ?

SIREN : Architecture qui utilise le sinus comme fonction d'activation périodique. Une couche sinus est définie comme suit :

$$\Phi(x) = W_n(\phi_{n-1} \circ \phi_{n-2} \circ \dots \circ \phi_0)(x) + b_n$$

avec $x_i \mapsto \phi_i(x_i) = \sin(W_i x_i + b_i)$

1. Initialisation des SIRENs

Distribution de sortie d'un neurone sinus unique :

Input : $x \sim \mathcal{U}(-1, 1)$

Output : $y = \sin(ax + b) \quad \forall a > \frac{\pi}{2}, y \sim \text{arcsin}(-1, 1)$

Distribution de sortie d'un neurone sinus :

Input : $x \in \mathbb{R}^n, w \in \mathbb{R}^n$

Output : $y = \sin(w^T x + b)$

Cas d'un neurone sinus situé sur la 2ème couche :

Input : Distribution $x \sim \text{arcsin}(-1, 1)$ et $w_i \sim \mathcal{U}\left(-\frac{c}{\sqrt{n}}, \frac{c}{\sqrt{n}}\right)$

Alors $w^T x \sim \mathcal{N}(0, c^2/6)$

2. Initialisation des SIRENs

Finalement, on tire les poids selon : $w_i \sim \mathcal{U}\left(-\sqrt{\frac{6}{n}}, \sqrt{\frac{6}{n}}\right)$

Initialisation de la première couche : $\sin(\omega_0 Wx + b)$

1. Résoudre une équation différentielle

Exemple de l'équation d'onde

$\phi(x, y, t)$ doit vérifier :

$$\frac{\partial^2 \phi}{\partial t^2} = c^2 \Delta \phi \text{ avec } \Delta \phi = \frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2}$$

Condition initiale de neumann : $\frac{\partial \phi(0, x, y)}{\partial t} = 0$

Condition initiale de dirichlet : $\phi(0, x, y) = f(x, y)$

2. Résoudre une équation différentielle

On en tire la fonction de perte suivante :

$$\underbrace{\|\frac{\partial^2 \phi}{\partial t^2} - c^2 \Delta \phi\|_2^2}_{\text{equation}} + \lambda_1 \underbrace{\|\frac{\partial \phi(0, x, y)}{\partial t}\|_2^2}_{\text{neuman}} + \lambda_2 \underbrace{\|\phi(0, x, y) - f(x, y)\|_2^2}_{\text{dirichlet}}$$

Avec λ_1, λ_2 des constantes de dosages.

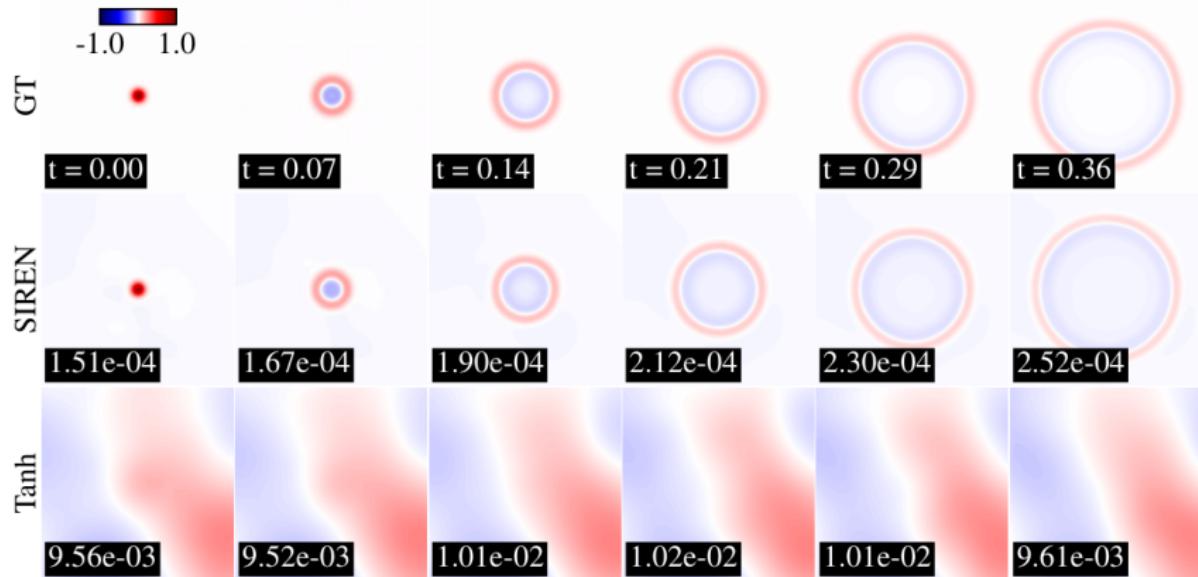
3. Résoudre une équation différentielle

L'espace des réseaux de neurones est stable par dérivation par rapport à l'entrée.

En particulier, l'espace des *SIREN* est stable par dérivation par rapport à l'entrée.

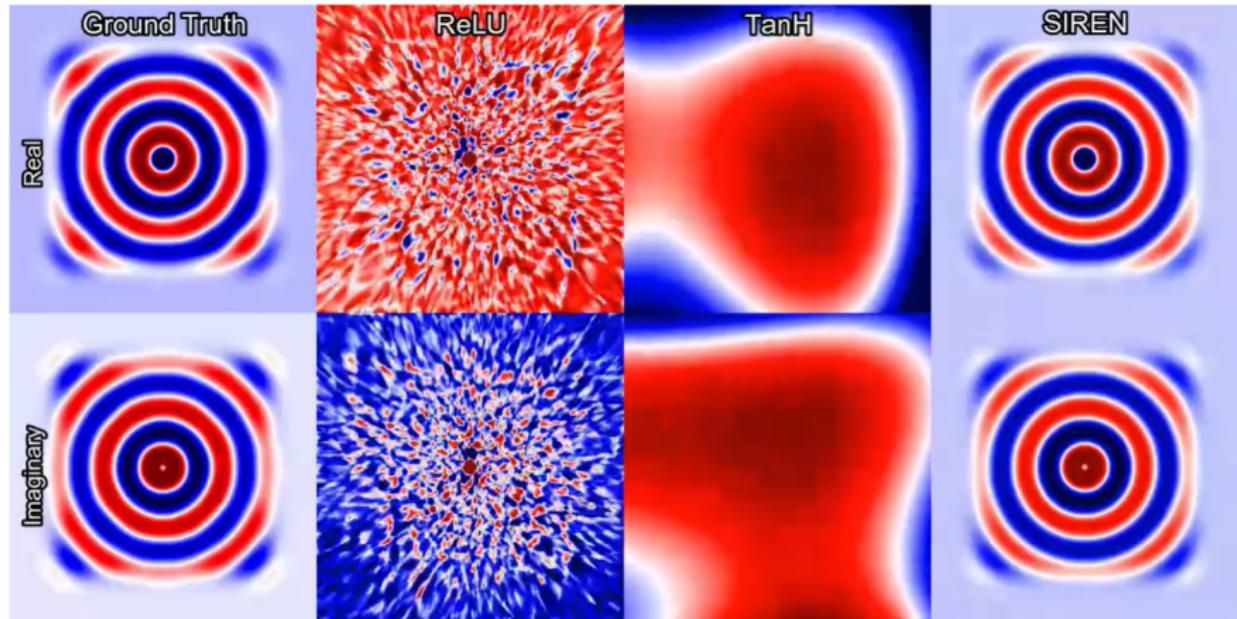
4. Résoudre une équation différentielle

Résultats pour l'équation d'onde



5. Résoudre une équation différentielle

$$\text{Equation de helmholtz : } \Delta\phi + \frac{1}{c(x,y)^2}\phi = -f(x,y)$$



1. Inpainting

Image originale B masquée pour obtenir une image b .

B est paramétrée par $\phi(x, y)$ où x, y sont les coordonnées du pixel.

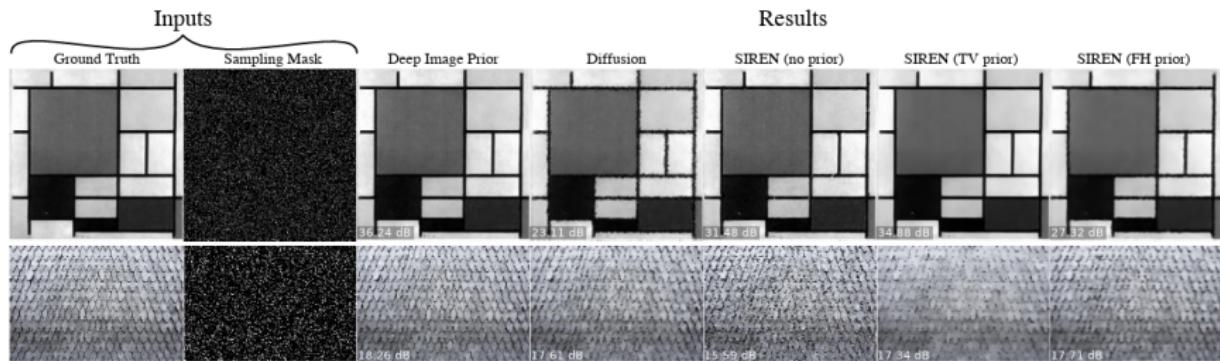
Fonction de perte : RMSE(ϕ, b) + a priori $\rightarrow \phi$ représente B implicitement !

Exemples d'a prioris :

- Variation totale : $\frac{1}{N} \|\nabla \phi(x, y)\|_1$
- Norme de Frobenius de la hessienne : $\frac{1}{N} \|hess(\phi(x, y))\|_F$

où N est le nombre de points échantillonnés.

2. Inpainting



3. Inpainting

Anti-aliasing : on remplace $\phi(x, y)$ par $h * \phi(x, y)$ où h est un noyau de convolution aléatoire :

$$h * \phi(x, y) = \frac{1}{N} \sum_{i=1}^N \phi(x + x_i, y + y_i)$$

où $x_i, y_i \sim \mathcal{P}$ de densité $p(x_i, y_i) = \frac{1}{2} \max(0, 1 - |x_i|) \max(0, 1 - |y_i|)$

Fonction de perte finale :

$$\|h * \phi - b\|_2^2 + \text{a priori}$$

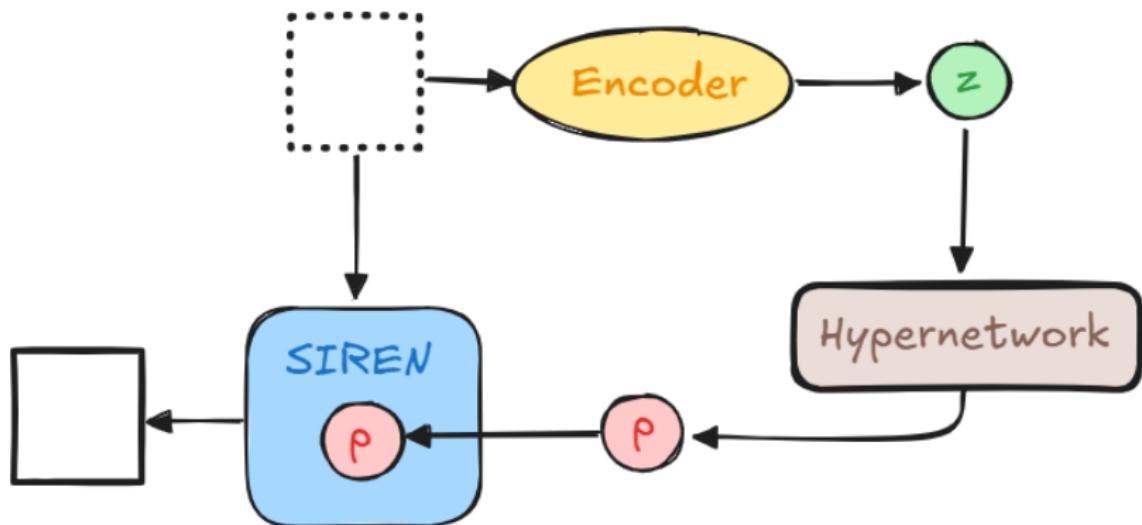
1. Inpainting sur un ensemble d'images

Problème : si on entraîne un seul réseau à compléter plusieurs images :

- Soit le réseau est trop petit et il apprend à compléter une image "moyenne" (pas assez de mémoire);
- Soit le réseau est très volumineux.

2. Inpainting sur un ensemble d'images

Solution : apprendre un espace de fonctions implicites.



3. Inpainting sur un ensemble d'images

Fonction de perte :

$$\frac{1}{H \times W} \underbrace{\|\mathcal{H}(\mathcal{C}(b)) - B\|_2^2}_{\text{vraisemblance}} + \underbrace{\lambda_1 \frac{1}{k} \|z\|_2^2}_{\text{régularisation de } z} + \underbrace{\lambda_2 \frac{1}{l} \|p\|_2^2}_{\text{régularisation de } p}$$

H, W : hauteur, largeur de l'image ;

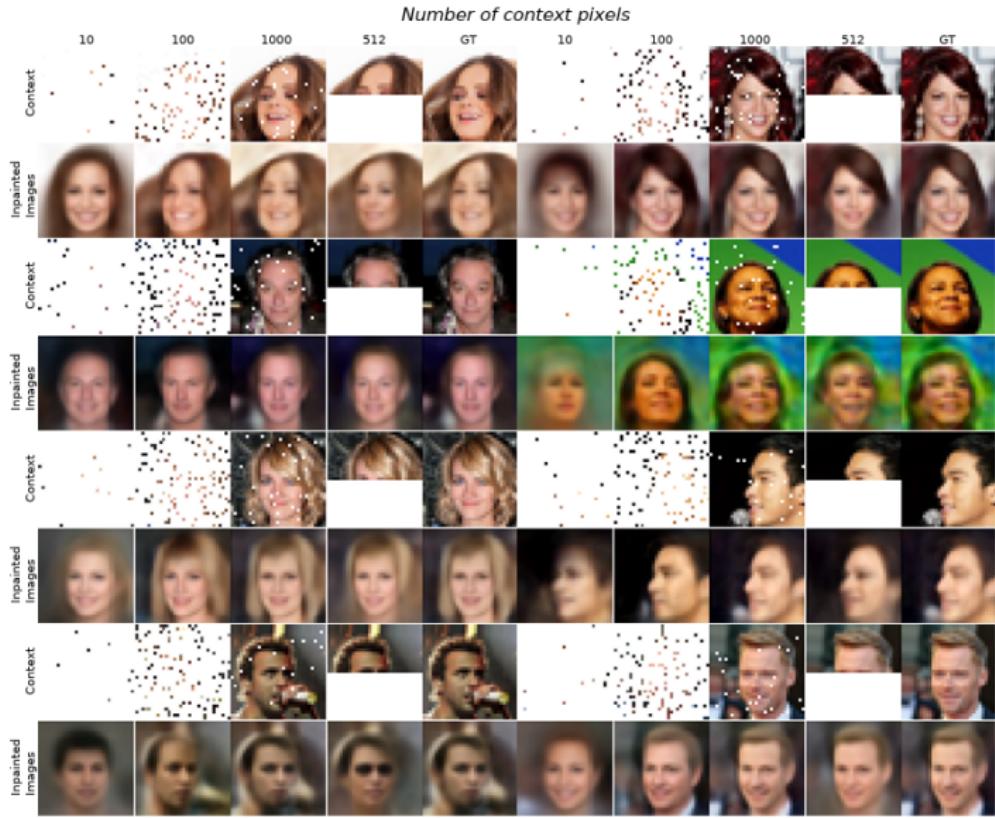
\mathcal{H}, \mathcal{C} : hyper-réseau, encodeur ;

B, b : image originale, bruitée ;

z, k : code latent et sa dimension ;

p, l : vecteur des poids et sa dimension.

4. Inpainting sur un ensemble d'images



5. Inpainting sur un ensemble d'images

Performances des encodeurs : MSE par pixel.

Number of Context Pixels	10	100	1000	512 (Half)	1024
CNP [16]	0.039	0.016	0.009	-	-
Sine Set Encoder + Hypernet.	0.035	0.013	0.009	0.022	0.009
ReLU Set Encoder + Hypernet.	0.040	0.018	0.012	0.026	0.012
PConv CNN Encoder + Hypernet.	0.046	0.020	0.018	0.060	0.019
CNN Encoder + Hypernet.	0.033	0.009	0.008	0.020	0.008

Quand les SIRENs ne marchent pas...

- Difficultés avec les données non-lisses (graphes, langage naturel) ;
- Overfit rapidement les hautes fréquences sans précautions ;
- Inefficace pour la classification ;
- Choix des hyperparamètres (ω_0 , λ) ;
- Difficultés pour extrapoler au-delà de leur domaine.

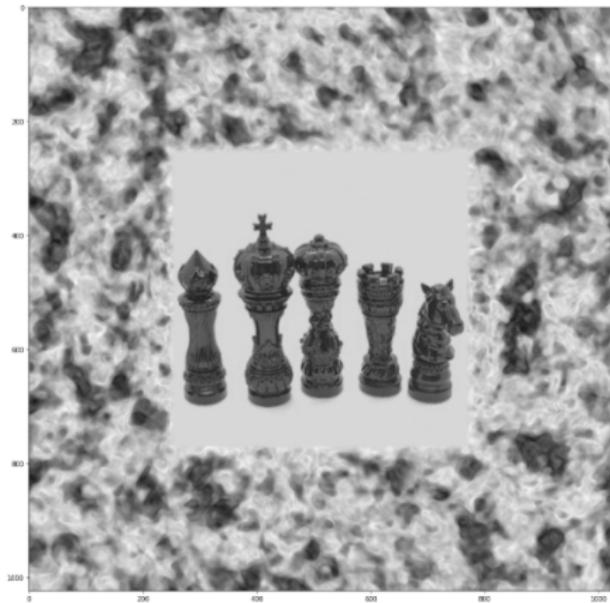
Exemple : deep learning challenge !

Quant les SIRENs ne marchent pas...

Interpolation $\backslash[-0.5, 0.5]$



Extrapolation $\backslash[-2, 2]$



Conclusion

Expérimenter !

- Changer les fonctions d'activations (exponentielles, densités de probabilités, splines, sinus tronqué...) ;
- Changer les produits scalaires (normes, puissances de produits scalaires...).

