
Deep Image Prior & Neural Fields

Representing images by neural networks

Supervised Learning

- Empirical distribution of training data $(x, y) \sim \hat{p}(x, y)$

$$(\mathbb{X}, \mathbb{Y}) = \{(x^{(1)}, y^{(1)}), \dots, (x^{(m)}, y^{(m)})\}$$

$$\begin{aligned}\boldsymbol{\psi}_{ML} &= \arg \max_{\boldsymbol{\psi}} q(\mathbb{Y} | \mathbb{X}, \boldsymbol{\psi}) \\ &= \arg \max_{\boldsymbol{\psi}} \sum_{i=1}^m \log q(y^{(i)} | x^{(i)}, \boldsymbol{\psi})\end{aligned}$$

$$\boldsymbol{\psi}_{ML} = \arg \max_{\boldsymbol{\psi}} \mathbb{E}_{(x, y) \sim \hat{p}(x, y)} \log q(y | x, \boldsymbol{\psi})$$

Supervised Learning

- Training data: $(x, y) \sim \hat{p}(x, y)$
- Choose distribution $q(y|x; \psi)$
- NN represents a function $f_\theta(x)$, which is not a direct prediction of y , but of ψ (e.g. mean)
- ML principle give us the cost function as
 - $-\log q(y|x; f_\theta(x))$

Example – Gaussian Model

- Training data distribution: $p(x, y) = p(y|x)p(x)$
- Estimator of mean y : $f(x)$
- Model distribution: $q(y|x) \equiv N(y|f(x), \sigma^2)$

$$\hat{f}(x) = \arg \min_{f(x)} \mathbb{E}_{p(x,y)}[-\log q(y|f(x))]$$

$$L = \int \left[\int (y - f(x))^2 p(y|x) dy \right] p(x) dx + c$$

$$\frac{\partial L}{\partial f} \propto \int (y - f(x)) p(y|x) dy = 0$$

$$\hat{f}(x) = \int y p(y|x) dy = \mathbb{E}_{p(y|x)}[y]$$

Example – Laplace Model

- Training data distribution: $p(x, y) = p(y|x)p(x)$
- Estimator of mean y : $f(x)$
- Model distribution: $q(y|x) \equiv \text{Laplace}(y|f(x), \gamma)$

$$\hat{f}(x) = \arg \min_{f(x)} \mathbb{E}_{p(x,y)}[-\log q(y|f(x))]$$

$$L = \int \left[\int |y - f(x)| p(y|x) dy \right] p(x) dx + c$$

$$\frac{\partial L}{\partial f} \propto \int \text{sign}(y - f(x)) p(y|x) dy = 0$$

$$\hat{f}(x) = m \quad \dots \text{median: } P(y \leq m|x) = P(y \geq m|x)$$

Deep Unrolling

- Sparse coding:

$$\min_{\mathbf{x}} \frac{1}{2} \|\mathbf{W}\mathbf{x} - \mathbf{y}\|_2^2 + \lambda \|\mathbf{x}\|_1$$

- ISTA:

$$\mathbf{x}_{k+1} = S_{\lambda/\mu} \left(\frac{1}{\mu} \mathbf{W}^T (\mathbf{y} - \mathbf{W}\mathbf{x}_k) + \mathbf{x}_k \right)$$

Deep Unrolling of ISTA (LISTA)

Iterative Shrinkage and Thresholding Algorithm

Algorithm: Input x^0 , Output x^L

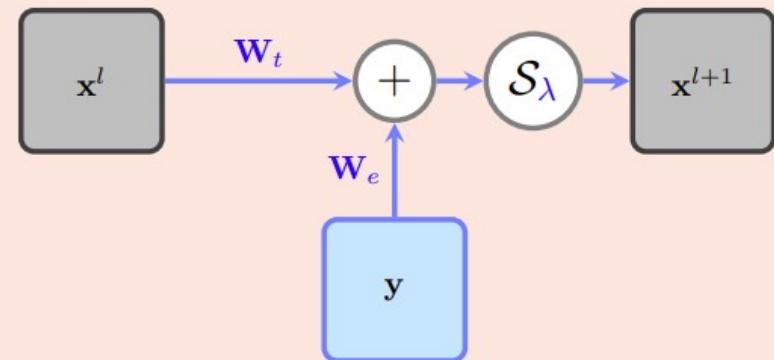
for $l = 0, 1, \dots, L - 1$ **do**

$$x^{l+1} = \mathcal{S}_\lambda \left(\left(I - \frac{1}{\mu} \mathbf{W}^T \mathbf{W} \right) x^l + \frac{1}{\mu} \mathbf{W}^T \mathbf{y} \right)$$

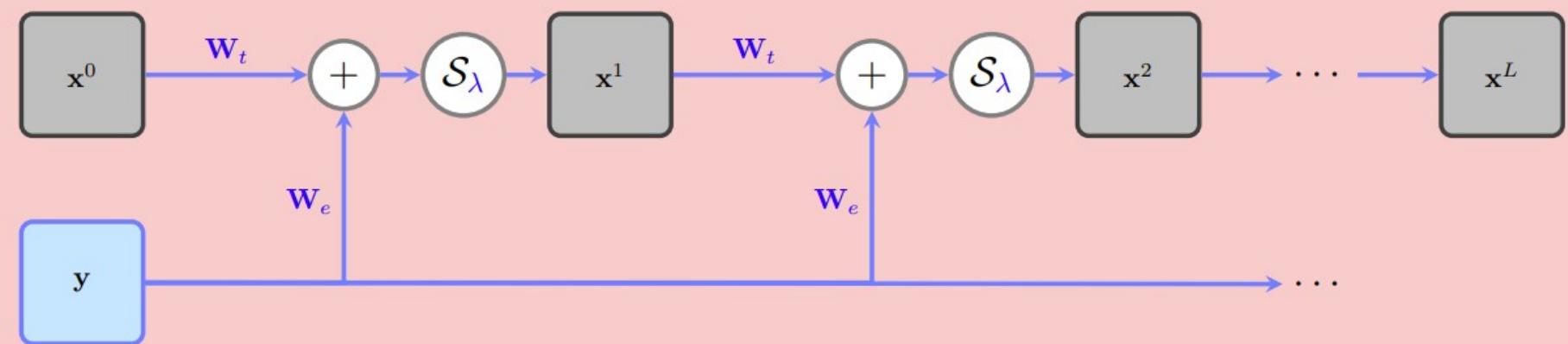
end for

$$\mathbf{W}_t = I - \frac{1}{\mu} \mathbf{W}^T \mathbf{W}$$
$$\mathbf{W}_e = \frac{1}{\mu} \mathbf{W}^T$$

A Single Network Layer

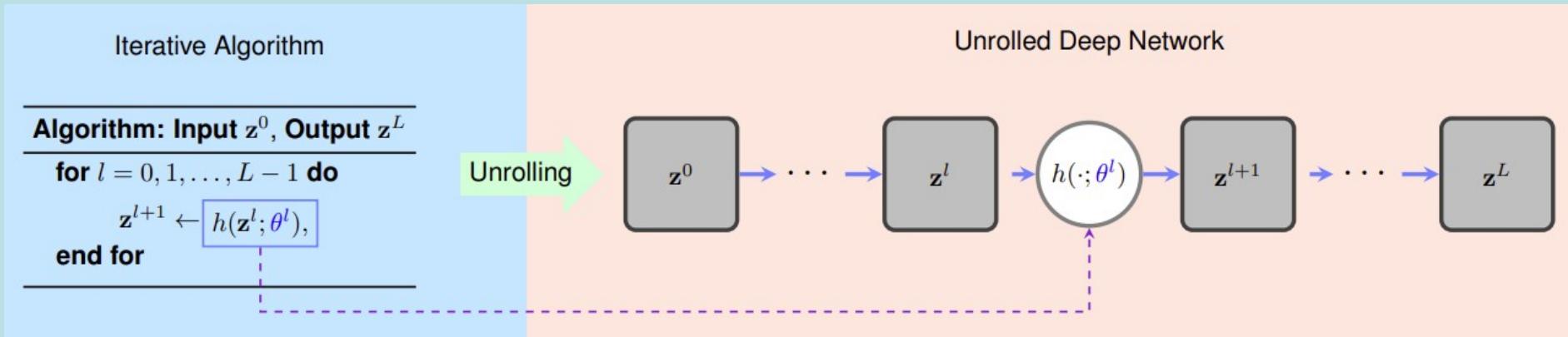


Unrolled Deep Network



Deep Unrolling

- General idea



ADMM - example

- Deconvolution with TV regularization

$$\min_{\mathbf{x}} (1/2) \|\mathbf{Hx} - \mathbf{g}\|_2^2 + \lambda \|\mathbf{Dx}\|_1$$

- Augmented Lagrangian

$$L(\mathbf{x}, \mathbf{z}, \mathbf{v}) \propto (1/2) \|\mathbf{Hx} - \mathbf{g}\|_2^2 + \lambda \|\mathbf{z}\|_1 + (\rho/2) \|\mathbf{Dx} - \mathbf{z} + \mathbf{v}\|_2^2$$

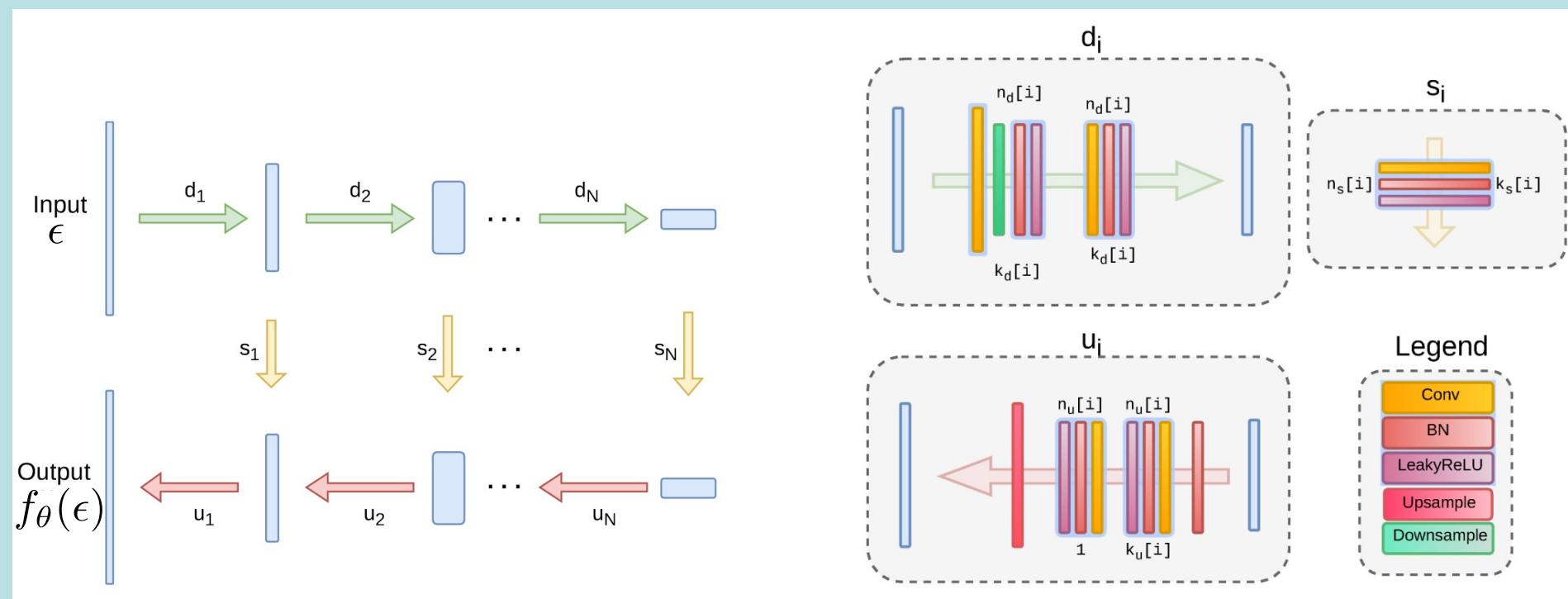
- ADMM

- 1) $\mathbf{x} \leftarrow \arg \min_{\mathbf{x}} L(\mathbf{x}, \mathbf{z}, \mathbf{v})$ System of linear equations (CG):
$$\mathbf{x} \leftarrow (\mathbf{H}^T \mathbf{H} + \rho \mathbf{D}^T \mathbf{D})^{-1} (\mathbf{H}^T \mathbf{g} + \rho \mathbf{D}^T (\mathbf{z} - \mathbf{v}))$$
- 2) $\mathbf{z} \leftarrow \arg \min_{\mathbf{z}} L(\mathbf{x}, \mathbf{z}, \mathbf{v})$ Proximal operator (soft-thresholding)
$$\mathbf{z} \leftarrow S_{\lambda/\rho}(\mathbf{Dx} + \mathbf{v})$$
- 3) $\mathbf{v} \leftarrow \mathbf{v} + (\mathbf{Dx} - \mathbf{z})$

Deep Image Prior

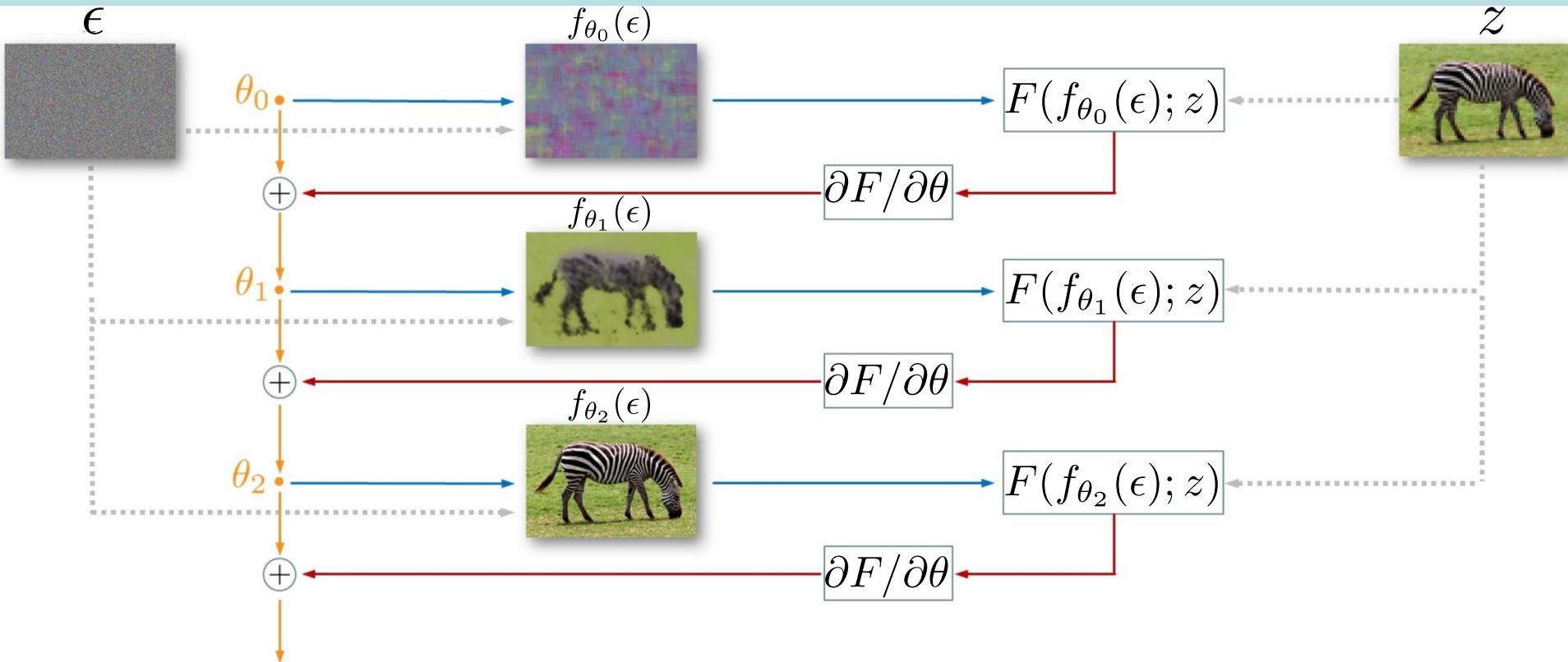
- Denoising $z = u + n \quad \min_u F(u; z)$
- no prior $F(u; z) = \|u - z\|^2$
- with prior $F(u; z) = \|u - z\|^2 + R(u)$
- with Deep Image Prior $\min_\theta F(f_\theta(\epsilon); z)$
 $F(f_\theta(\epsilon); z) = \|f_\theta(\epsilon) - z\|^2$

- $f_\theta(\epsilon)$ is U-Net architecture

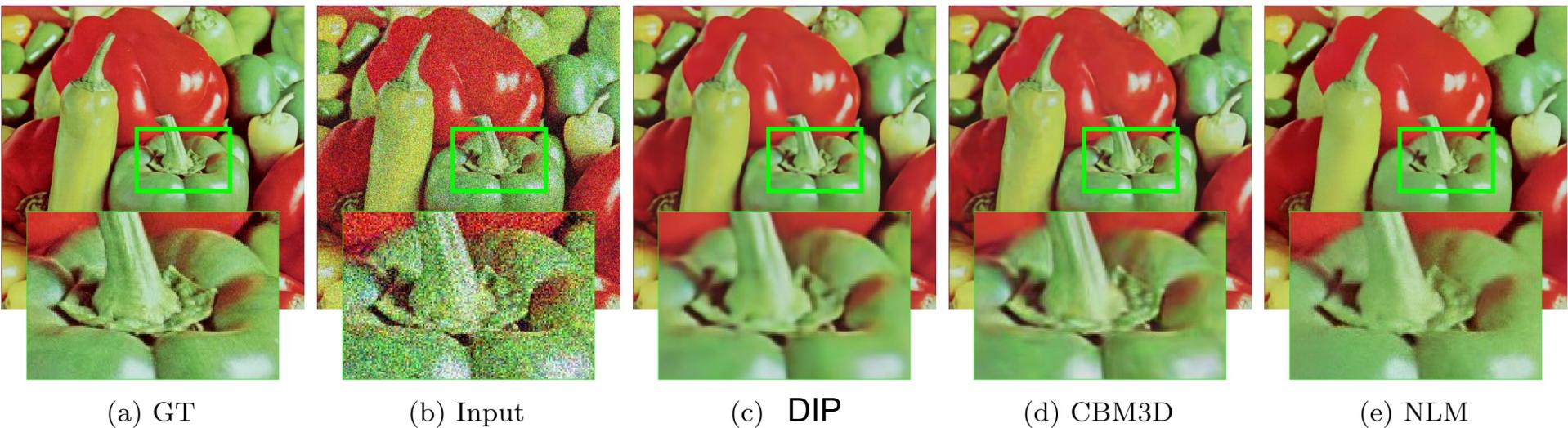


Zero-shot learning

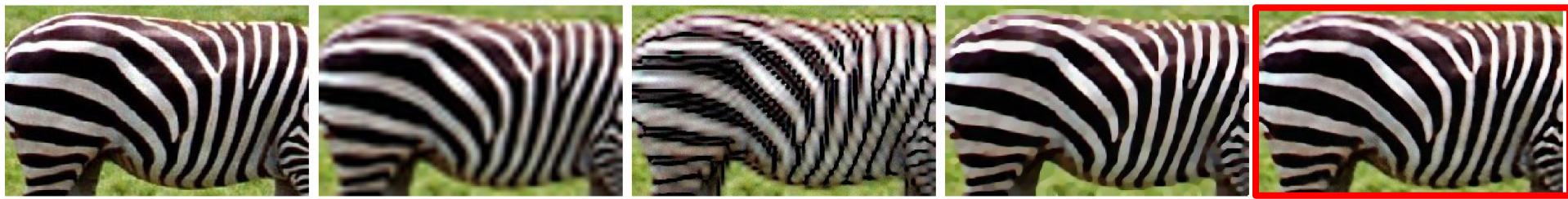
- Gradient descent



Denoising example

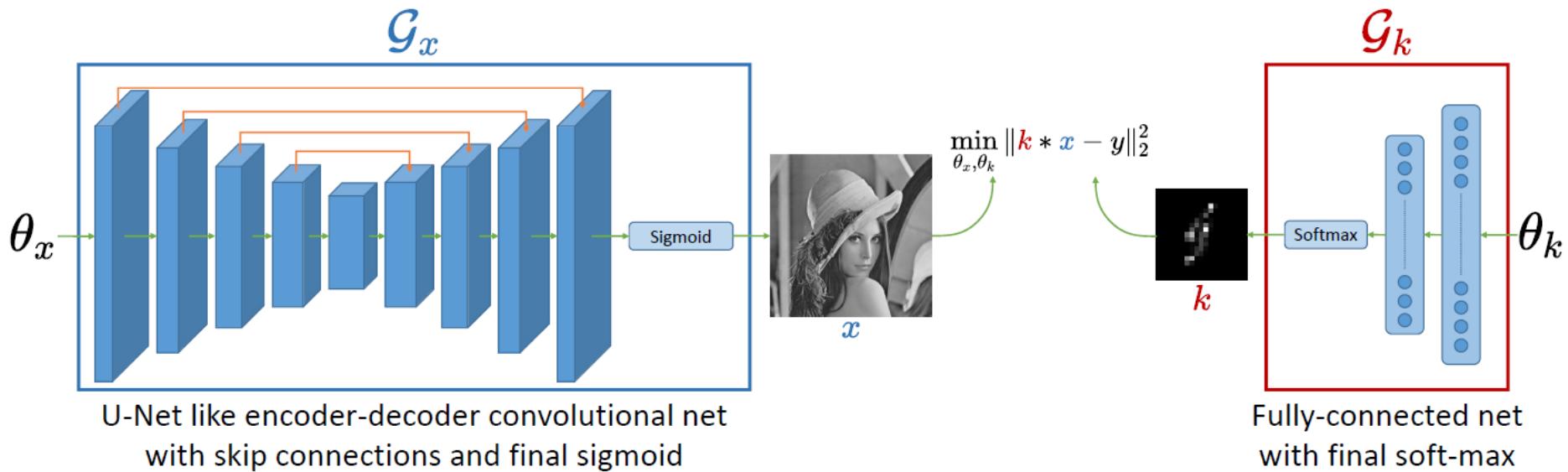


Super-resolution example



Blind Deconvolution

$$\min_{\theta_x, \theta_k} \|k * x - y\|_2^2, \quad \text{s.t. } 0 \leq x \leq 1 \text{ and } 0 \leq k, \|k\|_1 = 1, \quad \text{where } x = \mathcal{G}_x(\theta_x), k = \mathcal{G}_k(\theta_k)$$



Implicit Neural Representation

- also called Neural Fields
- represent an image(object) function with MLP

$$f(x) : \mathbb{R}^m \rightarrow \mathbb{R}^n$$



– e.g. color image $f(x) : \mathbb{R}^2 \rightarrow \mathbb{R}^3$

Zero-shot learning



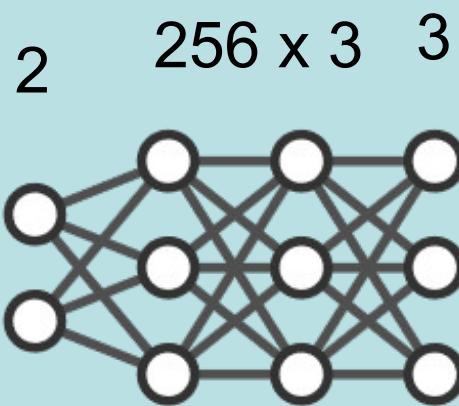
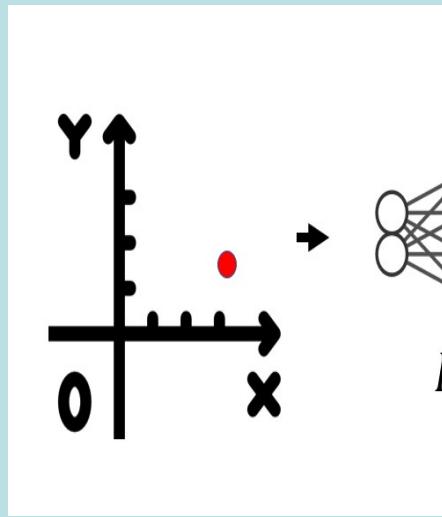
- e.g. fitting image $u(x)$:

$$\min_{\theta} \sum_i \|f_\theta(x_i) - u(x_i)\|^2$$

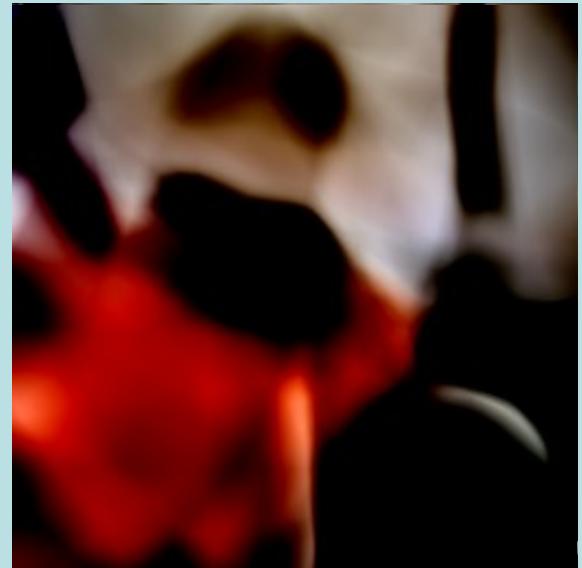
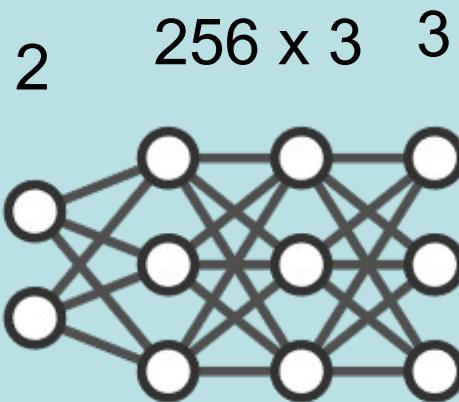
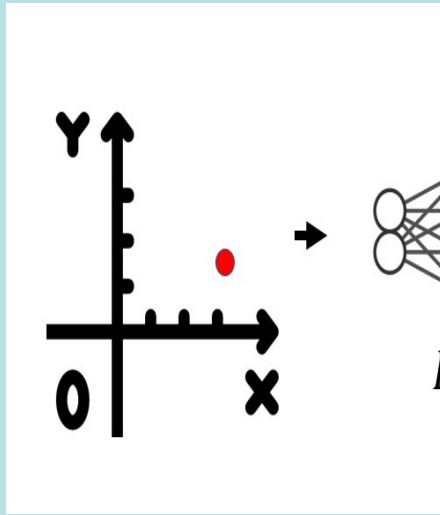
- or more complex losses

Neural Fields

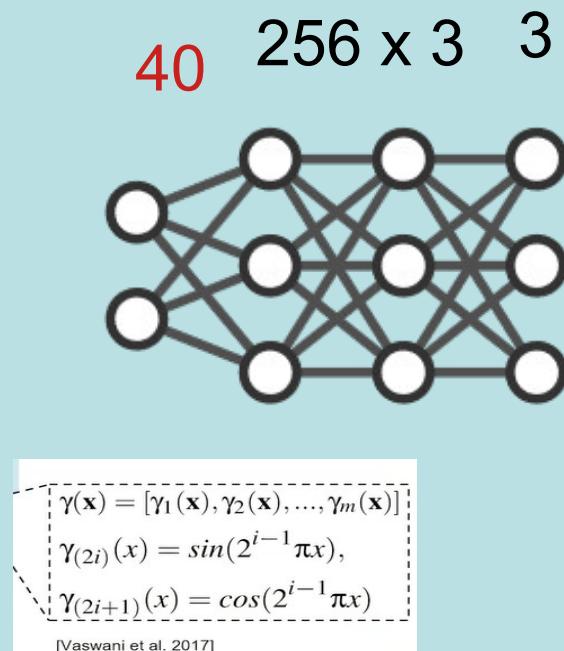
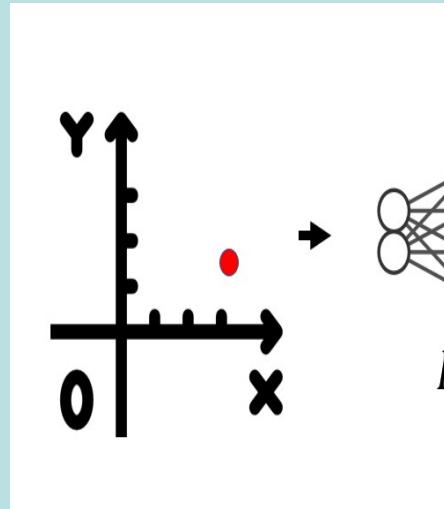
- Pros:
 - Compressed representation
 - Analytic derivatives $\nabla f(x)$ using back-propagation
- Cons:
 - train for every case



- ReLU



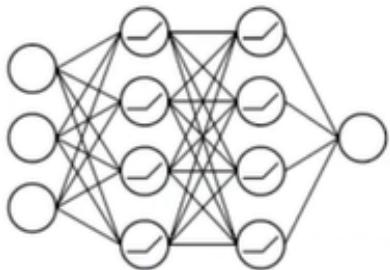
• Positional encoding



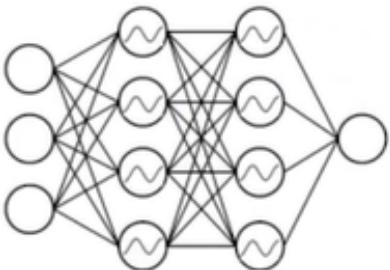
SIREN

A) Network Φ

ReLU



SIREN



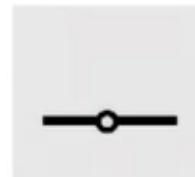
[Sitzmann et al. 2021]

Derivatives

1st (Φ')

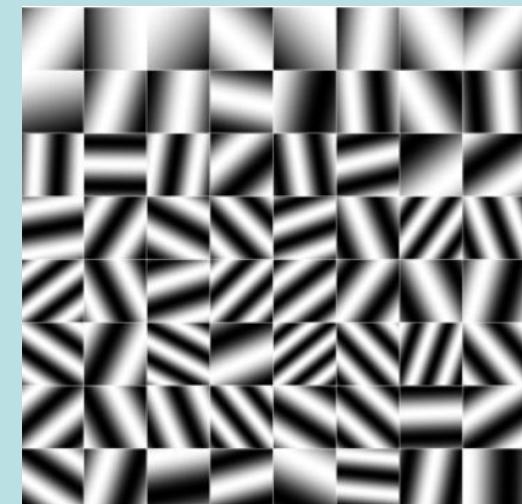


2nd (Φ'')

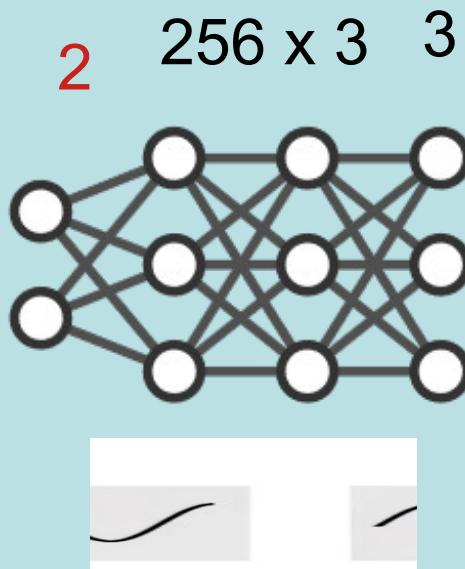
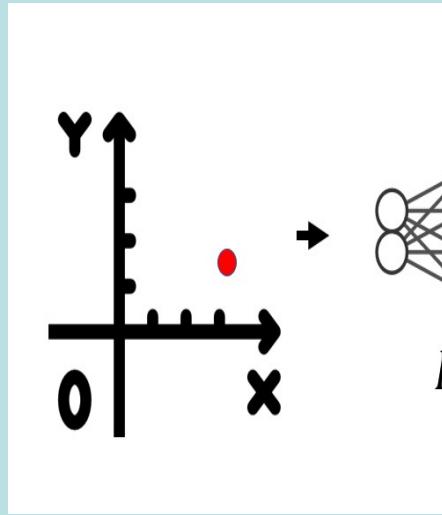


$\text{ReLU}(\mathbf{Wx} + \mathbf{b})$

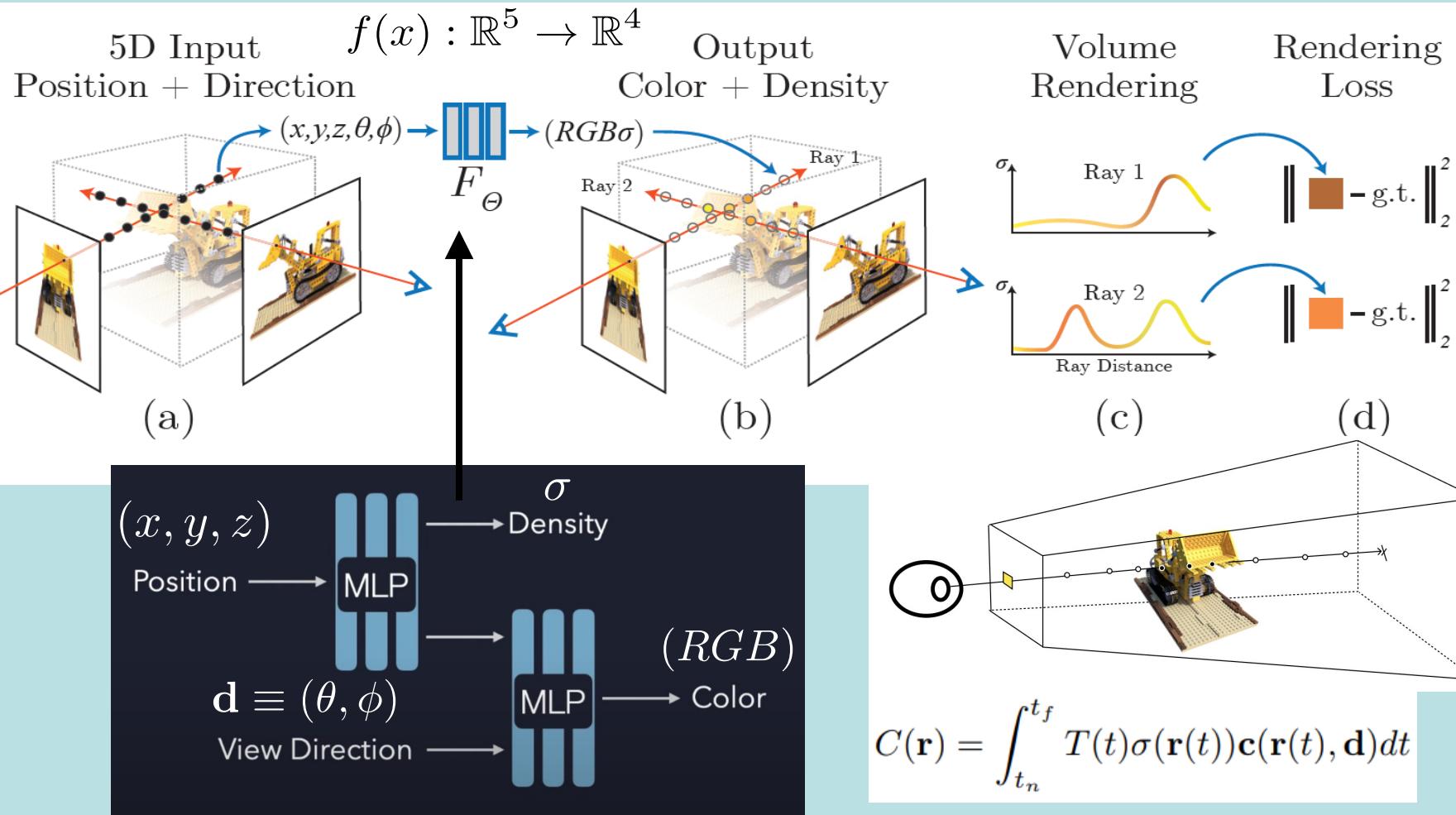
$\sin(\mathbf{Wx} + \mathbf{b})$



- Sinus activation



NeRF



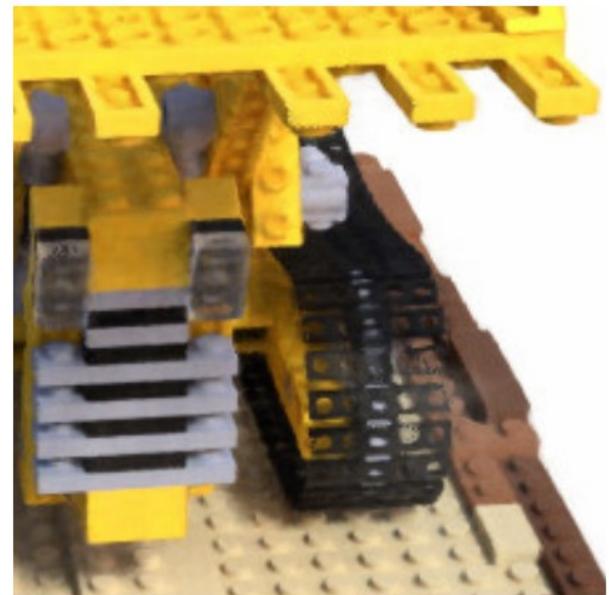
NeRF: arxiv.org/pdf/2003.08934.pdf



Ground Truth



Complete Model



No View Dependence

NVIDIA Instant NeRF



Input Images

NVIDIA NGP
Instant NeRF



CURE

- Video

