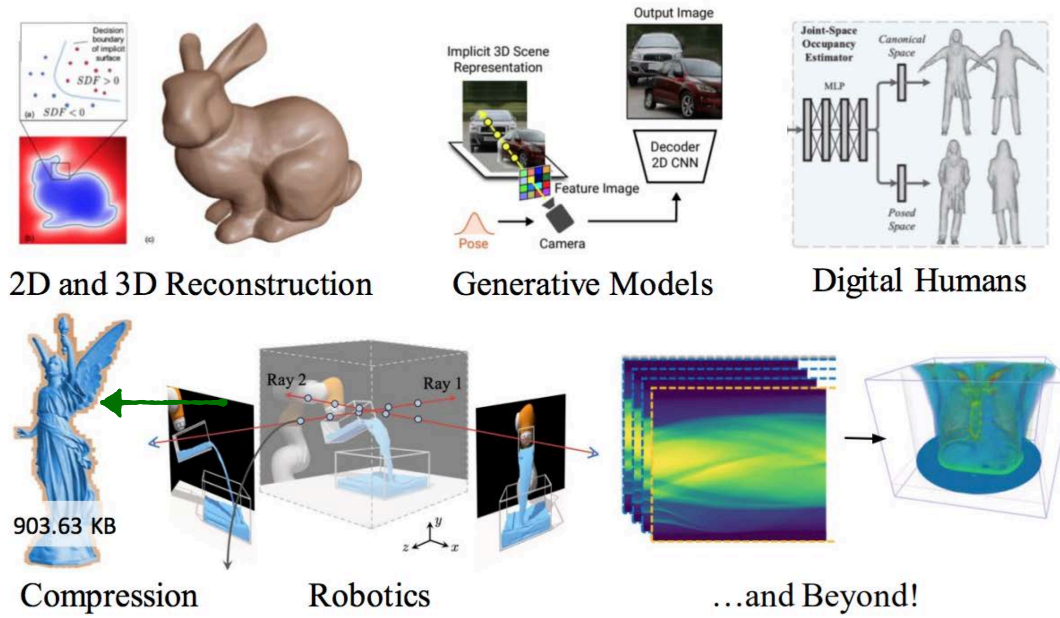
Towards an efficient representation using sinusoidal neural networks

by
Diana Aldana Moreno

Impa
2024

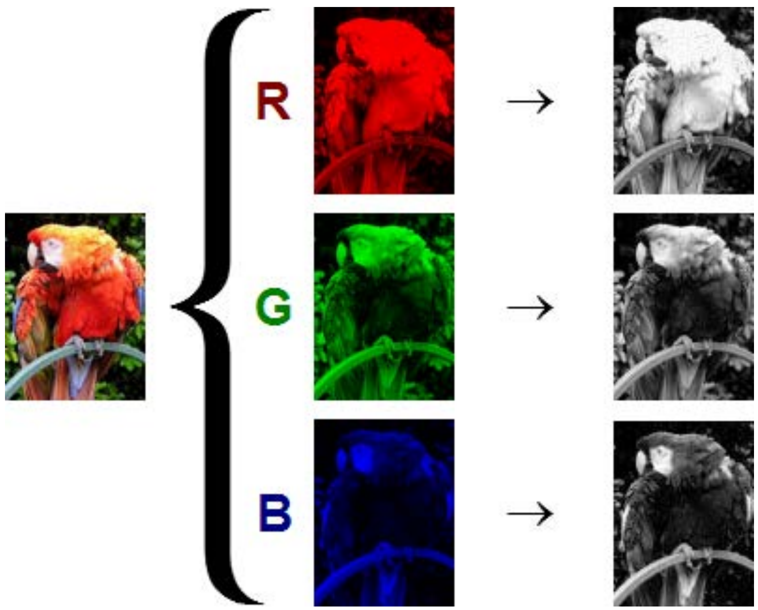
Implicit neural representations (INR)

- Neural fields and beyond website
- Awesome implicit representations Github
- Google scholar
- etc…



Xie, Yiheng, et al. "Neural fields in visual computing and beyond." *Computer Graphics Forum*. Vol. 41. No. 2. 2022.

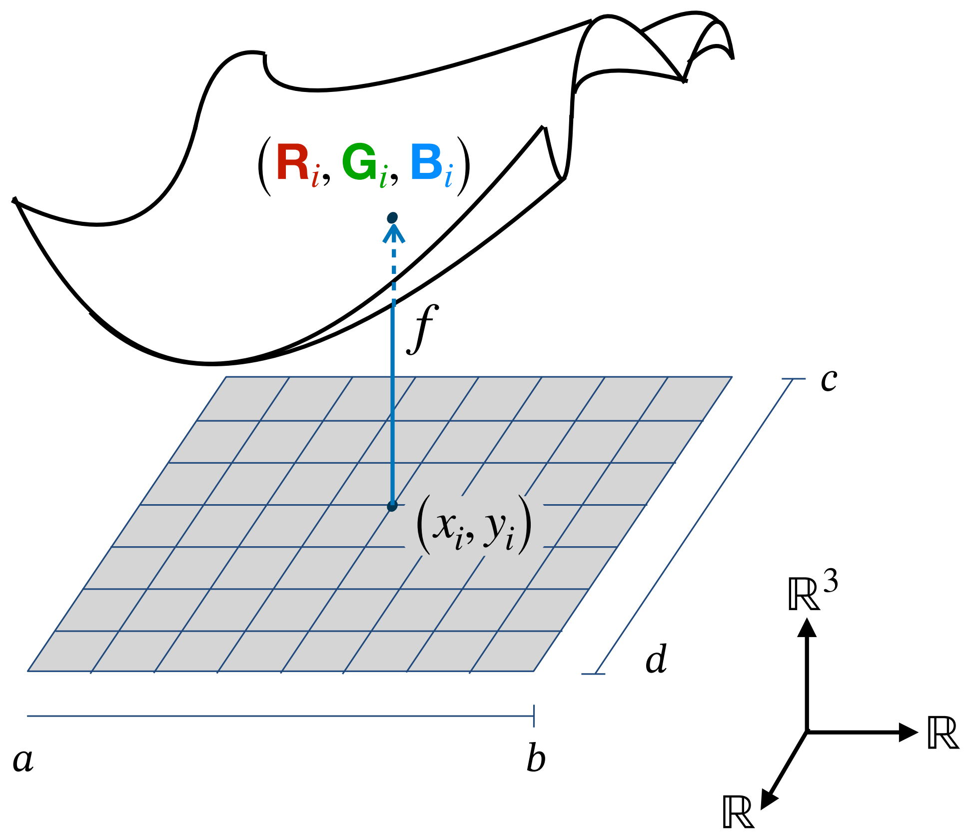
Regression problem



Ground truth f

- Discrete representation
- Grid dependant

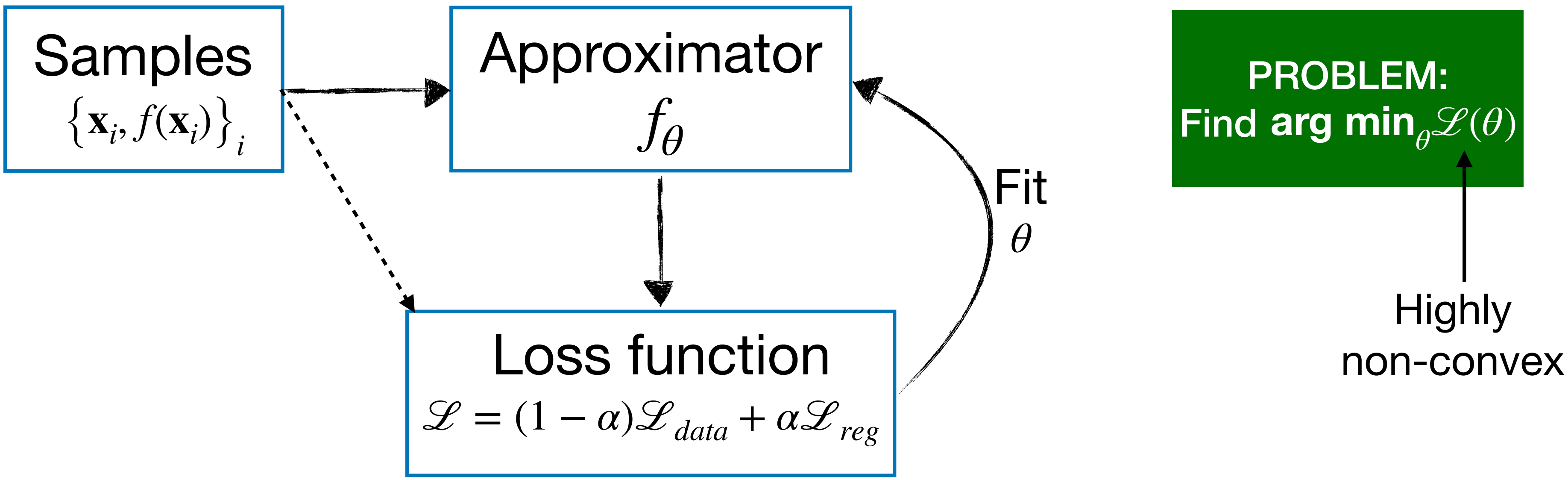
Approximation f_θ



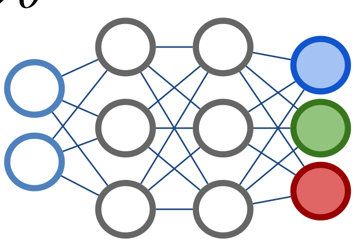
Samples $\{\mathbf{x}_i, f(\mathbf{x}_i)\}_i$

Image taken from: https://en.m.wikipedia.org/wiki/File:RGB_channels_separation.png

Pipeline



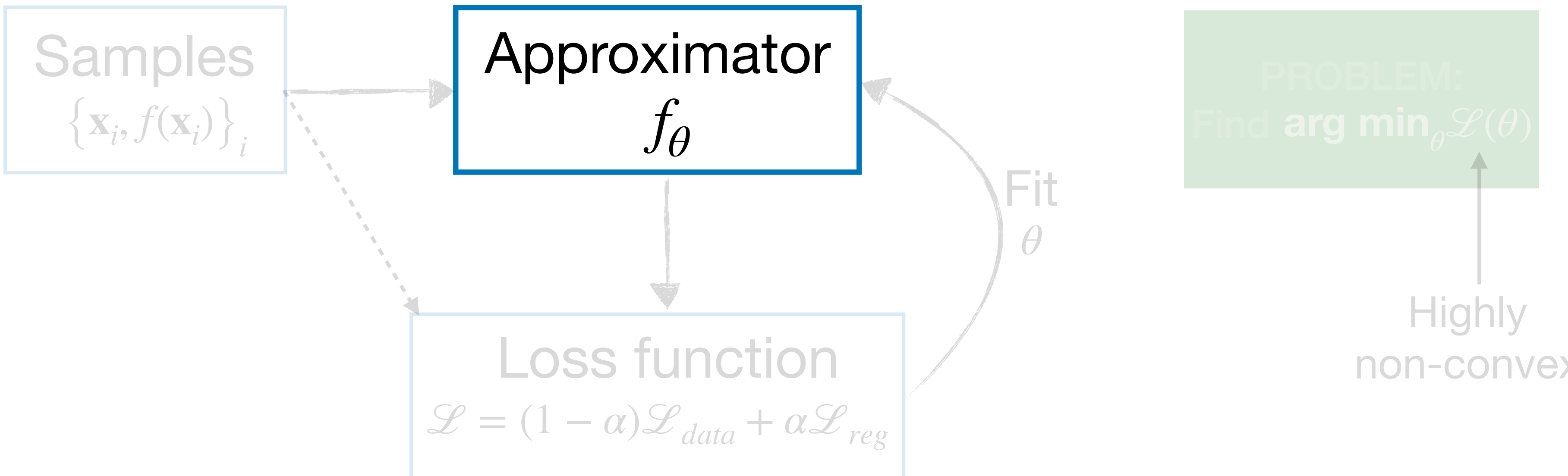
$f_\theta : \mathbb{R}^2 \to \mathbb{R}^3$



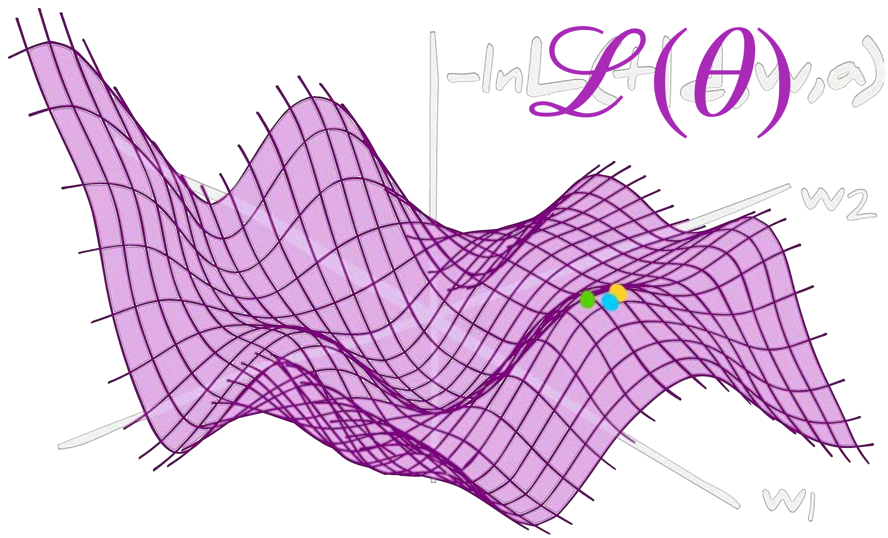
- Sine activation function
- Single hidden layer

- Initialization: θ_0
- $\mathscr{L}_{data}(\theta) = \sum_i \left(f(\mathbf{x}_i) - f_\theta(\mathbf{x}_i)\right)^2$

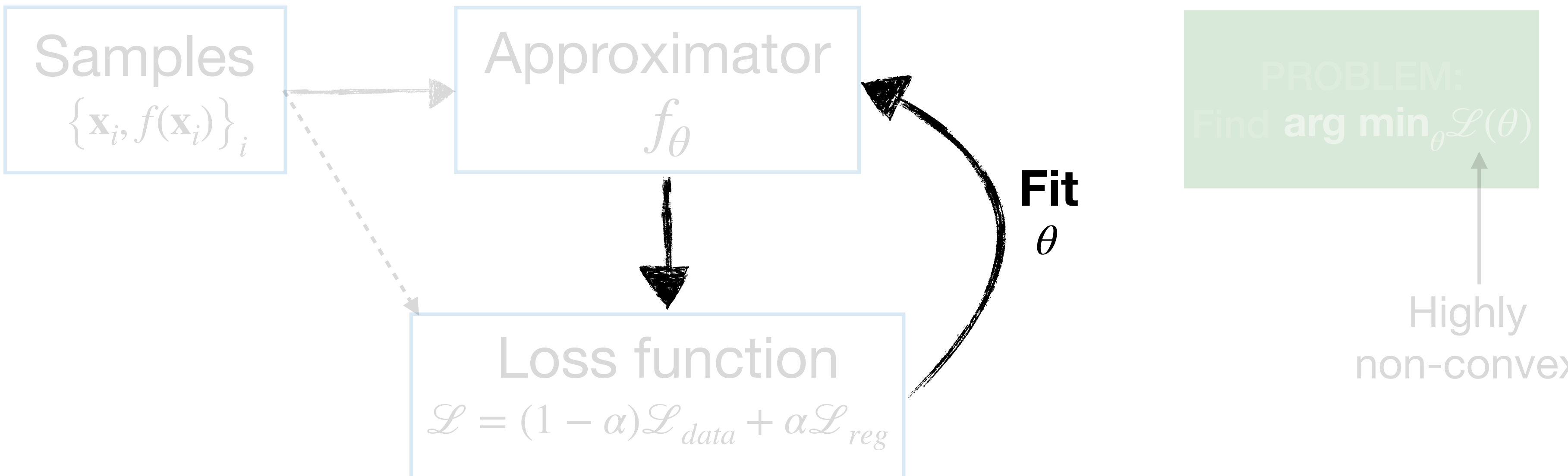
Pipeline



- **Understanding of f_θ**
- **Initialization:**
  - **First layer**
  - **Hidden layer**

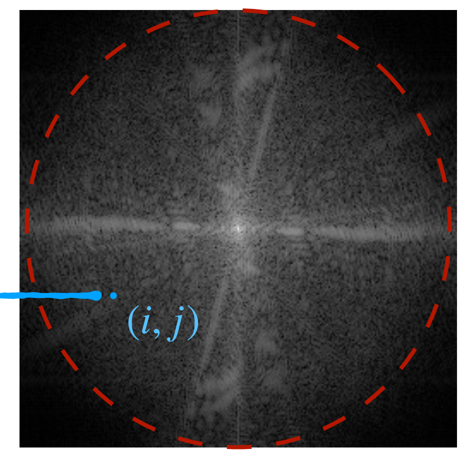


Pipeline

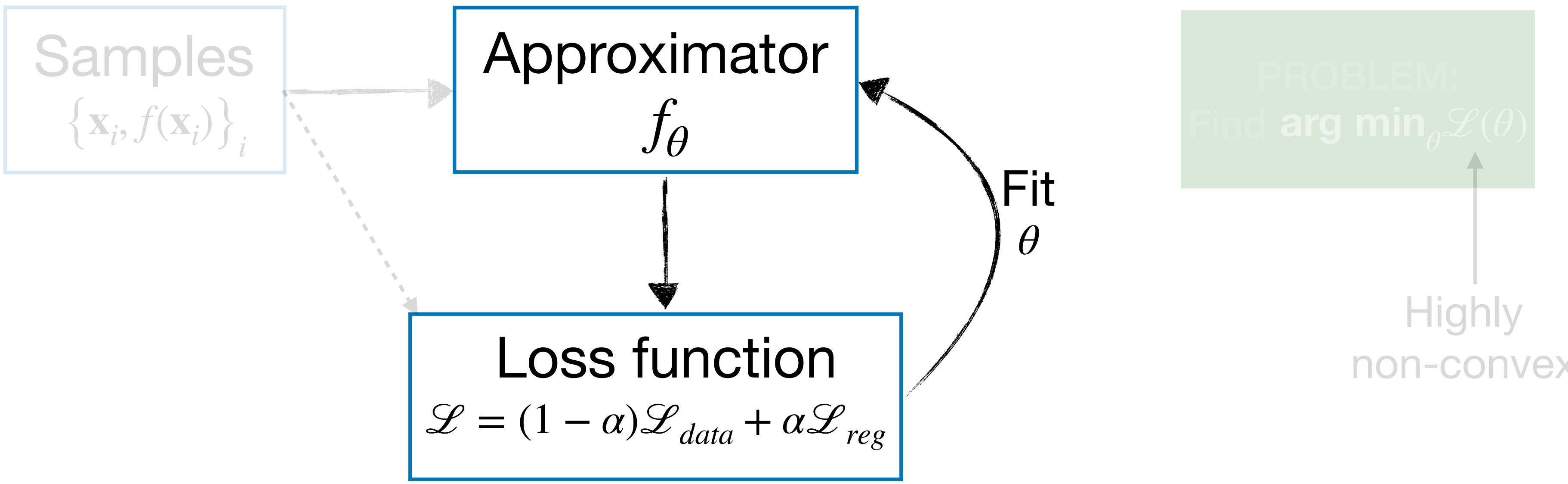


Bound the reconstruction's spectrum

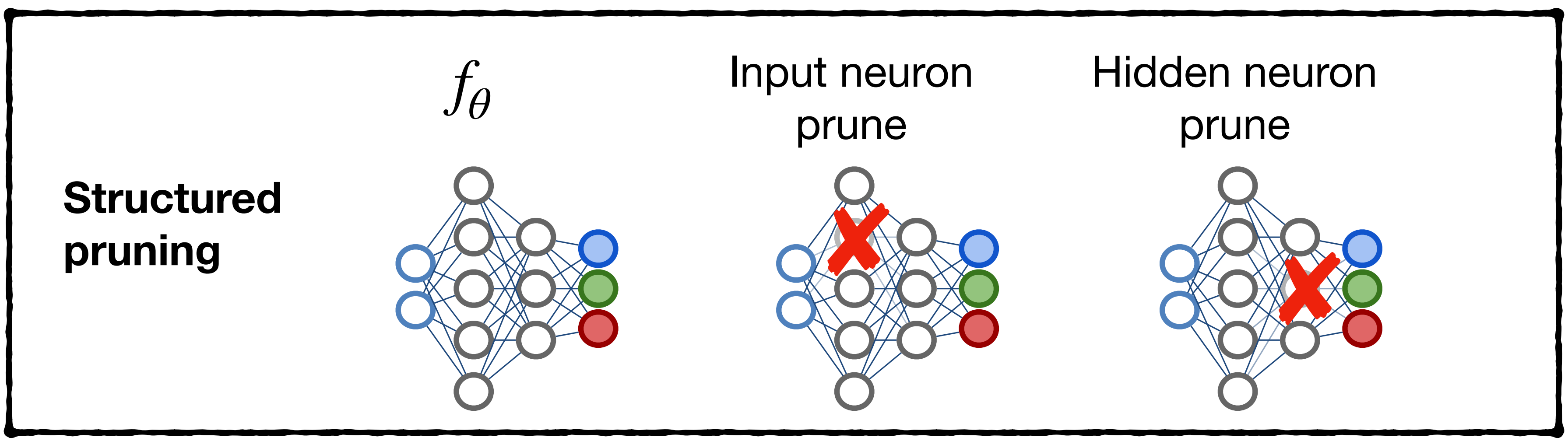
Intensity indicates amplitude of frequency (i, j)



Pipeline

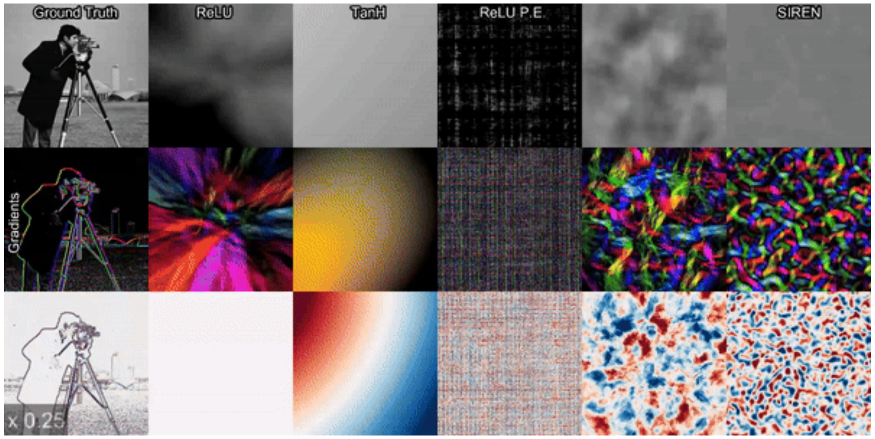


Samples
$\{\mathbf{x}_i, f(\mathbf{x}_i)\}_i$
Approximator
f_θ
Fit
θ
Loss function
$\mathscr{L} = (1-\alpha)\mathscr{L}_{data} + \alpha\mathscr{L}_{reg}$
PROBLEM:
Find **arg min**$_\theta \mathscr{L}(\theta)$
Highly
non-convex



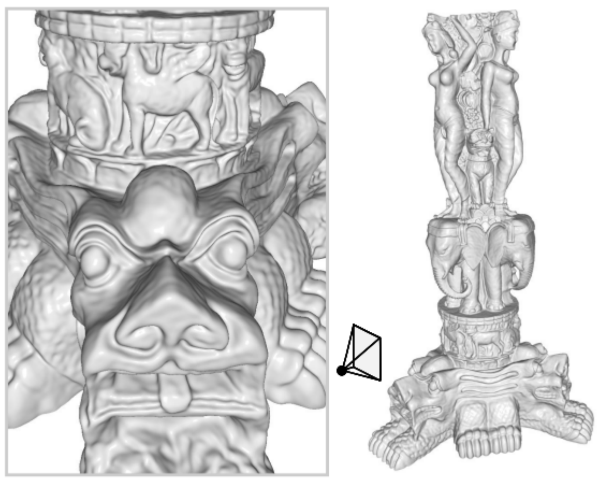
Structured pruning
f_θ
Input neuron prune
Hidden neuron prune

Why sinusoidal INRs?



Fast convergence, highly detailed

Stochastic / stratified sampling



Higher dimension representations

Compatible with different pipelines.

Sitzmann, Vincent, et al. "Implicit neural representations with periodic activation functions." Advances in neural information processing systems 33 (2020): 7462-7473.
Novello, Tiago, et al. "Neural Implicit Surface Evolution." Proceedings of the IEEE/CVF International Conference on Computer Vision. 2023.
Schardong, Guilherme, et al. "Neural Implicit Morphing of Face Images." Proceedings of the IEEE / CVF Computer Vision and Pattern Recognition Conference. 2024.

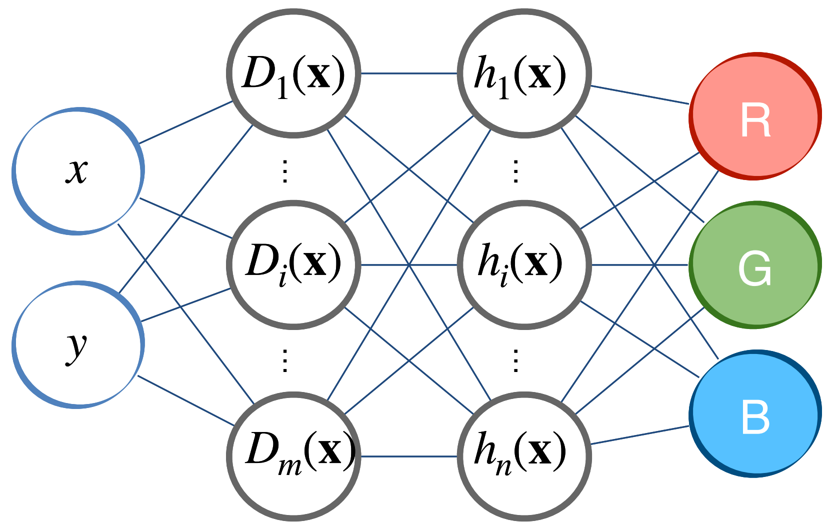
Goals

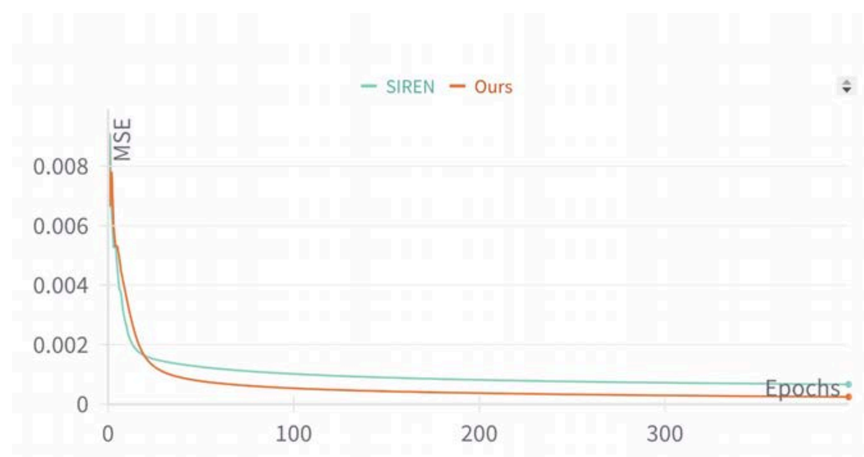
- Understand the training of sinusoidal INRs
- Control the generated frequencies (noise)
- Find an adequate size for f_θ
- Speed up training

m? n? $m \gg n$? $n \gg m$?

$$f_\theta$$

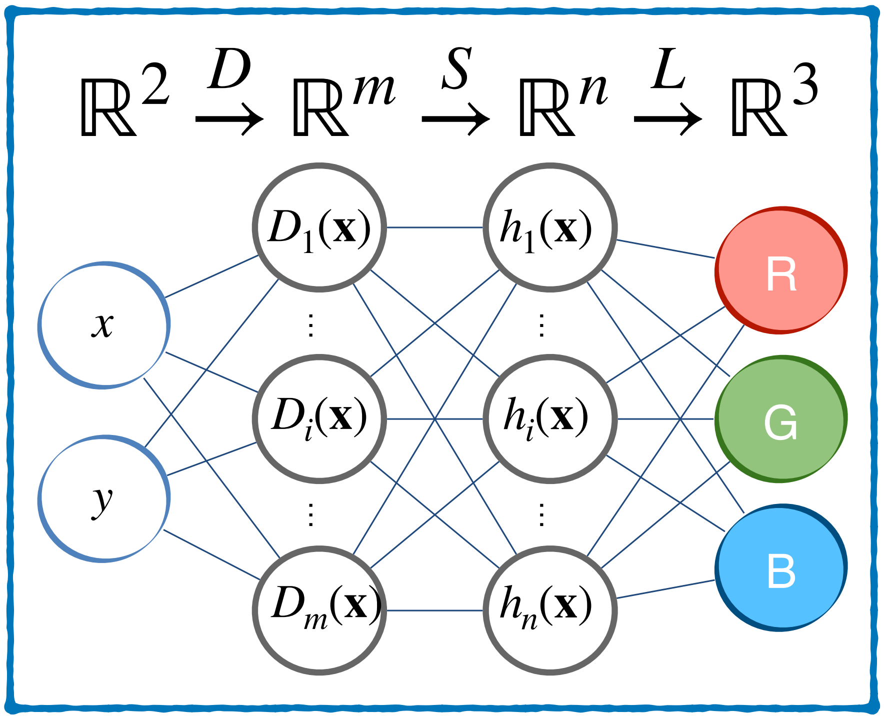
$$\mathbb{R}^2 \xrightarrow{D} \mathbb{R}^m \xrightarrow{S} \mathbb{R}^n \xrightarrow{L} \mathbb{R}^3$$





Sinusoidal INR's structure

f_θ



$$D(\mathbf{x}) = \left[\sin(\omega_j \mathbf{x} + \varphi_j)\right]_j$$

$$S(\mathbf{x}) = \left[\sin\left(\mathbf{W}_i \mathbf{x} + \mathbf{b}_i\right)\right]_i$$

$$L(\mathbf{x}) = \langle \mathbf{C}, \mathbf{x} \rangle + \mathbf{d}$$

$$f_\theta(\mathbf{x}) = L \circ S \circ D(\mathbf{x})$$

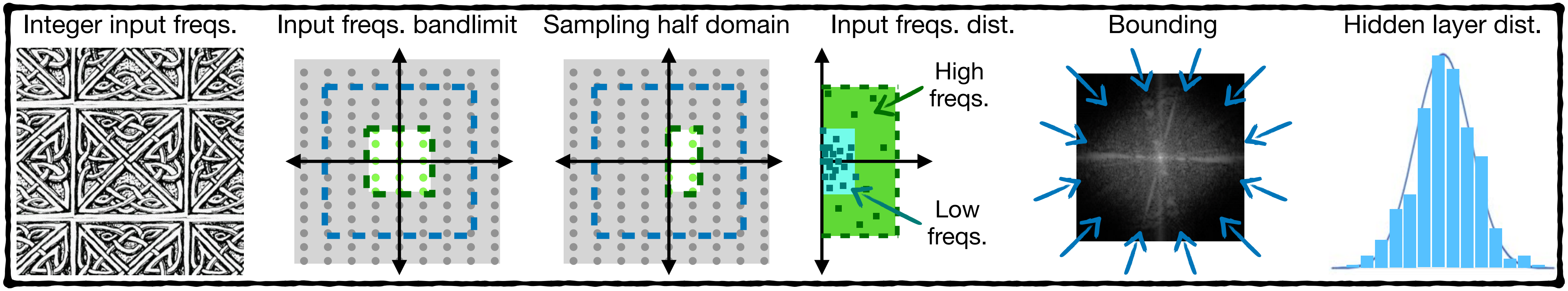
$$S \circ D(\mathbf{x}) = \left[\sum_{\mathbf{k} \in \mathbb{Z}^m} \alpha_{\mathbf{k}}\left(\mathbf{W}_i\right) \sin\left(\beta_{\mathbf{k}}(\omega)\, \mathbf{x} + \lambda_{\mathbf{k}}\right)\right]_i$$

Each canal of f_θ is a sum of infinite sines

The frequencies $\beta_{\mathbf{k}}(\omega) = \sum_{j=1}^{m} k_j \omega_j$ are integer combinations of ω.

Novello, Tiago. "Understanding sinusoidal neural networks." *arXiv preprint arXiv:2212.01833* (2022).

Initialization

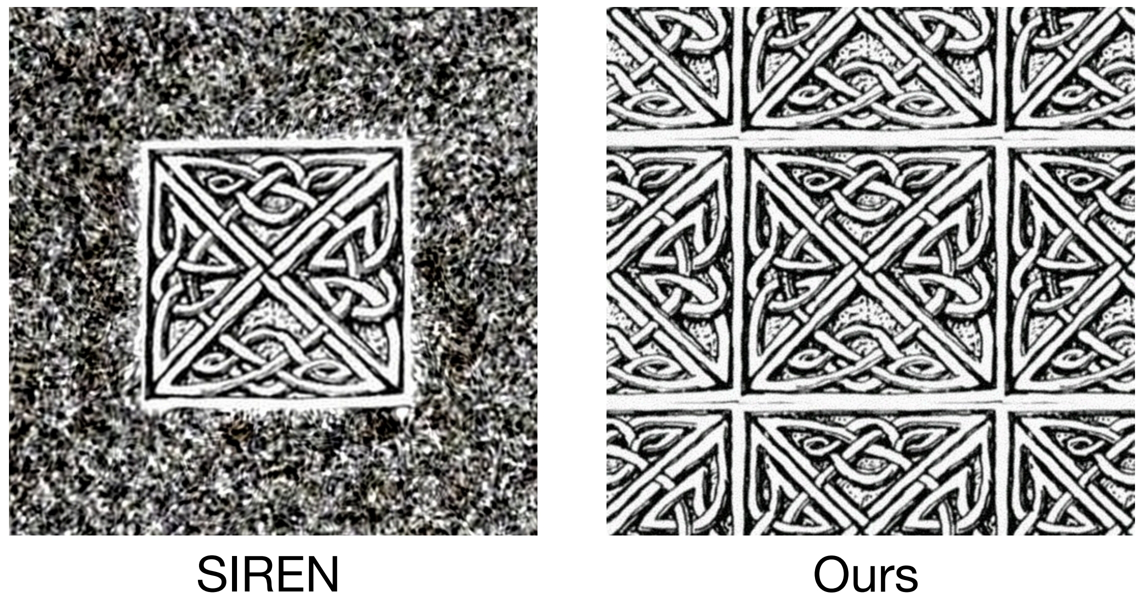


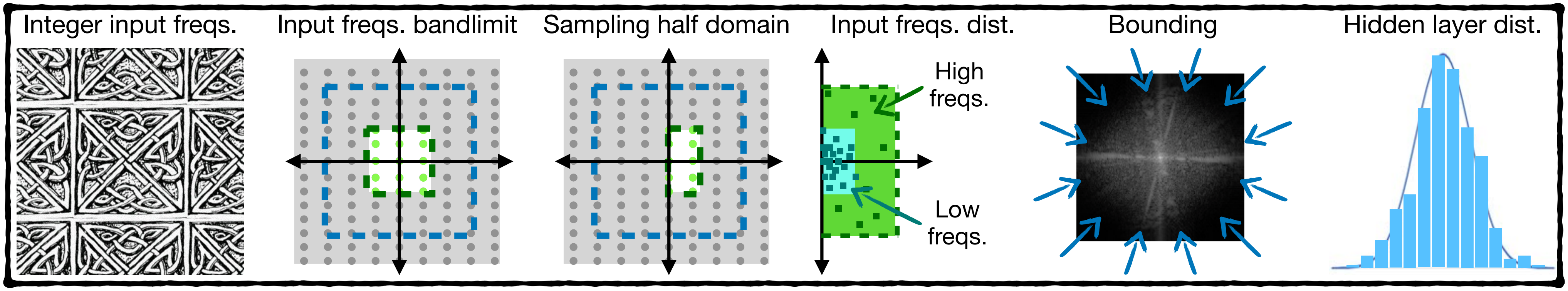
Generated frequencies $\beta_{\mathbf{k}}(\omega) = \sum_{j=1}^{m} k_j \omega_j$ with $k_1, \ldots, k_m$ integers.

$$\omega_i \in \frac{2\pi}{p}\mathbb{Z} \longrightarrow \beta_{\mathbf{k}}(\omega) \in \frac{2\pi}{p}\mathbb{Z}$$

$$f_\theta(\mathbf{x}) = L \circ \underbrace{S \circ D(\mathbf{x})}_{\text{Periodic (period } p)}$$

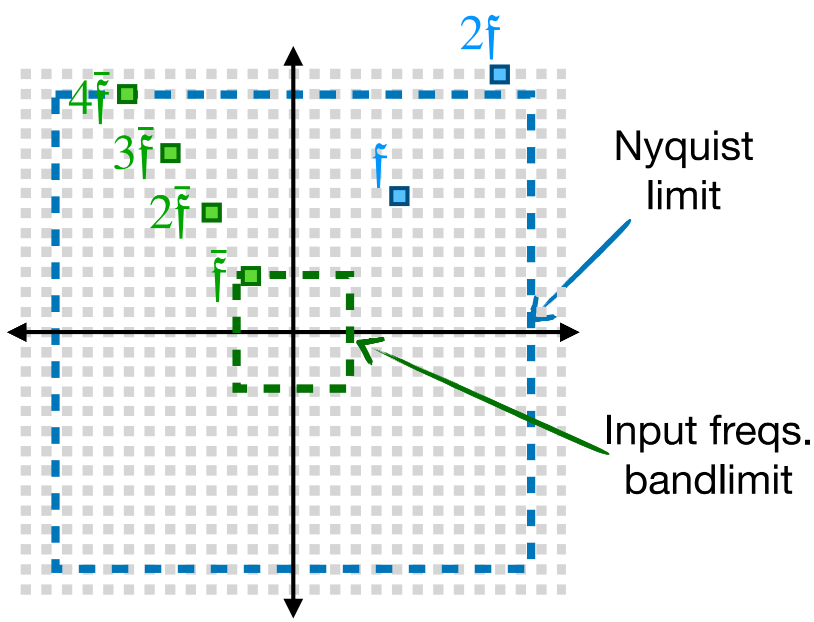
f_θ is p-periodic



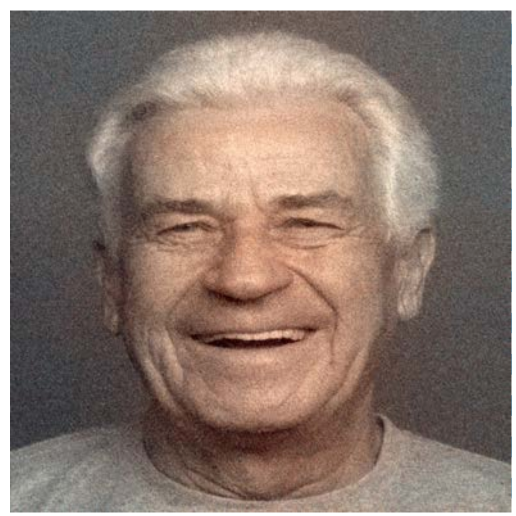


Generated frequencies $\beta_{\mathbf{k}}(\omega) = \sum_{j=1}^{m} k_j \omega_j$ with $k_1, \ldots, k_m$ integers.

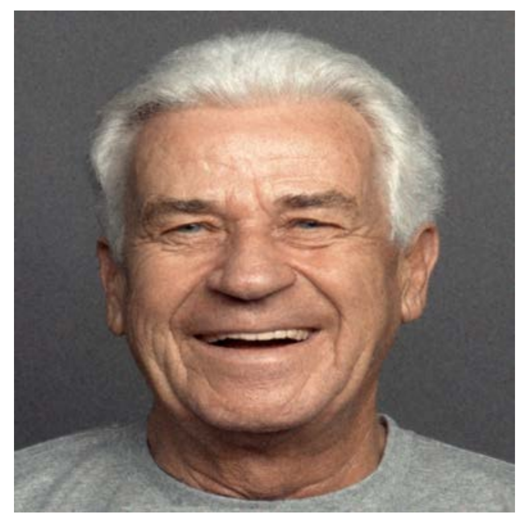
$$\omega = \frac{2\pi}{p}\mathfrak{f} \longrightarrow \beta_k(\omega) = \frac{2\pi}{p}k\mathfrak{f}$$



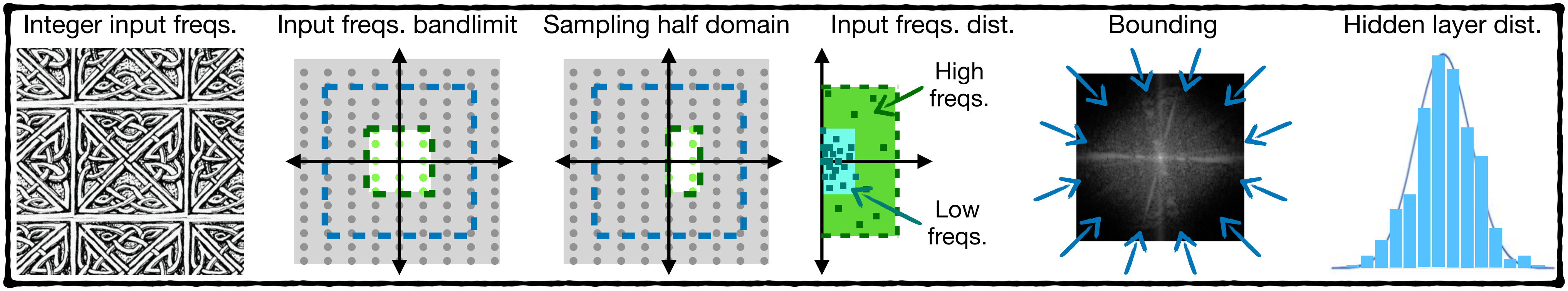
Nyquist limit 512



Maximum freq. 170



Maximum freq. 85



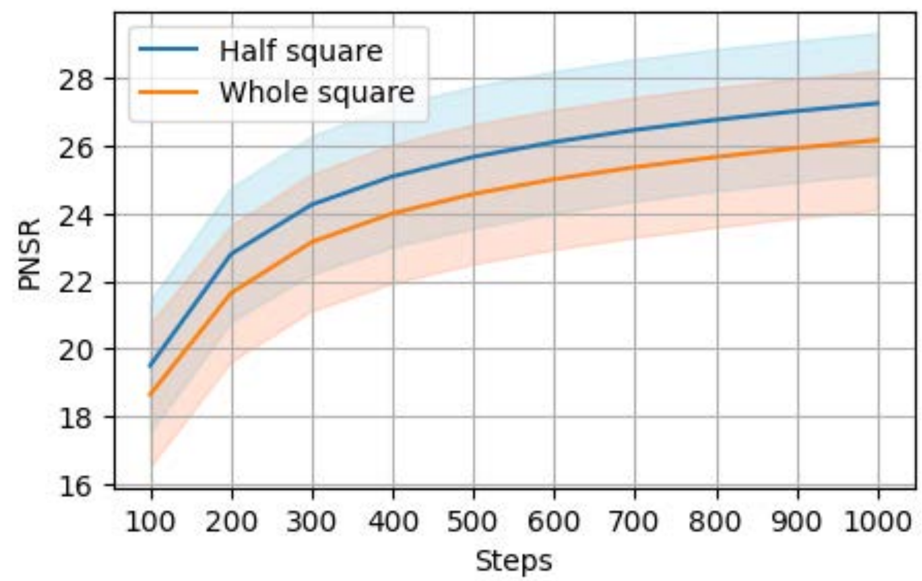
$$\sin(\omega\mathbf{x}+\varphi) = \sin(\omega\mathbf{x})\cos(\varphi)+\sin(\varphi)\cos(\omega\mathbf{x})$$

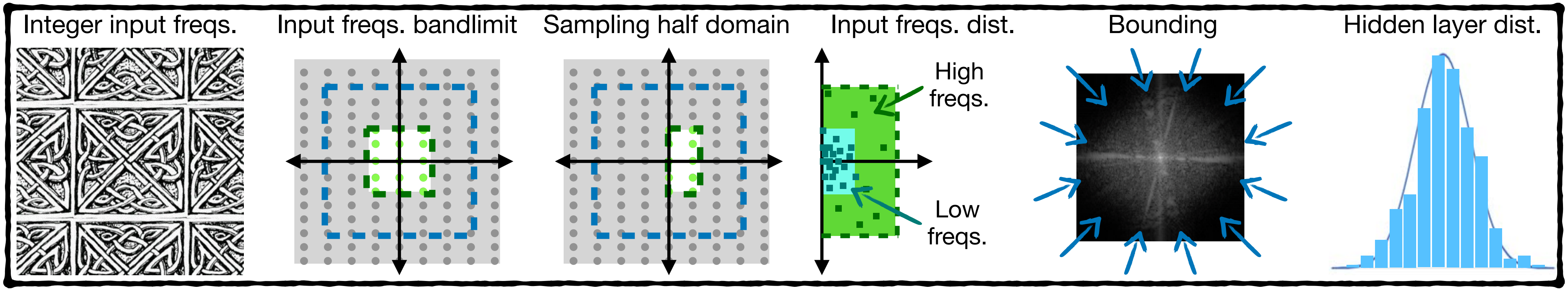
$$-\sin\left(-\omega\mathbf{x}+\frac{\pi}{2}\right)$$

$$D(\mathbf{x}) = \left[\sin(\omega_j\mathbf{x}+\varphi_j)\right]_j$$

Linear combination of sines with frequencies ω_j and $-\omega_j$.

Choose $\omega, -\omega$ or choose $\omega, \tilde{\omega}$?



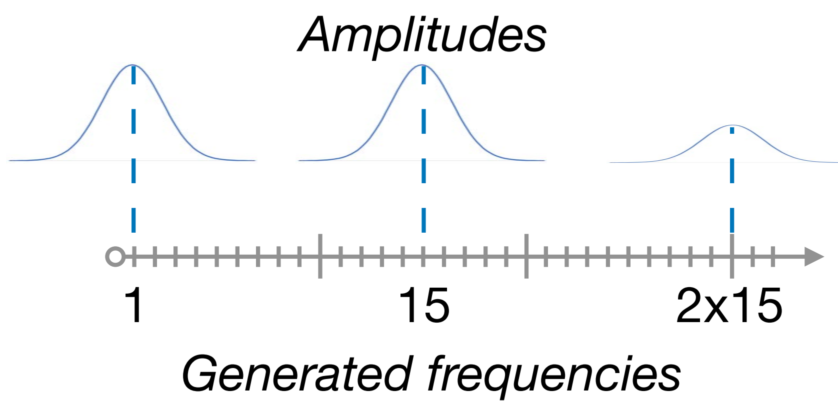


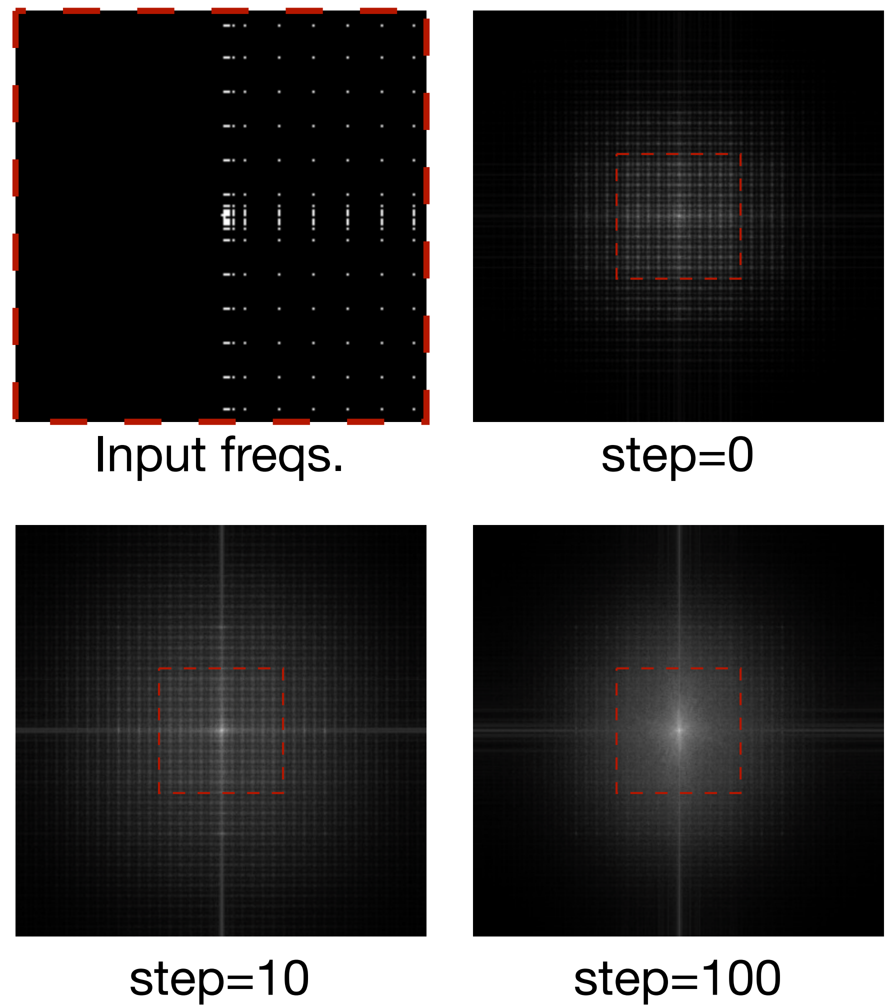
Generated frequencies $\beta_{\mathbf{k}}(\omega) = k_1\omega_1 + \ldots + k_m\omega_m$ have low amplitudes for $\|\mathbf{k}\|_\infty \geq 5$

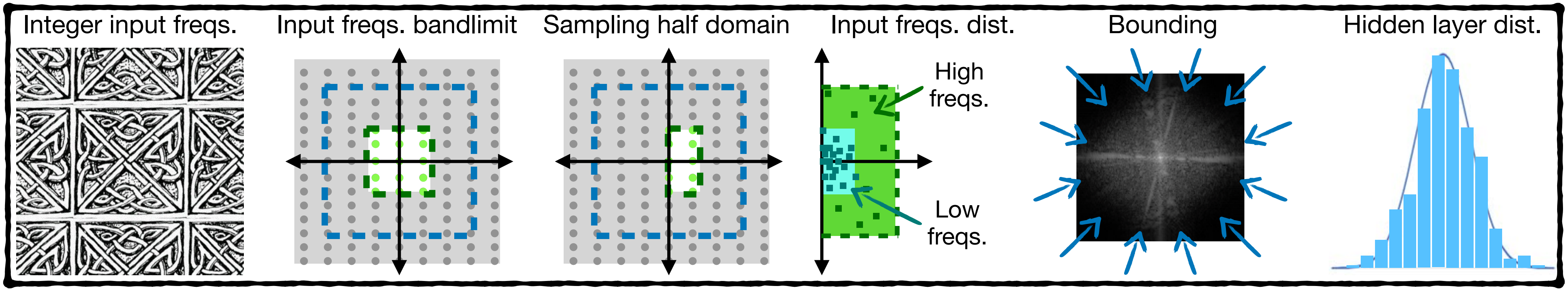
Example in $\mathbb{R}$

Input frequencies $\omega = \frac{2\pi}{p}[1, 15]$

Generated frequencies $\beta(\omega) = \frac{2\pi}{p}(k \cdot 1 + l \cdot 15)$





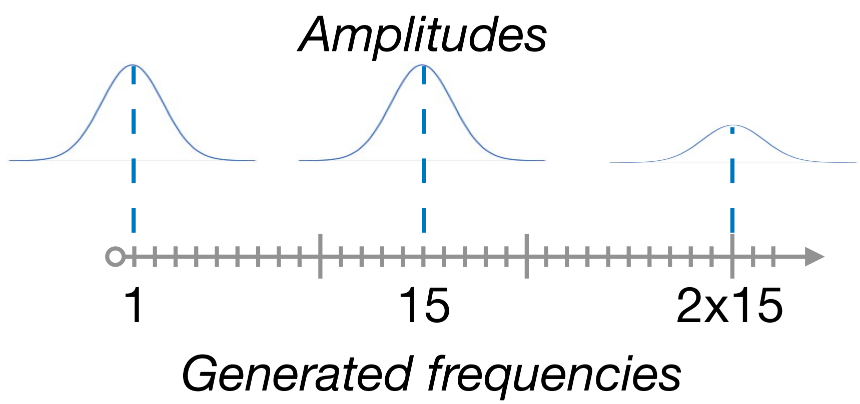


Generated frequencies $\beta_{\mathbf{k}}(\omega) = k_1\omega_1 + \ldots + k_m\omega_m$ have low amplitudes for $\|\mathbf{k}\|_\infty \geq 5$

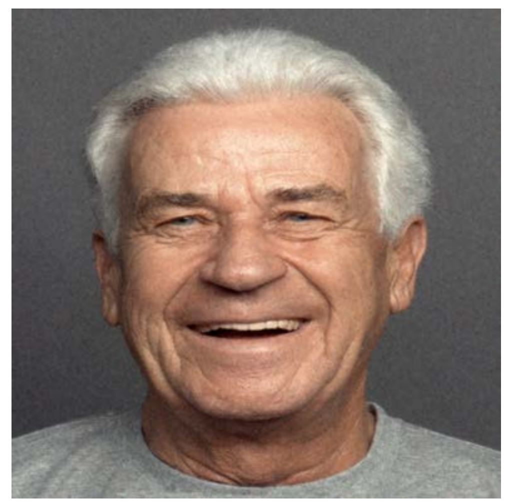
Example in $\mathbb{R}$

Input frequencies $\omega = \frac{2\pi}{p}[1, 15]$

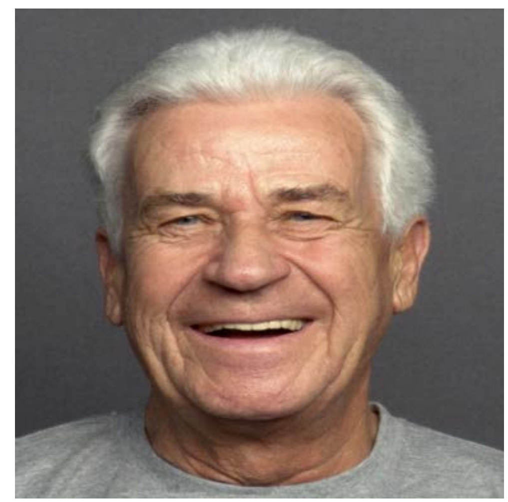
Generated frequencies $\beta(\omega) = \frac{2\pi}{p}(k \cdot 1 + l \cdot 15)$



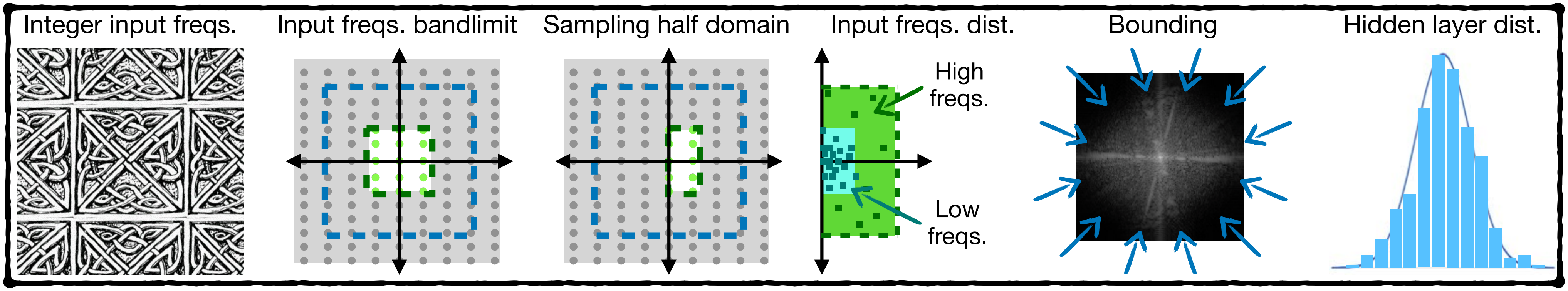
Maximum frequency of 85



Uniform
PSNR: 33.07



Our initialization
PSNR: 35.52



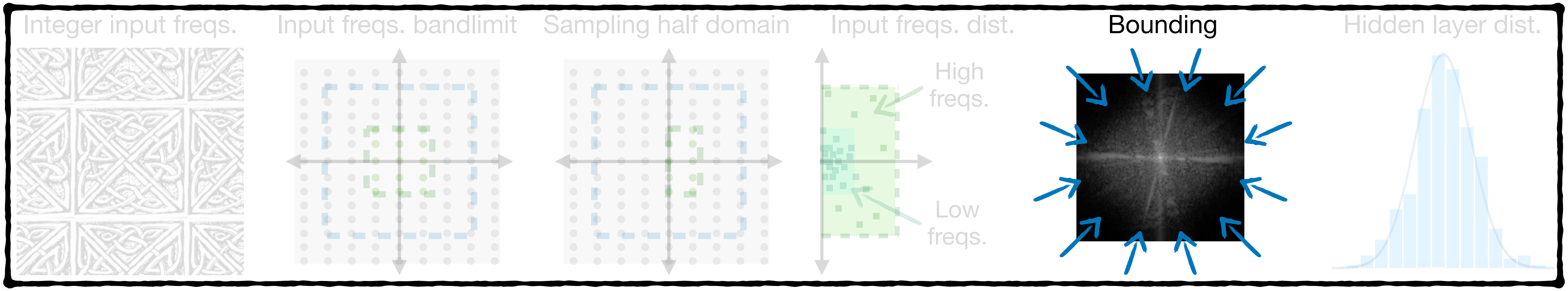
Generated frequencies $\beta_{\mathbf{k}}(\omega) = k_1\omega_1 + \ldots + k_m\omega_m$ with amplitudes $\alpha_{\mathbf{k}}(\mathbf{W}_i)$

Corollary

$$\left|\alpha_{\mathbf{k}}(\mathbf{W}_i)\right| < \prod_{j=1}^{m} \frac{\left(\frac{|W_i|}{2}\right)^{|k_j|}}{|k_j|!} \quad \leq 1$$

- If $\|\mathbf{W}_i\|_\infty \leq 2$ the amplitudes decrease at least exponentially with respect to $\|\mathbf{k}\|_\infty$
- SIREN initialization satisfies this equation (for m>6)

Novello, Tiago. "Understanding sinusoidal neural networks." *arXiv preprint arXiv:2212.01833* (2022).



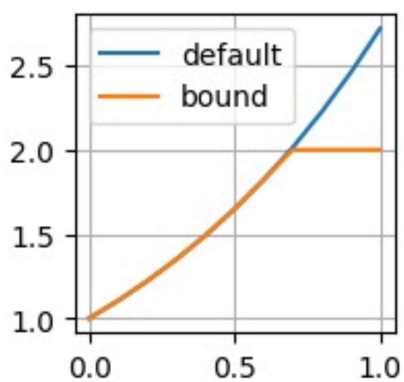
Generated frequencies $\beta_{\mathbf{k}}(\omega) = k_1\omega_1 + \ldots + k_m\omega_m$ with amplitudes $\alpha_{\mathbf{k}}(\mathbf{W}_i)$

Corollary

$$\left|\alpha_{\mathbf{k}}(\mathbf{W}_i)\right| < \prod_{j=1}^{m} \frac{\left(\frac{|W_{ij}|}{2}\right)^{|k_j|}}{|k_j|!}$$

$$\|\mathbf{W}\|_\infty \leq c,\ c \in (0,2]$$

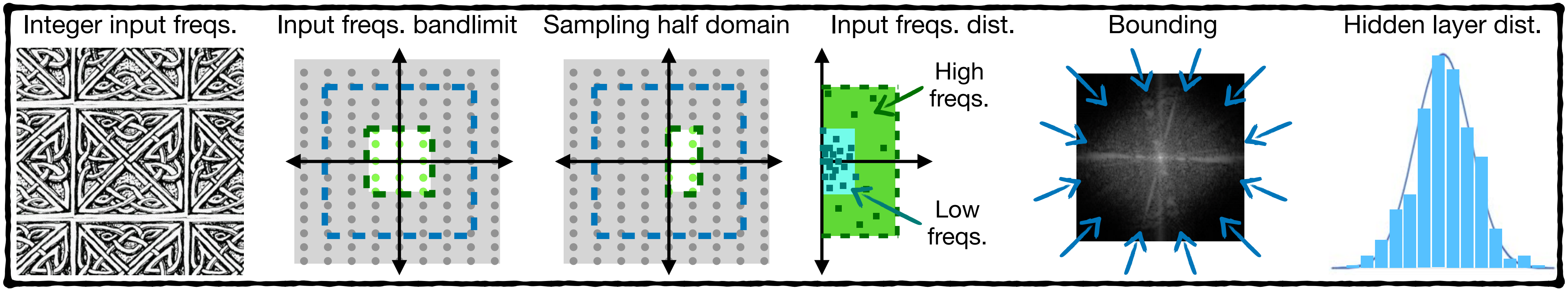
$$\left|\alpha_{\mathbf{k}}(\mathbf{W}_i)\right| < \frac{\left(\frac{c}{2}\right)^{\|\mathbf{k}\|_1}}{\prod_{j=1}^{m}|k_j|!}$$



c small

c big

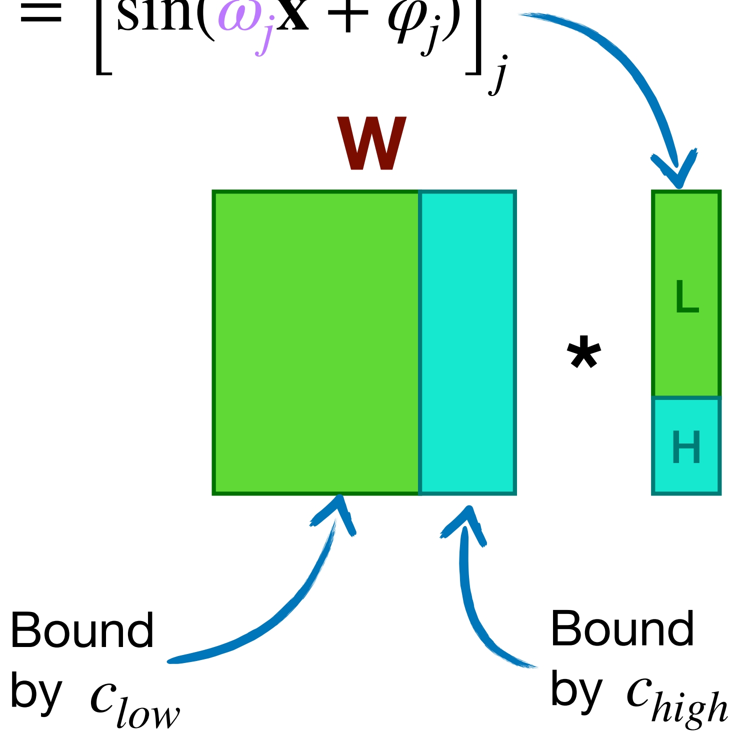
Novello, Tiago. "Understanding sinusoidal neural networks." *arXiv preprint arXiv:2212.01833* (2022).



Generated frequencies $\beta_{\mathbf{k}}(\omega) = k_1\omega_1 + \ldots + k_m\omega_m$ with amplitudes $\alpha_{\mathbf{k}}(\mathbf{W}_i)$

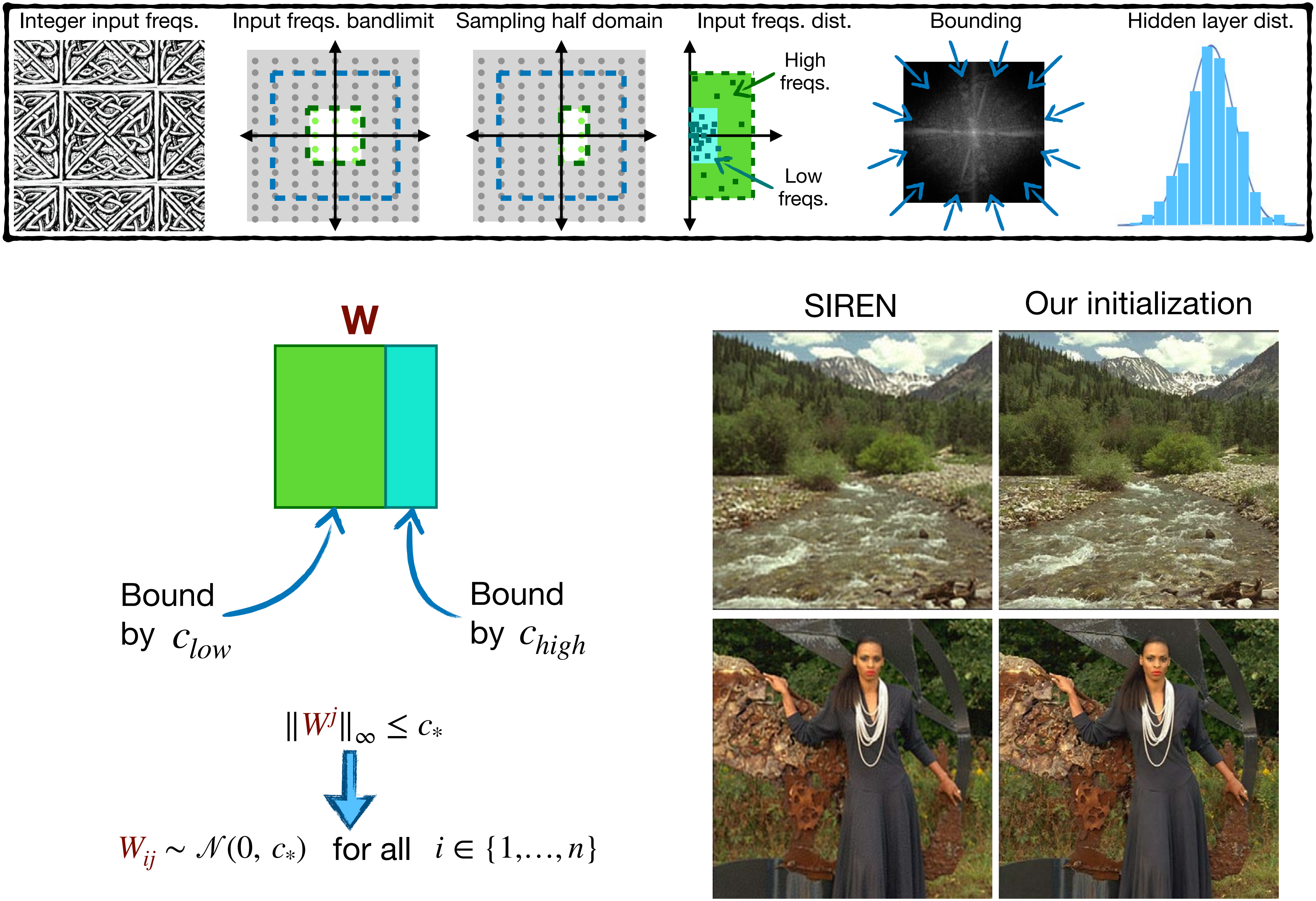
$$S \circ D(\mathbf{x}) = \sin\left(\mathbf{W}D(\mathbf{x}) + \mathbf{b}\right)$$

$$D(\mathbf{x}) = \left[\sin(\omega_j \mathbf{x} + \varphi_j)\right]_j$$



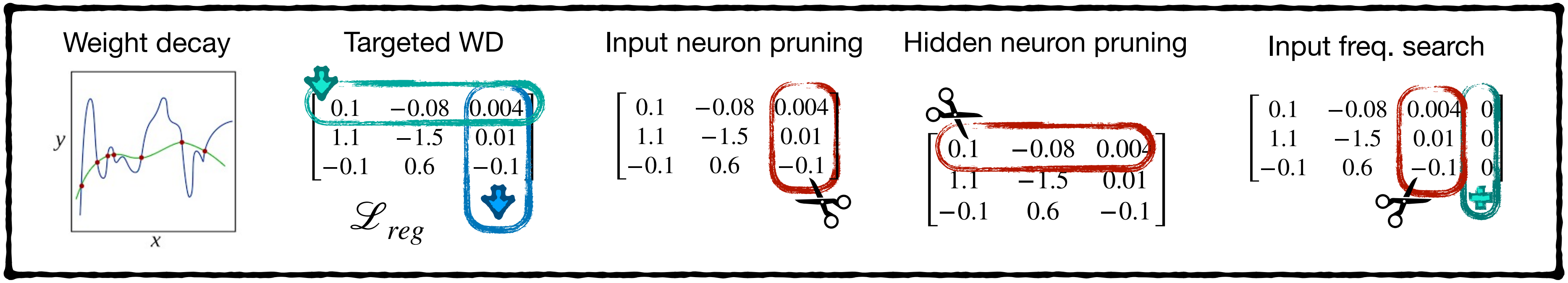
$c_{low} = 0.2,$
$c_{high} = 0.2$

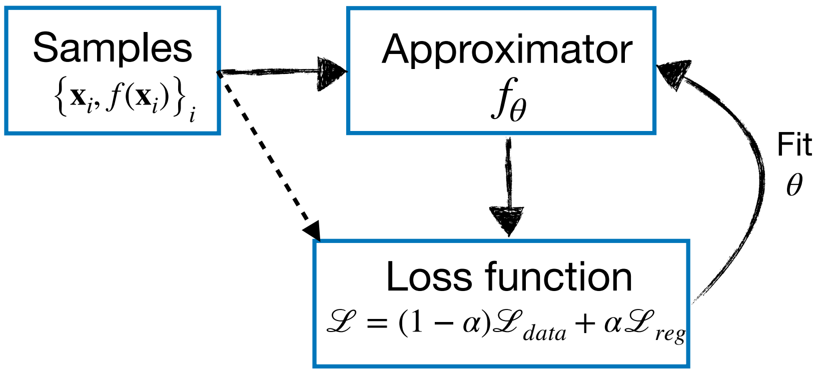
$c_{low} = 1.5,$
$c_{high} = 0.5$



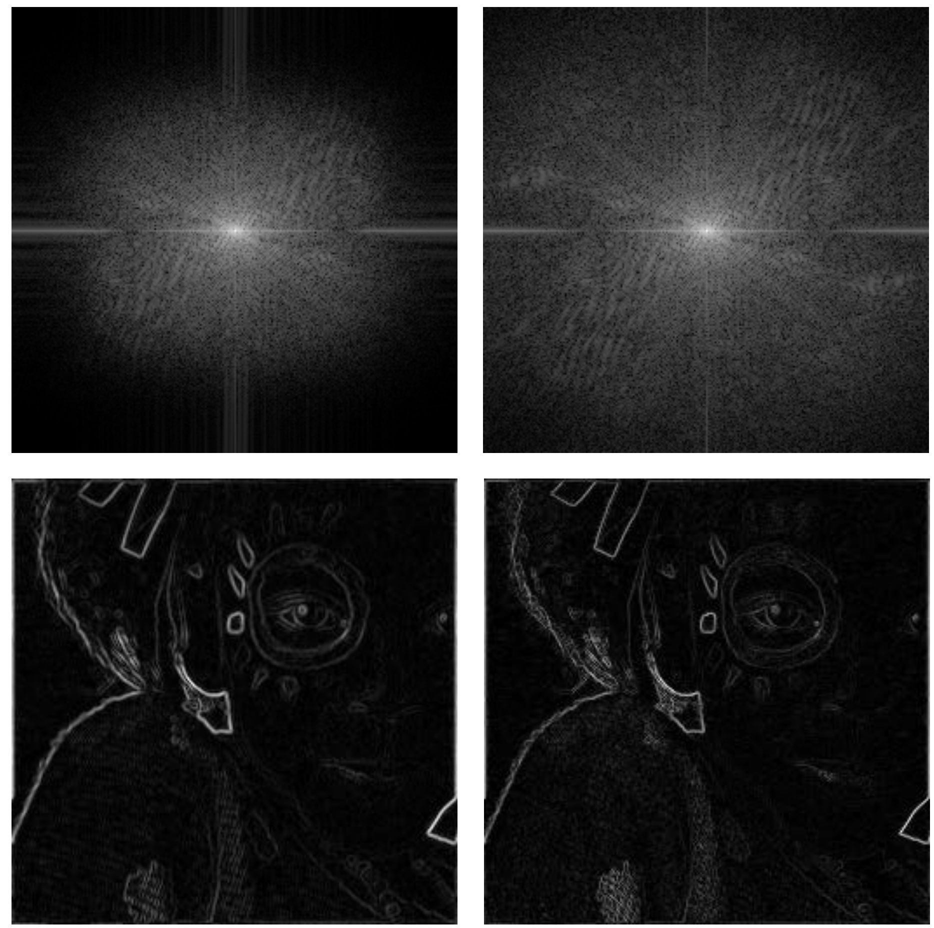
Integer input freqs.
Input freqs. bandlimit
Sampling half domain
Input freqs. dist.
High freqs.
Low freqs.
Bounding
Hidden layer dist.
W
Bound by c_{low}
Bound by c_{high}
$\|W^j\|_\infty \leq c_*$
$W_{ij} \sim \mathcal{N}(0, c_*)$ for all $i \in \{1,\ldots,n\}$
SIREN
Our initialization

Pruning

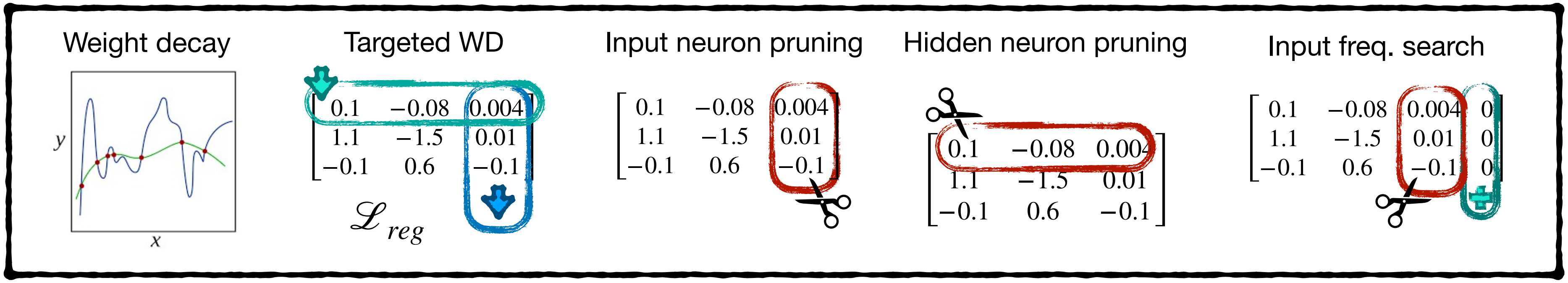




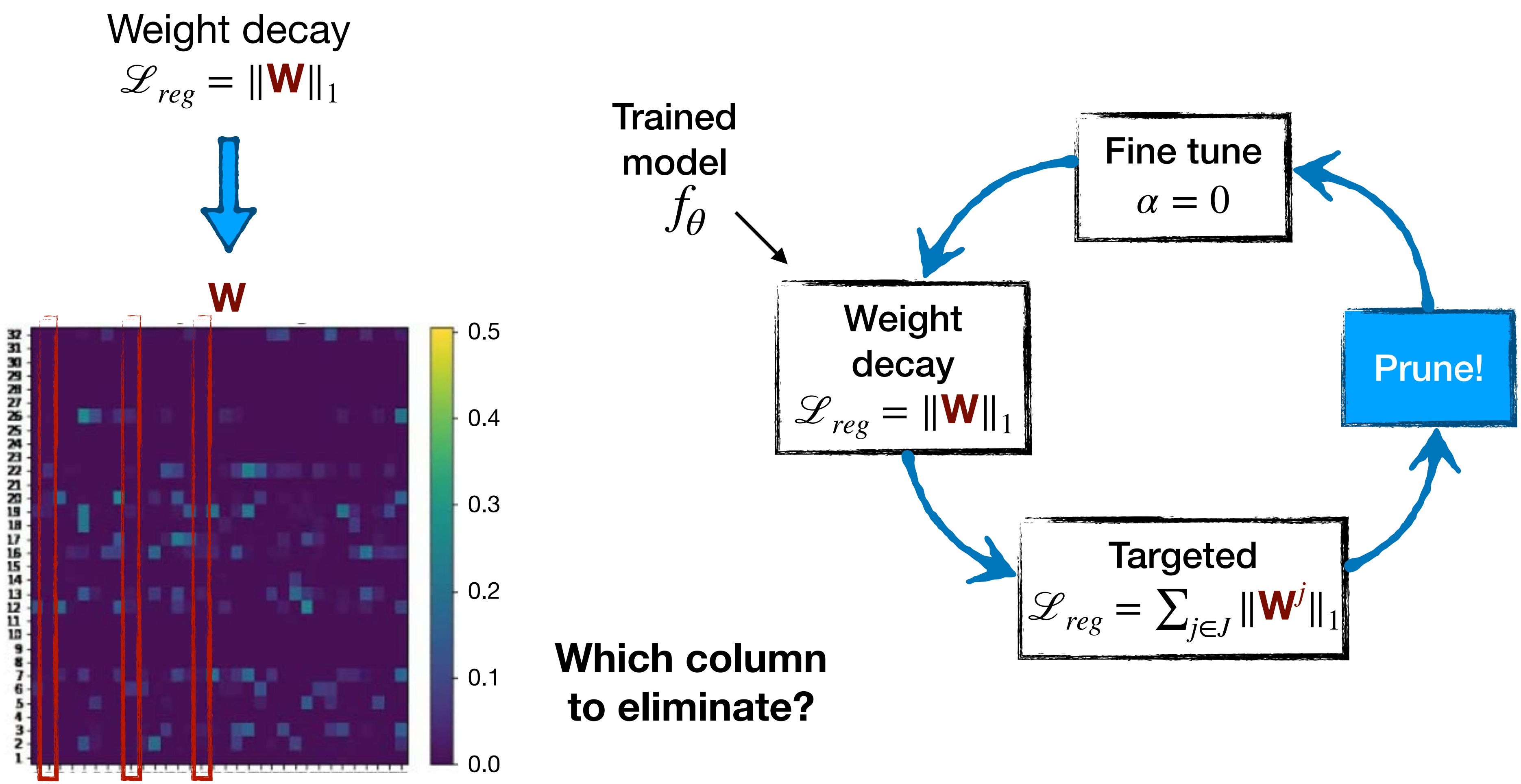
- $\mathscr{L}_{reg} = \|\mathbf{W}\|_1$
- Softening of the function
- Reduces values that are not as informative



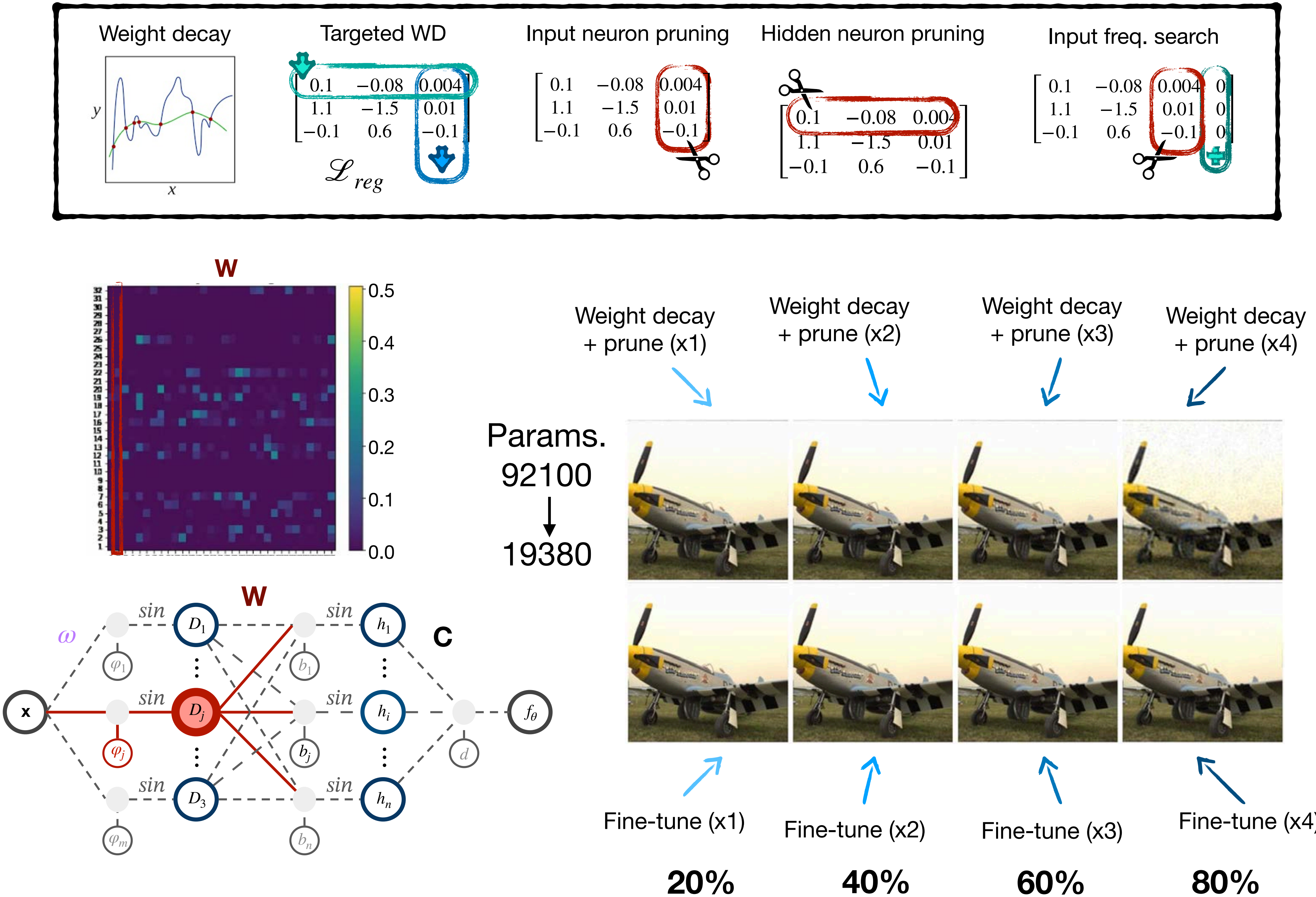
$\alpha = 0.00001$ $\alpha = 0.0000001$



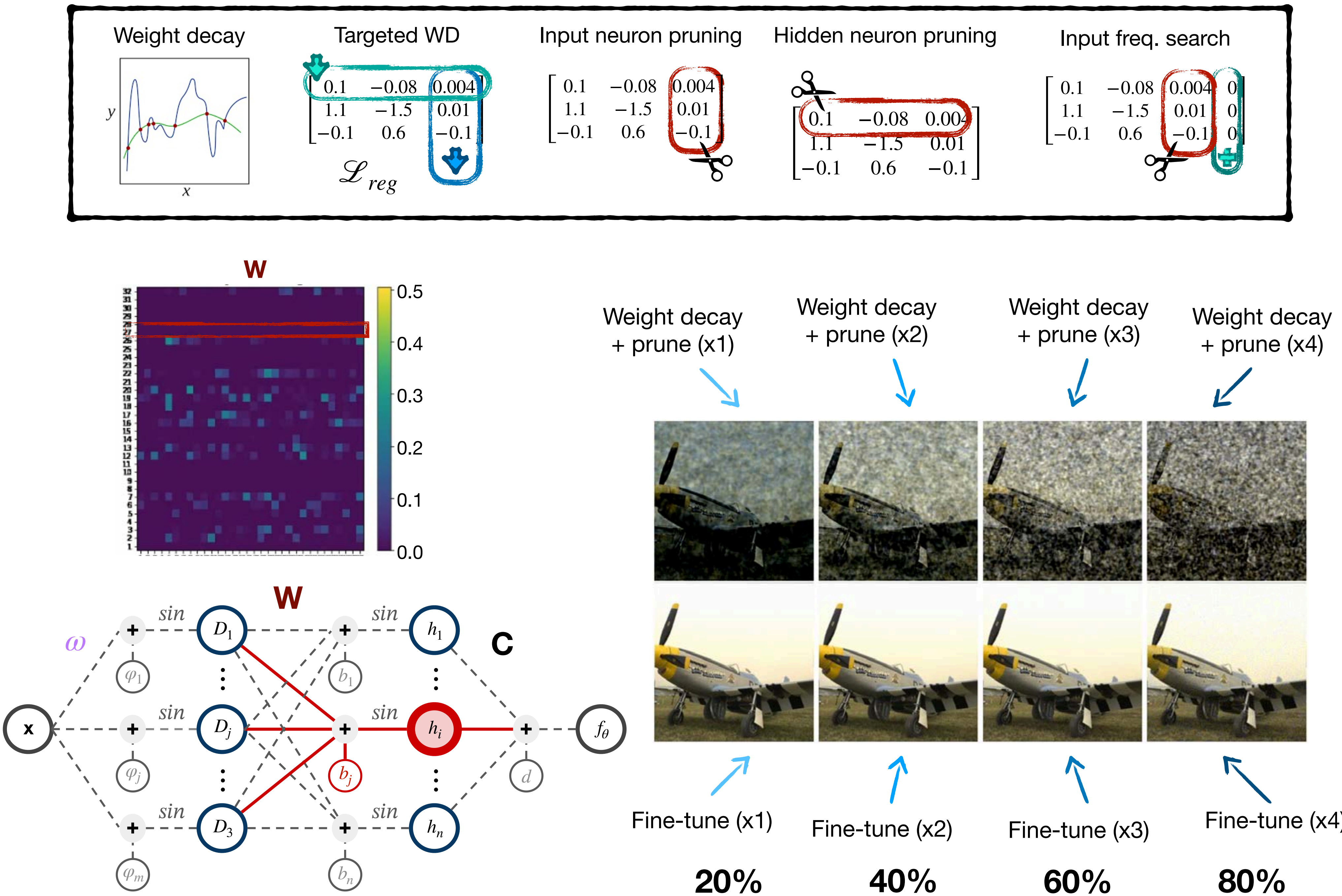
Weight decay
Targeted WD
Input neuron pruning
Hidden neuron pruning
Input freq. search
y
x
0.1 −0.08 0.004
1.1 −1.5 0.01
−0.1 0.6 −0.1
$\mathscr{L}_{reg}$
0.1 −0.08 0.004 0
1.1 −1.5 0.01 0
−0.1 0.6 −0.1 0

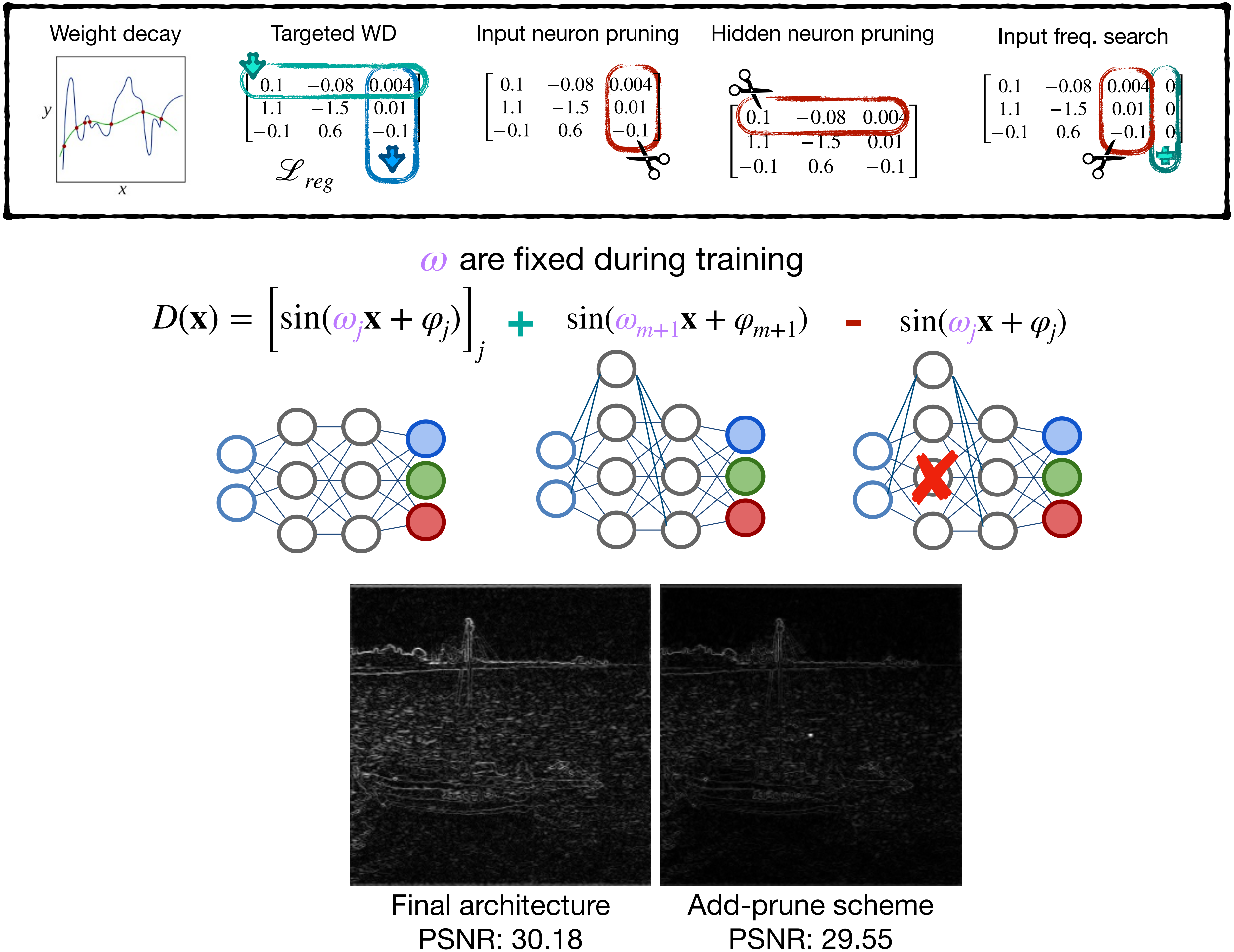


Weight decay
$\mathscr{L}_{reg} = \|\mathbf{W}\|_1$
W
0.5
0.4
0.3
0.2
0.1
0.0
Trained model
f_θ
Which column to eliminate?
Fine tune
$\alpha = 0$
Weight decay
$\mathscr{L}_{reg} = \|\mathbf{W}\|_1$
Prune!
Targeted
$\mathscr{L}_{reg} = \sum_{j \in J} \|\mathbf{W}^j\|_1$

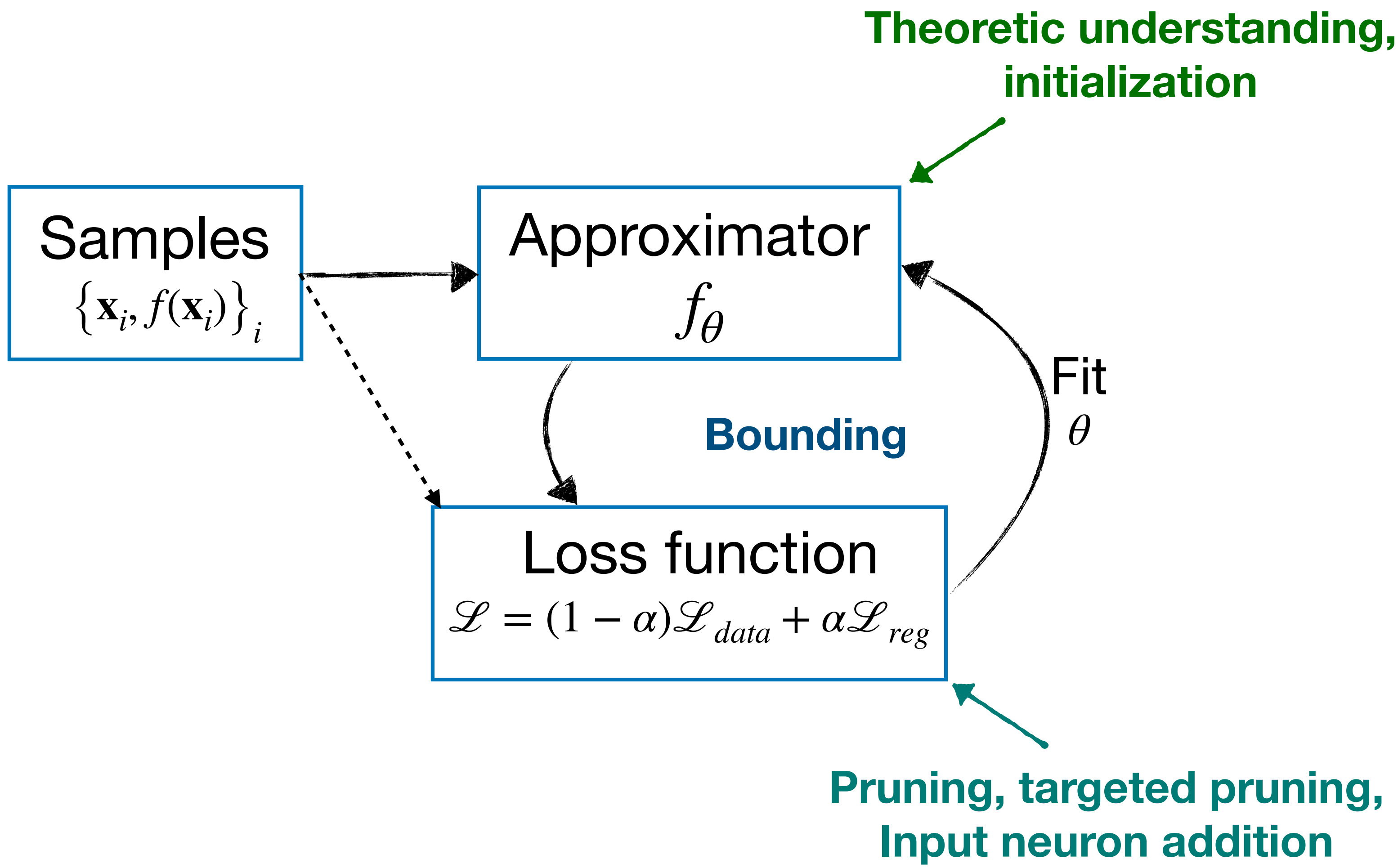


Weight decay
Targeted WD
Input neuron pruning
Hidden neuron pruning
Input freq. search
W
0.5
0.4
0.3
0.2
0.1
0.0
Weight decay + prune (x1)
Weight decay + prune (x2)
Weight decay + prune (x3)
Weight decay + prune (x4)
Params.
92100
19380
W
C
sin
x
Fine-tune (x1)
Fine-tune (x2)
Fine-tune (x3)
Fine-tune (x4)
20%
40%
60%
80%


Weight decay
Targeted WD
Input neuron pruning
Hidden neuron pruning
Input freq. search
W
Weight decay + prune (x1)
Weight decay + prune (x2)
Weight decay + prune (x3)
Weight decay + prune (x4)
Fine-tune (x1)
Fine-tune (x2)
Fine-tune (x3)
Fine-tune (x4)
20%
40%
60%
80%
C


Weight decay
Targeted WD
Input neuron pruning
Hidden neuron pruning
Input freq. search
$\mathcal{L}_{reg}$
ω are fixed during training
$D(\mathbf{x}) = \left[\sin(\omega_j \mathbf{x} + \varphi_j)\right]_j + \sin(\omega_{m+1}\mathbf{x} + \varphi_{m+1}) - \sin(\omega_j \mathbf{x} + \varphi_j)$
Final architecture
PSNR: 30.18
Add-prune scheme
PSNR: 29.55

Pipeline



Questions?