

# Sobolev acceleration for neural networks

Hwijae Son

joint work with Jong Kwon Oh and Hanbaek Lyu

Konkuk University

Kias AI Workshop, 2025

## Sobolev training

Sobolev training, first introduced in 2017 [1], refers to a training algorithm that leverages **derivative information** of the target function.

Let  $f$  denotes the target function and  $g(x; w)$  denotes a neural network with parameter  $w$ .

Then, the conventional  $L^2$  loss function is defined as:

$$\mathcal{L}(w) = \mathbb{E}_{x \sim p} \left[ (g(x; w) - f(x))^2 \right] \approx \frac{1}{N} \sum_{i=1}^N (g(x_i; w) - f(x_i))^2$$

**Sobolev training** leverages the  $H^1$  (or higher-orders) loss function:

$$\begin{aligned} \mathcal{H}(w) &= \mathbb{E}_{x \sim p} \left[ (g(x; w) - f(x))^2 + (\nabla_x g(x; w) - \nabla_x f(x))^2 \right] \\ &\approx \frac{1}{N} \sum_{i=1}^N (g(x_i; w) - f(x_i))^2 + (\nabla_x g(x_i; w) - \nabla_x f(x_i))^2 \end{aligned}$$

# Sobolev training

Sobolev training has shown strong empirical success across various scientific domains where derivative information is naturally available:

- ▶ Knowledge distillation [1].
- ▶ Physics-informed machine learning [2, 4].
- ▶ Tasks dealing with images (approximated derivative) [6].

So far, it has been (empirically) observed that

- ▶ Sobolev training results in a much smaller test error.
- ▶ Sobolev training makes networks learn more complicated features.
- ▶ Sobolev training accelerates the convergence of training dynamics.

## Sobolev training

- Sobolev Training results in a much smaller test error [1].

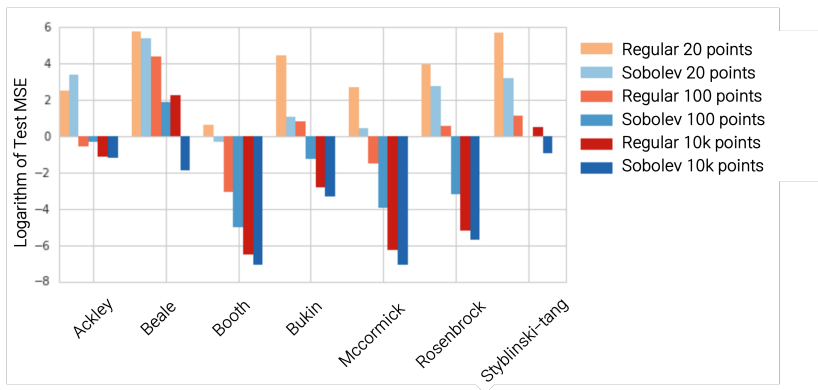


Figure: Test errors of Sobolev training compared to the  $L^2$  training.

## Sobolev training

- Sobolev Training makes networks learn more complicated features [5]:  
Spectral Bias, Frequency Principle, Frequency Bias

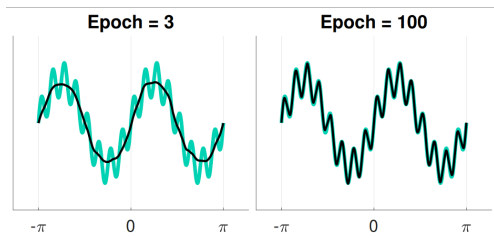


Figure: Spectral bias.

## Sobolev training

$$\begin{aligned}\mathcal{F}(f')(\xi) &= \int e^{-2\pi i \xi t} f'(t) dt = e^{-2\pi i \xi t} f(t) \Big|_{-\infty}^{\infty} + \int 2\pi i \xi e^{-2\pi i \xi t} f(t) dt \\ &= 2\pi i \xi \mathcal{F}(f)\end{aligned}$$

$$\|g(x; w) - f(x)\|_{H^s(\mathbb{R}^n)} = \|\mathcal{F}^{-1} \left[ (1 + |\xi|^2)^{s/2} \mathcal{F}(g(x; w) - f(x)) \right]\|_{L^2(\mathbb{R}^n)}$$

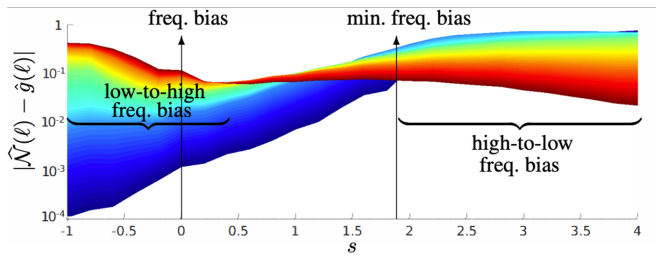


Figure: Changes in spectral bias [5].

## Sobolev acceleration

- ▶ Moreover, it has been demonstrated that Sobolev training can significantly accelerate the convergence of the  $L^2$  error.
- ▶ For instance, [2] provided empirical evidence of such 'Sobolev acceleration'.

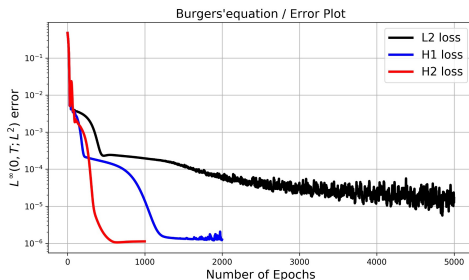


Figure: Sobolev acceleration for PINNs.

# Sobolev acceleration

Since then, Sobolev acceleration has been repeatedly observed in the literature. However, there is no theoretical evidence so far.

- ▶ We present a theoretical justification of Sobolev acceleration and quantify the acceleration, covering both  $H^1$  and  $H^2$  norms, for a specific class of ReLU-activated networks in the student–teacher framework.
- ▶ We illustrate our analysis through numerical examples, thereby demonstrating its generalization to a practical scenario.
- ▶ We also apply Sobolev training to modern deep learning benchmark problems, including denoising autoencoders and diffusion models, and demonstrate both convergence acceleration and improved performance.



## Sobolev training for linear models

As an illustrative example, we prove the Sobolev acceleration of gradient descent for the linear model.

### Proposition

Let  $X \in \mathbb{R}^{N \times d}$  denote the given data matrix and  $y = Xw^* + \epsilon \in \mathbb{R}^N$  be corresponding labels, where  $\epsilon \sim \mathcal{N}(0, \sigma^2 I)$ . Consider the linear model  $g(x; w) = w^T x$ . Define the following loss functions:

$$\begin{aligned}\mathcal{L}(w) &= \frac{1}{2} \sum_{i=1}^N (w^T x_i - w^{*T} x_i)^2 = \frac{1}{2} \|Xw - Xw^*\|^2, \\ \mathcal{H}(w) &= \frac{1}{2} \sum_{i=1}^N \left[ (w^T x_i - w^{*T} x_i)^2 + \lambda \|w - w^*\|^2 \right], \\ &= \frac{1}{2} [\|Xw - Xw^*\|^2 + \lambda \|w - w^*\|^2].\end{aligned}$$

Let  $\kappa(\cdot)$  denote the condition number of a matrix. Then,  $\kappa(\nabla_w^2 \mathcal{H}) < \kappa(\nabla_w^2 \mathcal{L})$ . Hence, Sobolev training improves the conditioning of the optimization problem and accelerates the convergence of gradient descent for the linear model.

# Sobolev training for linear models

## Proposition (Cont'd)

Let  $\hat{w}_{L^2}$  and  $\hat{w}_{H^1}$  be the optimal parameters that minimize  $\mathcal{L}(w)$  and  $\mathcal{H}(w)$ , respectively. Then  $\mathbb{E}_{x \sim P_{data}}(\hat{w}_{H^1}^T x - w^{*T} x)^2 < \mathbb{E}_{x \sim P_{data}}(\hat{w}_{L^2}^T x - w^{*T} x)^2$ , i.e., Sobolev training improves generalization error.

## Sketch of proof.

We can easily compute the Hessians:  $\nabla_w^2 \mathcal{L} = X^T X$  and  $\nabla_w^2 \mathcal{H} = X^T X + \lambda I$ .

Moreover, both optimal parameters are unbiased, i.e.,  $\mathbb{E}(\hat{w}_{L^2}) = \mathbb{E}(\hat{w}_{H^1}) = w^*$ .

Bias-variance tradeoff: generalization error =  $Bias^2 + Variance$ .

$Var_{H^1} = \sum_{i=1}^d \frac{\sigma_i^2}{(\sigma_i + \lambda)^2} < d = Var_{L^2}$ , where  $\sigma_i$  denotes the eigenvalues of  $X^T X$ . □

## Remark

Although the proposition relies on the idealized assumption that the true parameter  $w^*$  is known, it provides intuition for the Sobolev acceleration effect and the generalization ability.

# Assumptions

## Assumption (Two-layer ReLU network)

$g(x; w) = \sum_{j=1}^K \sigma(w_j^\top x)$  where  $w = [w_1, \dots, w_K] \in \mathbb{R}^{d \times K}$  and  $\sigma(t) = \max(0, t)$  with  $K \geq 1$  ReLU nodes in the hidden layer.

## Assumption (Student-teacher setting)

There exists an unknown *teacher parameter*  $w^*$  for which  $f(\cdot) = g(\cdot; w^*)$ .

## Assumption (Gaussian population)

The data distribution  $\mathcal{P}$  is the standard Gaussian  $N(0, I_{d \times d})$ .

## Definition

Let  $x_1, x_2, \dots, x_N$  be i.i.d. samples from  $\mathcal{P}$ . We define the population loss functions as:

$$\mathcal{L}(w) := \mathbb{E} \left( \frac{1}{2N} \sum_{j=1}^N (g(x_j; w) - g(x_j; w^*))^2 \right),$$

$$\mathcal{J}(w) := \mathbb{E} \left( \frac{1}{2N} \sum_{j=1}^N \|\nabla_x g(x_j; w) - \nabla_x g(x_j; w^*)\|^2 \right), \quad \mathcal{H}(w) = \mathcal{L}(w) + \mathcal{J}(w).$$

## Previous result

We begin by recalling a result of Tian [3] for the gradient flow for  $L^2$ -training with single ReLU node ( $K=1$ ):

### Theorem (Theorem 5 in [3])

Consider the gradient flow  $\dot{w} = -\nabla_w \mathcal{L}(w)$  and let  $V(w) = \|w - w^*\|^2$  denote the squared error. Suppose that an initial parameter  $w^0$  satisfies

$\|w^0 - w^*\| < \|w^*\|$ , then  $\frac{dV}{dt} = -(w - w^*)^\top \nabla_w \mathcal{L} < 0$  and  $w^t \rightarrow w^*$  as  $t \rightarrow \infty$ .

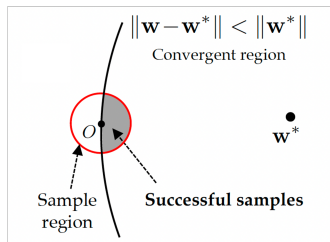


Figure: Convergent region

## Theoretical results

We show that by using the  $H^1$  loss function, the decay of  $V$  can be accelerated.

### Theorem ( $H^1$ acceleration for $K=1$ )

Consider the gradient flow  $\dot{w} = -\nabla_w \mathcal{H}(w)$ . Suppose that  $\|w^0 - w^*\| < \|w^*\|$ . Then,

$$\frac{dV}{dt} = -(w - w^*)^\top \nabla_w \mathcal{H} \leq -(w - w^*)^\top \nabla_w \mathcal{L} - \lambda(\theta)(\|w\|^2 + \|w^*\|^2) < 0,$$

where  $\lambda(\theta) = (2\pi - \theta) - \sqrt{\theta^2 + (2\pi - \theta)^2 \cos^2 \theta} \geq 0$  with  $\theta$  denoting the angle between  $w$  and  $w^*$ . Therefore, the decay of  $V$  is accelerated by using the Sobolev loss function. Moreover, since the rate of acceleration  $\lambda(\theta)$  is an increasing function of  $\theta \in [0, \pi/2)$ , the convergence  $w^t \rightarrow w^*$  is much more accelerated when  $\theta$  is large.

# Theoretical results

## Theorem (Comparison of the optimization landscapes)

Let  $\theta$  denotes the angle between  $w$  and  $w^*$ . Then we have

$$\begin{aligned}\nabla_w^2 \mathcal{L} &= \frac{1}{2}I - \alpha(uu^\top - \cos(\theta)vu^\top + \sin^2(\theta)I)(I - vv^\top), \\ \nabla_w^2 \mathcal{H} &= I - \alpha(2uu^\top - \cos(\theta)vu^\top + \sin^2(\theta)I)(I - vv^\top)\end{aligned}\tag{1}$$

where  $\alpha = \frac{\|w^*\|}{2\pi\|w\|\sin(\theta)}$ ,  $u = \frac{w^*}{\|w^*\|}$ , and  $v = \frac{w}{\|w\|}$ . Furthermore, if  $\frac{\|w^*\|\sin(\theta)}{\|w\|} < \frac{\pi}{2}$ , then

$$\kappa(\nabla_w^2 \mathcal{H}) = \frac{1}{1 - 4\alpha \sin^2(\theta)} < \frac{1}{1 - 3\alpha \sin^2(\theta)} = \kappa(\nabla_w^2 \mathcal{L}).$$

# Theoretical results

## Remark

Our result above gives an explicit impact of the  $H^1$  Sobolev training on the optimization landscape. Even in the single ReLU node setting, we observe that such an impact on the Hessian is nonlinear.

## Remark ( $\mathcal{L}$ vs. $2\mathcal{L}$ vs. $\mathcal{L} + \mathcal{J}$ )

Minimizing  $\mathcal{L}$  or  $2\mathcal{L}$  yields the same convergence rate, as both have identical condition numbers. In contrast, Sobolev training (via  $\mathcal{L} + \mathcal{J}$ ) improves the condition number of the Hessian, thereby accelerating convergence.

## Theoretical results

We now demonstrate the same effect for higher-order derivatives.

### Theorem ( $H^2$ acceleration)

*Suppose  $g(x; w) = (\sigma(w^\top x))^2$ , a two-layer network with a single ReLU<sup>2</sup> node.*

*Consider the gradient flow  $\dot{w} = -\nabla_w \mathcal{I}(w)$ , where*

*$\mathcal{I}(w) = \mathcal{I}_1(w) + \mathcal{I}_2(w) + \mathcal{I}_3(w)$  is the  $H^2$  population loss with  $\mathcal{I}_1 = \mathcal{L}$  and  $\mathcal{I}_2 = \mathcal{J}$ . If  $\|w^0 - w^*\| < \|w^*\|$  then,*

$$-(w - w^*)^\top \nabla_w \mathcal{I}_j(w) < 0, \text{ for } j = 1, 2, 3,$$

*and hence, the decay of  $V = \|w - w^*\|^2$  is accelerated under the gradient flow minimizing the higher order Sobolev loss functions.*



## Theoretical results

We now generalize the result to the ReLU networks with  $K > 1$  hidden nodes.

### Theorem

*If the teacher parameters  $\{w_j^*\}_{j=1}^K$  form an orthonormal basis and a student parameter  $W$  is initialized to be*

$$w_l = yw_1^* + \cdots + yw_{l-1}^* + xw_l^* + yw_{l+1}^* + \cdots + yw_K^* = (y, \dots, x, \dots, y),$$

*under the basis of  $\{w_j^*\}_{j=1}^K$ , where  $(x, y) \in \Omega = \{x \in (0, 1], y \in [0, 1], x > y\}$ , then the following holds.*

- 1. The student parameter  $w_l$  converges to  $w_l^*$  under the gradient flow  $\dot{w}_l = -\nabla_{w_l} \mathcal{H}(W)$ , i.e.,  $(x, y)$  converges to  $(1, 0)$ .*
- 2. Near  $(x, y) = (1, 0)$ , the gradient flows can be linearized to 2-d dynamical systems:*

$$L^2 \text{ gradient flow : } \begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix}_{L^2} \approx -M_3 \begin{pmatrix} x-1 \\ y \end{pmatrix},$$

$$H^1 \text{ gradient flow : } \begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix}_{H^1} \approx -2M_3 \begin{pmatrix} x-1 \\ y \end{pmatrix},$$

*where  $\lambda_1(M_3) = \frac{\pi}{2}$ , and  $\lambda_2(M_3) = \frac{\pi}{2}(K+1)$ . Hence,  $M_3$  is positive definite for all  $K$ , and the convergence is accelerated.*

# Theoretical results

## Theorem (Cont'd)

3. When  $x, y$  are initialized such that  $x = y = (0, 1]$ , the  $L^2$  gradient flow converges to the saddle point

$x = y = x_{L^2}^* = \frac{1}{\pi K}(\sqrt{K-1} - \arccos(\frac{1}{\sqrt{K}}) + \pi)$ , and the  $H^1$  gradient flow converges to the saddle point

$x = y = x_{H^1}^* = \frac{1}{2\pi K}(\sqrt{K-1} + 2\pi - 2\arccos(\frac{1}{\sqrt{K}}))$  and

$x_{L^2}(t) = (x(0) - x_{L^2}^*)e^{-K/2t}$  and  $x_{H^1}(t) = (x(0) - x_{H^1}^*)e^{-Kt}$ . Hence, the convergence is accelerated.

## Theoretical results

Lastly, we extend the results by allowing a more general initialization.

### Theorem

*If the teacher parameters  $\{w_j^*\}_{j=1}^K$  form an orthonormal basis and the student parameters are initialized as the symmetric Toeplitz matrix:*

$$\begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ \vdots \\ w_K \end{pmatrix} = \begin{pmatrix} t_1 & t_2 & t_3 & \cdots & t_K \\ t_2 & t_1 & t_2 & \cdots & t_{K-1} \\ t_3 & t_2 & t_1 & \cdots & t_{K-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ t_K & t_{K-1} & t_{K-2} & \cdots & t_1 \end{pmatrix} \begin{pmatrix} w_1^* \\ w_2^* \\ w_3^* \\ \vdots \\ w_K^* \end{pmatrix},$$

*then under the linearized  $L^2$  and  $H^1$  gradient flows,  $\mathbb{T}_{L^2} = (t_1 - 1, t_2, \dots, t_K)$  and  $\mathbb{T}_{H^1} = (t_1 - 1, t_2, \dots, t_K)$  follow*

$$\dot{\mathbb{T}}_{L^2} = -M\mathbb{T}_{L^2}, \quad \dot{\mathbb{T}}_{H^1} = -2M\mathbb{T}_{H^1}$$

*for a positive definite matrix  $M$ .*

# Experiments

- Sobolev acceleration effect persists under empirical loss minimization using SGD.

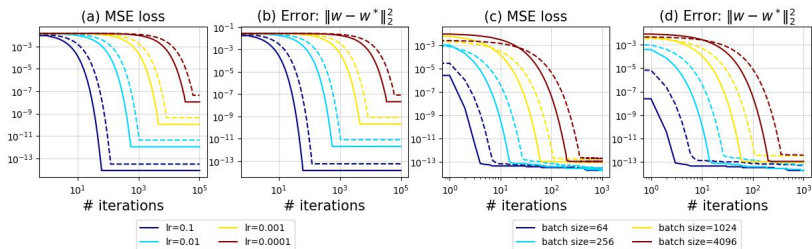


Figure: Comparison of convergence of  $L^2$  training(dashed lines) and  $H^1$  training(solid lines).

# Experiments

- ▶ Sobolev acceleration for MLPs (2-64-64-64-1) with different activations.
- ▶ Target function:  $f(x, y) = \sin(10(x + y)) + (x - y)^2 - 1.5x + 2.5y + 1$
- ▶ We train the networks by using the ADAM optimizer with a learning rate of  $1e - 4$ .

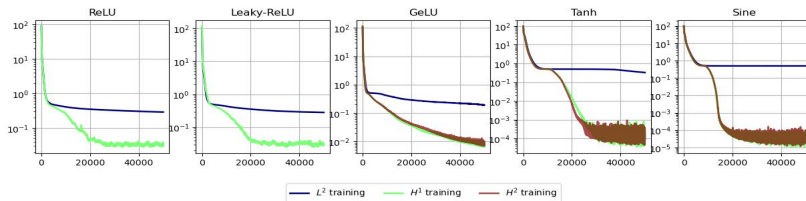


Figure: Results for various activation functions

# Experiments

- ▶ Sobolev acceleration for the Fourier feature networks and SIREN (both 1-64-64-64-1).
- ▶ Target function:  $f(x) = x + \sin(2\pi x^4)$
- ▶ We train the networks by using the ADAM optimizer followed by the L-BFGS optimizer.



Figure: Results for various architectures.

# Experiments

- ▶ Sobolev acceleration for the autoencoder task.
- ▶ Both the encoder and decoder are implemented using the ResNet-18.
- ▶ The model is trained for 300 epochs, and the Adam optimizer is used with a learning rate of  $1e - 3$ .

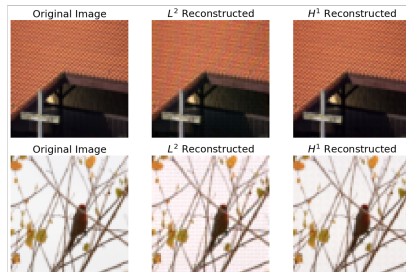
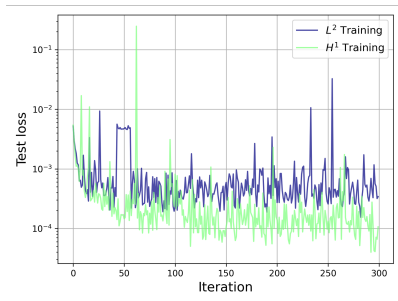


Figure: Autoencoder for the ImageNet dataset.

# Experiments

- ▶ Sobolev acceleration for the denoising autoencoder task.
- ▶ A simple autoencoder comprising an encoder and a decoder, each consisting of three convolution layers
- ▶ Trained with two types of additive noise: a Gaussian noise  $\epsilon_1 \sim N(0, 1/4)$  and a deterministic noise  $\epsilon_2 = 0.3 \sin(2\pi(x + y))$ .
- ▶ Tested on amplified noise  $\tilde{\epsilon}_1 \sim N(0, 1)$  and  $\tilde{\epsilon}_2 = 0.3 \sin(20\pi(x + y))$ .

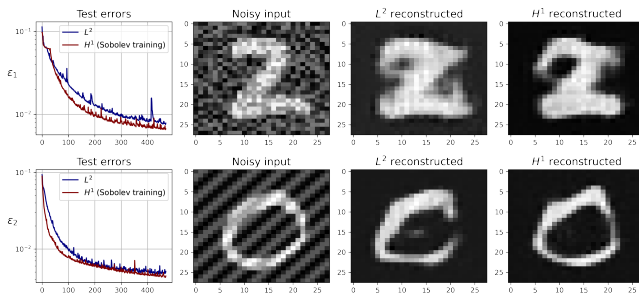


Figure: Denoising autoencoder for the MNIST dataset.



# Experiments

- ▶ Sobolev acceleration for the diffusion model.
- ▶ We trained a DDIM with a U-Net denoiser with 1,000 diffusion timesteps and 20 sampling steps.

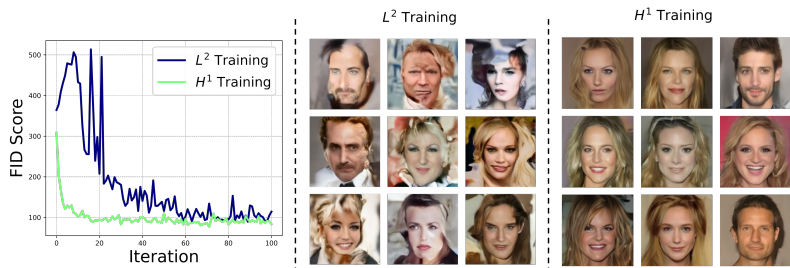


Figure: Diffusion model for the CelebA-HQ dataset.

# Summary

- ▶ Sobolev acceleration is a convergence acceleration phenomenon of training neural networks.
- ▶ We provide the first theoretical evidence of Sobolev acceleration by analyzing the Hessians of the loss landscapes and the gradient flow dynamics of the student–teacher setting for ReLU networks.
- ▶ We present several empirical observations suggesting that Sobolev acceleration is a general phenomenon across various deep learning tasks.
- ▶ We aim to extend the analysis of gradient dynamics in Sobolev training to encompass a wider range of neural network architectures, including deeper and more complex models

# References



Wojciech M Czarnecki, Simon Osindero, Max Jaderberg, Grzegorz Swirszcz, and Razvan Pascanu.

Sobolev training for neural networks.

*Advances in neural information processing systems*, 30, 2017.



Hwijae Son, Jin Woo Jang, Woo Jin Han, and Hyung Ju Hwang.

Sobolev training for physics informed neural networks.

*Communications in Mathematical Sciences*, 2023.



Yuandong Tian.

An analytical formula of population gradient for two-layered relu network and its applications in convergence and critical point analysis.

In *International conference on machine learning*, pages 3404–3413. PMLR, 2017.



Nikolaos N Vlassis and WaiChing Sun.

Sobolev training of thermodynamic-informed neural networks for interpretable elasto-plasticity models with level set hardening.

*Computer Methods in Applied Mechanics and Engineering*, 377:113695, 2021.



Annan Yu, Yunan Yang, and Alex Townsend.

Tuning frequency bias in neural network training with nonuniform data.

In *The Eleventh International Conference on Learning Representations*, 2023.



Wentao Yuan, Qingtian Zhu, Xiangyue Liu, Yikang Ding, Haotian Zhang, and Chi Zhang.

Sobolev training for implicit neural representations with approximated image derivatives.

In *European Conference on Computer Vision*, pages 72–88. Springer, 2022.

*Thank You!*