9 Inverse Problems & Deep Learning: Applications

Leon Herrmann

Stefan Kollmannsberger

Chair of Data Engineering in Construction

Bauhaus-Universität Weimar

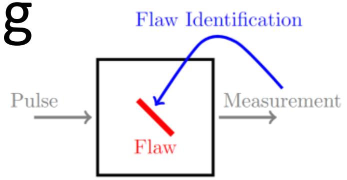
Deep Learning in Computational Mechanics – an introductory course, Herrmann et al. 2025

website

book

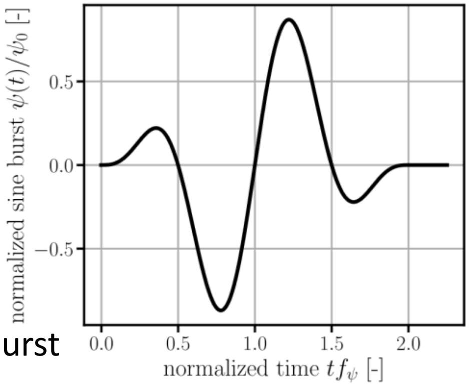
Contents

9.2 Ultrasonic Nondestructive Testing



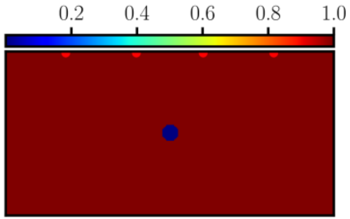
Ultrasonic Identification of flaws uses

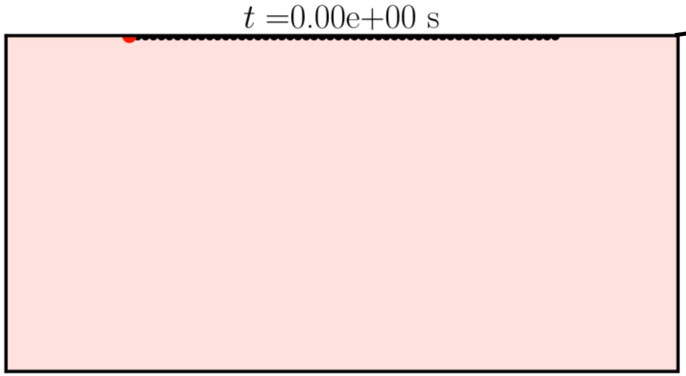
- Ultrasonic pulses (e.g., Ricker wavelet, or sinusoidal burst)
- Measurements of the pulses after traversal through the material

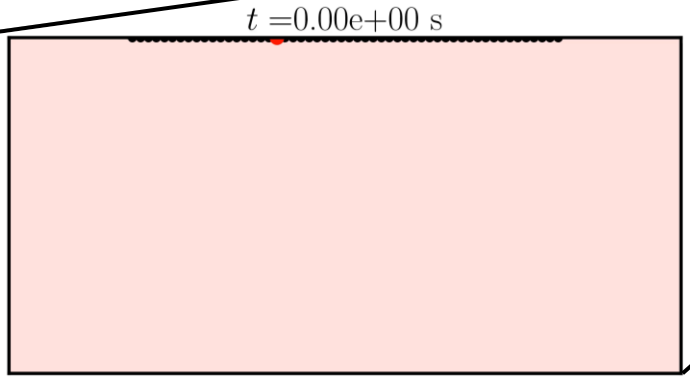


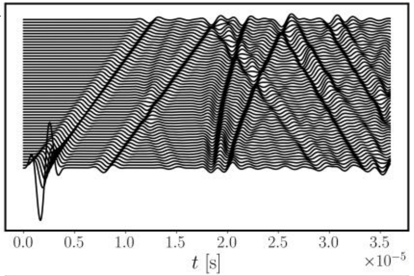
Sinusoidal burst

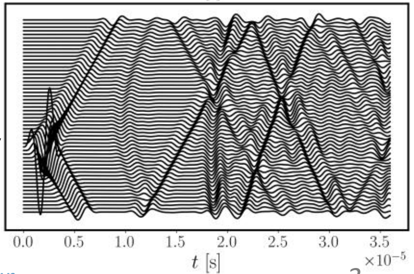
Plate with a hole











We will rely on a technique called **full waveform inversion** for the inversion (which relies on the wave equation)

9.2.1 Acoustic Wave Equation

- The linear acoustic response of a specimen subjected to a Sine burst can be described by the acoustic wave equation (simplification)

In principle, the elastic wave equation should be considered.

- The one-dimensional wave equation with homogeneous Neumann boundary conditions and homogeneous initial conditions reads

$$\frac{\partial^2 u(x,t)}{\partial t^2} - \frac{\partial}{\partial x}\left(c^2(x)\frac{\partial u(x,t)}{\partial x}\right) = p(x,t), \qquad \text{on } \mathcal{T} \times \Omega$$

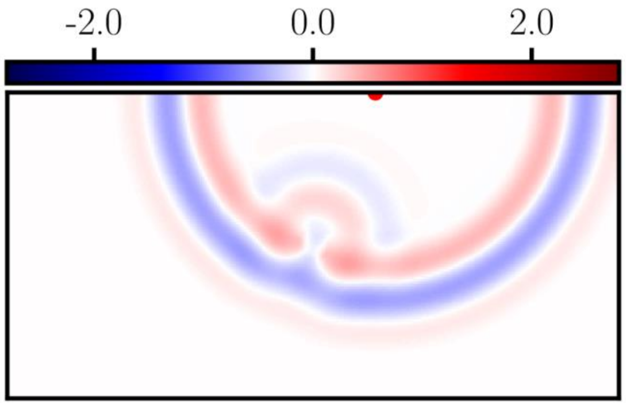
$$\frac{\partial u}{\partial x} = 0, \qquad \text{on } \mathcal{T} \times \Gamma_N$$

$$u(x,0) = u_t(x,0) = 0, \qquad \text{on } \Omega$$

- The primal variable in the wave equation is the wave pressure (wave field) u
- The wave speed c is related to the Young's modulus E and to the mass density ρ

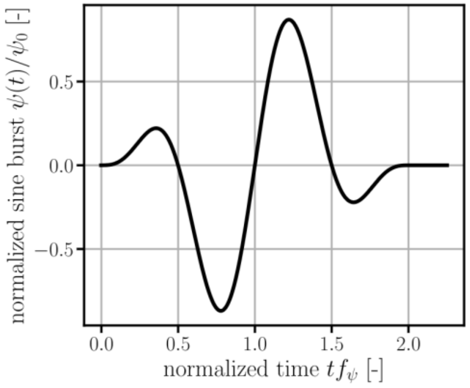
$$c = \sqrt{\frac{E}{\rho}}$$

- **Goal is to recover c from measurements of $\tilde{u}$**



9.2.1 Acoustic Wave Equation

$$\frac{\partial^2 u(x,t)}{\partial t^2} - \frac{\partial}{\partial x}\left(c^2(x)\frac{\partial u(x,t)}{\partial x}\right) = p(x,t), \qquad \text{on } \mathcal{T} \times \Omega$$



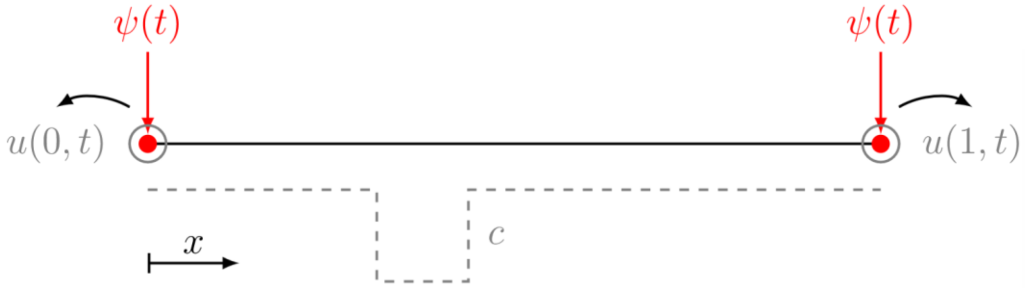
As source $p(x,t)$ a sinusoidal burst is applied

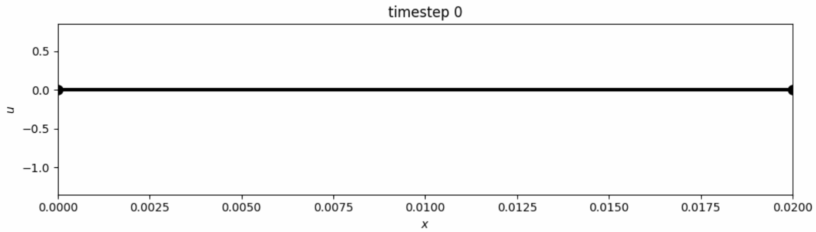
$$\psi(t) = \begin{cases} \psi_0 \sin(\omega t)\sin^2\left(\frac{\omega t}{2\, n_c}\right), & \text{for } 0 \leq t \leq \frac{2\pi n_c}{\omega} \\ 0, & \text{for } \frac{2\pi n_c}{\omega} < t \end{cases}$$

With an amplitude ψ_0, a frequency ω, n_c bursts, and at a specific location x_s (applied via a Dirac delta δ)

$$p(x,t) = \psi(t)\delta(x - x_s)$$

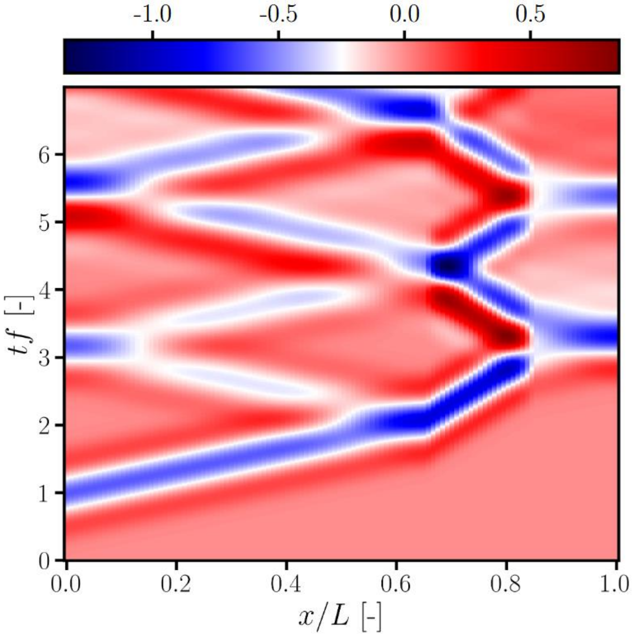
On a one-dimensional specimen the source can be applied as follows (measurements are extracted at $u(0,t), u(1,t)$)



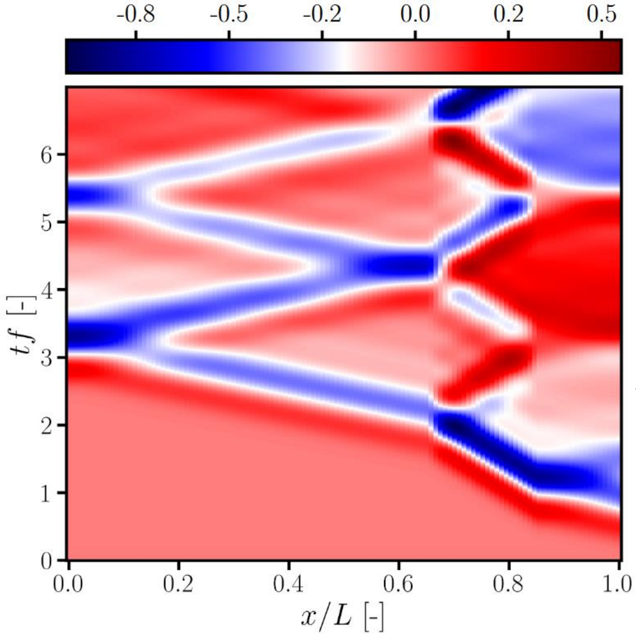


9.2.1 Acoustic Wave Equation – Results

The wave field can be computed with the finite difference method (see 9.2.2)

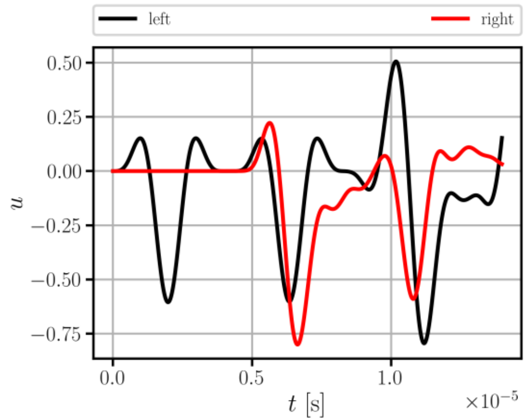


Sinusoidal burst applied on left boundary

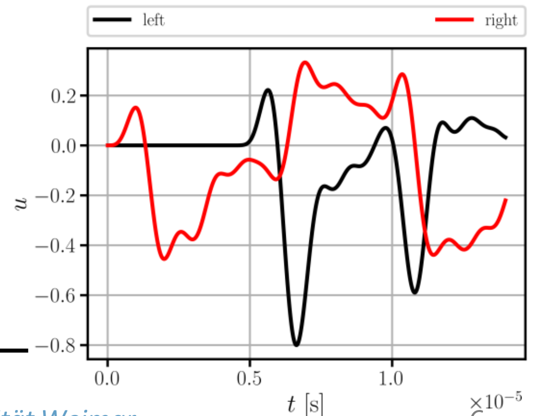


Sinusoidal burst applied on right boundary

Sinusoidal burst applied on left boundary



Sinusoidal burst applied on right boundary

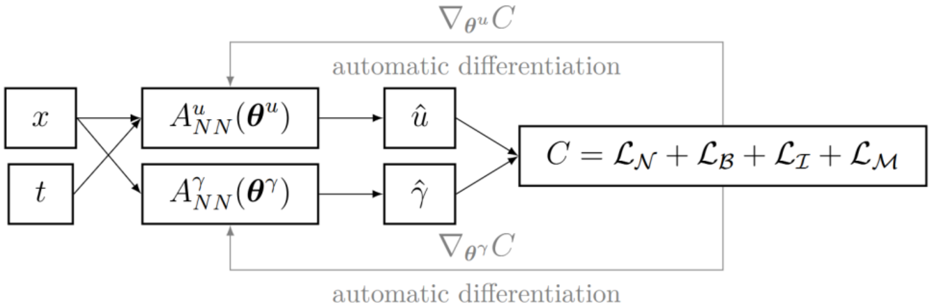


How to identify c from the 4 measurement signals of $u(0,t), u(1,t)$? (not the full field!)
Physics-informed neural networks, iterative forward solver, or data-driven solver?

9.2.3 Physics-Informed Neural Networks

On the use of neural networks for full waveform inversion, Herrmann et al. 2023

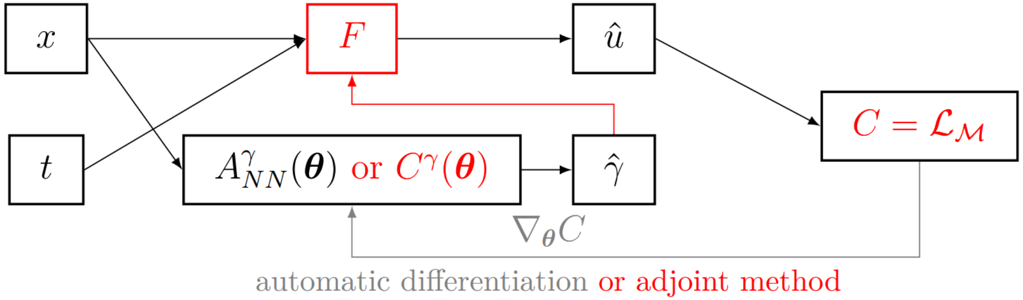
The physics-informed neural network for an inverse problem consists of 2 neural networks A_{NN}^{u} (for u) & A_{NN}^{γ} (for c)



Potentially many local optima

This poses a complex nested optimization (if C improves due to an improved u, C might increase if γ is changed)

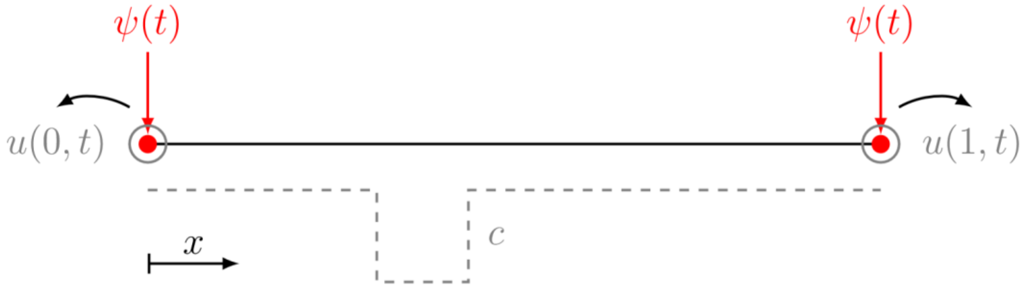
How about exchanging A_{NN}^{u} with an efficient forward solver?

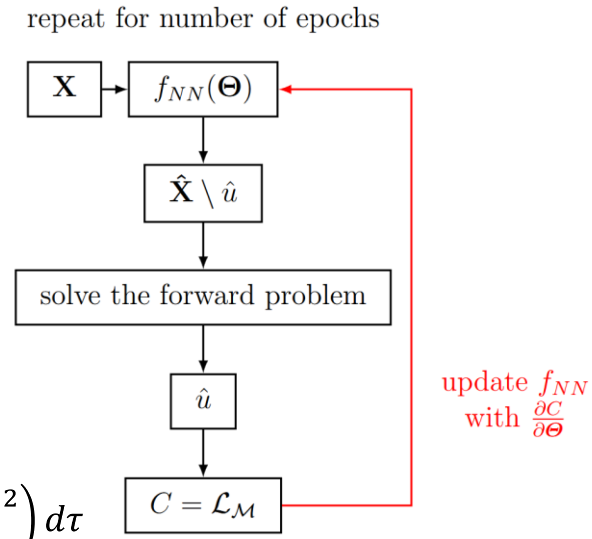


Therefore, we do not further consider physics-informed neural networks for inverse problems (exception: full-field data of u)

This leads to a more efficient scheme, which is analogous to the **iterative forward solver** (see 9.2.4)

9.2.4 Iterative Forward Solver (with Neural Network)





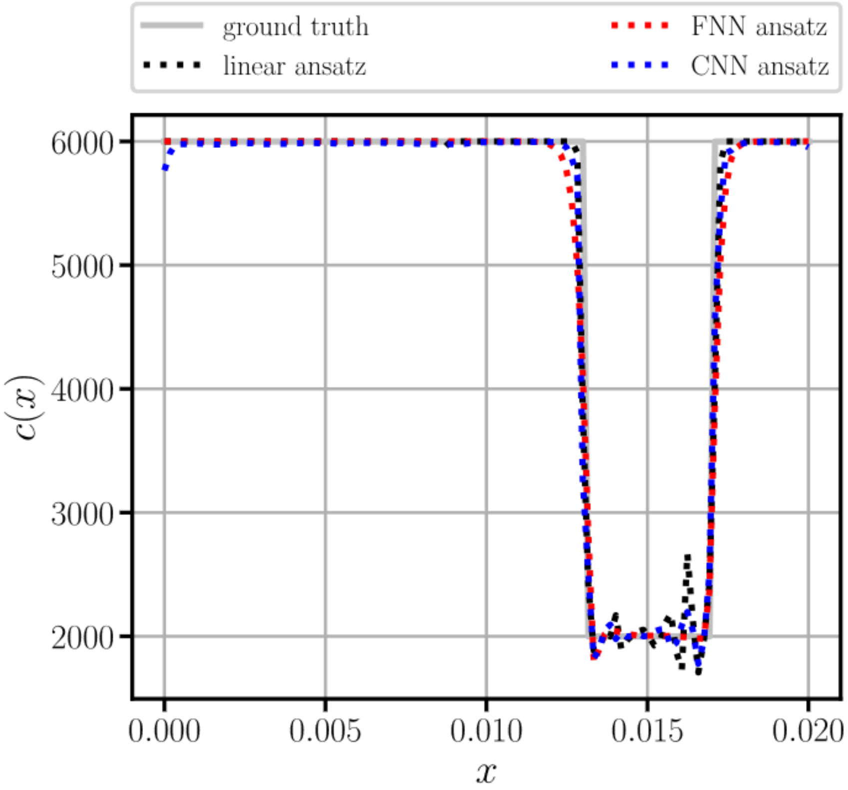
- Parametrization of c with neural network in the domain $x \in [0, L]$
  $\hat{c} = f_{NN}(x; \mathbf{\Theta})$
- Forward solver for $\hat{u}(\hat{c})$ via finite difference method
- Cost function as misfit between measurements $\tilde{u}$ and predictions $\hat{u}$

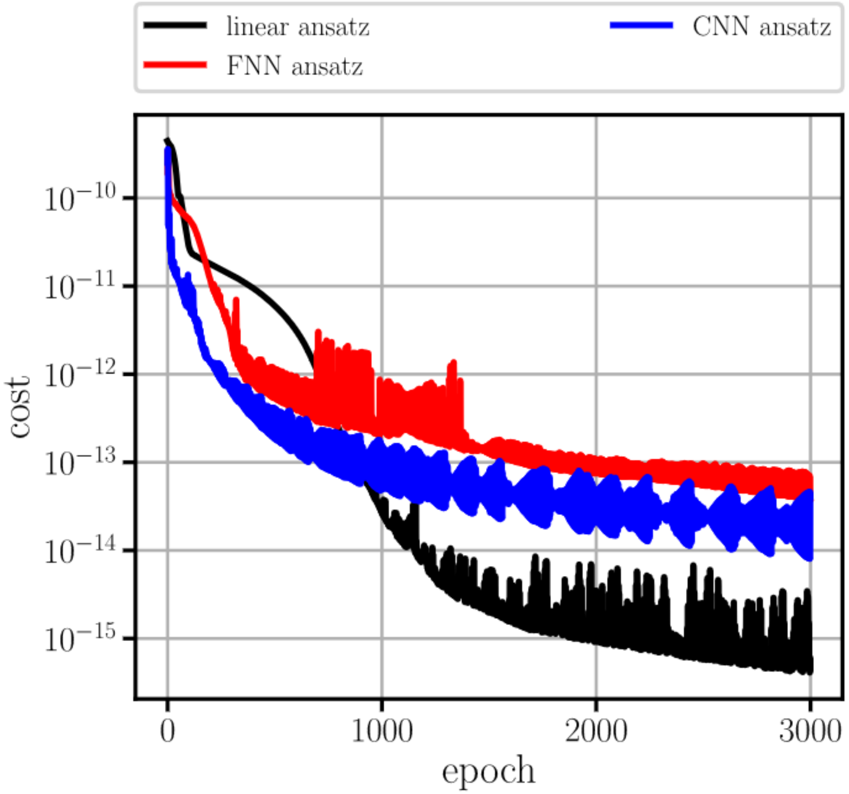
$$C(\tilde{\boldsymbol{u}}; \boldsymbol{p}; \mathbf{\Theta}) = \int_{\tau} \left(\left(\hat{u}(0, t; \hat{c}(\mathbf{\Theta}), \boldsymbol{p}) - \tilde{u}(0, t)\right)^2 + \left(\hat{u}(L, t; \hat{c}(\mathbf{\Theta}), \boldsymbol{p}) - \tilde{u}(L, t)\right)^2 \right) d\tau$$

$$C = \frac{1}{M} \sum_{j=1}^{M} C(\tilde{\boldsymbol{u}}_j; \boldsymbol{p}_j; \mathbf{\Theta})$$

- Where $M = 2$ is the number of experiments (achieved through different sources)
- Gradient $\partial C / \partial \mathbf{\Theta}$ through automatic differentiation (or the adjoint method)
- Gradient-based optimization to find a suitable $\hat{c}$ (defined in terms of $\mathbf{\Theta}$)

9.2.4 Iterative Forward Solver (with Neural Network)





Exercises

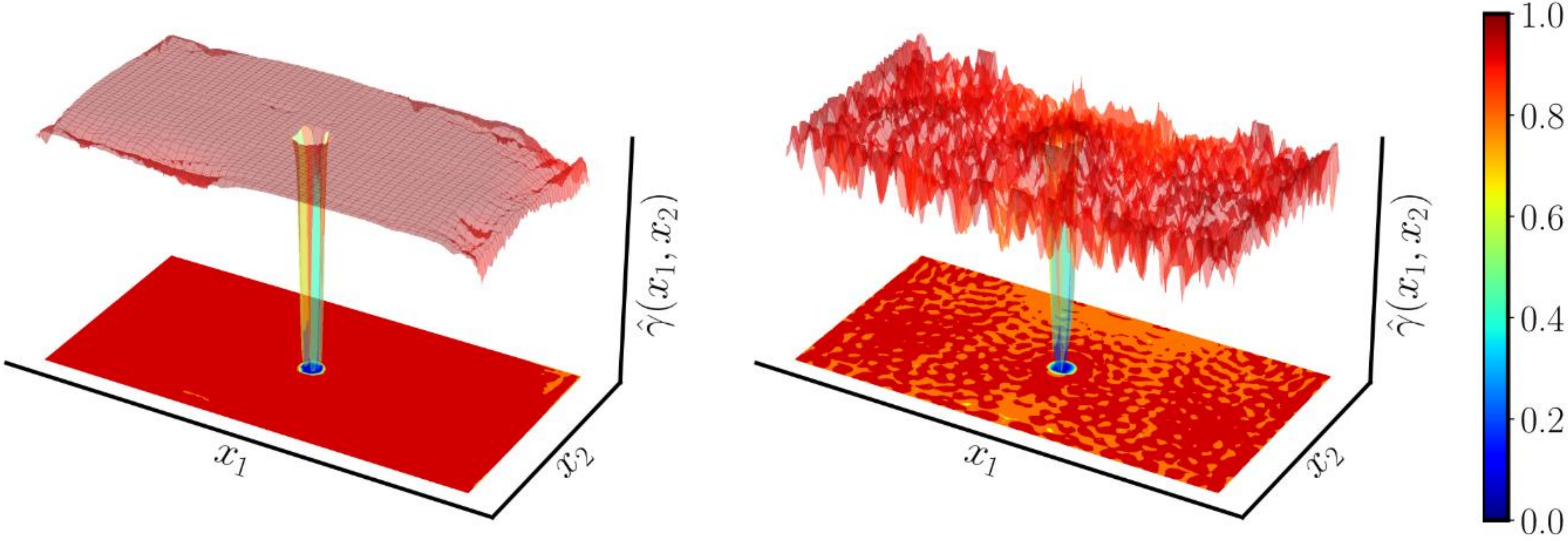
- E.33 Iterative Forward Solver (C)
  - Apply an iterative forward solver to a one-dimensional full waveform inversion example. Tune the hyperparameters and compare the three different ansatz functions.

9.2.4 Iterative Forward Solver with Neural Network

On the use of neural networks for full waveform inversion, Herrmann et al. 2023

Two-dimensional example: plate with a hole:

- Neural network has a regularizing effect, i.e., produces a smoother material field



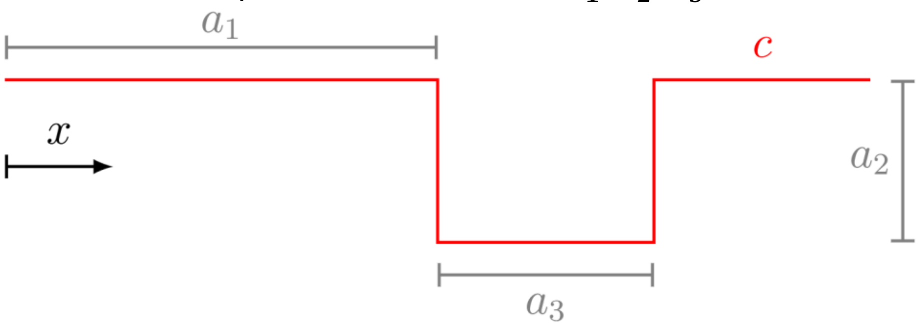
(Convolutional) neural network ansatz

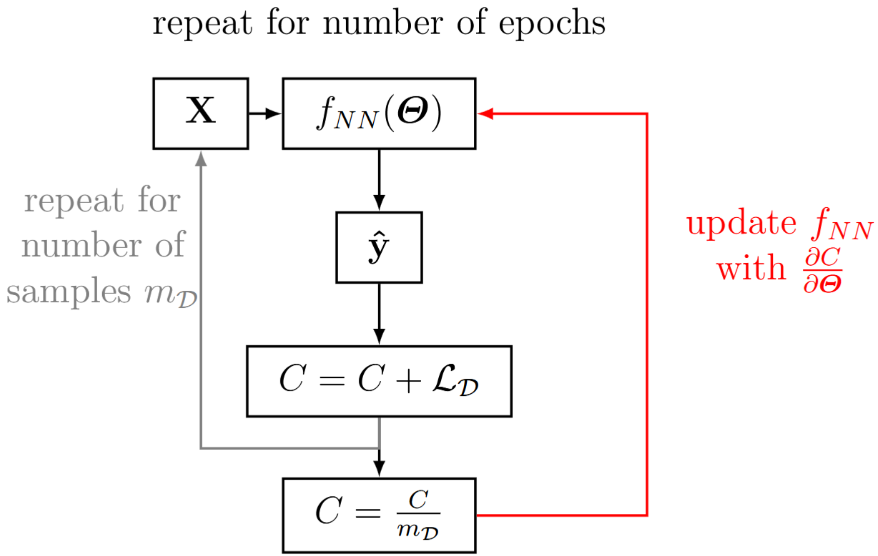
Linear ansatz

9.2.5 Data-Driven Solver

Goal is to learn a mapping between measurements $\tilde{\boldsymbol{u}}$ and the wave speed c

- The neural network relies on a dataset to learn the underlying physics
- A dataset with different c is needed.
  - Defect parametrization with a_1, a_2, a_3





  - Leading to $m = 10'000$ different wave fields
  - Corresponding wave fields $\tilde{\boldsymbol{u}} = \{\tilde{u}_i(0,t), \tilde{u}_i(L,t)\}_{i=1}^{m}$ are computed with the finite difference method
    - Dimensionality of wave speeds: $(10'000 \times 117)$ or $(10'000 \times 3)$

      Due to spatial discretization (117 points) & number of time steps (700)

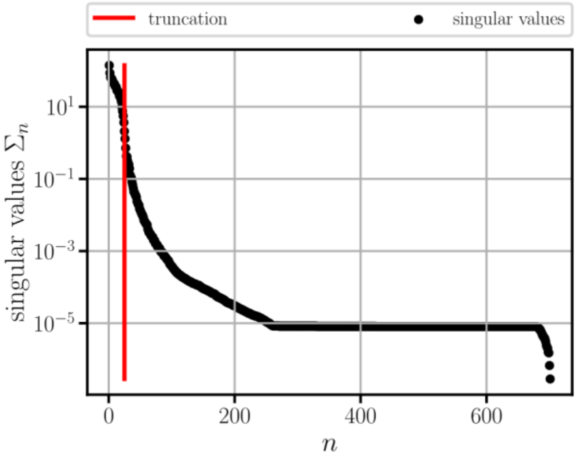
      Relying on a_1, a_2, a_3
    - Dimensionality of wave fields: $(10'000 \times 2 \times 2 \times 700)$

9.2.5 Data-Driven Solver

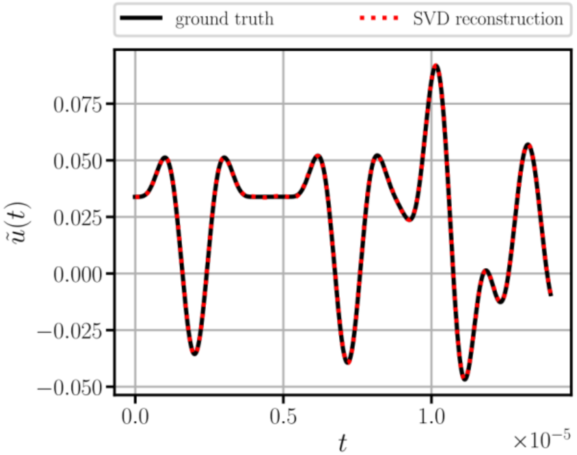
- Learning a mapping from $\tilde{u} \in (2 \times 2 \times 700)$ to $c \in (117)$ is challenging
- Dimensionality reduction techniques (from Chapter 6) might be beneficial
  - A low-dimensional representation of $\tilde{u}$ can be obtained through a singular value decomposition
  - This yields an orthogonal transformation matrix $\boldsymbol{V}$ (truncated: $\overline{\boldsymbol{V}}$), enabling the transformation to and from $\bar{\tilde{u}}$

$$\bar{\tilde{u}} = \tilde{u} \cdot \overline{\boldsymbol{V}}^T$$
$$\tilde{u} \approx \bar{\tilde{u}} \cdot \overline{\boldsymbol{V}}$$

  - At a truncation level of $r = 25$ $\tilde{u}$ can still be reconstructed
  - Thus $\tilde{u} \in (2 \times 2 \times 700)$ can be reduced to $\bar{\tilde{u}} \in (2 \times 2 \times 25)$



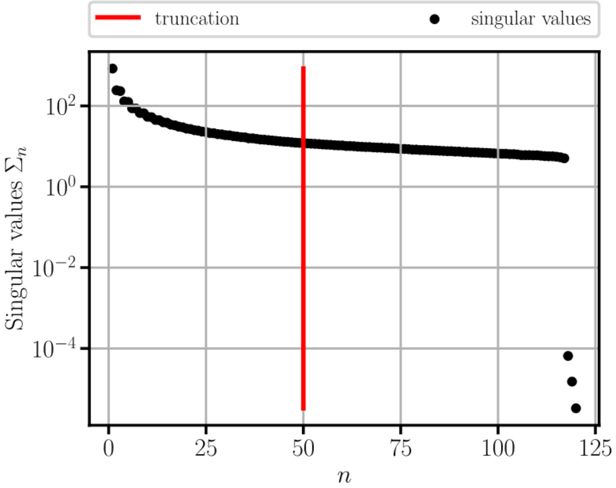
Singular values



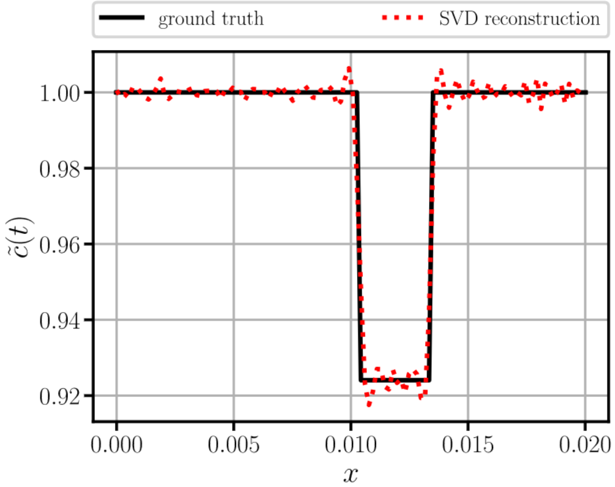
Reconstruction at truncation level $r = 25$

9.2.5 Data-Driven Solver

Can we compress $c \in (117)$ with a singular value decomposition?



Singular values



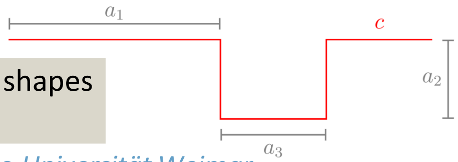
Reconstruction at truncation level $r = 50$

No, because the discontinuities require a nonlinear transformation

Could be achieved by nonlinear techniques, such as autoencoders (see Chapter 8)

For simplicity, we rely on the chosen parametrization a_1, a_2, a_3, leading to a reduction of $c \in (117)$ to $c \in (3)$

This limits the solver to only identify shapes represented by a_1, a_2, a_3

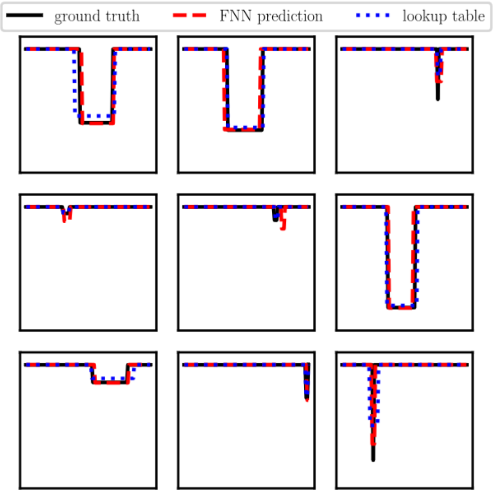


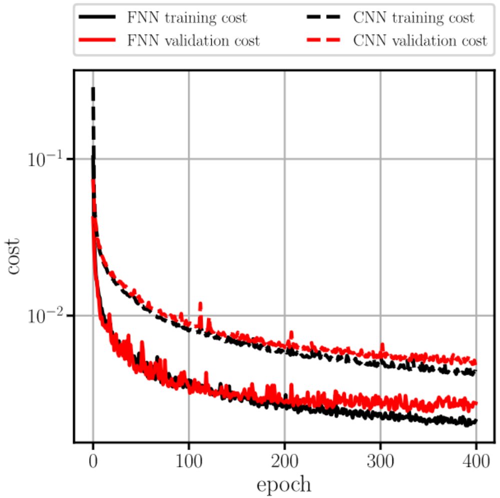
9.2.5 Data-Driven Solver

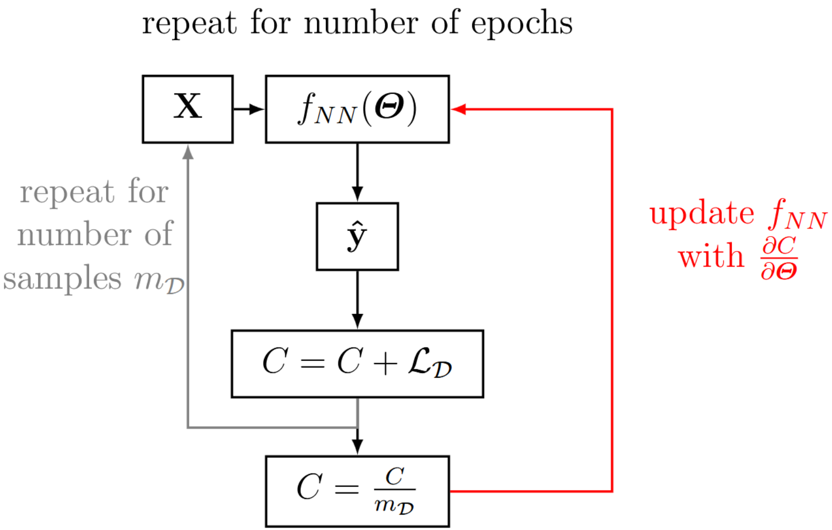
- Mapping between $\tilde{\bar{u}} \in (2 \times 2 \times 25)$ (i.e., (200)) and $c \in (3)$ via fully connected neural network (200 input neurons & 3 output neurons)
- Data-driven cost function

$$C = \frac{1}{m}\sum_{i=1}^{m}\left(\left(\tilde{a}_{1_i} - \hat{a}_{1_i}\right)^2 + \left(\tilde{a}_{2_i} - \hat{a}_{2_i}\right)^2 + \left(\tilde{a}_{3_i} - \hat{a}_{3_i}\right)^2\right)$$

- Minimization via Adam







Exercises

- E.34 Data-Driven Solver (C)
  - Train a data-driven solver to learn the mapping between wave field and the spatial wave velocity distribution. Begin with a dataset generation and a linear dimensionality reduction using singular value decomposition.

9.2.5.1 Transfer Learning

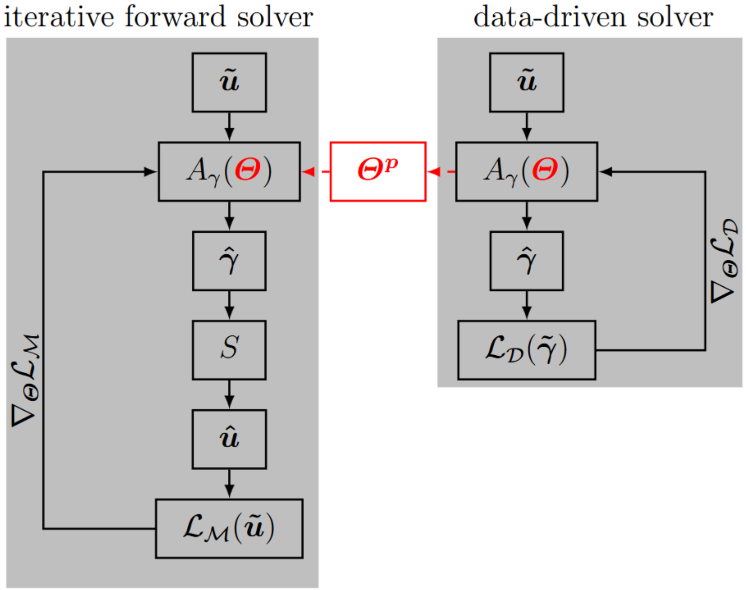
Accelerating full waveform inversion by transfer learning, Singh et al. 2025

- The data-driven solver enables fast online prediction, but is potentially unreliable
- The iterative forward solver is reliable, but expensive (1 gradient-based optimization per inversion)

Can we combine the two methods to form a reliable and fast method?

Transfer Learning

- is a technique in which knowledge learned from a task is reused to boost performance on a related task
- The initial training is called pretraining



- Idea is to use the data-driven solver as pretraining

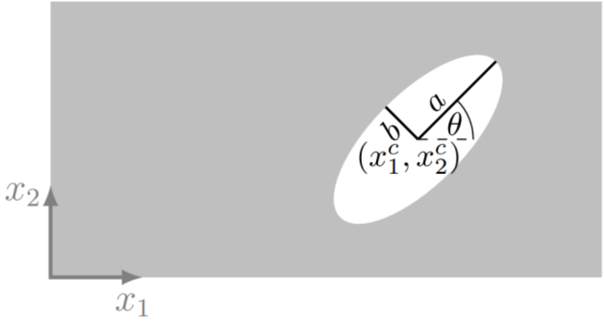
$$\mathcal{L}_D = \frac{1}{2m_D}\sum_{i=1}^{m_D}\int_\Omega \left(\hat{\gamma}_i(\boldsymbol{\Theta};\boldsymbol{x}) - \tilde{\gamma}_i(\boldsymbol{x})\right)^2 d\Omega$$

- Ensure a reliable identification with subsequent iterative forward solver

$$\mathcal{L}_M = \frac{1}{2}\int_\mathcal{T}\int_\Omega \sum_{i=1}^{m_M}\left(\hat{u}(\hat{\gamma}(\boldsymbol{\Theta};\boldsymbol{x}),t) - \tilde{u}(\boldsymbol{x}_i,t)\right)^2 \delta(\boldsymbol{x}-\boldsymbol{x}_i)\,d\Omega\, d\mathcal{T}$$

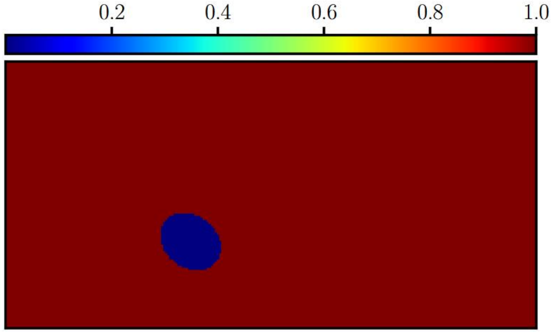
9.2.5.1 Transfer Learning

Dataset consisting of ellipsoidal defects for the data-driven solver (parametrization via $a, b, x_1^c, x_2^c, \theta$)

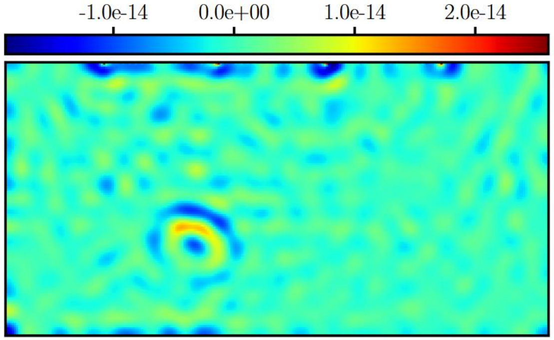


Data generation is cheap in comparison to solving the inverse problem (each forward computation is ~ 100 cheaper than solving 1 inverse problem) → more favorable breakeven threshold τ

Mapping via convolutional neural network (U-Net) between initial gradient $\nabla_\gamma C$ and true material distribution γ



Material distribution γ



Initial gradient $\nabla_\gamma C$

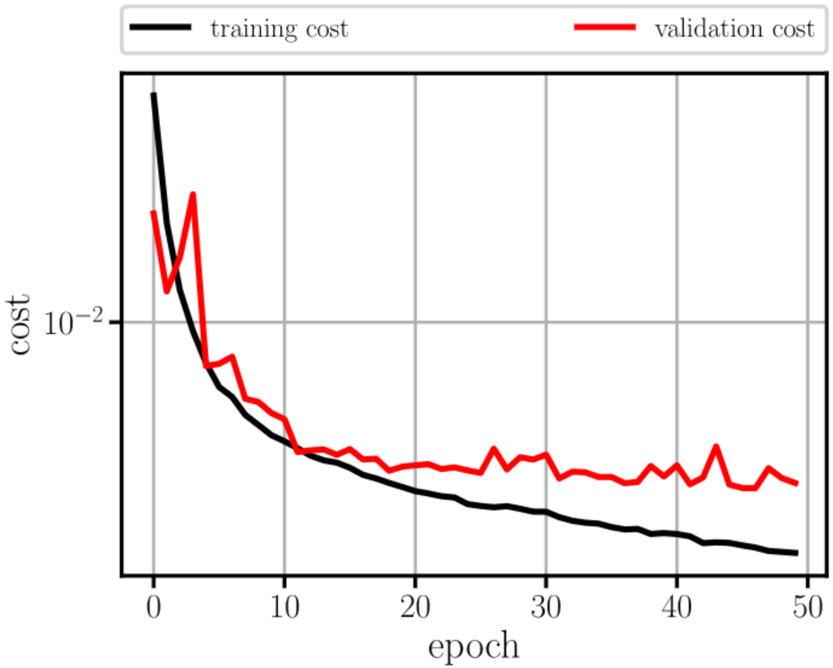
Input and output have the same shape (signals have shape (# of sensors × # of sources × # of timesteps), which is challenging to convert to (# of points in x × # of points in y)

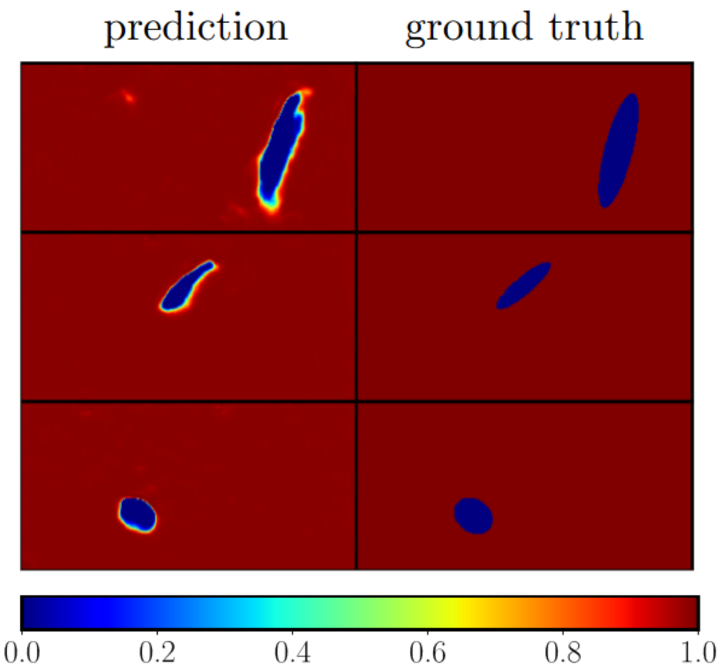
9.2.5.1 Transfer Learning

Accelerating full waveform inversion by transfer learning, Singh et al. 2025

Predictions of the data-driven solver after training with 1‘800 samples

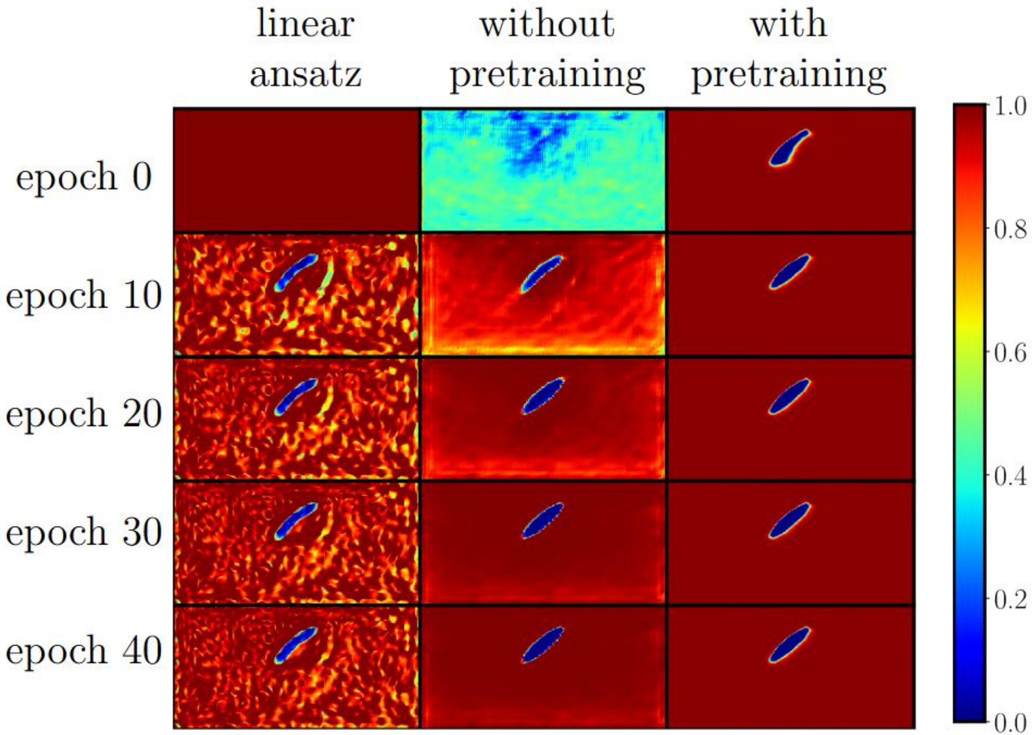
~100 samples are sufficient to produce similar results





9.2.5.1 Transfer Learning

Acceleration through transfer learning

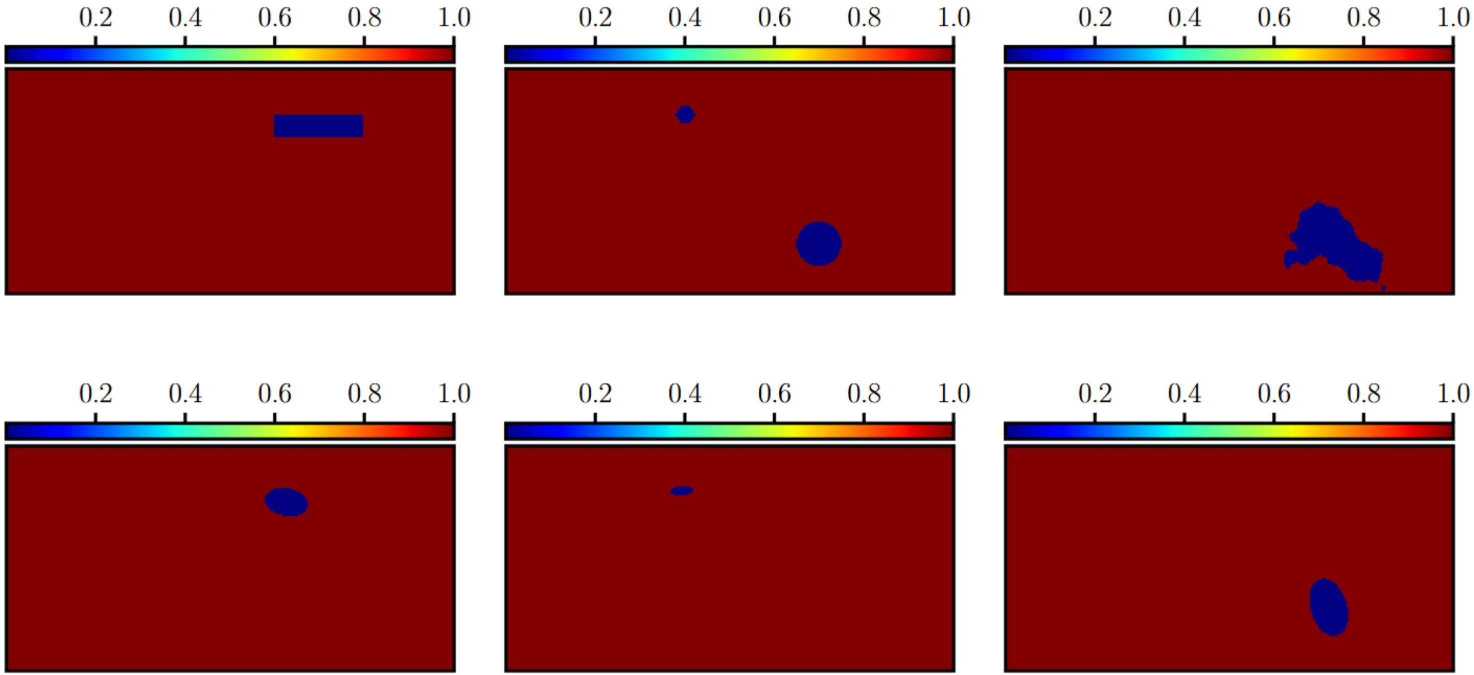


9.2.5.1 Transfer Learning

But what about material outside of the training dataset?

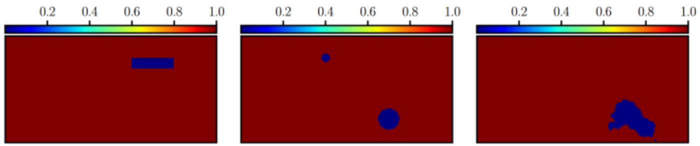
Test cases

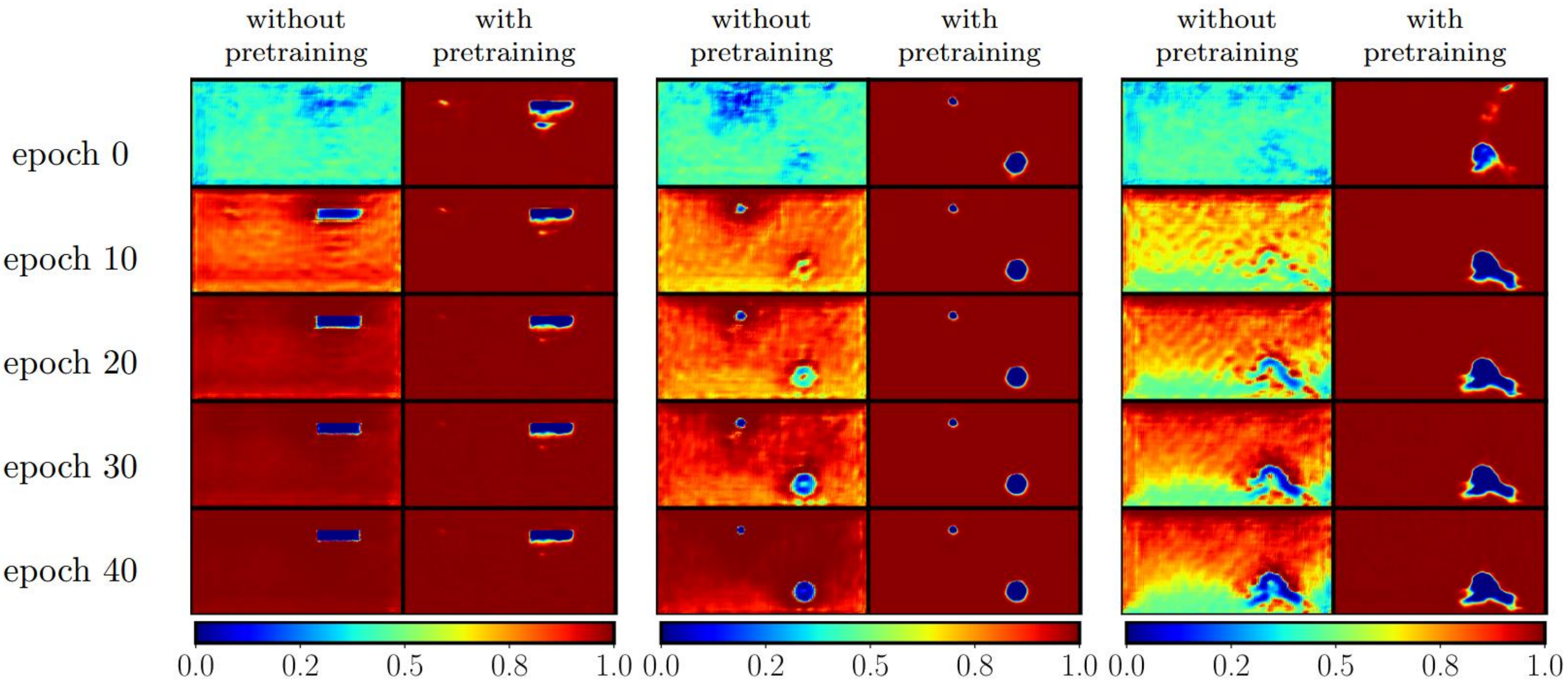
Closest material distribution in training dataset



9.2.5.1 Transfer Learning

Acceleration through transfer learning persists





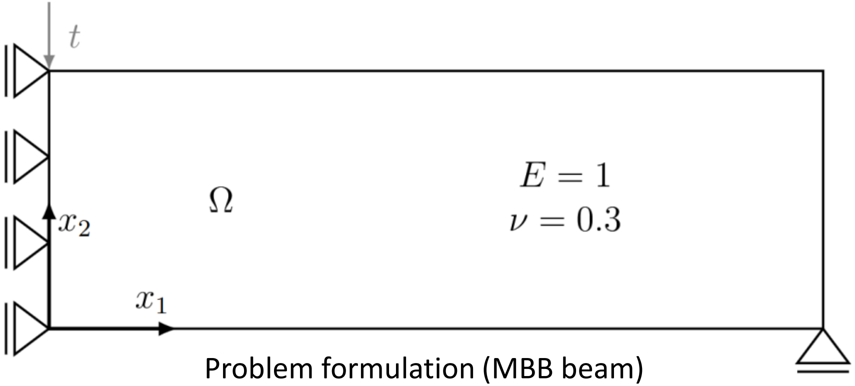
9.3 Topology Optimization

Topology Optimization: Theory, Methods, and Applications, Bendsøe et al. 2004

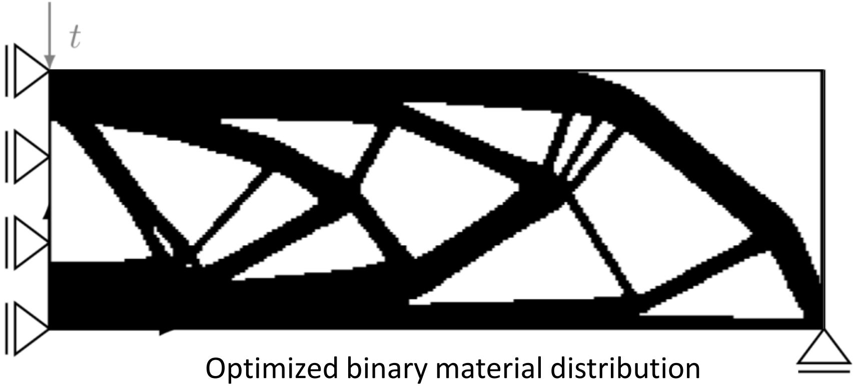
Topology optimization is a task from computational mechanics that can be framed as minimization problem to

- optimize material layout/distribution
- based on an objective $\mathcal{L}_O$ (e.g., stiffness maximization)
- under constraints (e.g., material volume, boundary conditions, material properties…)

- Typically (most efficiently) achieved through gradient-based optimization



Problem formulation (MBB beam)



Optimized binary material distribution

9.3 Topology Optimization

On the use of artificial neural networks in topology optimisation, Woldseth et al. 2022

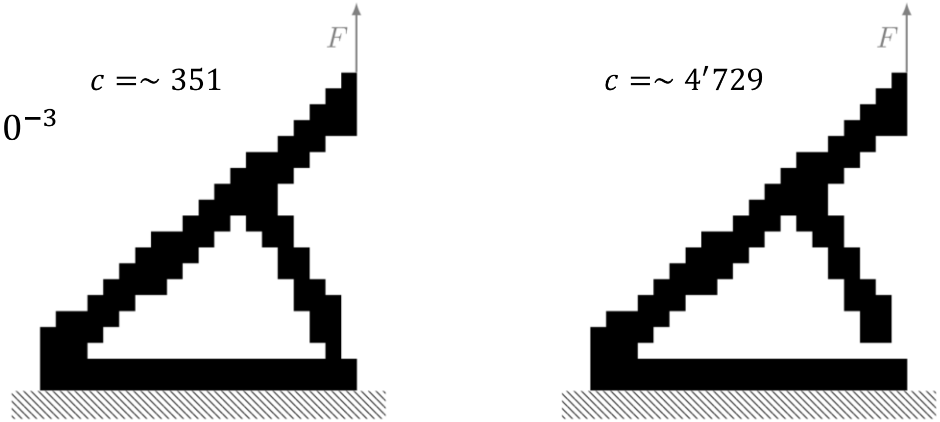
Unviable to learn forward and inverse operator in a nested optimization

How to integrate neural networks into topology optimization?

- Physics-informed neural networks → unsuitable for inverse problems without full-domain knowledge
- Data-driven solvers → minor errors lead to major structural discrepancies (& too expensive data generation)

Compliance $c = \boldsymbol{F}^T \boldsymbol{u}$

Difference in MSE: $\sim 2.5 \cdot 10^{-3}$



Transfer learning infeasible because in topology optimization:

- a full inversion must be performed for each data pair to be learnt → lack of good pretrained models

Generative approaches do exist, (not treated here, see Chapter 8)

What about a neural network ansatz in an iterative forward solver?

9.3.3.1 Compliance Optimization

Maximization of stiffness is formulated as the minimization of the compliance

$$c = \boldsymbol{F}^T \boldsymbol{u}$$

Extracted from the FEM system $\boldsymbol{K}\boldsymbol{u} = \boldsymbol{F}$

Parametrization of the domain via (linear) finite elements (K_e is the stiffness matrix of element e)

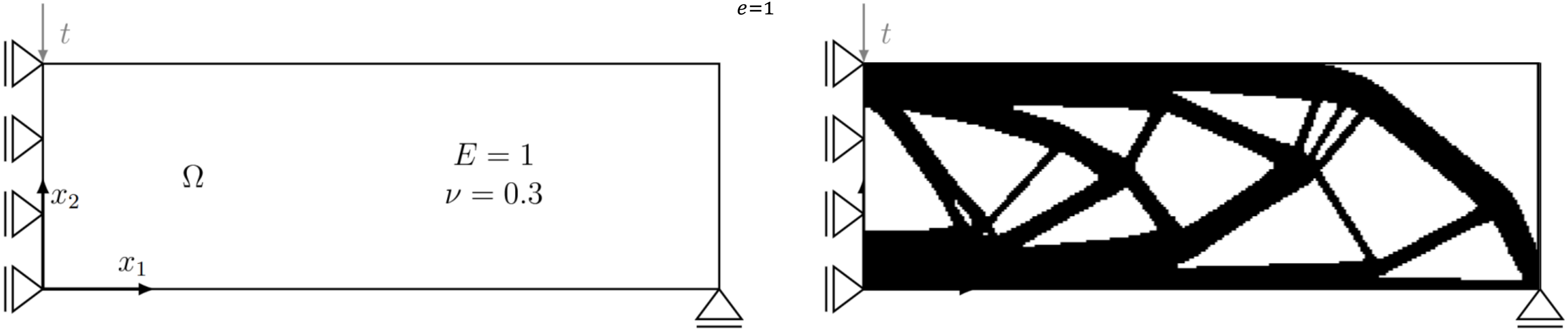
$$K = \sum_{e=1}^{N} \gamma_e^p K_e$$

γ_e scales the stiffness of the element i.e. is the design variables

γ_e is the density that defines the material distribution & $p \geq 3$ is an exponent that punishes material between 0 and 1.

The optimization is subject to a volume constraint

$$V = \sum_{e=1}^{N} v_e \gamma_e \leq \tilde{V}$$



9.3.3 Iterative Forward Solver with Neural Network

TOuNN: Topology Optimization using Neural Networks, Chandrasekhar et al. 2021

Instead of optimizing the design variables γ_e directly, a neural network ansatz is employed

$$\hat{\gamma}_e = f_{NN}(\boldsymbol{x}; \boldsymbol{\Theta})$$

Difference to full waveform inversion:

- Full waveform inversion does not require a constrained optimizer
- Topology optimization for the compliance problem needs a constrained optimizer (e.g., volumetric constraint)
  - Volume constraint is typically enforced strongly through the optimizer (e.g., optimality criterion (OC))
  - practically impossible with the neural network relying on Adam
    (exact updates of γ_e are not transferable to $\boldsymbol{\Theta}$, only directions $\nabla_{\boldsymbol{\Theta}} C$)
  - With a neural network ansatz, the volume constraint is incorporated via a penalty term

$$C = c + \kappa\left(\sum_{e=1}^{N} v_e \gamma_e - \tilde{V}\right)^2$$

  - Where κ is the penalty factor that increases during training

  - After training, gray values of γ_e remain (i.e., values that are not 0 or 1): These are handled by thresholding the result to a binary distribution

9.3.3 Iterative Forward Solver with Neural Network

Optimized structure after 500 epochs

Linear ansatz

Neural network ansatz

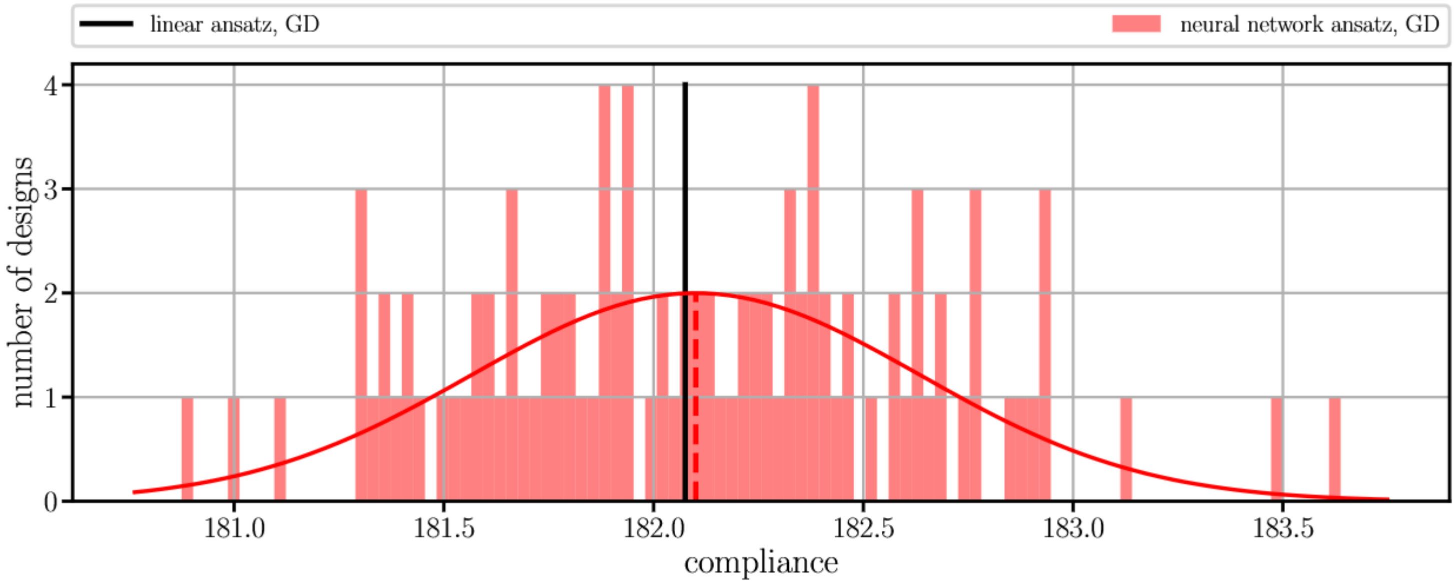
	linear ansatz, GD	neural network ansatz, GD	linear ansatz, OC
c before thresholding	$\sim$187.7	$\sim$190.3	$\sim$189.3
c after thresholding	$\sim$182.1	$\sim$181.5	$\sim$185.2
V after thresholding	0.502	0.500	0.500

Minor improvement at a slightly greater expense through gradient descent with Adam instead of optimality criterion

But are these improvements consistent? (different designs are possible through different initializations of $\boldsymbol{\Theta}$)

9.3.3 Iterative Forward Solver with Neural Network

Statistical evaluation of 100 samples generated with the neural network ansatz



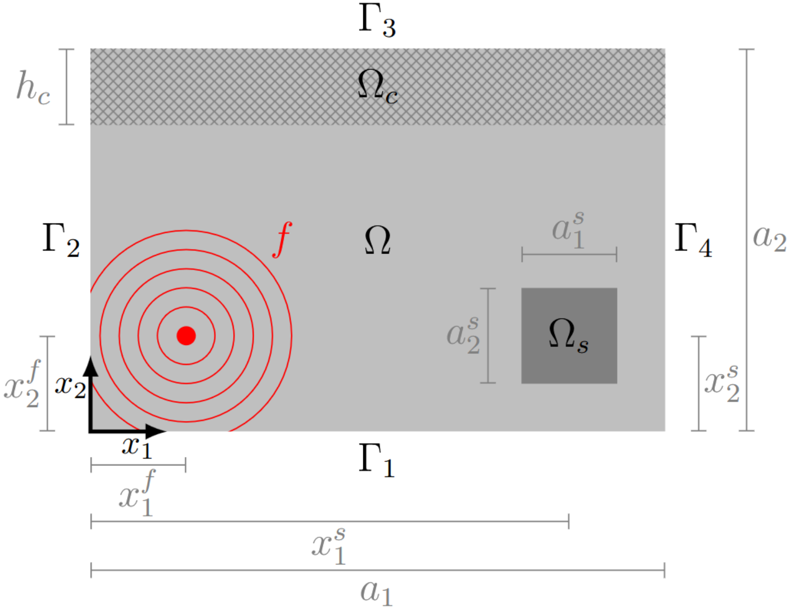
Limited improvements ($< 1.5\%$), despite higher computational effort & no consistent improvements

Benefit is limited: Neural network does not find better local optima, as the optimization problem is not challenging

9.3.3.2 Acoustic Topology Optimization

Acoustic design by topology optimization, Dühring et al. 2008

- The goal is to distribute material to achieve an acoustic target (e.g., noise supression)
- Example: topology optimization of a ceiling for noise supression



Ω_c is the domain to be optimized, i.e., the ceiling
Ω_s is the domain to be suppressed
Ω is the computational domain in which the sound propagates
f is the frequency of the source p

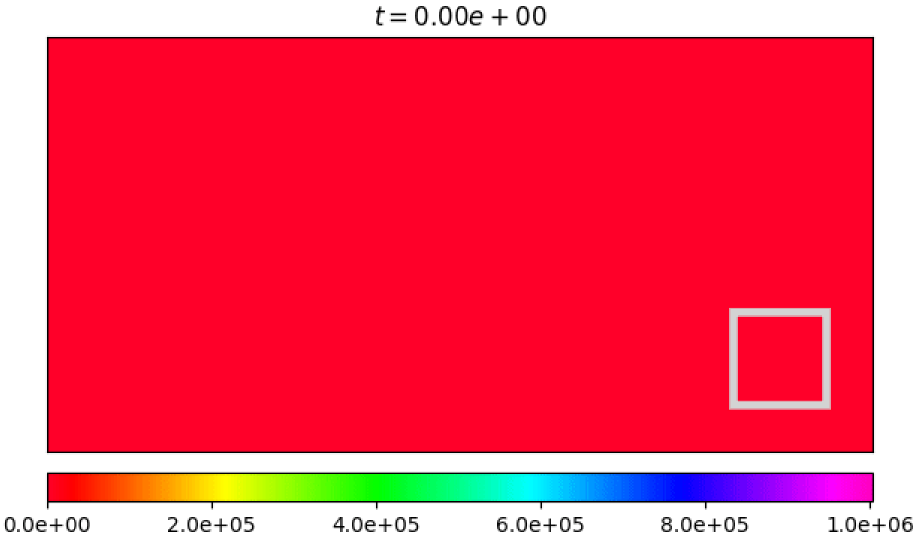
- Acoustic topology optimization
  - Is prone to end in premature local optima (very different designs possible)
  - Is an unconstrained optimization problem

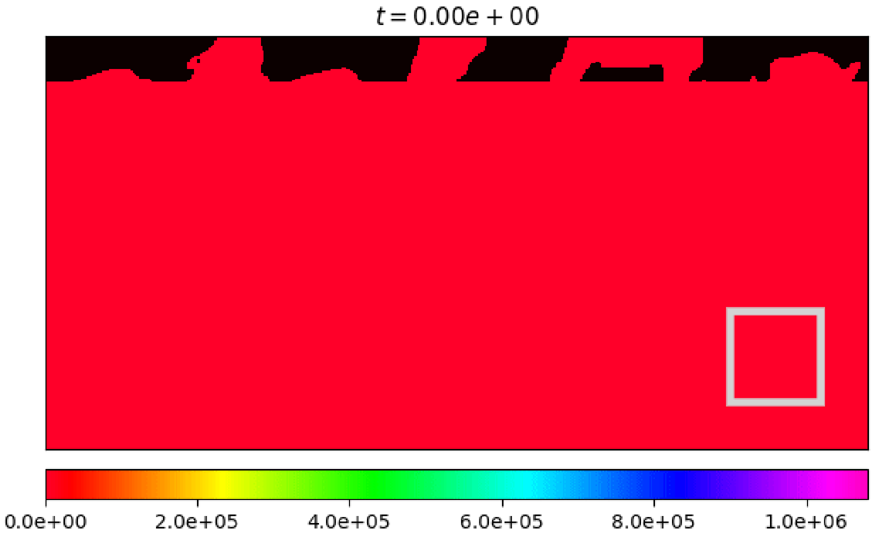
→ suitable for a neural network ansatz

Bauhaus-Universität Weimar

9.3.3.2 Acoustic Topology Optimization

Optimized ceiling





9.3.3.2 Acoustic Topology Optimization

Optimization is achieved through the minimization of the wave pressure u in the domain Ω_s

$$C = \frac{1}{\int_{\Omega_s} d\Omega_s} \int_{\Omega_s} |u|^2 d\Omega_s$$

We are only optimizing for a source with a single frequency

The wave pressure is obtained by solving the Helmholtz equation (wave equation in the frequency domain)

$$\nabla \cdot \left(\tilde{\rho}^{-1} \nabla u(x)\right) + i\tilde{\omega}\eta_d \tilde{\kappa}^{-1} u(x) + \tilde{\omega}^2 \tilde{\kappa}^{-1} u(x) = p(x)$$

The mass density & bulk modulus are parametrized via the material γ

$$\tilde{\rho}^{-1}(\gamma) = 1 + \gamma \left(\frac{\rho_1}{\rho_2} - 1\right)$$

$$\tilde{\kappa}^{-1}(\gamma) = 1 + \gamma \left(\frac{\kappa_1}{\kappa_2} - 1\right)$$

$\tilde{\omega}$ is the normalized angular frequency

η_d is a damping factor

$\tilde{\kappa}$ is the normalized bulk modulus

$\tilde{\rho}$ is the normalized mass density

Subindex 1 indicates properties of air

Subindex 2 indicates properties of the solid

p_0 is the reference pressure

Quality of the optimized ceiling is assessed by the sound pressure level L_p in the domain Ω_s

$$L_p = 10 \log_{10} \left(\frac{|p|^2}{p_0^2}\right)$$

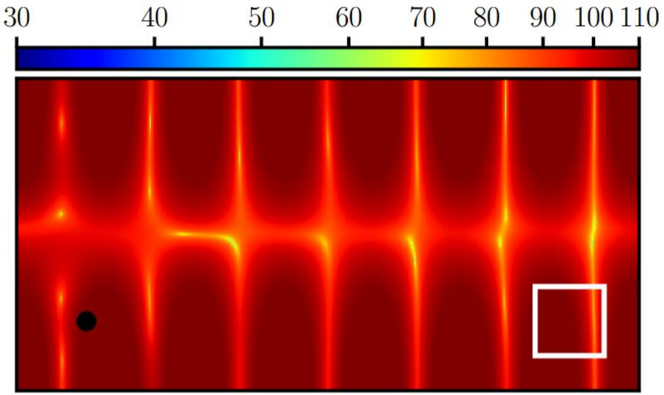
In the iterative forward solver, the material distribution γ is represented by a neural network

$$\hat{\gamma} = f_{NN}(x; \mathbf{\Theta})$$

9.3.3 Iterative Forward Solver with Neural Network

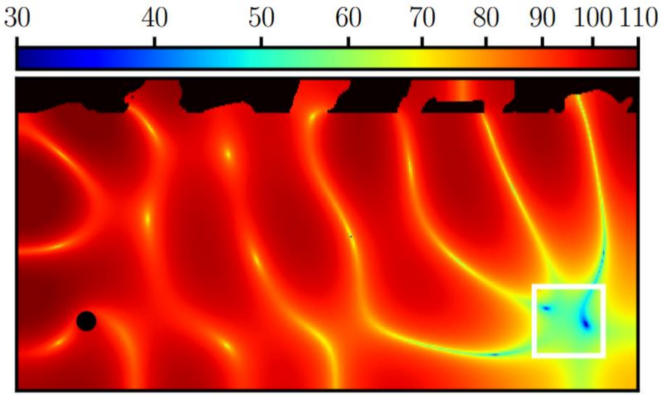
On Neural Networks for Generating Better Local Optima in Topology Optimization, Herrmann et al. 2024

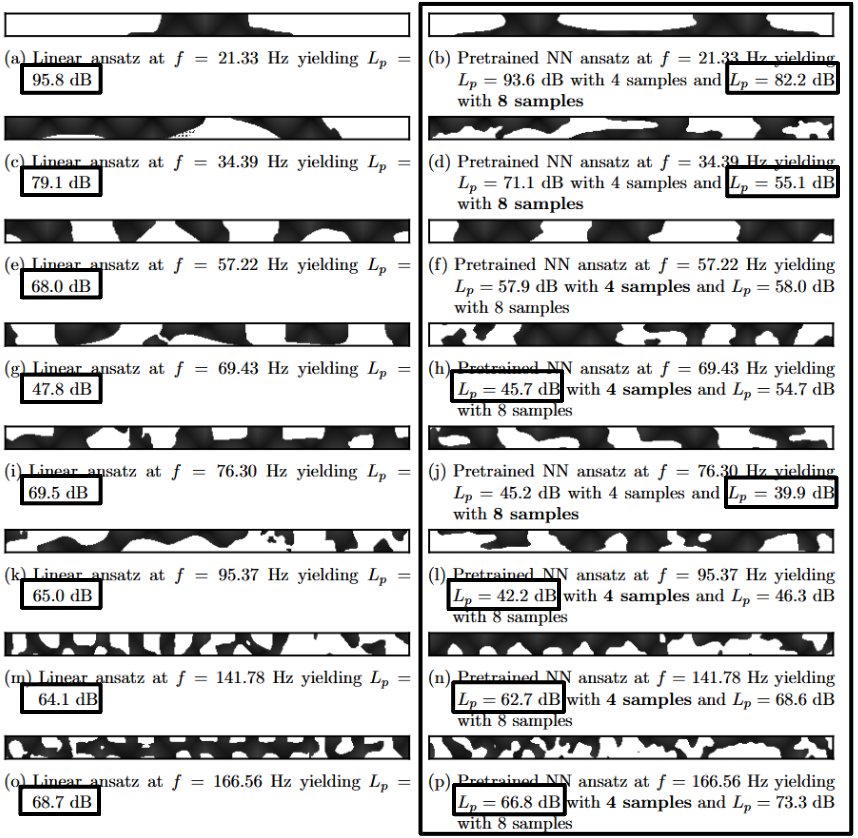
Without optimized ceiling: $L_p = 110$ dB



For $f = 69.43$ Hz

With optimized ceiling: $L_p = 61.3$ Hz



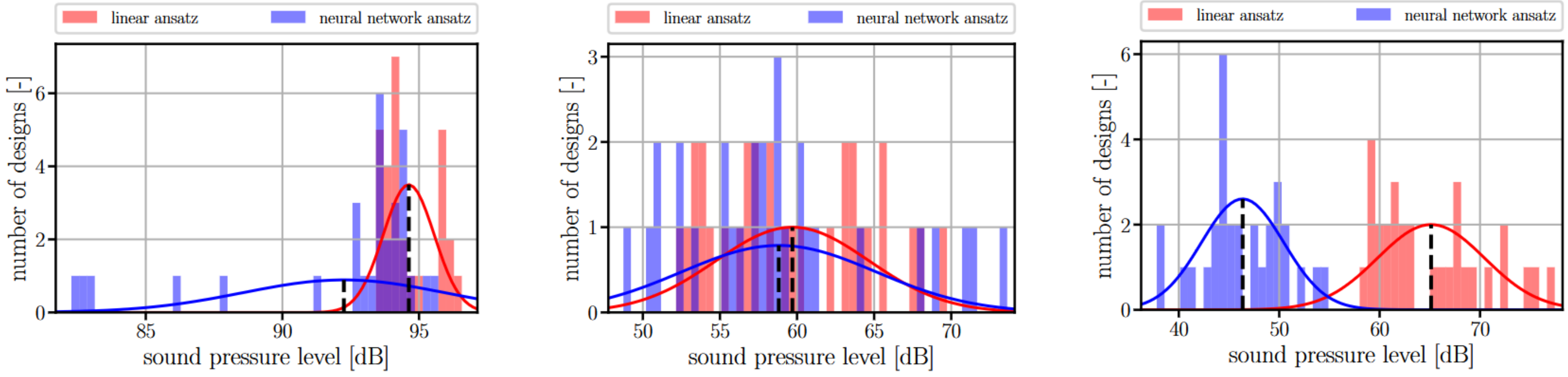


9.3.3 Iterative Forward Solver with Neural Network

On Neural Networks for Generating Better Local Optima in Topology Optimization, Herrmann et al. 2024

Statistical evaluation of 30 samples generated with

- The neural network ansatz (different initializations of $\boldsymbol{\Theta}$)
- The linear ansatz (different uniform initializations between 0 and 1)



$f = 21.33$ Hz $f = 57.22$ Hz $f = 76.30$ Hz

For the acoustic topology optimization problem, the <u>neural network ansatz is beneficial</u>, as better designs can be (and are frequently) generated

Contents

Bauhaus-Universität
Weimar

9 Inverse Problems & Deep Learning: Applications

Leon Herrmann

Stefan Kollmannsberger

Chair of Data Engineering in Construction

Bauhaus-Universität Weimar

Deep Learning in Computational Mechanics – an introductory course, Herrmann et al. 2025

website

book