Neuroexplicit Models for Data Processing

A Dissertation Submitted Towards
the Degree Doctor of Natural Sciences (Dr. rer. nat.)
of the Faculty of Mathematics and Computer Science of
Saarland University

submitted by
Karl Schrader
Saarbrücken, 2025

DAY OF COLLOQUIUM:
26.02.2026

DEAN OF FACULTY:
Prof. Dr. Roland Speicher

CHAIR OF THE COMMITTEE:
Prof. Dr. Isabel Valera

REVIEWERS:
Prof. Dr. Joachim Weickert
Prof. Dr. Guy Gilboa

ACADEMIC ASSISTANT:
Dr. Vassillen Chizhov

to my parents

UNIVERSITÄT
DES
SAARLANDES

Eidesstattliche Versicherung

Hiermit versichere ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus anderen Quellen oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet. Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder ähnlicher Form in einem Verfahren zur Erlangung eines akademischen Grades vorgelegt.

Declaration of original authorship

I hereby declare that this dissertation is my own original work except where otherwise indicated. All data or concepts drawn directly or indirectly from other sources have been correctly acknowledged. This dissertation has not been submitted in its presentor similar form to any other academic institution either in Germany or abroad for the award of any other degree.

Saarbrücken, 1.10.2025

gez. / signed
Karl Schrader

Short Abstract

This thesis explores hybrid approaches that integrate neural networks into model-based methods for image processing. Model-driven techniques based on differential equations and variational methods offer strong theoretical foundations and mathematical guarantees. If one expresses their goals and the algorithms to achieve them as mathematical equations, they become explicit and human-interpretable. Conversely, neural networks excel across a wide range of tasks but still remain black boxes. To bridge this gap, we investigate how the potential of neural networks can be leveraged by model-based approaches while preserving their interpretability. First, we focus on the model-based side. We propose a new derivation for a class of discretisations, which can then be implemented using typical neural network building blocks for a fast implementation. Next, we demonstrate that even simple neural solvers can effectively address numerically challenging problems, achieving good results for the inpainting of images and sound fields. Finally, we propose a neural mask optimisation framework for diffusion-based inpainting. It leads to the first neural method capable of handling high-resolution images by harmoniously combining model-based principles with neural optimisation. By embedding neural solutions within the structure of established model-based frameworks, our work advances the field of neuroexplicit computing and constructs interpretable yet performant approaches.

Kurzzusammenfassung

Diese Arbeit untersucht hybride Ansätze, die neuronale Netze in modellbasierte Verfahren der Bildverarbeitung integrieren. Modellgetriebene Techniken, die auf Differenzialgleichungen und Variationsmethoden basieren, bieten robuste theoretische Grundlagen und mathematische Garantien. Wenn deren Ziele und eingesetzte Algorithmen als Gleichungen ausgedrückt werden, sind diese explizit beschrieben und lassen sich klar interpretieren. Im Gegensatz dazu liefern neuronale Netze bei einer Vielzahl von Aufgaben gute Ergebnisse, bleiben aber weiterhin „Black Boxes“. Um beide Welten zusammenzuführen, untersuchen wir, wie neuronaler Netze in modellbasierte Ansätze integriert werden können, ohne deren Interpretierbarkeit zu beeinträchtigen. Wir schlagen eine neue Herleitung für eine Klasse von Diskretisierungen vor, die sich mittels typischer Bausteine neuronaler Netze effizient implementieren lässt. Weiterhin zeigen wir, dass bereits einfache neuronale Löser numerisch anspruchsvolle Probleme effektiv bewältigen und dabei gute Ergebnisse beim Inpainting von Bildern und Schallfeldern erzielen können. Schließlich stellen wir ein neuronales Maskenoptimierungs-Framework für diffusionsbasiertes Inpainting vor. Die harmonische Kombination von modellbasierten Prinzipien und neuronaler Optimierung ermöglicht hier die Verarbeitung von hochauflösenden Bildern. Durch die Einbettung neuronaler Lösungen in etablierte modellbasierte Frameworks leistet unsere Arbeit einen Beitrag zu neuroexpliziten Methoden und liefert interpretierbare und zeitgleich performante Ansätze.

Abstract

This thesis investigates the integration of neural networks into variational or partial-differential-equation-based approaches for image processing to achieve high-quality yet interpretable algorithms.

Decades of research into model-based methods have built a rich mathematical theory around variational, diffusion, and other models. It offers strong guarantees on the behaviour of various methods in terms of explicit concepts like stability or convergence guarantees. Furthermore, the models themselves are designed based on human insights and expressed in terms of mathematical equations.

Conversely, deep learning leaves the discovery of connections and patterns in data to neural networks. While the last decade has shown that this makes them extremely capable at a large range of tasks, they remain black boxes: even experts can usually not explain how the network arrived at its result.

This thesis seeks to use the power of neural networks and their frameworks within model-based approaches while maintaining the interpretability of the latter. The three approaches in this thesis span a spectrum from predominantly model-based methods to hybrid solutions that combine data-driven and model-driven strategies in equal measure:

The first approach focuses on the numerics of anisotropic diffusion processes, analysing a large class of finite difference discretisations and providing a rigorous stability analysis. The resulting discretisation is linked to the ResNet architecture, enabling highly parallel GPU-accelerated implementations of anisotropic diffusion.

The second approach employs neural solvers to address two challenging numerical problems in inpainting. In one, we use a neural prior and the minimisation capabilities of modern deep learning frameworks to overcome the poor conditioning of Euler's elastica. The other uses the flexibility of physics-informed neural networks to incorporate multiple optimisation goals into a simple architecture.

The third approach utilises large image datasets to construct optimised masks for homogeneous diffusion inpainting. The presented technique trains a deep learning model to quickly generate high-quality masks, integrating numerical solvers within the deep learning pipeline to capitalise on the advantages of both paradigms.

The contributions in this thesis demonstrate that there is a middle ground between fully neural and fully model-based approaches. Our algorithms confine neural models within frameworks defined by mathematical models. This allows us to retain their interpretability while still being able to harness the power of deep learning. Our work contributes to our vision of neuroexplict models which combine both neural and explicit, human-interpretable parts.

Acknowledgements

First of all, I would like to thank my advisor, *Prof. Joachim Weickert*, for five years of invaluable guidance. He not only directed my efforts, but also provided a broader perspective on both teaching and research. I am especially grateful for the freedom and responsibility I was given in my teaching activities.

This work was financially supported by the *European Research Council (ERC)* under the European Union's Horizon 2020 research and innovation programme (grant agreement no. 741215, ERC Advanced Grant INCOVID). Furthermore, I am grateful for the stimulating research environment provided by the GRK 2853/1 "Neuroexplicit Models of Language, Vision, and Action", funded by the *Deutsche Forschungsgemeinschaft (DFG, German Research Foundation)* under project number 471607914.

My thanks go out to my collaborators. *Dr. Tobias Alt-Veit* introduced me to neuroexplicit models before that term even existed, which lead to many enjoyable collaborations. *Dr. Pascal Peter* provided valuable input and context on everything from research to teaching. *Prof. Shoichi Koyama* opened my eyes to a completely new (to me) field of research. *Niklas Kämper*'s CUDA programming is ingenious. *Michael Ertel*, *Dr. Matthias Augustin*, and *Michael Krause* provided essential mathematical insights. Furthermore, substantial thanks are in order for *Kristina Schaefer* and *Dr. Vassillen Chizhov* for their comments on various research projects.

Beyond joint publications, I am very grateful to *Dr. Tobias Alt-Veit, Daniel Gaa, Dr. Pascal Peter* and *Kristina Schaefer* for proofreading this thesis and providing feedback.

In addition, I am grateful to all current and former members of the Mathematical Image Analysis group that provided a comfortable environment: *Sarah Andris, Dr. Leif Bergerhoff, Dr. Kireeti Bodduna, Peter Franke, Ferdinand-Dennis Jost, Hyoseung Kang, Andrea Kreutzer, Dr. Rahul Mohideen Kaja Mohideen, Jan Schmitz, Jón Arnar Tómasson, Aaron Wewior* and *Ellen Wintringer*.

Finally, beyond academia, both my parents *Katrin Ludwig-Schrader* and *Dr. Uwe Schrader*, as well as my wife *Ludmilla Schrader* were all instrumental in helping me get this thesis across the finish line.

Contents

1 Introduction

Deep learning methods have demonstrated a remarkable capacity for discovering patterns in large datasets, often making use of relationships that are neither detectable by nor interpretable for humans [63, 78, 265]. This ability has led to state-of-the-art performance on a wide range of benchmark tasks, including image classification, semantic segmentation, or generation [3, 44, 184, 267]. Consequently, these models are being integrated into more and more aspects of life, from large language models [3] over photorealistic image generation [31, 175] to autonomous driving [116].

Despite their empirical success, the internal mechanics of most deep learning models remain opaque [79, 108]. While they can be fully described as series of simple linear algebra operations, the intermediate values calculated by these networks largely carry no meaning that can be readily understood even by experts. The desire to change this motivates the field of explainable artificial intelligence (XAI) [149], which mostly works on post-hoc, local explanations of large pre-trained models [30, 140, 180]. We wholeheartedly subscribe to the general objective as a core tenant of the scientific method, and believe that understanding how and why a model works is just as important as its overall performance. However, this work approaches the problem from another direction: It aims to build models which are interpretable by design.

To this end, it makes use of classical model-driven methods, which can offer many of the properties purely neural approaches lack. As they are founded on explicit mathematical principles such as partial differential equations (PDEs) and variational models, they are inherently transparent: The model objectives and behaviours are clearly defined and can be rigorously analysed. However, this interpretability can be accompanied by practical limitations. Model-driven approaches may exhibit lower performance on certain tasks compared to neural networks. Furthermore, the numerical solution of these models often presents significant challenges, including the introduction of discretisation artefacts, the difficulty of minimising non-convex energy functionals, and the high computational cost of solving large-scale optimisation problems. Such issues frequently necessitate the design of highly specialised and involved solvers by domain experts.

The individual strengths of these two paradigms present an opportunity: To employ neural networks as a means to address the computational and optimisation challenges inherent in model-based frameworks. This hybrid approach seeks to retain

the interpretability of the underlying mathematical models while making use of the power of neural approaches to solve them.

The research presented in this dissertation is situated within the domain of neuroexplicit models [10, 89, 114]. This domain involves the systematic integration of neural building blocks into explicit, human-interpretable, model-based structures. While the degree of integration can vary, the focus of this thesis is predominantly on the model-driven perspective. We begin with transparent mathematical models and strategically incorporate neural components to overcome specific limitations or challenges.

The particular problems we consider in this thesis mostly originate from the domain of image processing, which offers a large variety of powerful but challenging models. Another contribution stems from acoustics, but the practical difference is not as significant as one might think: After stripping away the domain-specific terminology, we are left with a mathematical model to solve in either case. Thus, the neuroexplict strategies we develop in this work at the hand of specific problems can serve as blueprints to build performant, interpretable models.

1.1 Contributions

This thesis investigates three distinct areas of neuroexplicit approaches, each incorporating neural and model-driven components to different degrees. The contributions in this work are ordered by the amount of neural ideas used: It begins with an approach that is fully model-driven but can be translated into a neural architecture, and ends with a model consisting of data-driven and explicit components in equal amounts.

Numerical Solvers Translated into Neural Architectures A growing body of work has highlighted the deep connections between numerical schemes and modern neural architectures, ranging from applications like nonlinear diffusion processes over wavelets methods to variational models, see e.g. [11] for an overview. Building upon this perspective, we derive and analyse a class of anisotropic diffusion schemes [242, 250] that generalises several existing numerical approaches. For this class, we derive a fine-grained stability theory and establish connections to a widely used neural network architecture. This perspective enables the development of an efficient implementation using deep learning frameworks.

Neural Networks as Numerical Solvers The second area explores the use of neural networks as solvers for challenging variational problems and PDEs. We first

address Euler's elastica inpainting [154], an energy-based model whose poor conditioning renders classical solvers slow, unstable, or reliant on highly specialised techniques. We propose a method that integrates a theoretically well-founded discretisation with deep neural image representations [229], coupled with optimisation strategies originating from deep learning. This yields results comparable to specialised state-of-the-art methods, while using only simple ingredients. In addition, we move beyond image analysis to the domain of acoustics. There, we consider sparse inpainting of the sound field magnitude, which is governed by a PDE. By adapting physics-informed neural networks [178] to this domain, we demonstrate the ability to obtain physically plausible reconstructions in spite of very sparse data.

Neural Networks for Mask Optimisation The final chapter moves further towards data-driven learning, while retaining an explicit model-based structure. In it, we develop a neuroexplicit framework for optimising masks for homogeneous diffusion inpainting [32]. The approach incorporates both image reconstruction quality and the satisfaction of the underlying PDE directly into the neural training objective. As part of our architecture, we train a neural surrogate capable of approximately solving the inpainting PDE within the optimisation loop. We further extend this framework by allowing the integration of a classical weight-free solver into the deep learning pipeline, which yields additional improvements. Moreover, we introduce a domain decomposition strategy derived from theoretical insights that enables scalability to high-resolution images without sacrificing performance.

Together, these contributions demonstrate scenarios in which neural networks serve to enhance the capabilities of explicit models without significantly affecting their interpretability.

1.2 Organisation of the Thesis

Following the motivation and overview of our contributions in this introduction, Chapter 2 introduces notation, conventions, and mathematical preliminaries. Afterwards, Chapter 3 reviews topics from machine learning which are required throughout this thesis, and introduces analytical and numerical aspects of diffusion. In addition, it offers a brief primer on acoustics. Each of the following chapters then contains their own related work specific to the considered problem.

The main contributions are presented in Chapters 4–6, each corresponding to one of the three research directions outlined in Section 1.1. They are written to be largely self-contained and may be read independently.

Chapter 4 studies a class of discretisations for anisotropic diffusion. We provide a detailed stability analysis, connect the derivation to well-known neural architectures,

and show how this link enables efficient, parallel implementations within deep learning frameworks.

Chapter 5 investigates neural networks as solvers for numerical problems. First, we tackle Euler's elastica inpainting, combining multiple neural strategies for variational optimisation into a capable solver. Second, we study sparse inpainting of sound field magnitudes by adapting physics-informed neural networks to the governing PDE of sound fields.

Finally, Chapter 6 deals with mask optimisation for diffusion-based inpainting. We introduce a neural framework that learns both mask generation and inpainting jointly. Furthermore, we extend it by integrating a numerical solver into the pipeline, and by developing a domain decomposition strategy to extend the method to high-resolution images.

The thesis concludes in Chapter 7 with a summary and an outlook on future research. Afterwards, we list our publications, and present the bibliography, a glossary, as well as lists of symbols and figures.

2 Notations, Conventions, and Preliminaries

This chapter establishes the mathematical preliminaries for the thesis, beginning with notational and typesetting conventions before presenting the key definitions that will be used in subsequent chapters.

2.1 Notations and Conventions

We use a consistent typesetting for scalars, vectors, matrices and other mathematical expressions for ease of readability.

Scalars are denoted by lower-case letters like x, y or t.

Vectors use bold lower-case letters like $\boldsymbol{f}$ with elements $\boldsymbol{f} = (f_1, \dots, f_n)^\top \in \mathbb{R}^n$.

Matrices are typeset using bold upper-case letters $\boldsymbol{A}$. Here, the element in row i and column j is referred to by $a_{i,j}$.

Functions use lower-case letters. Scalar-valued functions use roman letters, while vector-valued or matrix-valued functions use bold face. Exceptions are energy functionals, which are given by E.

Neural networks are also functions, but are set using calligraphic font like $\mathcal{N}$. Similarly, their loss functions are denoted by $\mathcal{L}$.

Partial derivatives use multiple notations. Consider a function $f : \mathbb{R}^n \to \mathbb{R}$ with parameters $x_1, \dots, x_n$. Then, the partial derivative with respect to x_i can be denoted as $\frac{\partial f}{\partial x_i}$, $\partial_{x_i} f$, or f_{x_i}.

The gradient of a function $f : \mathbb{R}^n \to \mathbb{R}$ is written as $\boldsymbol{\nabla} f = (f_{x_1}, \dots, f_{x_n})^\top$.

Directional derivatives of a function $f : \mathbb{R}^n \to \mathbb{R}$ along a normalised vector $\boldsymbol{n} \in \mathbb{R}^n$ are given by $\partial_{\boldsymbol{n}} f = \boldsymbol{n}^\top \boldsymbol{\nabla} f$.

The divergence of a vector-valued function $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^n$, $\boldsymbol{f} = (f_1, \ldots, f_n)^\top$ is the sum of the partial derivatives of each component w.r.t. its corresponding coordinate **div** $f = \sum_{i=1}^n \partial_{x_i} f_{x_i}$.

The Laplacian of a scalar-valued function $f : \mathbb{R}^n \to \mathbb{R}$ is the sum of second derivatives $\Delta f = \sum_{i=1}^n \partial_{x_i x_i} f$.

Further information on the common usage of variable names can be found in the list of symbols at the end of this thesis.

2.2 Norms

We will encounter norms repeatedly both in the context of stability as well as error measures. Relevant ones are introduced in the following.

2.2.1 Vector Norms

For vectors, we only require the p-norm. For a vector $\boldsymbol{f} \in \mathbb{R}^n$ and $1 \leq p < \infty$, it is given by

$$\|\boldsymbol{f}\|_p = \left(\sum_{i=1}^n |f_i|^p \right)^{\frac{1}{p}}. \tag{2.1}$$

In particular, we make use of the p-norm for $p \in \{1, 2\}$. For $p = 1$, we obtain the L^1-norm, which is the sum of absolute values:

$$\|\boldsymbol{f}\|_1 = \sum_{i=1}^n |f_i| \,. \tag{2.2}$$

For $p = 2$, we get the Euclidean norm given by

$$\|\boldsymbol{f}\|_2 = \sqrt{\sum_{i=1}^n f_i^2} \,. \tag{2.3}$$

2.2.2 Matrix Norms

We require the spectral norm for stability analysis. It is the induced norm of the Euclidean norm, defined as

$$\|\boldsymbol{A}\|_2 = \max_{\|\boldsymbol{x}\|_2 = 1} \|\boldsymbol{A}\boldsymbol{x}\|_2 \,. \tag{2.4}$$

It can be understood as the largest possible elongation factor which can be achieved by multiplying a normalised vector $\boldsymbol{x} \in \mathbb{R}^n$ with the matrix $\boldsymbol{A}$. For stability analysis in Chapter 4, we make use of an equivalent definition. To that end, denote the spectral radius ρ of a square matrix $\boldsymbol{B} \in \mathbb{R}^{n\times n}$ by

$$\rho(\boldsymbol{B}) = \max\{|\lambda_1|, \dots, |\lambda_n|\} \tag{2.5}$$

where λ_i are the eigenvalues of $\boldsymbol{B}$. With this notation, the spectral norm of a matrix $\boldsymbol{A} \in \mathbb{R}^{m\times n}$ can be written as

$$\|\boldsymbol{A}\|_2 = \sqrt{\rho(\boldsymbol{A}^\top \boldsymbol{A})}\,. \tag{2.6}$$

For symmetric matrices $\boldsymbol{B} \in \mathbb{R}^{n\times n}$, this simplifies further to

$$\|\boldsymbol{B}\|_2 = \rho(\boldsymbol{B})\,. \tag{2.7}$$

One theorem that we will make use of in this context is the Gershgorin circle theorem [74, 252], which relates the eigenvalues of a matrix to its entries.

Theorem 1 (Gershgorin Circle Theorem)**.** *Consider a square matrix $\boldsymbol{B} \in \mathbb{R}^{n\times n}$ and define the disks*

$$K_i \coloneqq \left\{ x \in \mathbb{C} \;\middle|\; |x - b_{i,i}| \leq \sum_{i\neq j} b_{i,j} \right\}. \tag{2.8}$$

Then, all eigenvalues of $\boldsymbol{B}$ lie within in the union of the disks $\bigcup_{i=1}^{n} K_i$.

When the matrix $\boldsymbol{B}$ is symmetric, all its eigenvalues are real and the circles collapse to intervals on the real line.

2.3 Images

All our mathematical models are first defined in the continuous setting, before then being discretised. Here, we shortly discuss the representations of signals and images in both settings.

2.3.1 Continuous Setting

Signals are modelled as functions from an interval $[a, b] \subset \mathbb{R}$ to the real numbers. For greyscale images, we consider functions of type $f : \Omega \to \mathbb{R}$, where Ω is the

image domain. While most continuous models only assume that $\Omega \subset \mathbb{R}^2$ is a closed set, their discrete counterparts are easier to define if we restrict Ω to be a rectangle $[a, b] \times [c, d]$.

For colour images, the dimension of the co-domain increases to the number of channels. For example, for typical *red-green-blue* (RGB) images, the co-domain becomes $\mathbb{R}^3$.

2.3.2 Discrete Setting

A discrete signal $\boldsymbol{f} \in \mathbb{R}^n$ is obtained from a continuous signal $f : [a, b] \to \mathbb{R}$ by sampling uniformly with distance h:

$$f_i = f\left(a + \left(i - \frac{1}{2}\right)h\right). \tag{2.9}$$

Correspondingly for an image $f : [a, b] \times [c, d] \to \mathbb{R}$, and distances h_x, h_y in x- and y-direction respectively, we sample the discrete image $\boldsymbol{f} \in \mathbb{R}^{n_x \times n_y}$ through

$$f_{i,j} = f\left(a + \left(i - \frac{1}{2}\right)h_x, c + \left(j - \frac{1}{2}\right)h_y\right). \tag{2.10}$$

This corresponds to sampling at the centre of our pixels $[a + ih_x, a + (i + 1)h_x] \times [c + ih_y, c + (i + 1)h_y]$. Commonly, we assume that our pixels are square, giving $h_x = h_y = h$.

So far, we have represented discrete images as matrices, which require most operations to be described using tensors. While many implementations will still store and manipulate images this way, notation can be simplified significantly by vectorising the image representations. To this end, we arrange all pixels into a vector $\boldsymbol{f} \in \mathbb{R}^{n_x n_y}$ by concatenating the image rows. This allows us to express operations using simple matrix-vector multiplications.

2.4 Error Measures

We discuss the error measures we use throughout this thesis, starting with general purpose error measures and then moving to image-specific ones.

2.4.1 Mean Absolute Error

As the name suggests, the *mean absolute error* (MAE) of two signals $\boldsymbol{u}, \boldsymbol{f} \in \mathbb{R}^n$ is given by

$$\mathrm{MAE}(\boldsymbol{u}, \boldsymbol{f}) = \frac{1}{n}\|\boldsymbol{u} - \boldsymbol{f}\|_1 = \frac{1}{n}\sum_{i=1}^{n}|u_i - f_i| \,. \tag{2.11}$$

This definition extends to images by assuming that they are represented as vectors. It scales linearly with error size, which means that it is robust to outliers. When used in combination with gradient descent-based optimisation, the nondifferentiability for $u_i = f_i$ can become an issue. In this case, popular deep learning frameworks [1, 164] use a subgradient [183] instead.

2.4.2 Mean Squared Error

The *mean squared error* (MSE) is given by

$$\mathrm{MSE}(\boldsymbol{u}, \boldsymbol{f}) = \frac{1}{n}\|\boldsymbol{u} - \boldsymbol{f}\|_2 = \frac{1}{n}\sum_{i=1}^{n}(u_i - f_i)^2 \,. \tag{2.12}$$

As before, we can extend this definition to images and structures of arbitrary dimensions by vectorising them. For 2-D greyscale images, this results in the Frobenius norm. Compared to the MAE, the MSE penalises large deviations harder, but tolerates small errors. Both MAE and MSE are typically adequate choices to measure the distance between images, with small differences based on user preference and the considered application. Therefore, we make use of both throughout this thesis.

2.4.3 Peak Signal-to-Noise Ratio

The MSE and MAE always require the user to consider the underlying grey value range to interpret results. One metric which is independent of it is the *peak signal-to-noise ratio* (PSNR). For a grey value range of $[0, c_{\max}]$, it is defined as

$$\mathrm{PSNR}(\boldsymbol{u}, \boldsymbol{f}) = 10\log_{10}\left(\frac{c_{\max}^2}{\mathrm{MSE}(\boldsymbol{u}, \boldsymbol{f})}\right) . \tag{2.13}$$

PSNR values are reported in decibels (dB). Contrary to the previous metrics, higher values indicate greater similarity. In addition, it scales logarithmically, with an increase from 20 dB to 30 dB being much more noticeable than an increase from 30 dB to 40 dB. The PSNR is most commonly used for lossy image compression, which is why we use it in Chapter 6.

Though widely used, the metrics presented here can diverge significantly from human perception [240]. In response, a large range of perceptual metrics [39, 240, 266] have been proposed over the years. None of them has established itself as a universally accepted standard so far, with each having different blind spots. As such, we rely solely on the metrics presented here.

2.5 Convolution

Convolutions appear in this thesis both as a building block for neural networks and as part of continuous and discrete image processing.

In the continuous setting, the convolution of two functions $f, g : \mathbb{R}^n \to \mathbb{R}$ is given as

$$(f * g)(\boldsymbol{x}) = \int_{\mathbb{R}^n} f(\boldsymbol{x} - \boldsymbol{y}) g(\boldsymbol{y}) \, \mathrm{d}\boldsymbol{y} \, . \tag{2.14}$$

For our purposes, one of the functions is usually an image, and the other a so-called convolution kernel.

The discrete convolution of two tensors $(f_{\boldsymbol{i}})_{\boldsymbol{i} \in \mathbb{Z}^n}, (g_{\boldsymbol{i}})_{\boldsymbol{i} \in \mathbb{Z}^n}$ is defined as

$$(f * g)_{\boldsymbol{i}} = \sum_{\boldsymbol{j} \in \mathbb{Z}^n} f_{\boldsymbol{i} - \boldsymbol{j}} g_{\boldsymbol{j}} \, . \tag{2.15}$$

For signals and images, we will only need the 1-D and 2-D versions of this equation. Within neural networks however, this equation is also used for tensors of up to four dimensions.

3 Related Work

While the applications vary between chapters, they share their use of neural networks, and both Chapters 4 and 6 involve diffusion. We introduce those topics here. In addition, we provide a brief introduction to acoustics through the lens of image processing.

Neural networks and machine learning have significantly advanced image processing in recent years [63, 78, 265], and are one of the two key ingredients of this thesis. We introduce key neural architectures and how they are trained, followed by an analysis of some trends within the field.

Diffusion [98, 167, 242] has applications from denoising [13, 167, 242] over inpainting [71, 206] to compression [71, 199]. In this thesis, it appears in two ways: Chapter 4 connects numerics for diffusion with neural architectures. In Chapter 6, it is part of homogeneous diffusion inpainting and the associated mask optimisation problem.

3.1 Machine Learning

Neural networks have revolutionised many areas of science over the last two decades. Early approaches started out with handcrafted features which were used as inputs to simple architectures like support vector machines [43]. However, enabled by ever larger datasets and computational capabilities, the field moved to models which learn features directly from data. In combination with a suitable mapping from features to output, they are now highly capable on a large number of tasks [78].

3.1.1 Key Components

Before we go into the details of neural network architectures, we introduce the components that make up a general machine learning problem. We base this structure loosely on Goodfellow et al. [78] and Zhang et al. [265].

Data

The data is what the model is allowed to learn from, and what we use to evaluate the model. For image processing, our *dataset* typically consists of images, which the

neural network receives as input. In *supervised learning* problems, an additional value is provided for each image. This *label* is not part of the model input, and will usually have to be predicted by the model. In a classification setting, this would be the type of object present in the image.

The data is typically divided into *training data*, *validation data* and *test data*. The training data is the subset based on which the network is allowed to learn. The validation set is used to check the performance of the model during training, and select the best model based on it. Finally, the model is evaluated on the unseen inputs of the test data to measure the final performance.

Task and Objective function

In the broadest sense, the task is what the machine learning algorithm should accomplish. It is useful to describe it in terms of the desired output given an input. Furthermore, to train the network, we need to be able to quantify how successful the model is at its task. This *objective function* can take many different shapes depending on the available data and task. When a lower value of the objective function indicates a better result, it is also called a *loss function*.

It can be challenging to measure success at a task adequately: We want our algorithm to perform well on all possible inputs, but are only able to minimise the loss on the finite number of samples in the training data. It is possible that our algorithm is then able to perform well on the training set, but shows a higher loss on the validation data. This difference is called the *generalisation error*, and is just as important as minimising the loss on the training set.

Note that a loss function serves a completely different purpose than an energy functional of a variational model. While a loss function measures success at a given task, an energy functional encodes the strategy for achieving it. For example, in a simple denoising model, a lower energy value indicates success at balancing two objectives: staying close to the noisy input while reducing sharp variations [23]. This measures adherence to the prescribed strategy, and only by proxy does it measure success at the underlying task. In contrast, a typical neural network loss, such as the MSE, directly measures task success but offers no guidance on how to achieve it.

Model

The *model* is a function which takes an element of the dataset as input, and tries to produce the desired output based on the task and objective function. A *neural network architecture* can be seen as a function class parametrised by weights $\boldsymbol{\theta}$. The goal of learning or training is then to determine values for the weights which lead to the best success in the task as measured by the objective function. If the considered model

is sufficiently powerful or concatenates a sufficient number of chained transformations, the term *deep learning* is used. However, there is no clearly defined rule when machine learning turns into deep learning [78].

In the following, we will discuss the training process and common neural network architectures in more depth.

3.1.2 Training

Consider a general neural network $\mathcal{N}_{\boldsymbol{\theta}} : \mathbb{R}^m \to \mathbb{R}^n$ parametrised by weights $\boldsymbol{\theta}$. We want to train it on a dataset $(\boldsymbol{x}_i, \boldsymbol{y}_i), \boldsymbol{x}_i \in \mathbb{R}^m, \boldsymbol{y}_i \in \mathbb{R}^n, i = 1, \dots, N$, where each pair $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ contains a network input and the corresponding expected output. This is the supervised learning case mentioned above. Our loss $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ compares the network output $\mathcal{N}_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$ to the expected output $\boldsymbol{y}_i$.

While the following descriptions consider supervised learning, they can be trivially modified for the unsupervised setting. In that case, the dataset only contains the inputs $(\boldsymbol{x}_i)$ without ground truth outputs. Accordingly, the loss function $\mathcal{L}(\mathcal{N}_{\boldsymbol{\theta}}(\boldsymbol{x}))$ only received the network prediction as input.

The process of training now describes optimising the weights $\boldsymbol{\theta}$ such that the loss over the dataset is minimised. This is usually done by variants of gradient descent. Independent of the concrete method, the training time can be measured in *epochs*. One epoch has elapsed once every sample has been used exactly once to update the network weights.

Gradient Descent

We start by only considering a single sample $(\boldsymbol{x}, \boldsymbol{y})$. As the name suggests, gradient descent updates the weights by moving them into the direction of steepest descent w.r.t. the loss function. This is repeated until a minimum is reached. Denote the weights at step k by $\boldsymbol{\theta}^k$, and let the step size be given by $\tau > 0$. Then, each step performs the update

$$\boldsymbol{\theta}^{k+1} = \boldsymbol{\theta}^k - \tau \frac{\partial \mathcal{L}(\mathcal{N}_{\boldsymbol{\theta}}(\boldsymbol{x}), \boldsymbol{y})}{\partial \boldsymbol{\theta}^k} = \boldsymbol{\theta}^k - \tau \boldsymbol{\nabla}_{\boldsymbol{\theta}^k} L(\mathcal{N}_{\boldsymbol{\theta}}(\boldsymbol{x}), \boldsymbol{y}) \,. \tag{3.1}$$

Here, $\partial \mathcal{L} / \partial \boldsymbol{\theta} = \boldsymbol{\nabla}_{\boldsymbol{\theta}} \mathcal{L}$ denotes a column vector which contains the partial derivatives of the loss $\mathcal{L}$ with respect to all elements of $\boldsymbol{\theta}$. It is given by $(\partial \mathcal{L} / \partial \boldsymbol{\theta})_j = \partial_{\theta_j} \mathcal{L}$.

Stochastic Gradient Descent

For stochastic gradient descent [182], such a descent step is performed for each sample from the dataset one after the other:

$$\boldsymbol{\theta}^{k+1} = \boldsymbol{\theta}^k - \tau \frac{\partial \mathcal{L}(\mathcal{N}_{\boldsymbol{\theta}}(\boldsymbol{x}_k), \boldsymbol{y}_k)}{\partial \boldsymbol{\theta}^k} \,. \tag{3.2}$$

On large datasets, this tends to perform well overall, but gradients can vary wildly between samples. On one hand, this allows the optimisation to escape local minima or saddle points [49]. On the other, it can also cause the optimisation to diverge, especially if the learning rate τ is too large.

Batch Gradient Descent

The other extreme is to average gradients across the whole dataset for each update, leading to the equation

$$\boldsymbol{\theta}^{k+1} = \boldsymbol{\theta}^k - \tau \frac{1}{N} \sum_{i=1}^{N} \frac{\partial \mathcal{L}(\mathcal{N}_{\boldsymbol{\theta}}(\boldsymbol{x}_i), \boldsymbol{y}_i)}{\partial \boldsymbol{\theta}^k} \,. \tag{3.3}$$

This approach leads to a very smooth optimisation, but is more likely to get stuck in local minima. Furthermore, computational resources typically do not suffice to calculate gradients for a large dataset at once.

Mini-Batch Gradient Descent

Mini-batch gradient descent offers a middle ground: The training set is divided into smaller (mini-) batches of data over which the gradients are averaged. For a batch size of n_b, the update reads

$$\boldsymbol{\theta}^{k+1} = \boldsymbol{\theta}^k - \tau \frac{1}{n_b} \sum_{i=1}^{n_b} \frac{\partial \mathcal{L}(\mathcal{N}_{\boldsymbol{\theta}}(\boldsymbol{x}_{kn_b+i}), \boldsymbol{y}_{kn_b+i})}{\partial \boldsymbol{\theta}^k} \,. \tag{3.4}$$

Note that this notation assumes that the samples from the dataset are processed in order. Typically, the order would be randomised aver each epoch to avoid biases.

The batch size n_b is an important hyperparameter of training that can significantly influence convergence speed. It is chosen such that stability and computational efficiency are balanced.

Adam

The variants of gradient descent discussed so far can converge quite slowly, both in practice and in the mathematical analysis of convergence rates. Network parameters can live on different scales, and the size of gradients can change over the course of training, which means that a fixed learning rate is suboptimal [220].

As such, machine learning typically employs advanced momentum-based strategies. For a comprehensive overview, see e.g. [158, 220]. Over the last decade, *adaptive moment estimation* (Adam) [119] has emerged as a very popular choice. It combines the advantages of RMSprop [88] and AdaGrad [56] to build a one-size-fits-most algorithm. It maintains approximations of the mean $\boldsymbol{v}$ and uncentred variance $\boldsymbol{s}$ of the gradient through exponential moving averages. For smoothing factors β_1, β_2, it tracks

$$\boldsymbol{v}_k = \beta_1 \boldsymbol{v}_{k-1} + (1-\beta_1)\frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}^k}, \qquad \boldsymbol{v}_0 = \mathbf{0}\,, \tag{3.5}$$

$$\boldsymbol{s}_k = \beta_2 \boldsymbol{s}_{k-1} + (1-\beta_2)\left(\frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}^k}\right)^2, \qquad s_0 = \mathbf{0}\,. \tag{3.6}$$

The initialisation bias is remedies trough normalisation:

$$\widehat{\boldsymbol{v}_k} = \frac{\boldsymbol{v}_k}{1-\beta_1^k}\,,$$
$$\widehat{\boldsymbol{s}_k} = \frac{\boldsymbol{s}_k}{1-\beta_2^k}\,.$$

This leads to the weight update

$$\boldsymbol{\theta}^{k+1} = \boldsymbol{\theta}^k - \tau \frac{\widehat{\boldsymbol{v}_k}}{\sqrt{\widehat{\boldsymbol{s}_k}} + \varepsilon}\,. \tag{3.7}$$

As can be seen, it normalises the averaged gradient by the standard deviation. As such, it uses both momentum and is robust to sparse or noisy gradients.

The authors suggest to use $\beta_1 = 0.9$, $\beta_2 = 0.99$, and $\varepsilon = 10^{-8}$ for the hyperparameters [119]. As Adam is quite robust to these choices, they are rarely adjusted.

Automatic Differentiation

All optimisation methods discussed so far rely on efficient computation of the gradient of the loss w.r.t. the weights. In all modern deep learning frameworks, this is accomplished using *automatic differentiation* [19, 81, 133, 215, 253]. In fact, the dis-

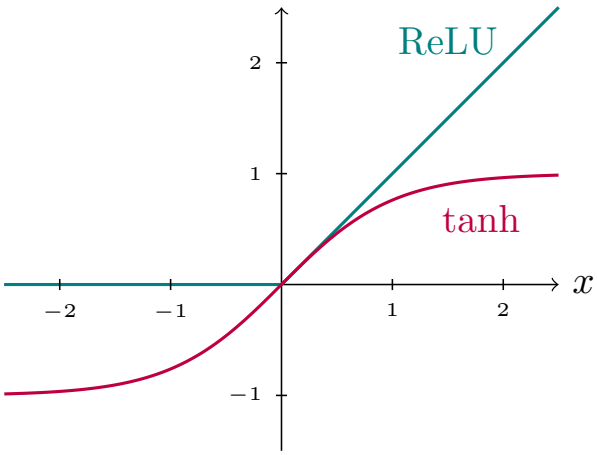


Figure 3.1: Visualisation of ReLU and tanh.

covery and implementation of it are one of the ingredients which enabled efficient training of large models in the first place.

Most deep learning frameworks [1, 164] rely on backpropagation [191] for automatic differentiation, which boils down to repeated applications of the chain rule. Just as deep learning frameworks hide the complexity of automatic differentiation from the user, we refer to [19] for an overview.

3.1.3 Architectures

While there are a plethora of neural architectures, this thesis builds upon only two archetypes: *multilayer perceptrons* (MLPs) [186, 187] and *convolutional neural networks* (CNNs) [70, 127]. We discuss their building blocks here.

Multilayer Perceptrons

The core building block of MLPs are *fully connected layers*. For an input $\boldsymbol{x} \in \mathbb{R}^n$, weights $\boldsymbol{K} \in \mathbb{R}^{m \times n}$, bias $\boldsymbol{b} \in \mathbb{R}^m$, and *activation function* $\sigma : \mathbb{R} \to \mathbb{R}$, they are defined by

$$h(\boldsymbol{x}) = \sigma(\boldsymbol{K}\boldsymbol{x} + \boldsymbol{b}) \tag{3.8}$$

where the activation σ is applied element-wise.

Many choices for the activation function are possible, with common choices being the *rectified linear unit* (ReLU) [76, 157] and sigmoid functions. The ReLU function is given by

$$\mathrm{ReLU}(x) = \begin{cases} x & \text{if } x > 0, \\ 0 & \text{otherwise.} \end{cases} \tag{3.9}$$

Since it is not differentiable in $x = 0$, gradient descent requires that a derivative in 0 is chosen. Surprisingly, the choice acts as a network hyperparameter [24].

One representative of sigmoids is the hyperbolic tangent given by

$$\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}\,. \tag{3.10}$$

Both functions are visualised in Figure 3.1.

A multilayer perceptron is constructed from a series of fully connected layers. All but the final one are called *hidden layers*, and the final one is the *output layer*. Its output dimension m is chosen such that it matches the task. For example, a scalar regression problem would have an output dimension of $m = 1$.

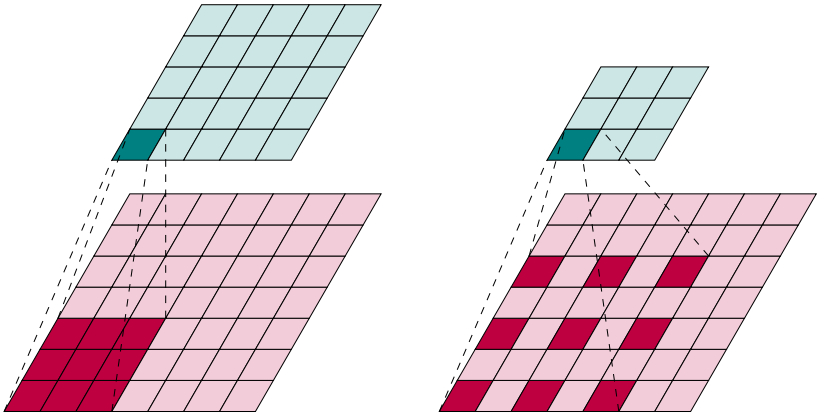
It has been shown that MLPs with a single hidden layer of sufficient size are already capable of approximating functions with arbitrary precision [45], which underscores their usefulness as an architecture. However, deeper networks typically perform better in practice [48, 62].

Convolutional Neural Networks

Discrete greyscale images are most naturally represented by matrices. However, they need to be vectorised when they are used as input for an MLP, as those are only able to take vectors as input. In fact, we could use any fixed, arbitrary permutation of the image pixels and would expect identical performance as the network has no inherent understanding of spatial structure. This discards multiple pieces of prior knowledge and introduces inefficiency:

1. Pixels which are close to each other are typically correlated. This is used heavily for tasks such as denoising or inpainting [242]. An MLP has no access to this structure, and has to learn such relations from scratch.

2. Many image operations, such as denoising, should behave identically independent of the position within the image. However, fully connected layers use different weights for every pixel, which can lead to position-dependent behaviour.



(a) regular convolution (b) dilated convolution with rate 2

Figure 3.2: Visualisation of a regular and a dilated convolution. While both use 9 weights, the regular convolution has a receptive field of 3×3 while the dilated one has one of 5×5. Note that this visualisation assumes no padding and a stride of 1.

3. Using MLPs on images can require large numbers of parameters. Considering an image of size 256×256 and a hidden layer which preserves the dimension, the weight matrix $\boldsymbol{K}$ alone contains $256^4 \approx 4$ billion parameters. This exceeds the size of all networks used in this thesis by two orders of magnitude, and leads to unnecessarily long training times.

A solution to these issues is the *convolution layer* [70, 127]. For an input $\boldsymbol{x} \in \mathbb{R}^n$ and a convolution kernel $\boldsymbol{k} \in \mathbb{R}^m$, the output is computed as

$$\boldsymbol{y} = \sigma((\boldsymbol{k} * \boldsymbol{f}) + \boldsymbol{b}) \tag{3.11}$$

where σ is an activation function and $\boldsymbol{b}$ is a bias. At the boundaries, most neural architectures rely on constant padding with zeroes unless otherwise mentioned.

The convolution layer remedies the issues mentioned above: It only uses local information, behaves the same everywhere, and uses a significantly smaller number of weights. One can then build a complete network by chaining multiple convolution layers. However, it can have problems with the propagation of information across the image: Assuming kernels of width $m = 3$, information can only travel by one pixel per layer.

One option to extend this so-called *receptive field* is through the use of dilated convolutions [96, 261]. They space out the pixels involved in the convolution on a regular

grid, see Figure 3.2 for a visualisation of the 2-D case. As a result, one convolution layer is able to incorporate information from much further away. We employ architectures which utilise dilated convolutions in Chapters 5 and 6.

Another technique for the spatial aggregation of information is *pooling*. Here, the tensor is divided into patches, for which a summary statistic is calculated. For max-pooling of a signal $\boldsymbol{f}$ with a patch size k and identical stride k, its output $\boldsymbol{u}$ is computed by

$$u_i = \max_{0 \leq a \leq k} f_{ki+a} \,. \tag{3.12}$$

The output will then be smaller by a factor of k. Max-pooling is especially popular compared to other options like average-pooling as it preserves the strongest response of a neuron with an area.

The inverse operation is upsampling. The simplest option is to just duplicate each value k times:

$$u_i = f_{\lfloor i/k \rfloor} \,. \tag{3.13}$$

Besides nearest-neighbour upsampling, another popular option is to learn the upsampling operation, see e.g. [205, 264].

By selectively down- and upsampling between convolutional layers, one can accelerate the propagation of information, and reduce computational cost. A popular architecture which makes use of this is the U-net [185]. However, we need to discuss batches and channels before we can introduce it.

Batches and Channels

While the concepts involved in neural networks are easier to explain for 1-D signals, most practical architectures for image processing operate on 4-D tensors.

Two of those dimensions represent the spatial extent, and a third is the channel dimension. If the network takes colour images as input, these channels will usually correspond to the colour channels of the image. However, within the hidden layers of neural architectures, the number of channels varies and generally does not correspond to colours any more. Instead, individual channels can represent various high-level concepts, which are typically only partially human-interpretable.

Convolution layers operate in this space, and can vary the number of channels. Consider a 3-D input tensor of size $n_x \times n_y \times n_i$ where $n_x \times n_y$ is the spatial extent and n_i is the number of channels. Correspondingly, let the output be of size $n_x \times n_y \times n_o$ where n_o is the number of output channels.

Then, each of the n_o output channels is computed by a single convolution kernel of shape $h \times w \times n_i$, where h and w are the spatial kernel dimensions. As such, the total number of weights in a convolution layer is given by $h \cdot w \cdot n_i \cdot n_o$.

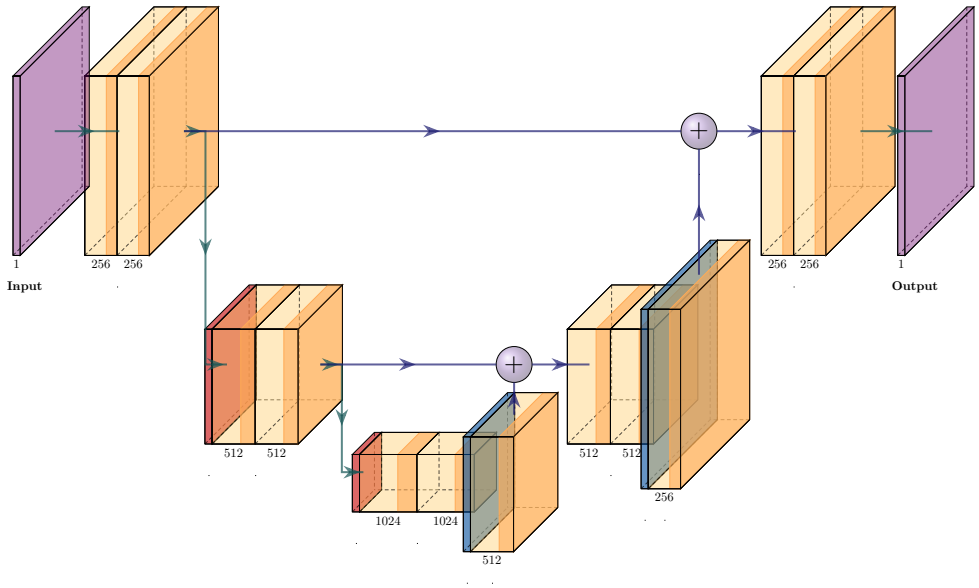


Figure 3.3: Visualisation of a U-Net with three scales. Each time the resolution is halved, the number of channels is doubled. Purple blocks denote input and output, yellow and orange ones represent convolutions with activations. Red and blue layers denote down- and up-sampling respectively. Adapted from [82].

Furthermore, as introduced above, networks are trained on (mini-)batches. To that end, the operations inside the neural network need to be computed for multiple input images at once. This leads to the introduction of the batch dimension and 4-D tensors of shape $n_b \times n_x \times n_y \times n_i$ where n_b is the batch size.

Given the high number of dimensions, precise mathematical descriptions of network architectures make it hard to understand their structure. Instead, visualisations can be used, which represent 3-D tensors by rectangular cuboids, where the length of each side represents the size of the dimension. One such visualisation can be seen for the U-net architecture [185] in Figure 3.3.

U-net

The U-net was originally introduced by Ronneberger et al. [185] for medical image segmentation, but has since proven to be successful on many image-to-image tasks. It consists of a downward pass and an upward pass, which are typically visualised in a U-shape. The downward pass alternates convolutional layers with pooling layers, which reduce the spatial dimension but increase the channel dimension. The upward pass alternates upsampling and convolutions while incorporating information from the

downwards pass until the original resolution is reached. We visualise this architecture in Figure 3.3.

ResNet

When building increasingly deep networks, one eventually encounters the vanishing gradient problem [21]. It leads to the gradients computed via backpropagation to be close to zero, which causes the training to stagnate. Experiments by He et al. [86, 87] were even able to show that at some point, networks started to perform worse when the number of layers is increased beyond a certain point. Their solution is the *residual block*, which computes for an input $\boldsymbol{x}$:

$$\boldsymbol{y} = \sigma_2(\boldsymbol{x} + \boldsymbol{k}_2 * \sigma_1(\boldsymbol{k}_1 * \boldsymbol{x} + \boldsymbol{b}_1) + \boldsymbol{b}_2)\,. \tag{3.14}$$

Here, $\sigma_{1,2}$ are activation functions, $\boldsymbol{k}_{1,2}$ are discrete convolution kernels, and $\boldsymbol{b}_{1,2}$ are biases. Notice in particular the addition of the input $\boldsymbol{x}$ before the second activation. This so-called *skip connection* allows for information to bypass both convolutions, and reduces the network task to learning the difference between input and output for each layer.

This approach can be combined with the original U-nets, where during upsampling information is combined using element-wise addition instead of concatenation along the channel dimension. Besides using the resulting performant architecture, we also make use of their architectural connection to explicit schemes in Chapter 4.

3.1.4 Recent Trends

Given the speed with which machine learning research is progressing today, trying to provide a comprehensive overview either means excessive breadth or insufficient depth. Furthermore, it would be outdated within months. As such, we restrict ourselves on a view of machine learning models through the lens of data, since this is one of the fundamental issues neuroexplicit models are trying to resolve.

In 2019, Richard Sutton published his essay "The Bitter Lesson" [221], which sparked a considerable debate on the state of AI research. Sutton's central thesis was that general-purpose approaches relying on massive computational resources and data consistently outperform sophisticated methods developed by human experts using domain-specific knowledge. He argued that carefully crafted solutions, regardless of their theoretical elegance, are ultimately superseded by approaches that simply scale computational power and training data.

His arguments were well supported by empirical evidence: The success of deep convolutional networks such as ResNet-152 [86], which achieved leading performance on ImageNet [192] mainly through architectural depth rather than domain-

specific design, underscores the power of this approach. Similarly, the transformer architecture [16, 231] demonstrated high performance across many natural language processing tasks through general attention mechanisms rather than task-specific architectural components. Reinforcement learning systems like AlphaGo Zero [210] managed to beat humans in complex games primarily through massive computational resources and self-play, rather than incorporation of expert strategic knowledge.

However, several researchers identified limits to this approach. Breunig [27] challenged Sutton's assumptions, arguing that "The Bitter Lesson is dependent on high-quality data." We too believe that this is a fundamental limitation in this paradigm. For domains like medical imaging, researchers can count themselves lucky if they have thousands of images of mixed quality available, see e.g.[153], compared to the billions used to train modern diffusion models [203]. The same holds for e.g. autonomous driving [36] and other real-world problems, where data collection is costly, and edge cases are rare but critical.

Further critiques [144] focused on data efficiency. They observed that machine learning models demonstrate markedly lower data efficiency compared to human learning, often requiring orders of magnitude more examples to acquire equivalent competency [126, 135]. Achieving greater data efficiency, they argued, necessitates incorporating strong inductive biases into learning systems, rather than relying exclusively on pattern recognition from large datasets. Our work contributes to this goal, with our inductive biases coming in the form of mathematical models.

The current landscape of artificial intelligence development presents both validation of and challenges to Sutton's original thesis. Large language models such as GPT-4 [3] and advanced video generation systems like Veo 3 [80] are indeed trained on massive datasets, utilising substantial portions of the publicly accessible internet [146, 232]. These systems demonstrate that, when available, more can be indeed better.

However, practical limitations of this paradigm have become increasingly apparent. High-quality training data sources are expected to be saturated in the coming decade [232]. Additionally, the internet increasingly contains AI-generated content, creating recursive training scenarios where models learn from synthetic data produced by previous AI systems. This phenomenon, termed *model collapse*, can result in worse model performance [7, 208, 209].

These developments indicate that future AI progress cannot rely solely on data scaling, contrary to Sutton's thesis. Even general-purpose models will encounter fundamental constraints imposed by finite data. Currently still data-rich domains will likely become data-limited in the coming years as high-quality sources are saturated.

This constraint necessitates exploration of alternative approaches to AI development. Multiple research directions addressing data efficiency have gained promi-

nence, including few-shot learning ideas [213, 214, 233], transfer learning methodologies [162, 270], and self-supervised learning paradigms. Additionally, synthetic data generation [18] and domain adaptation methods [162] offer ways to stretch existing data further.

Nowadays, there are already areas where neuroexplicit models are state of the art in highly competitive fields, even if they do not go by this name [27]. Consider for example the chess engine Stockfish [216]. Is consists of a highly optimised state space search algorithm, combined with an originally handcrafted position evaluation function. In 2020, the authors replaced the evaluation function by a tiny network with just 17,500 parameters inside its hidden layers. However, it has been beating the fully neural runner-up LCZero [134] consistently since then [225]. While Stockfish does not call itself "neuroexplicit", it is clearly combining knowledge of human experts and insights with AI models to great success, even in the presence of almost unlimited training data through self-play.

Neuroexplicit models can be part of the answer to limited data. By combining the learning capabilities of neural networks with the interpretability and efficiency of explicit mathematical models, neuroexplicit approaches can augment learned information with knowledge.

3.2 Diffusion

Diffusion describes the equilibration of particle concentrations within a closed space. Fick [67] first described the governing equations for concentrations of salts in 1855. His work is closely related to Fourier's work on heat transfer [69] and Ohm's work on electrical conduction [160]. The notion of diffusion in image processing treats grey values as concentrations, and describes their behaviour over time as an initial boundary value problem. Here, we cover diffusion models of various capabilities and complexities, starting with the simplest one.

3.2.1 Linear Homogeneous Diffusion

Linear homogeneous diffusion for image processing has been discovered independently multiple times, with the first description by Japanese researcher Iijima in 1959 [97, 98, 99], and western discoveries following 20 years later [120, 254].

The process considers an evolving image $u : \Omega \times [0, \infty) \to \mathbb{R}$ on a domain $\Omega \subset \mathbb{R}^2$. It is defined by the initial boundary value problem

$$\partial_t u = \Delta u \qquad \text{on } \Omega \times [0, \infty), \tag{3.15}$$

$$\partial_{\boldsymbol{n}} u = 0 \qquad \text{on } \partial\Omega \times [0, \infty), \tag{3.16}$$

$$u(\boldsymbol{x}, 0) = f(\boldsymbol{x}) \qquad \text{on } \Omega. \tag{3.17}$$

The diffusion equation (3.15) describes the image evolution over time. It leads to the equilibration of the grey values in the limit $t \to \infty$. The boundary conditions (3.16) define reflecting boundary conditions and dictate that no mass is transported across the image boundary given by the outer normal vector $\boldsymbol{n}$. Together with the diffusion equation (3.15), this leads to a preservation of the average grey value. The initial condition (3.17) initialises the image u at time $t = 0$ to the input image f.

As the diffusion equation (3.15) acts the same everywhere in the image, it is considered to be *homogeneous*. Furthermore, it does not depend on the local image structure, making it *linear* as well. These properties give this type of diffusion its name.

Linear homogeneous diffusion is parameter-free, which makes it simple to use. However, while it is a useful tool in downstream tasks like sparse inpainting [111], it tends to perform poorly for denoising. There, it blurs both noise and structures.

3.2.2 Nonlinear Isotropic Diffusion

Nonlinear isotropic diffusion [35, 75, 167] seeks to remedy this by inhibiting diffusion at edges, which are detected as regions with high gradient magnitude:

$$\partial_t u = \mathbf{div}\left(g(|\boldsymbol{\nabla} u|^2)\boldsymbol{\nabla} u\right). \tag{3.18}$$

The *diffusivity* g is a positive, decreasing function that controls the strength of diffusion in dependence of the squared gradient magnitude. Many diffusivities were proposed, for example the exponential Perona-Malik diffusivity [167]

$$g(s^2) = \exp\left(-\frac{s^2}{2\lambda}\right) \tag{3.19}$$

with a contrast parameter $\lambda > 0$, which even allows for the enhancement of edges.

The original formulation is ill-posed and sensitive to noise [118]. This can be remedied through presmoothing the gradient inside the diffusivity argument [33]:

$$\partial_t u = \mathbf{div}\left(g(|\boldsymbol{\nabla} u_\sigma|^2)\boldsymbol{\nabla} u\right). \tag{3.20}$$

The smoothing is $u_\sigma = K_\sigma * u$ is done with a Gaussian kernel K_σ with mean 0 and standard deviation σ.

As this process adapts the strength of diffusion based on the local structure of the evolving image, it is called *nonlinear*. However, it still treats all directions equally, making it *isotropic*.

While it performs better at denoising than homogeneous diffusion, it still struggles to denoise edges: It either smooths them away, or preserves them along with the noise present there depending on the parameter choice.

3.2.3 Nonlinear Anisotropic Diffusion

Anisotropic diffusion allows for different amounts of diffusion in different directions. Most practically relevant anisotropic models are also nonlinear, and adapt the diffusion to the local image structure. They do so by replacing the scalar-valued diffusivity g of nonlinear isotropic diffusion with a diffusion tensor $\boldsymbol{D} \in \mathbb{R}^{2\times 2}$:

$$\partial_t u = \mathbf{div}\left(\boldsymbol{D}(\boldsymbol{\nabla} u_\sigma)\boldsymbol{\nabla} u\right). \tag{3.21}$$

The diffusion tensor is a symmetric positive definite matrix which is commonly defined through its eigenstructure. For example for *edge-enhancing anisotropic diffusion* (EED) [246], it is constructed as follows: The first eigenvector $\boldsymbol{v}_1 \parallel \boldsymbol{\nabla} u_\sigma$ points across the local edge, and the second $\boldsymbol{v}_2 \perp \boldsymbol{\nabla} u_\sigma$ is parallel to it. To allow for diffusion along the edge but not across, the eigenvalues are then chosen as $\lambda_1 = g(|\boldsymbol{\nabla} u_\sigma|^2)$ and $\lambda_2 = 1$ respectively. The complete diffusion tensor is then given as

$$\boldsymbol{D}(\boldsymbol{\nabla} u_\sigma) = g(|\boldsymbol{\nabla} u_\sigma|^2) \cdot \boldsymbol{v}_1 \boldsymbol{v}_1^\top + 1 \cdot \boldsymbol{v}_2 \boldsymbol{v}_2^\top. \tag{3.22}$$

Both homogeneous diffusion (3.15) and nonlinear isotropic diffusion (3.18) can be written using the anisotropic diffusion equation (3.21) by setting $\boldsymbol{D} = \boldsymbol{I}$ and $\boldsymbol{D} = g(|\boldsymbol{\nabla} u|^2)\boldsymbol{I}$ respectively. We provide a visual overview of the three models we introduced in Figure 3.4. We see that linear homogeneous diffusion removes noise and structure everywhere. Nonlinear isotropic diffusion with the exponential Perona-Malik diffusivity preserves the edges, but also the noise at edges. EED is then also capable of preserving edges while denoising.

Beyond EED, several more anisotropic diffusion models [106, 172, 189, 196, 226, 243, 247] exist with different objectives and strengths.

3.2.4 Discretisations

So far, we have only discussed the different types of diffusion in the continuous setting. Here, we will introduce implementations using explicit schemes and finite dif-

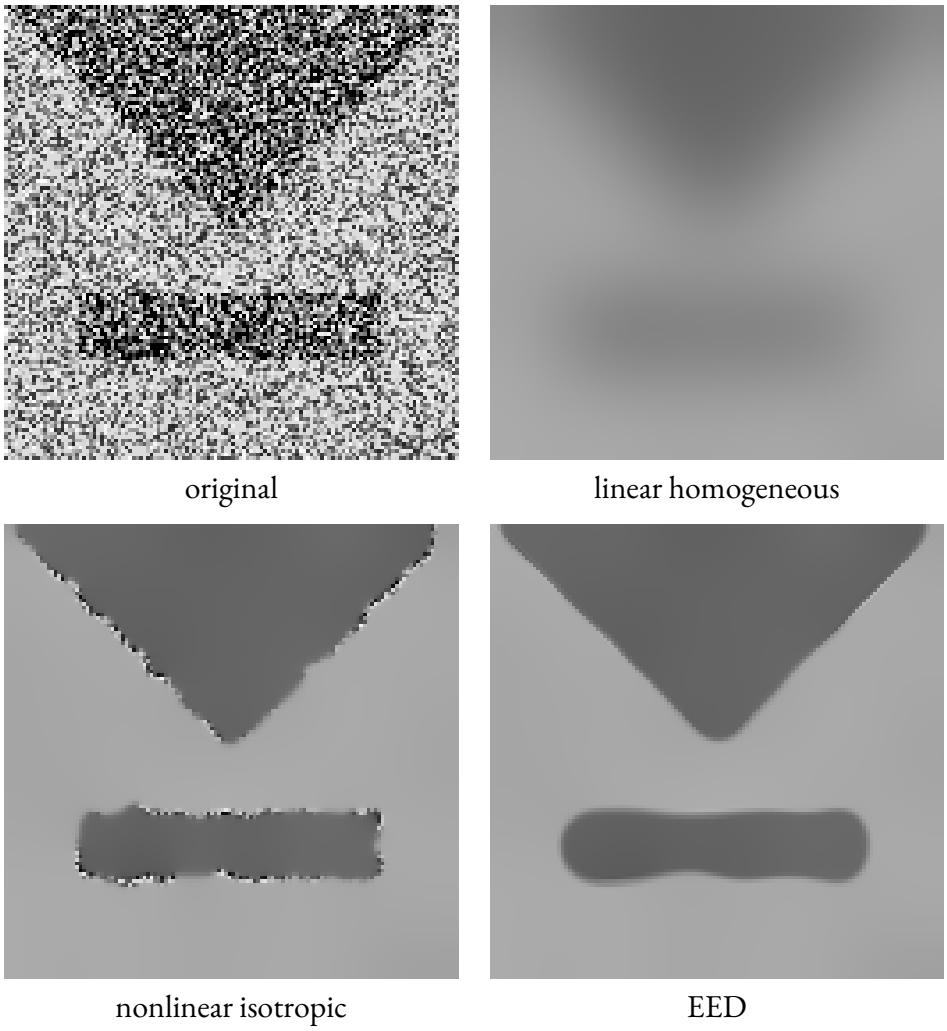


Figure 3.4: Visual comparison of the different diffusion methods on *pruebab*. All use a stopping time of $t = 80$. Nonlinear isotropic diffusion and EED use the exponential Perona-Malik diffusivity for $\lambda = 3.5$ and $\sigma = 2$. Test image taken from [242].

ferences. They serve as the basis for discretisations used throughout this thesis, in particular for anisotropic diffusion in Chapter 4 and Euler's elastica in Chapter 5.

Finite Differences

Finite differences can be used for derivative approximation. We introduce them for the 1-D setting, but the same ideas extend to higher dimension. To this end, consider a discrete signal $\boldsymbol{f} \in \mathbb{R}^n$ sampled from a continuous signal with grid size h.

For the first derivative, three basic options exist:

$$f_i' \approx \frac{f_{i+1} - f_i}{h} \qquad \text{(forward difference)}, \tag{3.23}$$

$$f_i' \approx \frac{f_{i+1} - f_{i-1}}{2h} \qquad \text{(central difference)}, \tag{3.24}$$

$$f_i' \approx \frac{f_i - f_{i-1}}{h} \qquad \text{(backward difference)}. \tag{3.25}$$

Their names are derived from the pixels involved in the approximation: The forward difference only uses location index i and its successor, the backward difference the predecessor, and the central difference uses indices on both sides.

The quality of an approximation can be judged by describing its asymptotic error in terms of the grid size h. For the forward and backward difference, the error is in $\mathcal{O}(h)$, and for the central difference in $\mathcal{O}(h^2)$ where $\mathcal{O}$ is the Bachmann-Landau notation. The so-called *consistency order* is then given as the power of h in the asymptotic error. As such, the forward and backward differences have consistency order 1, and the central difference has order 2. Here, a higher consistency order denotes a better approximation quality.

For the second derivative f_i'', the central difference is given by

$$f_i'' \approx \frac{f_{i-1} - 2f_i + f_{i+1}}{h^2}\,. \tag{3.26}$$

It has consistency order 2.

As finite difference approximations utilise the same coefficients across the whole signal, we can visualise them in *stencils*. For the central difference for the second derivative (3.26), we obtain

$$\boxed{\tfrac{1}{h^2}}\boxed{-\tfrac{2}{h^2}}\boxed{\tfrac{1}{h^2}} = \frac{1}{h^2}\boxed{1}\boxed{-2}\boxed{1}\,. \tag{3.27}$$

This stencil contains the weights for the central element f_i and its neighbours. The action of this stencil on a vector can thus be described through a matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ which contains the stencil entries appropriately centred in each row:

$$\boldsymbol{f}'' \approx \boldsymbol{A}\boldsymbol{f} = \frac{1}{h^2} \begin{pmatrix} -1 & 1 & & & \\ 1 & -2 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & -2 & 1 \\ & & & 1 & -1 \end{pmatrix} \boldsymbol{f}\,. \tag{3.28}$$

The first and last row differ due to assumed reflecting boundary conditions. In the discrete setting, they can be modelled by extending the signal to the left and right through $f_0 \coloneqq f_1$ and $f_{n+1} \coloneqq f_n$. The contributions of f_0 and f_{n+1} thus get accumulated onto f_1 and f_n.

Explicit Schemes

Discretising the time derivative $\partial_t \boldsymbol{u}$ with a forward difference gives the approximation

$$\partial_t \boldsymbol{u} \approx \frac{\boldsymbol{u}^{k+1} - \boldsymbol{u}^k}{\tau}\,. \tag{3.29}$$

Here, τ is the time step size and $\boldsymbol{u}^k$ denotes the image after k time steps. When we describe the discretisation of the divergence term for the image $\boldsymbol{u}^k$ by $\boldsymbol{A}(\boldsymbol{u}^k)\boldsymbol{u}^k$, the fully discrete version of the diffusion equation (3.21) is given by

$$\frac{\boldsymbol{u}^{k+1} - \boldsymbol{u}^k}{\tau} = \boldsymbol{A}(\boldsymbol{u}^k)\boldsymbol{u}^k\,. \tag{3.30}$$

Solving for $\boldsymbol{u}^{k+1}$ gives the explicit scheme

$$\boldsymbol{u}^{k+1} = \big(\boldsymbol{I} + \tau \boldsymbol{A}(\boldsymbol{u}^k)\big)\boldsymbol{u}^k \tag{3.31}$$

where $\boldsymbol{I}$ is the identity matrix. Starting from the initial image $\boldsymbol{u}^0 = \boldsymbol{f}$, we can repeatedly evaluate the right hand side to obtain approximations of u at time $t = k\tau$.

Stability

In tasks like diffusion-based inpainting, we are interested in the steady-state of $\boldsymbol{u}$ for $t \to \infty$. As such, one would be tempted to choose the time step τ as large as possible. However, choosing τ too large can cause the values of $\boldsymbol{u}$ to diverge. The stability

theory of diffusion was explored comprehensively by Weickert [242] and determines limits on τ within which the grey values remain reasonable.

Two notions of stability are relevant for us. The first is the maximum-minimum principle, which states that the evolving image never exceeds the range of the initial one:

$$\min_j f_j \leq u_i^k \leq \max_j f_j \qquad \forall i,\, \forall k > 0\,. \tag{3.32}$$

This strong notion of stability leads to restrictive bounds on the time step size for anisotropic diffusion in particular. The nonnegativity discretisation [242] even imposes limits on the maximal anisotropy.

A weaker notion which is more appropriate for discretisations of anisotropic diffusion is stability in the L^2-norm [250]. Here, iterations need to be nonincreasing in the Euclidean norm:

$$\|\boldsymbol{u}^{k+1}\|_2 \leq \|\boldsymbol{u}^k\|_2 \qquad \forall k \geq 0\,. \tag{3.33}$$

For the discrete diffusion equation (3.31) and assuming a symmetric matrix $\boldsymbol{A}$, this is guaranteed for

$$\|\boldsymbol{I} + \tau\boldsymbol{A}\|_2 = \rho(\boldsymbol{I} + \tau\boldsymbol{A}) \leq 1\,. \tag{3.34}$$

Assuming furthermore that $\boldsymbol{A}$ is negative semi-definite, we obtain the restriction

$$\rho(\tau\boldsymbol{A}) \leq 2 \;\Leftrightarrow\; \tau \leq \frac{2}{\rho(\boldsymbol{A})}\,. \tag{3.35}$$

As such, the problem of guaranteeing stability in the Euclidean norm reduces to determining the spectral radius of $\boldsymbol{A}$. We will use this methodology to analyse a stencil class for anisotropic diffusion in Chapter 4.

3.3 Basics of Acoustics

While all but one of the publications incorporated in this thesis deal with image processing, Section 5.5 is placed in the domain of acoustics. The application there still boils down to solving a PDE, but the PDE itself and the error measure used are specific to the domain. We introduce basic context here with notation adapted to be similar to the one commonly used in the image domain.

3.3.1 Sound Fields

On a technical level, *sound* is vibrating particles in a medium, which causes a variation in pressure. These variations are relative to the ambient pressure in the medium, e.g.

atmospheric pressure in air. If considered at a specific location and time, this variation is called the *sound pressure*.

If we consider the sound pressure in a domain $\Omega \subset \mathbb{R}^3$ across time, this spatio-temporal distribution is called a *pressure distribution* or *sound field*. It can be a modelled as a function $f(\boldsymbol{x}, t) : \Omega \times [0, \infty) \to \mathbb{R}$. In lossless media, the pressure distribution adheres to the *homogeneous wave equation*

$$u_{tt} = c^2 \Delta u \,. \tag{3.36}$$

There, Δ denotes the Laplacian with respect to the spatial coordinates, and c is the speed of sound in the considered medium, for example $c \approx 343\ m/s$ in air at room temperature. While air is not perfectly lossless, it can be treated as such for most applications, see e.g. [125].

Microphones can measure the sound pressure at a spatial location over time. Here, we assume perfect omnidirectional microphones which pick up sound waves arriving from all locations equally.

For many applications, it is useful to consider the sound field in the frequency domain instead. It can be calculated as the time-domain Fourier transform of u

$$\widehat{u}(\boldsymbol{x}, \omega) = \int_{-\infty}^{\infty} u(\boldsymbol{x}, t) e^{i\omega t} \, \mathrm{d}t \tag{3.37}$$

where ω is the angular frequency, given in radians per second. It relates to the ordinary frequency f by $\omega = 2\pi f$. Intuitively, $\widehat{u}$ describes the amplitude and phase of a sound wave of a specific frequency ω at location $\boldsymbol{x}$. Both u and $\widehat{u}$ are referred to as sound field in the literature, with the distinction typically being clear from context [227].

By plugging $\widehat{u}$ into the wave equation (3.36), we obtain the governing equation for $\hat{u}$. The resulting PDE is the Helmholtz equation given by

$$\Delta \widehat{u} + k^2 \, \widehat{u} = 0 \,. \tag{3.38}$$

Here, $k = \frac{\omega}{c}$ is the wave number, which describes the spatial frequency.

3.3.2 Sound Field Estimation

Microphones placed in a room are only able to capture the sound pressure at a few spatial locations. However, applications such as spatial active noise control or spatial audio reproduction for virtual reality require the pressure distribution to be known inside the whole domain [125]. This motivates the field of *sound field estimation*,

which deals with the interpolation of sound fields from a discrete set of measurements.

A comprehensive overview of algorithms for this problem is beyond the scope of this thesis and would require the introduction of a large amount of numerical techniques which are not used elsewhere in this work. We recommend [227] for an overview of model-based methods, and [125] for neuroexplicit and neural methods.

In this thesis, we only consider the interior problem, which aims to estimate the sound field inside a region Ω. All sound sources are assumed to be outside Ω, that is Ω is said to be *source-free*.

While it is commonly phrased differently in the sound field estimation literature, we still state the mathematical description based on the terminology used in the rest of this thesis. This reveals the close connection to sparse inpainting as considered in Chapter 6. Let $f(\boldsymbol{x}, \omega) : \Omega \times [0, \infty) \to \mathbb{C}$ be a sound field in the frequency-domain representation, which is only known in a subset $K \subset \Omega$. Our goal is to reconstruct a function $u : \Omega \times [0, \infty) \to \mathbb{C}$ which coincides with f on K, and conforms to the Helmholtz equation (3.38) everywhere else. For a binary confidence function c with $c(\boldsymbol{x}) = 1$ for known positions in K and $c(\boldsymbol{x}) = 0$ otherwise, the reconstructed sound field should fulfil

$$c \cdot (u - f) - (1 - c) \cdot \left(\Delta u + k^2 u\right) = 0\,. \tag{3.39}$$

Up to the reaction term $k^2 u$, this formulation is identical to homogeneous diffusion inpainting. We will consider a variant of this problem in Section 5.5.

3.3.3 Error Measures

Just as in the image domain, there are also a range of different error measures in the acoustic domain. Besides MAE and MSE as general metrics, the *logarithmic spectral distance* (LSD) can be used as a perceptual metric [124, 260]. It compares the sound field magnitude $|u|$ for set of position and frequencies. Let $\boldsymbol{x}_1, \dots, \boldsymbol{x}_n$ be a set of locations, and $\omega_1, \dots, \omega_w$ be a set of frequencies. It is defined as

$$\mathrm{LSD}(u, f) = \frac{1}{n} \sum_{i=1}^{n} \sqrt{\frac{1}{w} \sum_{j=1}^{w} \left(20 \log_{10} \frac{|u(\boldsymbol{x}_i, \omega_j)|}{|f(\boldsymbol{x}_i, \omega_j)|}\right)^2}\,. \tag{3.40}$$

For synthetic datasets where the ground truth is available everywhere, the locations can be obtained by discretising the domain Ω just as for images in Equation 2.10. The frequencies can be chosen based on the desired application, for example by a sampling the range of human-audible frequencies. We use a version of this loss in Section 5.5.

4 Numerical Solvers Translated into Neural Architectures

In this chapter, we introduce our first integration of neural networks into numerical solvers, but still remain firmly rooted on the model side. Our largest contribution is a fine-grained stability proof for a class of discretisations for anisotropic diffusion. Afterwards, we relate the derivation of the discretisation itself to neural architectures, which leads to a fast implementation in deep learning frameworks.

The results in this chapter were presented at ECMI 2023 [202] and will appear in the proceedings. The proposal to split the anisotropic operator into four directions was made by Prof. Joachim Weickert. The specific formula for the free parameter delta can be found in an unpublished manuscript of Michael Krause. Prof. Joachim Weickert has also helped to sharpen and shorten the final version of the manuscript.

4.1 Introduction

Anisotropic diffusion models with a diffusion tensor have numerous applications in physics and engineering. Moreover, they also play a fundamental role in image analysis [242], where they are used for denoising, enhancement, scale-space analysis, and various interpolation tasks such as inpainting and superresolution. Sophisticated nonlinear models with appropriate directional behaviour can close interrupted structures and maintain or create sharp edges. However, to achieve results with only few dissipative artefacts and good rotation invariance, appropriate numerical approximations are needed. They should also come with provable stability guarantees and lead in a natural way to efficient implementations. Ideally they should also exploit the impressive parallelisation potential of modern GPUs. The goal of our contribution is to address these numerical issues.

4.2 Chapter Contributions

Motivated by image analysis applications, where one has a regular pixel grid and aims at simple numerical algorithms, we consider finite difference approximations on a 3×3 stencil. However, our results are also useful for anisotropic diffusion problems in other areas. Our contributions are threefold:

First, we study space discretisations of a general anisotropic diffusion operator on a 3×3 stencil. They split the 2-D anisotropic process into four 1-D diffusions. This class has one free parameter that can be used for quality optimisation. It covers the two-parameter stencil family of Weickert et al. [250], while removing its parameter redundancy and offering a simpler derivation. Moreover, it subsumes many previous discretisations with second-order consistency.

Our second contribution consists of a detailed stability analysis, where we establish fairly tight bounds on the spectral norm of the matrix associated with the stencil family. It allows to derive time step size restrictions for the corresponding explicit scheme (and accelerations that rely on it).

Last but not least, our stencil derivation based on a directional splitting enables the translation of the explicit anisotropic diffusion scheme into a ResNet block [86], which is a highly popular component of neural networks. This showcases that ideas are often shared between numerical schemes and neural architectures. More importantly, it allows simple and fast parallel implementations of anisotropic diffusion on GPUs using neural network libraries such as PyTorch.

4.3 Related Work

Many finite difference discretisations for anisotropic diffusion processes exist in the literature. Often they use spatial discretisations on a 3×3 stencil with consistency order two. The stencil class of Weickert et al. [250] comprises seven of them. Our findings offer a simpler derivation and representation of this family. Moreover, we extend the results from [250] by establishing concrete time step size limits for explicit schemes, connecting these algorithms to neural networks, and exploring simple and efficient parallelisations.

Our stencil family originates from a splitting 2-D anisotropic diffusion into four 1-D diffusions along fixed directions. Earlier splittings of this type intended to derive discretisations that are stable in the maximum norm [152, 242]. In general this is only possible for fairly mild anisotropies [242]. We consider stencils that offer stability in the Euclidean norm for all anisotropies.

Multiple recent works [11, 12, 188, 193] connect explicit schemes for PDEs to the ResNet [86] architecture. For example, Alt et al. [11] show that evolutions of dis-

cretised 1-D diffusion models with a scalar-valued diffusivity can be represented as ResNet blocks. In [12], they also explore the 2-D anisotropic case. However, their methodology is limited to evolution equations that arise as gradient descent of an energy functional. This excludes popular methods like edge-enhancing diffusion [242], for which Welk [251] has shown that no energy functional exists. We can translate these methods as well.

4.4 Organisation of the Chapter

In Section 4.5.1, we derive a class of finite difference discretisations on a 3×3 stencil. We establish stability results for the corresponding explicit scheme in Section 4.5.2. Section 4.5.3 shows how our splitting into 1-D diffusions leads to a translation of this scheme to a ResNet architecture, and it analyses its performance on a GPU. We conclude this chapter in Section 4.6.

4.5 Anisotropic Diffusion Stencils: From Simple Derivations over Stability Estimates to ResNet Implementations

4.5.1 Discretising Anisotropic Diffusion with the δ-Stencil

In this section, we study a simple and fairly general approach for a space discretisation of anisotropic diffusion on a 3×3 stencil. It is based on a directional splitting into four 1-D diffusion processes, which we discuss first.

1-D Diffusion To denoise a 1-D signal $f : [a, b] \to \mathbb{R}$, one can create simplified versions $\{u(x, t) \mid t \geq 0\}$ of it with the nonlinear diffusion process [167]

$$\partial_t u = \partial_x \Big(g\big((\partial_x u)^2\big)\, \partial_x u \Big) \qquad (t > 0), \tag{4.1}$$

$$u(x, 0) = f(x)\,. \tag{4.2}$$

For further details on nonlinear diffusion, we refer to Section 3.2.2. At the domain boundaries a and b, we impose reflecting (Neumann) boundary conditions. To prepare for the later translation to a neural architecture, we introduce the flux function $\Phi(\partial_x u) = g((\partial_x u)^2)\, \partial_x u$. It leads to the evolution equation $\partial_t u = \partial_x(\Phi(\partial_x u))$.

A finite difference discretisation of this 1-D process serves as building block for discretising anisotropic diffusion. To obtain a discrete signal $\boldsymbol{u} = (u_i) \in \mathbb{R}^N$, we sample u with grid size h. We discretise the derivatives with a forward difference in time and for the inner spatial derivative, and a backward difference for the outer one. This leads to the explicit scheme

$$\frac{u_i^{k+1} - u_i^k}{\tau} = \frac{1}{h}\left(\Phi\left(\frac{u_{i+1}^k - u_i^k}{h}\right) - \Phi\left(\frac{u_i^k - u_{i-1}^k}{h}\right)\right), \tag{4.3}$$

where $\tau > 0$ is the time step size, i denotes the location, and k the time level.

Anisotropic Diffusion In image analysis, anisotropic diffusion with a diffusion tensor [242] creates filtered versions $u(\boldsymbol{x}, t)$ of a scalar-valued (i.e. greyscale) image $f(\boldsymbol{x})$ by evolving it with the PDE

$$\partial_t u = \mathbf{div}\,(\boldsymbol{D}\,\boldsymbol{\nabla} u), \qquad \boldsymbol{D} = \begin{pmatrix} a & b \\ b & c \end{pmatrix} \tag{4.4}$$

where we initialise $u(\boldsymbol{x}, 0)$ with $f(\boldsymbol{x})$ and use reflecting boundary conditions. For further details on anisotropic diffusion, we refer to Section 3.2.3.

The $\boldsymbol{\delta}$-Stencil The discrete setting considers images $\boldsymbol{f}, \boldsymbol{u}^k \in \mathbb{R}^N$ obtained by sampling f and $u(., k\tau)$ with a grid size of h and arranging the pixel values into column vectors. The key idea of our discretisation is the decomposition of an anisotropic 2-D diffusion process into a sum of four nonlinear 1-D diffusions along the axial and diagonal directions

$$\boldsymbol{e}_0 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad \boldsymbol{e}_1 = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad \boldsymbol{e}_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \quad \boldsymbol{e}_3 = \frac{1}{\sqrt{2}}\begin{pmatrix} -1 \\ 1 \end{pmatrix}. \tag{4.5}$$

We determine directional diffusivities $w_0, \ldots, w_3$ for the corresponding directions by solving the system of three equations with four unknowns arising from

$$\mathbf{div}\left(\begin{pmatrix} a & b \\ b & c \end{pmatrix}\boldsymbol{\nabla} u\right) \stackrel{!}{=} \sum_{i=0}^{3} \partial_{\boldsymbol{e}_i}(w_i\,\partial_{\boldsymbol{e}_i} u). \tag{4.6}$$

Its solution has one free parameter which we call δ:

$$w_0 = a - \delta, \qquad w_1 = \delta + b, \qquad w_2 = c - \delta, \qquad w_3 = \delta - b\,. \tag{4.7}$$

All four 1-D diffusion processes can be discretised as before in (4.3). Each direction uses three pixels of its 3×3 neighbourhood. Discretising e.g. $\partial_{\boldsymbol{e}_1}(w_1\, \partial_{\boldsymbol{e}_1} u)$ in the pixels $(i-1, j-1)$, (i, j), and $(i+1, j+1)$ at distance $h\sqrt{2}$ gives

$$\frac{1}{h\sqrt{2}}\left((\delta + b)_{i+\frac{1}{2},j+\frac{1}{2}} \frac{u_{i+1,j+1} - u_{i,j}}{h\sqrt{2}} - (\delta + b)_{i-\frac{1}{2},j-\frac{1}{2}} \frac{u_{i,j} - u_{i-1,j-1}}{h\sqrt{2}} \right). \quad (4.8)$$

Incorporating all four directions yields the following ***δ*-stencil** for **div** $(\boldsymbol{D}\, \boldsymbol{\nabla} u)$:

$$\frac{1}{h^2} \cdot \begin{array}{|c|c|c|} \hline \frac{1}{2}(\delta - b)_{i-\frac{1}{2},j+\frac{1}{2}} & (c-\delta)_{i,j+\frac{1}{2}} & \frac{1}{2}(\delta + b)_{i+\frac{1}{2},j+\frac{1}{2}} \\ \hline (a-\delta)_{i-\frac{1}{2},j} & \begin{array}{c} -(a-\delta)_{i+\frac{1}{2},j} - (a-\delta)_{i-\frac{1}{2},j} \\ -\frac{1}{2}(\delta + b)_{i+\frac{1}{2},j+\frac{1}{2}} - \frac{1}{2}(\delta + b)_{i-\frac{1}{2},j-\frac{1}{2}} \\ -(c-\delta)_{i,j+\frac{1}{2}} - (c-\delta)_{i,j-\frac{1}{2}} \\ -\frac{1}{2}(\delta - b)_{i-\frac{1}{2},j+\frac{1}{2}} - \frac{1}{2}(\delta - b)_{i+\frac{1}{2},j-\frac{1}{2}} \end{array} & (a-\delta)_{i+\frac{1}{2},j} \\ \hline \frac{1}{2}(\delta + b)_{i-\frac{1}{2},j-\frac{1}{2}} & (c-\delta)_{i,j-\frac{1}{2}} & \frac{1}{2}(\delta - b)_{i+\frac{1}{2},j-\frac{1}{2}} \\ \hline \end{array} \quad (4.9)$$

where the x-axis points to the right, and the y-axis to the top. We assume that the diffusion tensor $\boldsymbol{D}$ is available in the staggered grid locations $\left(i \pm \frac{1}{2}, j \pm \frac{1}{2}\right)$. This is fairly natural if it relies on first-order derivatives, which can be computed with central differences in a 2×2 neighbourhood [250]. We obtain values in $\left(i \pm \frac{1}{2}, j\right)$ and $\left(i, j \pm \frac{1}{2}\right)$ by averaging:

$$(a-\delta)_{i\pm\frac{1}{2},j} = \frac{1}{2}\left((a-\delta)_{i\pm\frac{1}{2},j+\frac{1}{2}} + (a-\delta)_{i\pm\frac{1}{2},j-\frac{1}{2}} \right) \quad (4.10)$$

and similar for $(c-\delta)_{i,j\pm\frac{1}{2}}$. Then the δ-stencil family has consistency order two.

Incorporation of the Stencil Family of Weickert et al. [250] With (4.10) and $\delta = \alpha a + \beta b + \alpha c$, the δ-stencil family comprises that of Weickert et al. [250] that uses two parameters α and β. This shows that the parameters of the latter contain redundancy which we remove with the δ-stencil. Moreover, our stencil derivation is simpler than the one in [250] that has been obtained by discrete energy minimisation. In [250] it is shown that these stencils comprise seven discretisations from the literature. Since we are not aware of any second-order accurate discretisations on a 3×3 stencil that is not covered by this class, the δ-stencil family may even be more general.

4.5.2 Stability Theory for the δ-Stencil

Let the symmetric matrix $\boldsymbol{A} = \boldsymbol{A}(\boldsymbol{u}^k)$ act on an image $\boldsymbol{u}^k$ locally by applying the space-variant δ-stencil. Weickert et al. [250] have already established that $\boldsymbol{A}$ is negative semidefinite for $\alpha \leq \frac{1}{2}$ and $|\beta| \leq 1-2\alpha$. They have also replaced β by a parameter γ such that $\beta = \gamma(1–2\alpha)\,\mathrm{sgn}(b)$ and $|\gamma| \leq 1$. Choosing α close to $\frac{1}{2}$ and γ close to 1 improves rotation invariance and reduces dissipativity in experiments [250]. In practice, parameters $\alpha < 0$ are irrelevant and make a stability analysis more complicated. Thus, we exclude them from now on.

Consider an explicit anisotropic diffusion scheme $\boldsymbol{u}^{k+1} = (\boldsymbol{I} + \tau \boldsymbol{A}(\boldsymbol{u}^k))\,\boldsymbol{u}^k$ with unit matrix $\boldsymbol{I}$, time step size $\tau > 0$, and a negative semidefinite matrix $\boldsymbol{A} \neq \boldsymbol{0}$ such that for the spectral norm $\rho(\boldsymbol{A}) > 0$ holds. Then stability in the Euclidean norm in terms of $\|\boldsymbol{u}^{k+1}\|_2 \leq \|\boldsymbol{u}^k\|_2$ holds if

$$\tau \;\leq\; \frac{2}{\rho(\boldsymbol{A})}\,. \tag{4.11}$$

We can bound $\rho(\boldsymbol{A})$ as follows:

Theorem 2 (Bound on Spectral Norm)**.** *Let the eigenvalues of the diffusion tensor $\boldsymbol{D}$ be given by $\lambda_1 \geq \lambda_2 \geq 0$. Assume that $\delta = \alpha(a+c)+\beta b$ where $\beta = \gamma(1–2\alpha)\,\mathrm{sgn}(b)$ for $\alpha \in [0, \frac{1}{2}]$ and $|\gamma| \leq 1$. Then the spectral norm of the matrix $\boldsymbol{A}$ satisfies*

$$\rho(\boldsymbol{A}) \;\leq\; \frac{4\,(1-\alpha)\,(\lambda_1+\lambda_2) \;+\; 2\,(1-\gamma\,(1-2\alpha))\,(\lambda_1-\lambda_2)}{h^2}\,. \tag{4.12}$$

Proof. For the considered choice of α and β, we know from [250] that the symmetric matrix $\boldsymbol{A}$ is negative semidefinite. Thus, its spectral norm is determined by its smallest eigenvalue $\lambda_{\min}$ as $\rho(\boldsymbol{A}) = -\lambda_{\min}(\boldsymbol{A})$. Let us now bound $\lambda_{\min}(\boldsymbol{A})$ with Gershgorin's circle theorem [252] as introduced in Theorem 1. As $\boldsymbol{A}$ applies the δ-stencil, this theorem states that the smallest eigenvalue of $\boldsymbol{A}$ is bounded from below by the central stencil entry, minus the sum of absolute values of all other entries. Using (4.10) and grouping all terms by the four diffusion tensor locations $(i \pm \frac{1}{2}, j \pm \frac{1}{2})$ gives

$$\begin{aligned}\rho(\boldsymbol{A}) \;\leq\; \tfrac{1}{2h^2} \max_{a,b,c} \;\big(\;& ((a-\delta)+|a-\delta|+(\delta+b)+|\delta+b|+(c-\delta)+|c-\delta|)_{i-\frac{1}{2},j-\frac{1}{2}} \\ +\;& ((a-\delta)+|a-\delta|+(\delta-b)+|\delta-b|+(c-\delta)+|c-\delta|)_{i-\frac{1}{2},j+\frac{1}{2}} \\ +\;& ((a-\delta)+|a-\delta|+(\delta-b)+|\delta-b|+(c-\delta)+|c-\delta|)_{i+\frac{1}{2},j-\frac{1}{2}} \\ +\;& ((a-\delta)+|a-\delta|+(\delta+b)+|\delta+b|+(c-\delta)+|c-\delta|)_{i+\frac{1}{2},j+\frac{1}{2}}\big).\end{aligned} \tag{4.13}$$

Next, we bound the right hand side from above by assuming that the diffusion tensors at the different locations are independent. The notation

$$M_\pm \;:=\; (a-\delta) + |a-\delta| + (\delta \pm b) + |\delta \pm b| + (c-\delta) + |c-\delta| \tag{4.14}$$

allows us to rewrite the bound as

$$\rho(\boldsymbol{A}) \;\leq\; \frac{1}{h^2}\big(\max_{a,b,c}(M_+) + \max_{a,b,c}(M_-)\big)\,. \tag{4.15}$$

In the following, we determine the maximum of M_+. Calculations for M_- are analogous. Notice that in M_+, the three terms $a-\delta$, $\delta+b$, and $c-\delta$ appear pairwise with their absolute values. This will simplify the calculation of the maximum. Consider the sum of the three terms:

$$m_+ \;:=\; (a-\delta) \;+\; (\delta+b) \;+\; (c-\delta) \;=\; a+b+c-\delta\,. \tag{4.16}$$

If m_+ has a maximum in which $a-\delta$, $\delta+b$, and $c-\delta$ are all nonnegative, then $\max(M_+) = 2\,\max(m_+)$, since $x+|x| = 2x$ for $x \geq 0$. We now proceed to show that such a maximum of m_+ exists. To this end, we rewrite the entries a, b, and c of the positive semidefinite diffusion tensor $\boldsymbol{D}$ in terms of its normalised eigenvectors $(u,v)^\top$, $(v,-u)^\top$ and their eigenvalues $\lambda_1 \geq \lambda_2 \geq 0$:

$$a = \lambda_1 u^2 + \lambda_2 v^2\,, \qquad b = (\lambda_1-\lambda_2)\,uv\,, \qquad c = \lambda_2 u^2 + \lambda_1 v^2\,. \tag{4.17}$$

The possible ranges for eigenvalues may differ between diffusion models. Therefore, we determine the maximum of m_+, and by extension our limit on $\rho(\boldsymbol{A})$, as a function of λ_1 and λ_2. This leaves the entries u, v of the eigenvectors as the only variables to maximise m_+ over. Using (4.17) in (4.16) gives

$$\begin{aligned}
\max_{u,v}(m_+) \;&=\; \max_{u,v}\big((1-\alpha)\,(\lambda_1+\lambda_2) \;+\; (1-\beta)(\lambda_1-\lambda_2)\,uv\big) \\
&=\; \max_{u,v}\big((1-\alpha)\,(\lambda_1+\lambda_2) \;+\; (1-\gamma(1-2\alpha)\,\mathrm{sgn}(uv))\,(\lambda_1-\lambda_2)\,uv\big) \\
&=\; \max_{u,v}\begin{cases}
(1-\alpha)\,(\lambda_1+\lambda_2) + \underbrace{(1-\gamma(1-2\alpha))}_{\geq 0}\underbrace{(\lambda_1-\lambda_2)}_{\geq 0}\underbrace{uv}_{>0} & \text{for } uv > 0, \\
(1-\alpha)\,(\lambda_1+\lambda_2) + \underbrace{(1+\gamma(1-2\alpha))}_{\geq 0}\underbrace{(\lambda_1-\lambda_2)}_{\geq 0}\underbrace{uv}_{\leq 0} & \text{for } uv \leq 0.
\end{cases}
\end{aligned} \tag{4.18}$$

The case where $uv > 0$ always gives larger results than the second one. Thus,

$$\max_{u,v}(m_+) \;=\; \max_{u,v}\left((1-\alpha)\,(\lambda_1+\lambda_2) \;+\; (1-\gamma(1-2\alpha))\,(\lambda_1-\lambda_2)\,uv\right). \tag{4.19}$$

We maximise the second term by maximising uv. For our normalised eigenvectors, $u^2+v^2=1$ holds. Hence, $\max(uv)=\frac{1}{2}$ for $u=v=\pm\frac{1}{\sqrt{2}}$. Since we only need the maximal function value, we can consider only $u=v=\frac{1}{\sqrt{2}}$. This gives

$$\max_{u,v}(m_+) \;=\; (1-\alpha)\,(\lambda_1+\lambda_2) \;+\; \frac{1}{2}\,(1-\gamma(1-2\alpha))\,(\lambda_1-\lambda_2)\,. \tag{4.20}$$

We are not able to draw conclusions about the maximum of M_+ from the maximum of m_+ yet. It remains to show that $a-\delta$, $\delta+b$, and $c-\delta$ are all nonnegative in our maximum with $u=v=\frac{1}{\sqrt{2}}$. We start with $a-\delta$ and use (4.17):

$$\begin{aligned}
(a-\delta)|_{u=v=\frac{1}{\sqrt{2}}} \;&=\; \big(a-\beta b-\alpha(a+c)\big)|_{u=v=\frac{1}{\sqrt{2}}} \\
&=\; \frac{1}{2}\Big((\lambda_1+\lambda_2)-\gamma\underbrace{(1-2\alpha)}_{\geq 0}\underbrace{(\lambda_1-\lambda_2)}_{\geq 0}-2\alpha(\lambda_1+\lambda_2)\Big) \\
&\geq\; \frac{1}{2}\Big((\lambda_1+\lambda_2)-(1-2\alpha)(\lambda_1-\lambda_2)-2\alpha(\lambda_1+\lambda_2)\Big) \\
&=\; (1-2\alpha)\,\lambda_2 \;\geq\; 0\,.
\end{aligned} \tag{4.21}$$

In a similar way, one shows $(c-\delta)|_{u=v=\frac{1}{\sqrt{2}}} \geq 0$. For $b+\delta$ we verify

$$\begin{aligned}
(b+\delta)|_{u=v=\frac{1}{\sqrt{2}}} \;&=\; \big((1+\beta)\,b+\alpha(a+c)\big)|_{u=v=\frac{1}{\sqrt{2}}} \\
&=\; \frac{1}{2}\underbrace{(1+\gamma(1-2\alpha))}_{\geq 0}\underbrace{(\lambda_1-\lambda_2)}_{\geq 0}+\alpha\,(\lambda_1+\lambda_2) \;\geq\; 0\,.
\end{aligned} \tag{4.22}$$

As all three terms are nonnegative in the maximum, we can conclude that

$$\max_{u,v}(M_+) \;=\; 2\max_{u,v}(m_+) \;=\; 2\,(1-\alpha)\,(\lambda_1+\lambda_2)+(1-\gamma(1-2\alpha))\,(\lambda_1-\lambda_2)\,. \tag{4.23}$$

Analogous computations lead to the same maximum for M_-. Inserting both into (4.15) produces the claimed bound on the spectral norm. □

Using Theorem 2 within (4.11) directly gives the following time step size limit:

Corollary 1 (Stability of Explicit Scheme)**.** *An explicit anisotropic diffusion scheme* $\boldsymbol{u}^{k+1} = \big(\boldsymbol{I} + \tau \boldsymbol{A}(\boldsymbol{u}^k)\big)\boldsymbol{u}^k$*, where* $\boldsymbol{A}$ *satisfies the assumptions of Theorem 2 with* $\lambda_1 > 0$*, is stable in the Euclidean norm for*

$$\tau \;\leq\; \frac{h^2}{2\,(1-\alpha)\,(\lambda_1+\lambda_2) \;+\; (1-\gamma\,(1-2\alpha))\,(\lambda_1-\lambda_2)}\,. \tag{4.24}$$

While our proof does not guarantee that this bound is strict, our practical experience suggests that it is. Corollary 1 covers two important special cases:

1. In the **homogeneous diffusion case** with $\lambda_1 = \lambda_2 = 1$, this time step size limit simplifies to $\tau \leq \frac{h^2}{4(1-\alpha)}$. Moreover, setting $\alpha := 0$ turns the δ-stencil into the standard five point approximation of the Laplacian, which leads to the well-known 2-D time step size limit $\tau \leq \frac{h^2}{4}$; see e.g. [150].

2. In the **maximally anisotropic case** with $\lambda_1 = 1$, $\lambda_2 = 0$, and $\gamma = 1$, one performs 1-D diffusion along one eigendirection of $\boldsymbol{D}$. Then (4.24) becomes $\tau \leq \frac{h^2}{2}$. In spite of being in a 2-D setting, this coincides with the typical 1-D time step size limit [150], which is less restrictive. This shows that our scheme takes full advantage of the anisotropy. In image analysis, this result is relevant for coherence-enhancing nonlinear diffusion filters [242]. Similar findings have also been made with a recent numerical scheme for a maximally anisotropic backward parabolic PDE [195].

4.5.3 Translating Anisotropic Diffusion into ResNets

Let us now interpret our explicit scheme in the context of neural networks. This extends the result of Alt et al. [11] from the 1-D setting to the 2-D anisotropic case. We start by presenting their translation of 1-D diffusion into a ResNet block, and then build the anisotropic ResNet block from there.

ResNets ResNets [86] are very popular neural network architectures. They use ResNet blocks that compute an output $\boldsymbol{u}^{k+1}$ from an input $\boldsymbol{u}^k$ by

$$\boldsymbol{u}^{k+1} \;=\; \sigma_2\big(\boldsymbol{u}^k + \boldsymbol{K}_2\,\sigma_1\big(\boldsymbol{K}_1\boldsymbol{u}^k + \boldsymbol{b}_1\big) + \boldsymbol{b}_2\big) \tag{4.25}$$

for discrete convolution kernels $\boldsymbol{K}_1, \boldsymbol{K}_2$, bias vectors $\boldsymbol{b}_1, \boldsymbol{b}_2$, and nonlinear activation functions σ_1, σ_2. For further details on ResNets, see also our introduction in Section 3.1.3.

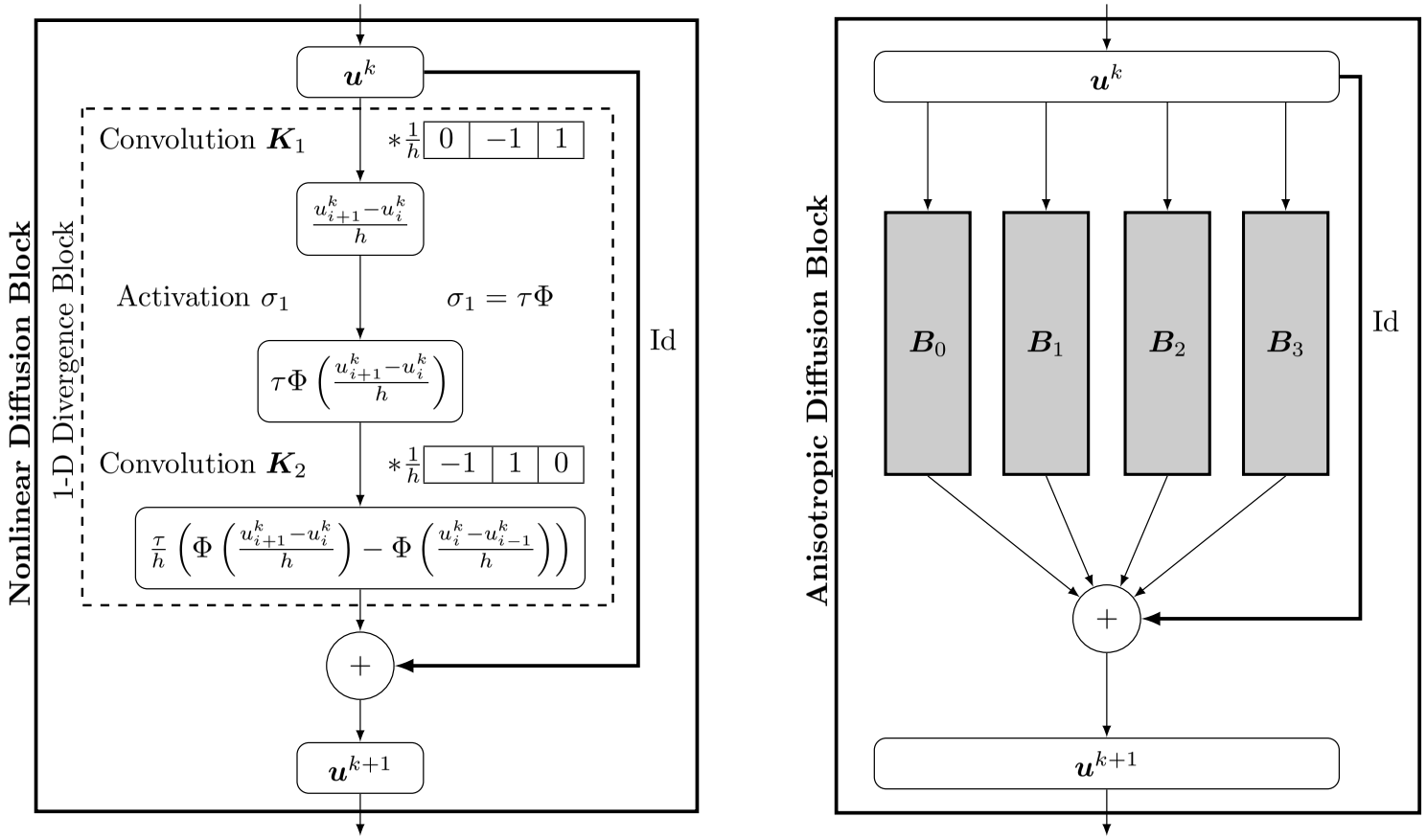


Figure 4.1: **(a) Left**: Translation of 1-D nonlinear diffusion into a ResNet block. Adapted from [11]. **(b) Right**: Anisotropic diffusion as a ResNet block with a sum of four 1-D divergence blocks. The blocks $\boldsymbol{B}_0, \ldots, \boldsymbol{B}_3$ correspond to the directions $\boldsymbol{e}_0, \ldots, \boldsymbol{e}_3$.

Translating 1-D Diffusion into ResNets The basic translation of 1-D diffusion into ResNets is surprisingly simple [11, 193]: In vector notation, the explicit scheme (4.3) becomes

$$\boldsymbol{u}^{k+1} \;=\; \boldsymbol{u}^k + \tau \boldsymbol{D}_h^- \big(\Phi\big(\boldsymbol{D}_h^+ \boldsymbol{u}^k\big)\big). \tag{4.26}$$

Here, $\boldsymbol{D}_h^+$ and $\boldsymbol{D}_h^-$ represent matrices computing forward and backward first order derivative approximations with grid size h. Comparing (4.26) to the ResNet block (4.25) reveals that both perform the same computations when identifying

$$\boldsymbol{K}_1 = \boldsymbol{D}_h^+, \quad \sigma_1 = \tau\Phi, \quad \boldsymbol{K}_2 = \boldsymbol{D}_h^-, \quad \boldsymbol{b}_1 = \boldsymbol{b}_2 = \boldsymbol{0}, \quad \sigma_2 = \mathrm{Id}\,. \tag{4.27}$$

The computational graph for this is shown in Figure 4.1(a).

Alt et al. [11] use this connection to advocate ResNet architectures with mirrored kernels $\boldsymbol{K}_1$ and $\boldsymbol{K}_2$ to guarantee stability in the Euclidean norm. Moreover, their experiments show advantages of nonmonotone activation functions.

Translating 2-D Anisotropic Diffusion into ResNets Our directional splitting allows also a natural translation of anisotropic diffusion into ResNets. We split the divergence term of 2-D anisotropic diffusion into a sum of four divergence terms of 1-D diffusion processes, and use the previous translation for each. This is illustrated in Figure 4.1(b). By appropriately concatenating the 2-D convolution kernels into 4-D tensors, we match the ResNet structure precisely.

Experiments Implementing numerical schemes for GPUs using CUDA can be labour-intensive and requires expertise. However, deep learning frameworks are capable of fully automatic and efficient parallelisation of user code. As we were able to decompose our discretisation into neural network primitives, we can use these frameworks to obtain an efficient implementation with little effort.

As a prototypical anisotropic diffusion process as described in (4.4) we use edge-enhancing image diffusion [242], for which we consider 10 iterations of an explicit scheme. We compare three implementations: The first uses C and runs on the CPU. It computes entries of the δ-stencil before applying it to the image. The second is an implementation in the PyTorch framework which follows the same strategy. As this style is uncommon in most neural networks, the implementation is fairly involved. The third also uses PyTorch, but follows our ResNet translation. It only requires two convolutions, one activation function, and a summation. This leads to a concise and simple implementation.

For an image with 2048×2048 pixels, our C code takes $1.6\ s$ on an AMD 5800X CPU. Both our PyTorch implementations perform one order of magnitude faster at

$0.16\,s$ and $0.15\,s$ respectively on an Nvidia 3090 GPU. This demonstrates that our ResNet translation is able to significantly accelerate EED with a straightforward parallel implementation. It is even as fast as the much more involved stencil-based GPU implementation. This behaviour is consistent across image and batch sizes, provided that the total pixel count is sufficiently large.

4.6 Chapter Conclusions

We have explored three aspects of anisotropic diffusion stencils. The first was an intuitive derivation of a large second-order stencil family based on directional splitting. While it covers the full stencil class of Weickert et al. [250], its derivation is simpler, and it requires only one free parameter (δ) instead of two. Therefore, we call it the δ-stencil family.

Secondly, we have established a rigorous spectral norm estimate of the matrix associated to this stencil family. It allows to derive fairly tight time step size limits of explicit schemes to guarantee stability in the Euclidean norm. We have restricted ourselves to explicit schemes, since they are structurally similar to feedforward neural networks. Moreover, they form the backbone of acceleration methods based on super time stepping [248] and extrapolation concepts [84]. Further multigrid-like acceleration options may arise from multiscale network features such as pooling operations and U-net structures [11].

Thirdly, the directional splitting from our derivation has been instrumental in linking anisotropic diffusion to ResNets. It paves the road to an effortless and efficient parallelisation with libraries such as PyTorch.

In the larger context of this thesis, this chapter uses the fewest neural components. In fact, this chapter did not contain any trainable weights at all. Our methodology only made use of the efficient implementations of deep learning frameworks and used them outside their usual scope.

However, future research could change this: Based on our implementation, iterations of anisotropic diffusion methods can be integrated in deep learning pipelines. Furthermore, the diffusion tensor entries could be made trainable, and adapt to the local image structure in more complex ways.

For the next chapter, we will take another step towards neural methods. Our models there use the representational power of neural networks to model discrete images and continuous functions, and combine it with gradient descent to solve mathematical models. However, these methods still do not require datasets and continue to rely on human-designed models.

5 Neural Networks as Numerical Solvers

This chapter explores how neural networks can be used as solvers for numerically challenging problems in two ways.

The first addresses the Euler's elastica inpainting problem and was published in the proceedings of and presented at EUVIP 2022 [200]. In addition, I also presented early results from this work at the SIAM Conference on Imaging Science 2022. All experiments are my own. Dr. Tobias Alt-Veit provided important input on theoretical and practical machine learning aspects. Prof. Joachim Weickert suggested the discretisation of the elastica energy, and provided valuable discussion and guidance throughout the preparation of the manuscript. Michael Ertel provided a theoretical justification for the used finite difference derivative approximations.

The second contribution in this chapter constitutes a deviation from the image domain to deal with inpainting of the sound field magnitude. It is under review at the time of submission of this thesis. All experiments and the proposed neural architecture are my own. Prof. Shoichi Koyama provided indispensable guidance on matters specific to the domain of acoustics, along with significant contributions to the manuscript. Dr. Tomohiko Nakamura and Prof. Mirco Pezzoli provided input in multiple discussions.

5.1 Introduction

Progress in solving PDEs or minimising variational energies using finite differences, finite elements, or even meshless methods has steadily advanced over the last decades. Nevertheless, poorly conditioned or very large problems can still require prohibitive amounts of computational power. Moreover, many problems can require complex tailored numerics with equally complex implementations.

Neural networks are able to address some of these challenges by being able to efficiently search high-dimensional domains and through a remarkable robustness to ill-posed problems. They are able to detect correlations in immense amounts of data and can use them for predictions, e.g. in the context of image generation or completion [184].

Still, purely data-driven approaches which do away with most priors commonly require more data than is available in many scientific applications such as medical image analysis. Furthermore, they can disobey physical laws, or generalise poorly to unseen data. This motivates the field of physics-informed machine learning [114], which aims to integrate physics into learning tasks. Here, some data is used in conjunction with fully or partially known dynamics.

A popular example of this family of methods are *physics-informed neural networks* (PINNs) [177, 178], which penalise deviation from a PDE through introducing appropriate loss functions. This strategy of soft penalty constraints is highly flexible, as nearly every desirable property can be encoded this way. In a similar spirit, deep energies [77] propose to employ energy functionals as loss functions for neural networks.

In this chapter, we consider one extreme of the gamut of physics-informed methods: The setting where the full physics of our model are known, but only data from a single sample is available. In this case, we do not rely on the ability of neural networks to infer correlations from large datasets. Instead, we essentially employ them as solvers for our numerical problems: We make use of the implicit prior defined by a neural architecture [54] and the ability of modern deep learning framework to efficiently minimise energies through gradient descent.

5.2 Chapter Contributions

Our first contribution considers image inpainting with the Euler's elastica energy functional. This variational model can be transparently derived from desirable properties for shape completion, but remains numerically challenging. Existing implementations often suffer from blurry edges or poor rotation invariance. As a remedy, we design the first neural algorithm that simulates inpainting with Euler's Elastica. We use the deep energy concept which employs the variational energy as neural network loss. Furthermore, we pair it with a deep image prior where the network architecture itself acts as a prior. This yields better inpaintings by steering the optimisation trajectory closer to the desired solution. Our results are qualitatively on par with state-of-the-art algorithms on elastica-based shape completion. They combine good rotation invariance with sharp edges. Moreover, we benefit from the high efficiency and effortless parallelisation within a neural framework. Our neural elastica approach only requires 3x3 central difference stencils. It is thus much simpler than other well-performing algorithms for elastica inpainting. Last but not least, it is unsupervised as it requires no ground truth training data.

In addition, we propose a method for estimating the magnitude distribution of an acoustic field from spatially sparse magnitude measurements. Such a method is useful when phase measurements are unreliable or inaccessible. Physics-informed

neural networks have shown promise for sound field estimation by incorporating constraints derived from governing PDEs into neural networks. However, they do not extend to settings where phase measurements are unavailable, as the loss function based on the governing PDE relies on phase information. To remedy this, we propose a phase-retrieval-based PINN for magnitude field estimation. By representing the magnitude and phase distributions with separate networks, the PDE loss can be computed based on the reconstructed complex amplitude. We demonstrate the effectiveness of our phase-retrieval-based PINN through experimental evaluation.

5.3 Related Work

Most approaches in physics-informed machine learning work by penalising deviation from assumptions or equations through additional loss terms. This very flexible approach allows to e.g. encourage adherence to PDEs [128, 178, 211] or algebraic equations [64] by penalising their squared residual.

PINNs [85, 178, 211] represent a widely used framework for initial boundary value problems. They model the target function with a neural network, which is trained to adhere to the desired PDE. A wide variety of extensions [101, 102, 151, 223, 236] were proposed over the years, aiming at increasing the quality or performance, see e.g. [142] for a recent overview. However, PINNs can still suffer from low convergence rates or instability on some problems [237, 238]. We employ PINNs in Section 5.5.

As PINNs focus by design on solving individual instances of PDEs, they are unsuitable for real-time inference. Neural operator methods [139] alleviate this by learning a mapping from an input function space, for example from an initial condition, directly to the PDE solution. Extensions move to learn the mapping in the Fourier domain [130] or in a latent space [121]. However, these approaches require datasets consisting of pairs of inputs and PDE solutions, which are not available for all problems. *Physics-informed neural operators* (PINO) [131] offer a middle ground by including a PDE loss function similar to PINNs. This reduces the number of training samples needed to learn the operator.

The *Deep Ritz* method [60] or *variational PINNs* (VPINNs) [65, 117] also aim to solve PDEs, but rely on reformulating the PDE as a variational energy. Such a formulation does not exist for all PDEs. Furthermore, *backwards stochastic differential equations* (BSDEs) rely on stochastic reformulations of PDEs [58, 59]. They excel at very high-dimensional problems commonly found in finance.

Methods for variational problems are closely related in spirit [57, 77]. Here, *deep energies* [77] describe the use of a discretised variational energy as loss function. Similar to neural operators, a network is trained to learn a mapping from an input to an

output which minimised the variational energy. Here however, no ground truth results are used for training, but exclusively the variational energy. We incorporate this concept in our solver for Euler' elastica in Section 5.4. Our use of a neural network to predict the solution of an inpainting PDE in Chapter 6 follows a similar philosophy.

In conjunction with the soft-constrained approaches above, network architectures can be deliberately designed to only be able to represent certain function classes. In the context of PINNs, this can be used for boundary conditions [55, 218]. There, networks are built such that all functions they can represent fulfil e.g. Neumann or Dirichlet boundary conditions. Other works ensure rotation invariance this way [12].

5.4 CNN-Based Euler's Elastica Inpainting with Deep Energy and Deep Image Prior

Inpainting [22, 61, 147] aims at reconstructing missing regions of an image in a semantically plausible way. In the last two decades, many inpainting techniques have been proposed using model-based [83] ideas as well as deep learning [165, 262]. A particularly well-researched variational model goes back to Mumford [154] which utilises Euler's elastica [66]. It minimises an energy with a total variation and a curvature term. Both terms offer a clear interpretation for shape completion tasks and lead to a particularly transparent model.

Unfortunately, the numerical minimisation of this nonconvex energy is highly nontrivial, since its gradient flow is a singular, anisotropic nonlinear partial differential equation (PDE) of order four. It is difficult to find efficient algorithms that produce sharp edges, offer a good approximation of rotation invariance, and can inpaint large gaps. Numerous numerical ideas have been proposed to tackle this challenging task, including multigrid techniques [29], discrete gradient methods [181], augmented Lagrangian approaches [222], operator splitting [261], and lifting concepts [34]; see [112] for an overview and additional references. To the best of our knowledge, however, no deep neural networks have been used for solving elastica inpainting. In view of the success of convolutional neural networks (CNNs) [78] in all visual computing fields as well as in numerical analysis, this is surprising. The goal of this section is to close this gap.

Our Contributions Our approach focuses on *deep energies* [77]. It uses the variational energy as loss function for training a neural network without ground truth data. For digital images, this only requires to discretise the energy. The task of its numerical minimisation via gradient descent is delegated to the neural network. In this way we benefit from its powerful optimisation algorithms and an effortless

parallelisation on GPUs. It should be noted that the discretisation of the elastica energy only involves first and second order derivatives. This is far more pleasant than discretising a fourth order gradient flow. We show that 3×3 central differences with optimised rotation invariance are sufficient for this task.

While our discrete energy is simple, it cannot penalise checkerboard-like artefacts. As a remedy, we employ a second neural concept, namely *deep image priors* [229]: This approach reparametrises an image as the output of a neural network and has a regularising effect. It efficiently prevents the introduction of unnatural artefacts into the resulting image.

Our experiments show that both neural components are essential for obtaining a good minimiser of the elastica energy. We even reach the quality of a sophisticated state-of-the-art algorithm. Our approach is efficient, offers sharp results with good rotation invariance, and can bridge large gaps.

Related Work Combinations of Euler's elastica and neural concepts are rare and restricted to areas beyond image inpainting. The use of elastica for supervised classification is investigated in [132]. There, they act as a regulariser on the level lines of a learned classifier. An elastica-based segmentation model where the minimiser is predicted by a neural network was explored in [37].

Inpainting models that rely on deep learning can produce visually realistic images; see e.g. [165, 262]. However, they require tremendous amounts of training data, and uncovering the learned model is infeasible due to the large number of trainable parameters. The concept of deep energies [77] follows a different philosophy by proposing to use classical variational energy models as loss functions for neural networks. Deep energy models require no ground truth training data, and the mathematical model is fully defined by the energy.

Deep image priors [229] have shown how the architecture of a neural network can act as a regularising prior on its output [54]. They preserve natural visual objects while attenuating noise. Therefore, their optimisation trajectories pass close by the desired solution in tasks such as noise or artefact removal. Many works build upon this concept and propose different strategies for stopping the iteration as close as possible to the desired solution; see e.g. [234] and the references therein.

Organisation of the Section In Section 5.4.1, we briefly review Euler's elastica model for inpainting. Afterwards, in Section 5.4.2, we introduce our neural algorithm. We evaluate it in Section 5.4.3 and present our conclusions in Section 5.4.4.

5.4.1 Review: Elastica Inpainting

Let $f : \Omega \to \mathbb{R}$ denote a continuous greyscale image that is only known on an *inpainting mask* K, a subset of the rectangular image domain Ω. Inpainting aims at reconstructing f in the *inpainting domain* $\Omega \setminus K$. Euler's elastica obtain such a reconstruction u as a minimiser of the energy

$$E(u) = \int_{\Omega \setminus K} \|\nabla u\|_2 \left(b + (1-b)\kappa^2\right) dx\, dy. \tag{5.1}$$

On the inpainting mask K, we enforce $u = f$. The energy (5.1) combines the total variation (TV) [190] penaliser given by $\|\nabla u\|_2$ with the level line curvature $\kappa = \mathbf{div}\left(\frac{\nabla u}{\|\nabla u\|_2}\right)$, and $\|\cdot\|_2$ denotes the Euclidean norm. The balance between the two components is steered by a parameter $b \in [0,1]$. It can produce results from pure contour length minimisation (for $b = 1$) to pure curvature minimisation ($b = 0$). Both components are highly desirable and psychophysically relevant for shape completion [113].

Although the elastica energy (5.1) is transparent, it involves many inherent problems: $\|\nabla u\|_2$ is nondifferentiable in $\mathbf{0}$, and dividing by it within the curvature term may create singularities. Moreover, since $\|\nabla u\|_2\, \kappa$ is the second derivative of u in level line direction, the elastica energy is highly anisotropic. It is also nonconvex and may thus have numerous local minimisers. Any minimiser is a steady state of the gradient flow PDE of the energy. Since the energy (5.1) is nonquadratic, its gradient flow is nonlinear. Moreover, (5.1) involves derivatives up to order two. This creates a gradient flow of order four; see [207] for an explicit formula. Fourth order PDEs are numerically much harder to solve than the widespread second order ones. Other challenges such as nondifferentiability, singular behaviour, and anisotropy are inherited.

As already mentioned, these problems directly lead to numerous numerical challenges. They have inspired many researchers to develop highly sophisticated algorithms for elastica inpainting. We show that the concepts of deep energies and deep priors help us to make these problems more manageable and lead to a simple and well-performing algorithm.

5.4.2 Solving Elastica with Deep Learning

We follows the *discretise then optimise* paradigm. Discretising the energy rather than its fourth order gradient flow allows us to restrict ourselves to derivatives up to order two. Moreoever, we know that the resulting algorithm minimises a discrete energy,

which we can use to monitor its success. Most importantly, however, we can exploit the capabilities of neural networks to minimise difficult nonconvex energies in an efficient way.

Discretising the Elastica Energy

The components of the energy (5.1) can be spelled out as

$$\|\nabla u\|_2 \;\approx\; \sqrt{u_x^2 + u_y^2 + \varepsilon^2}\,, \tag{5.2}$$

$$\kappa \;\approx\; \frac{u_y^2 u_{xx} - 2u_x u_y u_{xy} + u_x^2 u_{yy}}{(u_x^2 + u_y^2 + \varepsilon^2)^{\frac{3}{2}}}\,, \tag{5.3}$$

where subscripts denote partial derivatives. Regularising $\|\nabla u\|_2$ with $\varepsilon > 0$ avoids nondifferentiability and division by zero.

We obtain discrete images $\boldsymbol{u}$, $\boldsymbol{f}$ by sampling the continuous functions u, f on a uniform grid with distance h. We represent images as vectors by stacking the grey values into a column vector. Moreover, we consider a discrete binary inpainting mask $\boldsymbol{c}$ as an indicator image of the set K. A value of 1 marks known data, and 0 indicates the inpainting domain.

Finite difference discretisations of derivatives are always a compromise between many criteria: Ideally they are simple, offer good rotation invariance, do not introduce artificial blur, and have a high approximation quality for all frequencies. In practice, improving the performance w.r.t. one criterion often comes at the expense of another. Our discretisation prioritises *simplicity* and good *rotation invariance,* since both are frequent weaknesses of existing elastica algorithms. We shall also see that it does not create blurry edges. Its main drawback are checkerboard artefacts in the highest grid frequency. We will cure them with a deep image prior.

For *simplicity*, we approximate all derivatives by finite differences on a 3×3 stencil, which allows consistency order two. To guarantee that they all fit together, we obtain them jointly from the coefficients of a weighted least squares regression polynomial of order two. We achieve good *rotation invariance* with the tensor product of the binomial weights $[\frac{1}{4}, \frac{1}{2}, \frac{1}{4}]$ in x- and y-direction, which approximates a rotationally invariant 2-D Gaussian. This yields

$$\partial_x \approx \frac{1}{8h}\begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline -2 & 0 & 2 \\ \hline -1 & 0 & 1 \\ \hline\end{array}\,, \quad \partial_{xx} \approx \frac{1}{4h^2}\begin{array}{|c|c|c|}\hline 1 & -2 & 1 \\ \hline 2 & -4 & 2 \\ \hline 1 & -2 & 1 \\ \hline\end{array} \tag{5.4}$$

and corresponding stencils in y-direction. Note that our approximations for ∂_x and ∂_y coincide with Sobel operators, which are well-known for their rotation invariance. For the mixed derivative we obtain (with y-axis oriented upwards)

$$\partial_{xy} \approx \frac{1}{4h^2} \begin{array}{|c|c|c|} \hline -1 & 0 & 1 \\ \hline 0 & 0 & 0 \\ \hline 1 & 0 & -1 \\ \hline \end{array} . \tag{5.5}$$

With these stencils and by replacing the integral by a summation over all pixels i in the inpainting domain (i.e. in locations i with $c_i = 0$), we arrive at our discrete counterpart of (5.1):

$$E(\boldsymbol{u}) \;=\; \sum_i (1 - c_i) \, \|\boldsymbol{\nabla} u\|_{2,i} \left(b + (1-b)\kappa_i^2 \right) . \tag{5.6}$$

Minimisation via Deep Energies

To minimise this energy with advanced variants of gradient descent, we use it as a loss function in a neural network. Golts et al. [77] call this a *deep energy*. Modern deep learning frameworks allow us to implement the discrete energy $E(\boldsymbol{u})$ and use backpropagation [191] to evaluate $\boldsymbol{\nabla}_{\boldsymbol{u}} E(\boldsymbol{u})$. With this we can perform gradient descent:

$$\boldsymbol{u}^{k+1} = \boldsymbol{u}^k - \tau \boldsymbol{\nabla}_{\boldsymbol{u}^k} E(\boldsymbol{u}^k), \tag{5.7}$$

where τ denotes the step size and superscripts the iteration level. For Euler's elastica, this frees us from the need to find a good discretisation of a fourth-order PDE. Instead, only the energy which contains derivatives up to second order must be discretised. Its gradient is efficiently computed by the network by exploiting the parallelism of GPUs. Note that so far, our network is used as a pure optimisation tool. It has no trainable weights apart from the iteratively inpainted image $\boldsymbol{u}^k$.

Regularisation with a Deep Image Prior

By design, our finite difference stencils (5.4)–(5.5) are simple and offer good rotation invariance. However, it is easy to see that the discrete elastica energy does not penalise oscillations with the highest grid frequency, where both Sobel operators return zero. As a result, checkerboard artefacts can appear. We avoid them with the regularising properties of a deep image prior [54, 229]: It has been shown that typical neural networks converge rapidly towards natural images while attempting to avoid unnatural artefacts. As a consequence, our resulting model first recovers an image that fulfils

the desired elastica properties, and may introduce undesirable artefacts afterwards. Thus, stopping the minimisation at the right time is crucial.

We impose a deep prior on $\boldsymbol{u}$ by replacing it by the output of a network $\boldsymbol{u} = \mathcal{N}(\boldsymbol{c}, \boldsymbol{f}, \boldsymbol{\theta})$ with mask $\boldsymbol{c}$ and known grey values $\boldsymbol{f}$ as inputs, and parametrised by weights $\boldsymbol{\theta}$. Instead of searching for the minimiser $\boldsymbol{u}$ directly, we are now solving for the weights $\boldsymbol{\theta}$ which minimise $E(\mathcal{N}(\boldsymbol{c}, \boldsymbol{f}, \boldsymbol{\theta}))$. Gradient descent gives

$$\boldsymbol{\theta}^{k+1} = \boldsymbol{\theta}^k - \tau \nabla_{\boldsymbol{\theta}^k} E(\mathcal{N}(\boldsymbol{c}, \boldsymbol{f}, \boldsymbol{\theta}^k)). \tag{5.8}$$

Note the close connection to the explicit scheme (5.7), as the gradient w.r.t. the weights is computed via the chain rule. With $\boldsymbol{u}^k := \mathcal{N}(\boldsymbol{c}, \boldsymbol{f}, \boldsymbol{\theta}^k)$ and the Jacobian $\nabla_{\boldsymbol{\theta}^k} \boldsymbol{u}^k$, this reads

$$\nabla_{\boldsymbol{\theta}^k} E(\boldsymbol{u}^k) = (\nabla_{\boldsymbol{\theta}^k} \boldsymbol{u}^k)\, \nabla_{\boldsymbol{u}^k} E(\boldsymbol{u}^k). \tag{5.9}$$

The gradient $\nabla_{\boldsymbol{u}^k} E(\boldsymbol{u}^k)$ is calculated first before being distributed onto the contributing weights $\boldsymbol{\theta}^k$.

Our experiments will demonstrate that this deep prior regularisation efficiently attenuates checkerboard artefacts. Moreover, by avoiding irrelevant local minima, it brings us closer to the desired solution after a reasonable number of iterations.

Our Inpainting Network Architecture

In Figure 5.1 we outline the full pipeline of our neural algorithm for elastica inpainting. As the energy is only considered within the inpainting domain, we remask the network output $\boldsymbol{u}$ with the known data $\boldsymbol{f}$. The final reconstruction is passed to the deep energy loss function which evaluates its quality.

Our architecture consists of a small gated U-net [185, 262] as introduced in Section 3.1.3. Ulyanov et al. [229] found that variants of U-nets perform well as deep priors for tasks like inpainting or artefact removal. Furthermore, gated U-nets were used successfully for free-form inpainting as in our setting [262], and we confirm their suitability as deep priors in our experiments.

The defining feature of U-nets is their shape: The left part consists of a sequence of convolutions and downsampling operations, resulting in features on different scales. In a similar way, the right side repeatedly convolves and upsamples the features, and it concatenates them with features of the same scale from the downsampling pass. The gated U-net enhances the architecture by jointly evolving features and masks in each gated convolutional layer; see [262] for a detailed explanation.

Figure 5.1: The full network architecture. The flow of gradients during backpropagation is depicted with dashed lines. Notice that $\nabla_{\boldsymbol{u}} E(\boldsymbol{u})$ is still computed as part of the backpropagation, which shows the close connection to gradient descent.

5.4.3 Experiments

In this section we first present an ablation study which demonstrates that both neural components of our approach are essential. Afterwards we evaluate its performance for inpainting of natural images and shape completion tasks.

Experimental Setup

We prefer standard U-nets for fast inpaintings of natural images, and gated ones for shape completion with highest quality. Depending on the size of the image, three or four scales with two convolutional layers each are used. We start with up to 28 channels at the finest resolution and double them with each downsampling. Following [262], dilated convolutions on the coarsest scale of the gated U-nets are included. The inpainting region of the masked image is initialised with random uniform noise in the same range $[0, 1]$ as the image.

The network is trained with the Adam optimiser [119]. We start with a learning rate of $\tau = 0.001$ for natural images and $\tau = 0.00005$ for shape completion and decrease it in later epochs. The parameter b is chosen as to minimise the MAE w.r.t. the ground truth. We use the MAE since it better reflects our visual impression than the MSE, while remaining mathematically more transparent than pure perceptual measures.

Ablation Study

The ablation study in Figure 5.2 shows that the deep energy alone is insufficient for obtaining the desired inpainting result, regardless of the number of iterations. Incorporating the deep image prior is essential. It helps to escape from bad local minima: From the third to the fourth image, the discrete energy drops from 2.22 to the remarkably low value of 0.08.

Inpainting of Natural Images

Figure 5.3 shows how our inpainting algorithm performs on natural images. After a few thousand iterations, it produces the desired result that minimises the MAE. Additional iterations still reduce the energy, but deteriorate the MAE and the visual impression; see Figure 5.4. The checkerboard-like artefacts in the steady state confirm the previously discussed limitations of our discrete energy. Thus, it is helpful to stop earlier and benefit from the regularising qualities of the deep image prior. Stopping earlier also allows us to obtain our results faster. With an Nvidia GTX 1080 GPU, it takes 49 s for the 6,000 iterations needed for the 256×256 image *trui*. Finding an automatic stopping criterion is part of our future research.

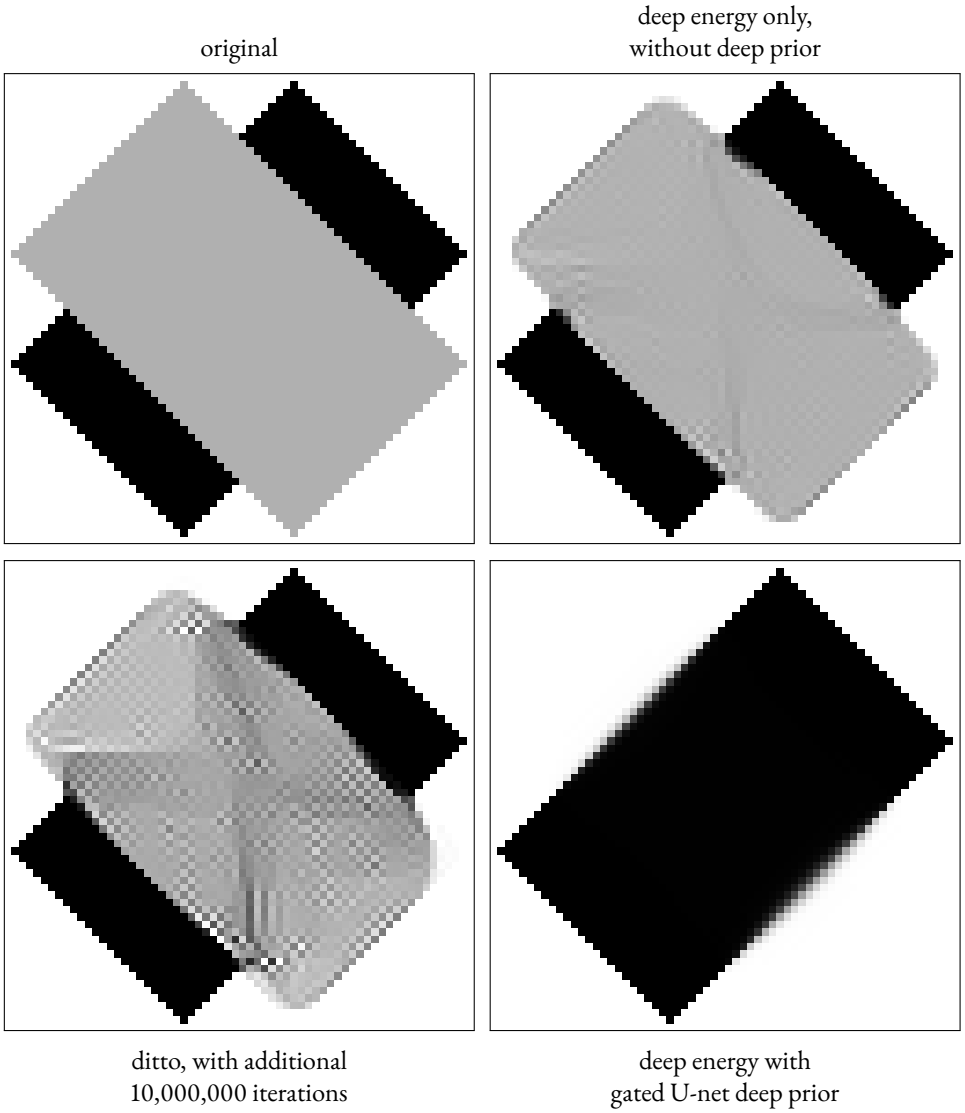


Figure 5.2: Comparison of results with and without deep prior ($b = 0.001, \varepsilon = 0.0001$, and 60,000 iterations). For simplicity, the inpainting region is initialised with the average gray value. The learning rate starts at $\tau = 0.00004$ and is halved every 20,000 iterations. The optimisation producing the third image ran for an additional 10,000,000 iterations at the final learning rate.

Figure 5.3: Inpainting results for *trui* and *peppers* ($b = 0.175$, $\varepsilon = 0.005$, $\tau = 0.001$). After 6000–7000 iterations, a good reconstruction with low mean absolute error (MAE) is obtained. Additional iterations produce checkerboard artefacts (digital zoom recommended).

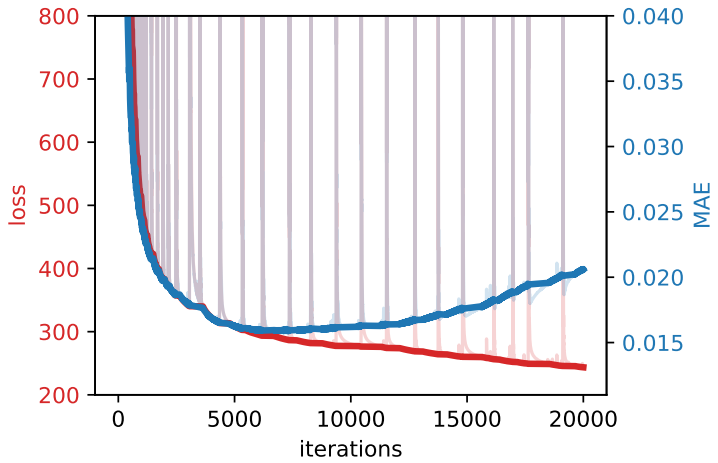
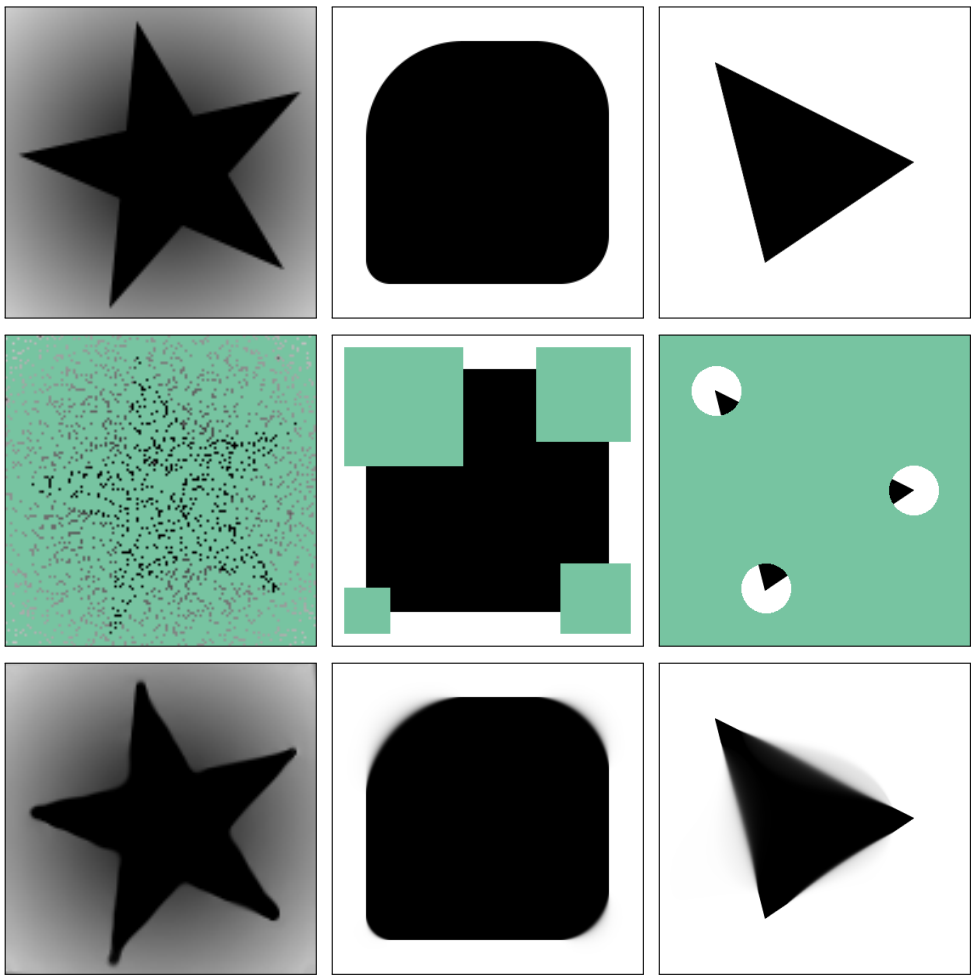


Figure 5.4: Energy and reconstruction error over time corresponding to *trui* in Figure 5.3. Spikes were filtered from the bold lines to make the general trends more recognisable. They are caused by brief, very large gradients generated by the curvature term which cause the network to erroneously change the global brightness, but are quickly recovered from.

Speed comparisons to other works remain difficult as they often do not disclose their runtimes, do not make use of the GPU, or cannot produce comparable quality. A rough comparison can be made to the augmented Lagrangian approach of Tai et al. [222]. For an 300×235 image and a fairly dense random mask with 60 % known pixels, their inpainting takes 78 s on an Intel Core 2 Duo P8600 @ 2.4 GHz CPU. Yashtini and Kang [261] propose an ADMM method and report 854 s for inpainting a 220×340 colour image with a random mask with 20 % density on a 2.5 GHz Intel Core i5 CPU. These numbers indicate that our algorithm offers a competitive speed.

Shape Completion

Shape completion allows us to demonstrate that our discretisation produces sharp and rotation invariant results. In Figure 5.5, we show that completion of straight edges, curves, and combinations thereof are handled adequately. Most noticeably, very large gaps with almost 200 pixels between the mask regions can be closed. In Figure 5.6 we compare to the state-of-the-art results obtained with the sophisticated lifting approach of Chambolle and Pock [34]. While our neural algorithm is simpler, it produces inpaintings of comparable visual quality.



128×128, $b = 0.02$, adapted from [222]

256×256, $b = 0.0005$

400×400, $b = 0.0001$, adapted from [198]

Figure 5.5: Neural inpainting results on shape completion tasks ($\varepsilon = 0.0001$). From top to bottom: Original, inpainting domain (in green), inpainting result.

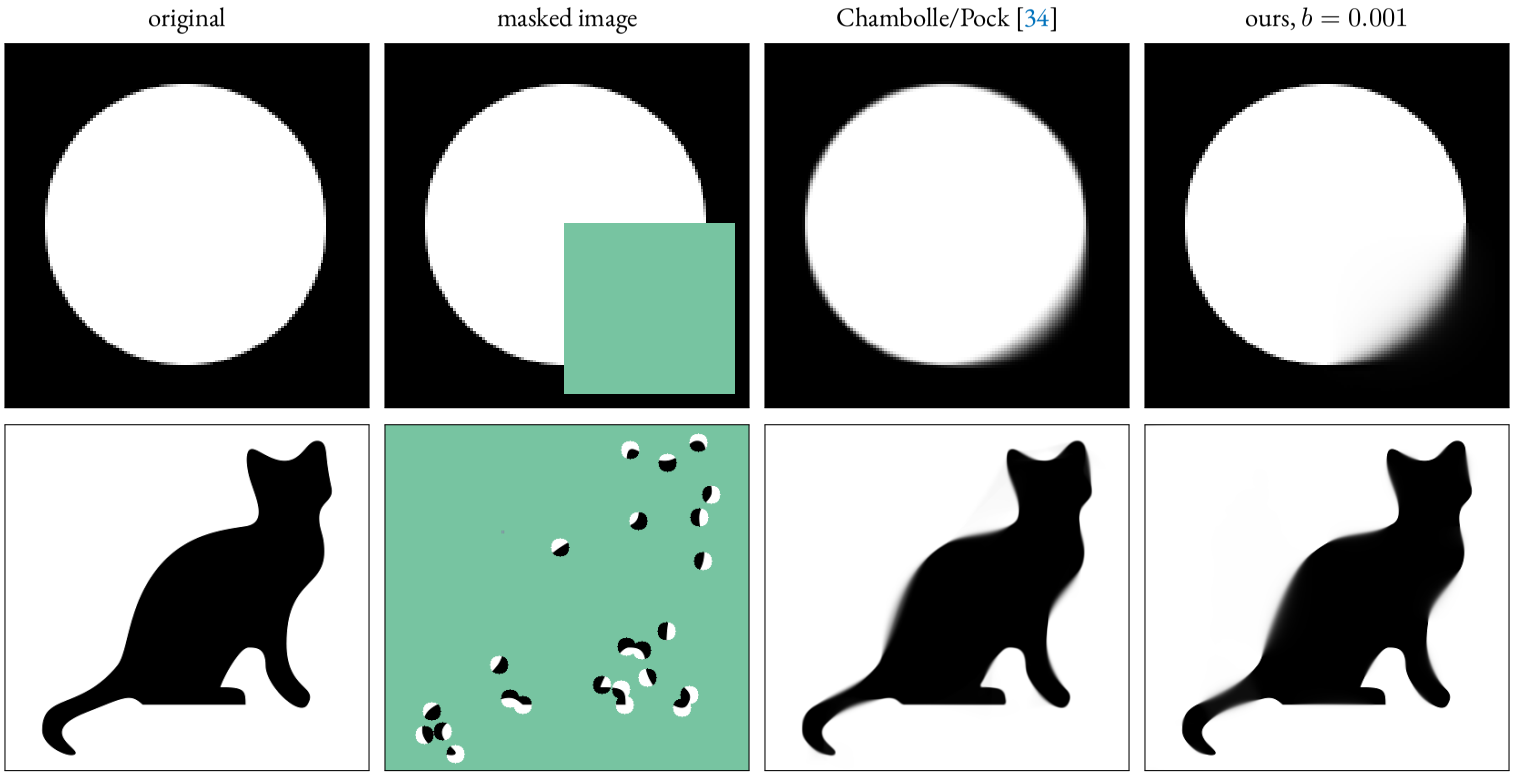


Figure 5.6: Comparison to the TSC shape completion results of Chambolle and Pock [34], kindly provided by the authors ($\varepsilon = 0.0001$). Following their experiment, we also initialised the inpainting region of the masked image with grey value 0.5.

5.4.4 Conclusions

We have proposed the first neural algorithm for Euler's elastica inpainting. Research on numerical methods for this attractive but difficult inpainting model has been pursued for more than two decades and produced highly sophisticated techniques. It is thus surprising that our very simple approach is qualitatively competitive to a leading elastica algorithm for shape completion. This speaks for the quality of its parts and the fruitful synergy between model-based concepts and deep neural networks. Our finite difference approximations for the first and second order derivatives of the elastica energy offer good rotation invariance and sharpness. Energy minimisation is accomplished automatically by a neural network that frees us from the burden of discretising the numerically challenging fourth order Euler–Lagrange PDE. Moreover, the regularising properties of a deep image prior avoid high-frequent artefacts that the discrete energy cannot penalise. In contrast to purely data-driven neural approaches, our hybrid algorithm requires neither ground truth inpaintings nor training data.

5.5 Phase-Retrieval-Based Physics-Informed Neural Networks for Acoustic Magnitude Field Reconstruction

Sound field estimation aims to estimate a spatial distribution of an acoustic field based on a discrete set of sensor observations. It is one of the fundamental techniques in acoustic signal processing and machine learning that can be applied to various downstream tasks, such as acoustic imaging [148], room acoustic analysis [163], and spatial audio reproduction [176].

Most methods for sound field estimation are targeted at the complex-valued amplitude distribution in the frequency domain or the real-valued amplitude distribution in the time domain, and assume that they can be observed at discrete positions of multiple sensors. One of the most widely used techniques is the basis-expansion-based method, which is based on the expansion representation of the acoustic field by plane wave functions, spherical wave functions, or equivalent sources [123, 163, 176]. The basis-expansion-based methods have been generalised as kernel-regression-based methods, which constrain the estimated function to satisfy the Helmholtz equation by properly-defined kernel functions [228]. Recently, neural methods have attracted attention due to their high flexibility and representational power [124, 137, 141, 161, 179]. Among those methods, physics-informed neural networks (PINNs) [114, 125] have shown promising results. They use an explicit constraint on physical properties, defined as the deviation of the predicted sound field from the governing PDEs, i.e., wave or Helmholtz equation [178]. This constraint allows the PINN to avoid overfitting by penalising functions that do not adhere to the governing PDE.

In contrast to many sound field estimation methods, the target of this study is the acoustic magnitude distribution. It is given by the absolute value of the complex-valued amplitude or pressure of the sound field in the frequency domain with its discrete measurements. The estimation techniques for acoustic magnitude distribution will be useful when the signals of each sensor are not synchronised. For example, in ad-hoc microphones, each sensor device usually operates independently [41], and when measuring the directivity of vibrating bodies, such as musical instruments, sequential recording of sound radiation is frequently used [4]. However, unlike the methods using the complex-valued amplitudes, it is difficult for the magnitude distribution estimation to incorporate the physical properties of the sound field: The governing PDE incorporates the phase, and cannot be computed without it. Therefore, the magnitude field estimation has typically relied on purely data-driven techniques [124].

We propose a magnitude field estimation method that incorporates phase retrieval and enables the use of a Helmholtz-equation-based loss even without access to phase

measurements. Specifically, our neural architecture jointly estimates magnitude and phase distributions so that the measured magnitudes are matched at sensor positions while the deviation from the Helmholtz equation is minimised. The effectiveness of our phase-retrieval-based PINN is evaluated through numerical experiments in simulated rooms.

5.5.1 Problem Statement and Prior Work

Acoustic Magnitude Field Estimation

We consider an arbitrary room with target region $\Omega \subset \mathbb{R}^3$. Inside the target region, the magnitude of the acoustic pressure $|u(\boldsymbol{x}, \omega)| : \Omega \times \mathbb{R} \to \mathbb{R}$ is known for a set of M measurement locations $\{\boldsymbol{x}_m^{(\mathrm{m})}\}_{m=1}^M \in \Omega$, and a fixed angular frequency ω. The goal is to infer the magnitude of the acoustic pressure at a set of N test locations $\{\boldsymbol{x}_n^{(\mathrm{t})}\}_{n=1}^N \in \Omega$ that are different from the measured ones. In the following, we will omit ω for notational simplicity. Note that this problem differs from the usual estimation of the complex-valued acoustic pressure in that only the magnitude is known, but not the phase. We offer further details on this problem from the perspective of image processing in Section 3.3.

Related Work

Most previous works consider the interpolation of the complex-valued pressure field instead of just its magnitude. The most widely used approach in the pressure interpolation is based on basis expansion, which relies on expanding the pressure distribution into plane waves, spherical wave functions, or point sources [42, 227]. This technique is generalised as kernel regression with a constraint of the governing PDE, which can be interpreted as an infinite-dimensional basis expansion [228].

Recently, many neural methods for sound field estimation and *head-related transfer function* (HRTF) interpolation, which is closely related to sound field estimation, have been proposed [100, 124, 137, 141, 161, 179, 194, 217, 268]. Several of these, particularly in HRTF interpolation, target the interpolation of magnitude distributions [100, 124, 137, 268]. PINN enables the combination of the high representational power of neural networks with physical prior knowledge [114, 125]. Based on an implicit neural representation or *neural field* (NF) [212], PINNs represents the spatial distribution using a neural network. It is then trained using a loss function that induces the output to satisfy the governing PDE, which is called PDE loss [178]. However, since the PDE loss is computed via automatic differentiation from the NF output [19], the NF output needs to be the complex-valued amplitude in the frequency domain. Otherwise, the governing PDE cannot be evaluated. Therefore,

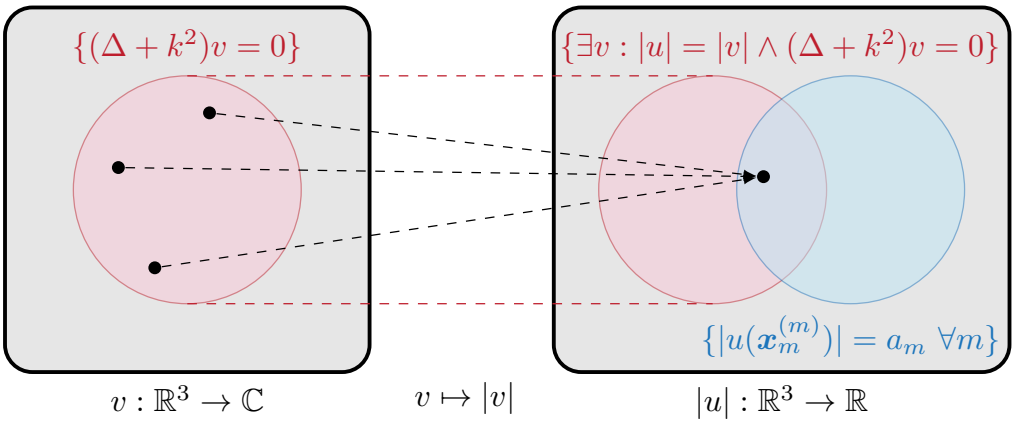


Figure 5.7: Relationship of pressure and magnitude fields. The PDE constraint on the pressure field (in red) implies a set of physically plausible magnitude distributions. The goal of our approach is to find a magnitude distribution which has an associated valid pressure distribution, and matches our observations (in blue).

largely only methods used for interpolating general real-valued functions have been applied to estimate magnitude distributions.

Still, even when estimating the magnitude distribution, the underlying complex-valued amplitude distribution should be constrained to the space spanned by functions satisfying the governing PDEs. This in turn implies a constraint on the space of valid magnitude distribution.

5.5.2 Phase-retrieval-based PINN for magnitude field estimation

We begin with a short introduction of PINNs at the example of a complex-valued pressure field estimation, before describing how we extend the approach to be able to apply it to magnitude field estimation.

PINN

PINNs [178] aim to solve *initial value problems* (IVP) by modelling the target function with a NF, and make use of automatic differentiation frameworks for the calculation of partial derivatives. The IVP that is considered for sound field estimation in the frequency domain is defined by the homogeneous Helmholtz equation

$$(\Delta + k^2)u(\boldsymbol{x}) = 0, \qquad u : \Omega \times \mathbb{R} \to \mathbb{C},\ \boldsymbol{x} \in \Omega, \tag{5.10}$$

where k is the wave number, together with the initial condition

$$u(\boldsymbol{x}_m^{(\mathrm{m})}) = s_m, \qquad m \in \{1, \dots, M\}\,. \tag{5.11}$$

Here, $\{s_m\}_{m=1}^M$ are the pressure measurements at the locations $\{\boldsymbol{x}_m\}_{m=1}^M$.

PINNs model the sound field $u(\boldsymbol{x})$ through a NF. Its input are spatial coordinates $\boldsymbol{x} \in \Omega$, and its output is the predicted sound field at that location. To ensure that the network adheres to the initial condition and Helmholtz equation, two loss functions are defined. The data loss $\mathcal{L}_{\text{data}}$ is used to penalise deviations from the initial condition. The PDE loss $\mathcal{L}_{\text{PDE}}$ penalises deviation from the governing PDE. To that end, locations $\{\boldsymbol{x}_p^{(\mathrm{p})}\}, p \in \{1, \dots, P\}$, are sampled from the spatial domain Ω. At those locations, the mean squared residual is considered:

$$\mathcal{L}_{\text{PDE}} = \frac{1}{P} \sum_{p=1}^{P} \left\| (\Delta + k^2) u(\boldsymbol{x}_p^{(\mathrm{p})}) \right\|_2^2 . \tag{5.12}$$

A weighting of both losses is used for training:

$$\mathcal{L} = \lambda_{\text{data}} \mathcal{L}_{\text{data}} + \lambda_{\text{PDE}} \mathcal{L}_{\text{PDE}}\,. \tag{5.13}$$

The weights $\lambda_{\text{data}}, \lambda_{\text{PDE}} > 0$ can be chosen to prioritise one loss over the other. A technique to adapt the weighting parameters is also proposed [256].

When training the network, the loss $\mathcal{L}$ is minimised through gradient descent. Partial derivatives inside the Helmholtz equation are calculated using automatic differentiation [19], which saves the user from having to discretise them. It is clear to see that when both losses are 0, the IVP is solved. While PINNs typically only reach this minimum approximately, they are still good approximations of the target function.

Notably, the setup described here cannot be directly applied to the magnitude field estimation. The Helmholtz equation (5.10) requires the complex-valued pressure field, which is not available in our setting. Additionally, no equivalent statement operating on the magnitude fields $|u|$ is known.

Motivation of Our Approach

The key insight of our approach is that there has to be some underlying sound field u which fulfils two conditions:

1. At the sensor locations $\boldsymbol{x}_m^{(\mathrm{m})}$, its magnitude should match the observed one $|s_m| := a_m$ for $m \in \{1, \dots, M\}$.

2. The sound field u adheres to the Helmholtz equation everywhere in the domain.

This sound field cannot be uniquely determined, as the phase component can be periodically shifted. Additionally, for a low number of sensors M, there can be further degrees of freedom. Accordingly, a sound field u fulfilling these conditions can still differ significantly from ground truth. We visualise this relationship in Fig. 5.7. However, as our experiments in Sec. 5.4.3 will show, using the Helmholtz equation as a regulariser in this way still improves reconstruction quality.

For our model, we encode the two conditions mathematically:

1. **Data Condition:**

$$|u(\boldsymbol{x}_m^{(\mathrm{m})})| = a_m \qquad m \in \{1, \dots, M\} \tag{5.14}$$

2. **Physics Condition:**

$$(\Delta + k^2)u(\boldsymbol{x}) = 0 \qquad \forall \boldsymbol{x} \in \Omega \tag{5.15}$$

In the following, we demonstrate how to train a PINN to find a sound field u fulfilling these conditions.

Proposed Architecture

Similar to the PINN approach for sound field estimation, we predict a sound field u, and encode our two conditions through loss functions. For the data condition, instead of using the mean square error, we selected the mean average error of the logarithmically scaled magnitude as a perceptually-motivated data loss [257]

$$\mathcal{L}_{\text{data}} = \frac{1}{M} \sum_{m=1}^{M} \left| 20 \log_{10} \left(\frac{a_m}{|u(\boldsymbol{x}_m^{(\mathrm{m})})|} \right) \right|, \tag{5.16}$$

which is equivalent to the *log-spectral distance* for a single frequency. For the physics condition, we penalise the squared residual (5.12).

Our final network architecture is illustrated in Fig. 5.8. At its core, it consists of two NFs with multilayer perceptrons, one for magnitude estimation and one for phase prediction. As input, both receive a *random Fourier feature* (RFF) embedding [223], which maps the sensor position into high-dimensional features using sinusoids with the randomly sampled frequencies $\boldsymbol{B}$. This greatly improves the network's ability to handle higher-frequency variations in the output.

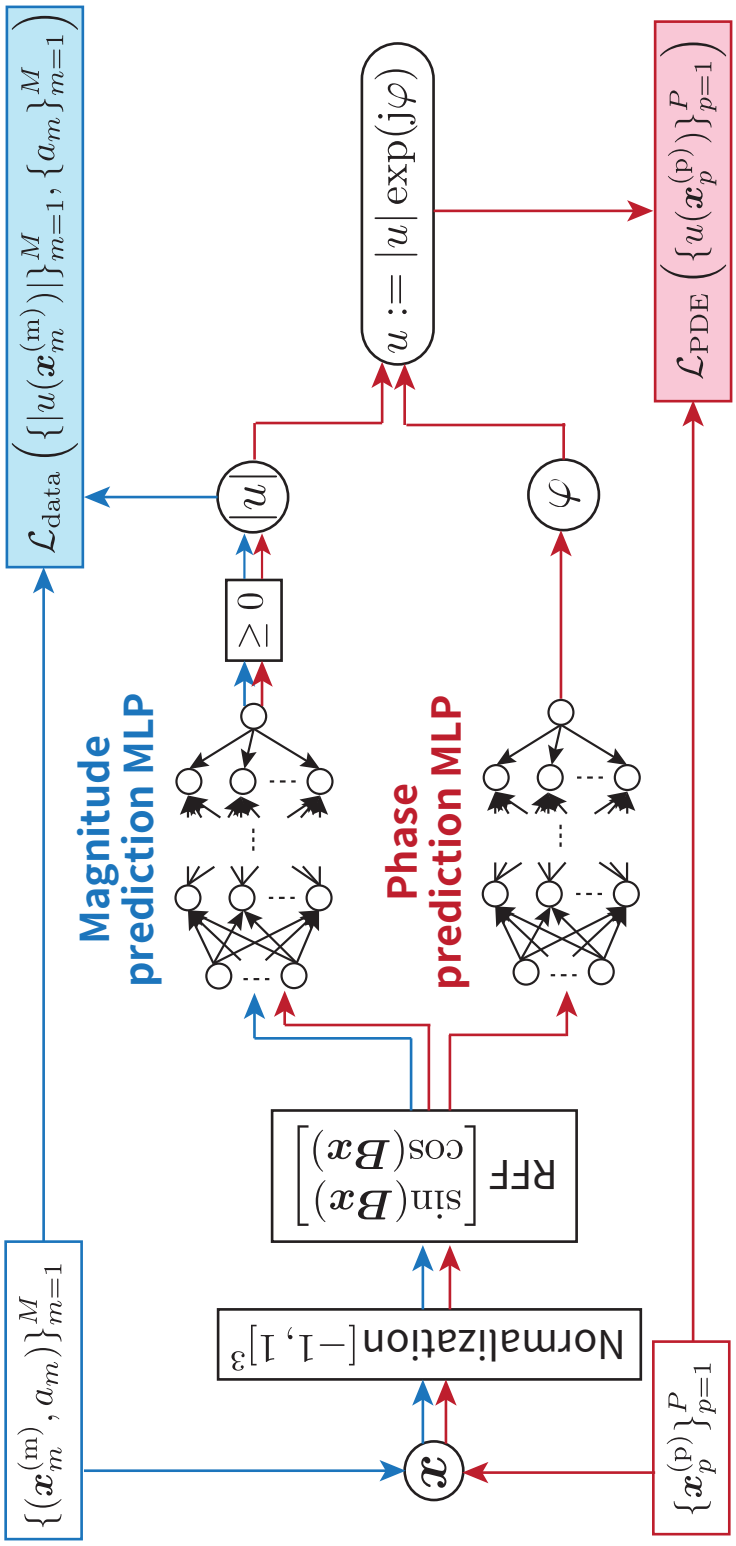


Figure 5.8: Visualisation of our network architecture. Blue arrows highlight the data flow of the magnitude dataset, and red arrows of the reconstructed complex-valued pressure dataset.

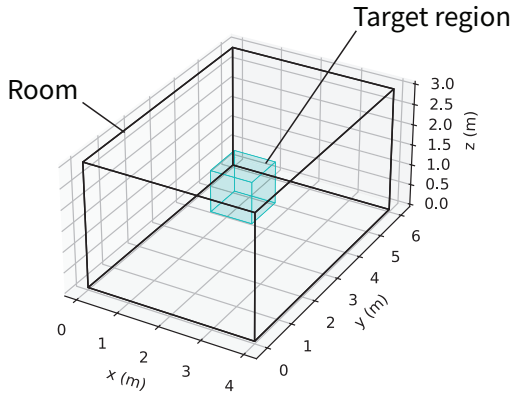


Figure 5.9: Experimental setup. Geometry of the room and target region Ω.

5.5.3 Experiments

Experimental Setup

Dataset We use a synthetic dataset generated with the image source method [8] using `pyroomacoustics` [197]. Our environment is a room of size $3\,\mathrm{m} \times 4\,\mathrm{m} \times 6\,\mathrm{m}$ with a reverberation time T_{60} of $200\,\mathrm{ms}$ as shown in Fig. 5.9. Inside, a cube lattice of 33^3 positions is placed in a unit cube ($1.0\,\mathrm{m}^3$) at the origin as the target region Ω. Of those, 5, 10, 20, or 50 measurement locations are randomly selected for training, and the rest are used for testing. 64 sources are placed randomly in the room outside Ω. Of those, half were used for hyperparameter optimisation, and half were used for the results presented here. Our experiments are performed for 200 Hz, 400 Hz, and 600 Hz.

Network Architecture For all experiments, we use MLPs with 4 hidden layers and 256 neurons per layer, with $\tanh$ activation. The RFF matrix $\boldsymbol{B} \in \mathbb{R}^{128\times 3}$ is sampled from a Gaussian distribution with unit variance.

Training We train each instance for 5×10^5 iterations with the AdamW [138] optimiser with an initial learning rate of 10^{-3}. Every 10^4 iterations, we decrease the learning rate by 10%. We employ a data loss weight $\lambda_{\text{data}} = 10^{-1}$ and a PDE loss weight of $\lambda_{\text{PDE}} = 10^{-3}$. Those weights have been optimised to achieve the lowest possible data loss at the expense of a slightly increased physics loss.

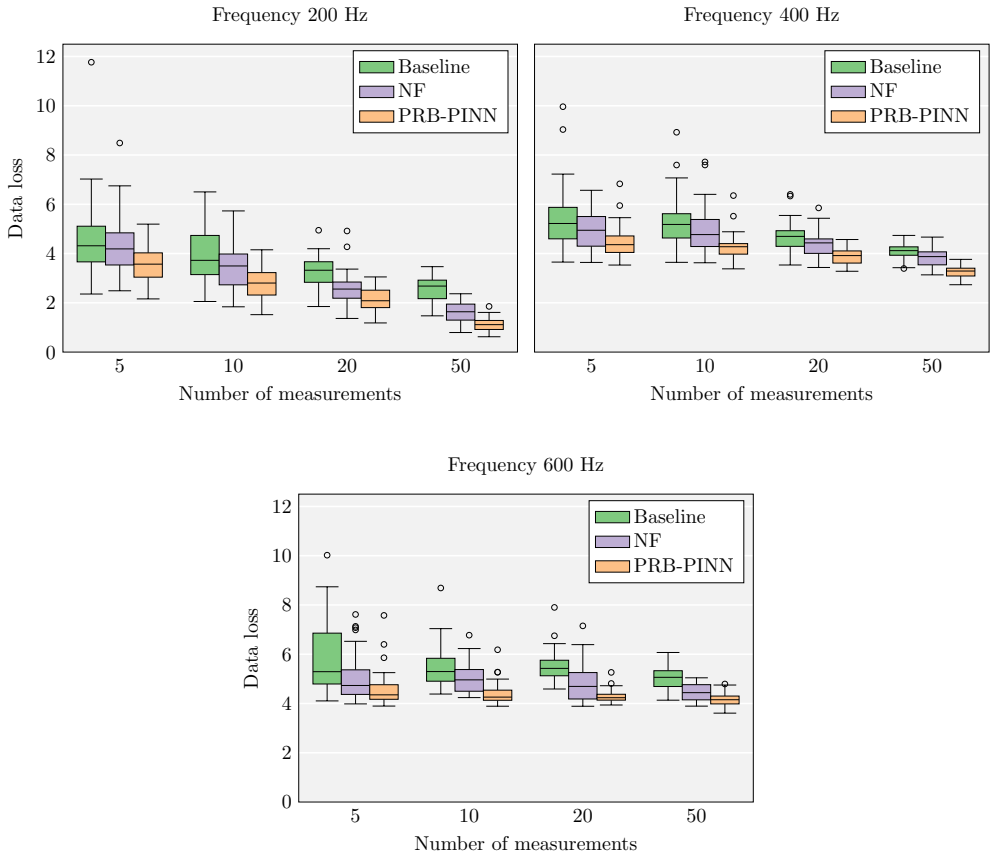


Figure 5.10: Plot of the test set data loss (5.16) for different frequencies and numbers of sources in the training set. The use of the physics loss is beneficial in all cases.

Results

We compare three different approaches. Our *Baseline* is simple nearest neighbour interpolation [155]. We compare against interpolation with a NF without physics loss (*NF*), which is trained solely with the data loss (5.16). This approach is only implicitly regularised through the biases inherent to the network architecture [229]. Our proposed approach is interpolation by the phase-retrieval-based PINN (PRB-PINN).

Reconstruction Quality We compare the reconstruction quality of the different methods in Fig. 5.10. Across all frequencies and numbers of sensors, there is a consistent ordering of methods: Baseline performs worst, followed by NF, and our PRB-PINN performs best.

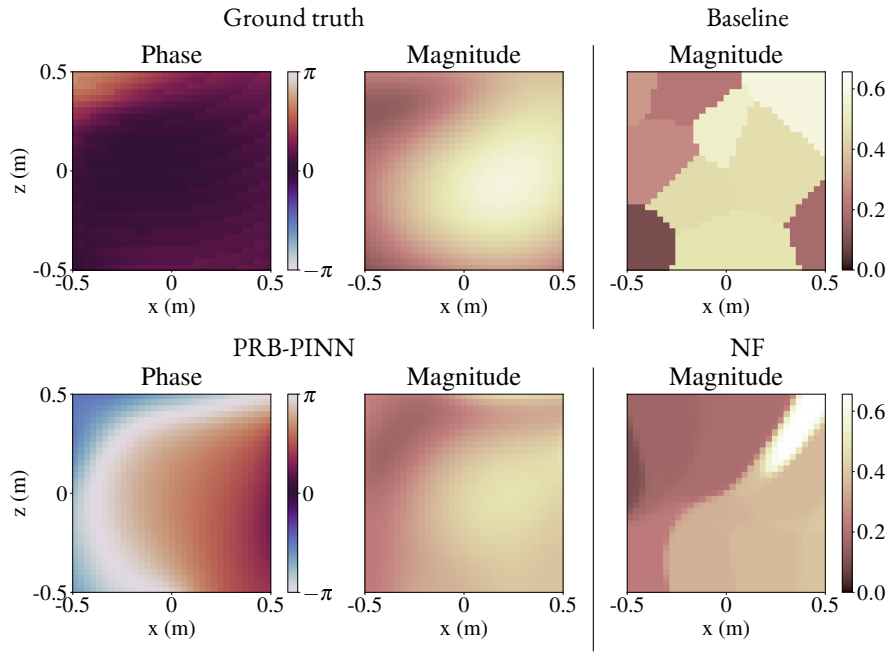
As expected, reconstruction quality increases with the number of measurements. Furthermore, the interpolation problem becomes harder with increasing frequency as the sound field varies more rapidly in space. This leads to a lower reconstruction quality for all considered methods.

Visual Evaluation In Fig. 5.11, we visualise the magnitude distributions in the x-z-plane of our target domain for different frequencies. Notably, we can see that the phase produced by the NN does not match the ground truth phase. However, it still leads to spatial variation with the correct frequency in the magnitude distribution, significantly increasing reconstruction quality. Especially at a low frequency of 200 Hz, due to the low amount of variation in space, this reconstruction then matches the ground truth well. For higher frequencies, the space of physically plausible phases increases as well. As such, there can be a greater mismatch between reconstruction and ground truth.

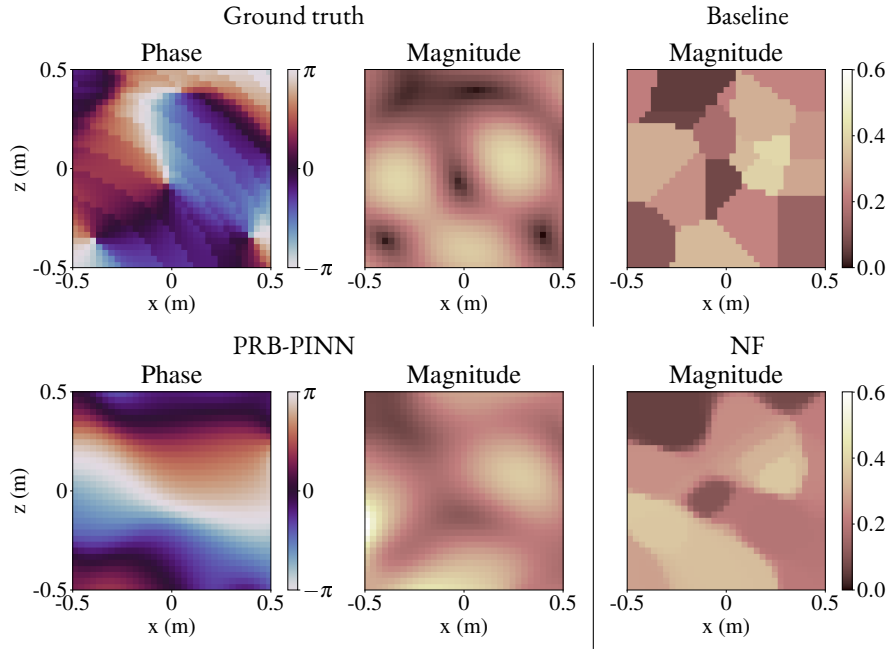
PDE Loss Fig. 5.12 explores the influence of the PDE loss weight λ_{PDE}. Especially for higher numbers of measurement locations M, increasing the weight too far can lead to a higher test loss, even though the physics loss still decreases. We believe that this is due to the data not perfectly adhering to the Helmholtz equation itself.

5.5.4 Conclusions

We proposed a phase-retrieval-based PINN for magnitude field estimation in the frequency domain. Due to the inaccessible phase data, current methods for the magnitude field estimation rely only on the measured magnitude data and do not employ the governing PDE that the original complex-valued function of the estimation target should satisfy. Our network architecture jointly estimates magnitude and phase



(a) 200 Hz, $m = 20$ measurements.



(b) 400 Hz, $m = 50$ measurements.

Figure 5.11: Visualization of the magnitude distribution in the x-z plane at 200 Hz and 400 Hz.

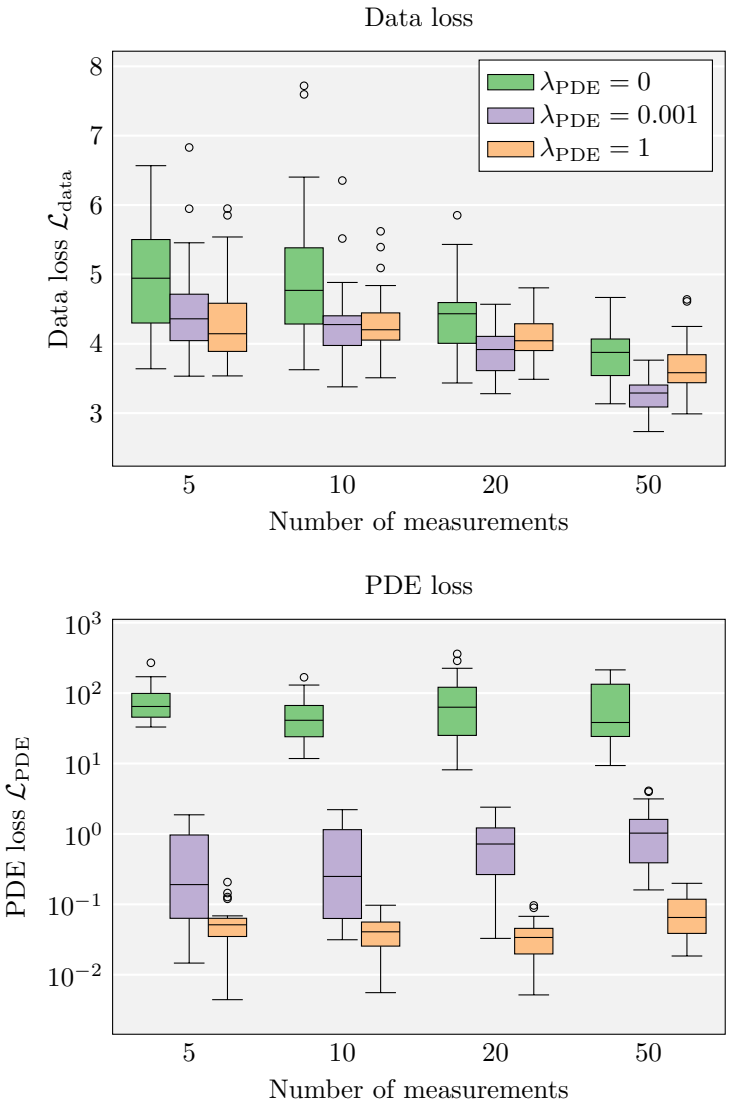


Figure 5.12: Visualisation of the impact of the PDE loss weight λ_{PDE}. A PDE loss either lower or higher than the chosen one results in worse reconstruction quality.

distributions using neural fields, and penalises the deviation of the reconstructed complex-valued function from the Helmholtz equation. We demonstrated that our phase-retrieval PDE loss is effective by comparing the proposed method with the NF without the PDE loss in the experiments.

5.6 Chapter Conclusions

The numerical problems in this chapter presented multiple challenges for purely model-based algorithms: For elastica inpainting, they are difficult to design, too slow, or too complex to be analysed. For magnitude field interpolation, the challenge arose from the complexity of multiple interlinked objectives.

However, this chapter has shown that just because a problem might be hard for one class of optimisation strategies, it might not be for another. In cases where purely model-driven solvers do not deliver satisfactory results, trying neural ideas is viable.

They offer advantages which compliment the strengths of mathematical models. First of all, automatic differentiation can take care of derivative calculation is situations where symbolic approaches become incomprehensible, as is the case for our discretised Euler's elastica energy. This allows the user to specify the energy, and let deep learning frameworks take care of the rest.

A similar synergy is achieved by PINNs. On the mathematical side, the user is free to specify a range of different goals from boundary conditions to PDEs. All those goals can then be transparently used as loss functions for the neural field, which attempts to satisfy them through gradient descent.

Within the class of neuroexplict models, this chapter is still largely on the model-driven side. While we now train neural networks in the sense that we are using gradient descent to optimise weights, this training happens per sample instead of on a dataset. This highlights that neural networks are capable of more than just extracting features from data. Their built-in regularisation capabilities and representational power, combined with minimisation through gradient descent, allow for effective solvers for energies and PDEs. This chapter contributes to further exploration in this area.

In the next and last chapter of this thesis, we finally use neural networks for their typical purpose of extracting patterns from data. However, we still ensure interpretability by embedding the networks into a mathematical framework.

6 Neural Networks for Mask Optimisation

For the final chapter of this thesis, we investigate the problem of mask optimisation for homogeneous diffusion with the help of neural networks. It contains materials from a journal paper published in PAA 2023 [174] and a conference paper in the proceedings of SSVM 2023 [201].

The journal paper [174] was written by Dr. Pascal Peter as first author. I extended the neural network code that was used by Dr. Tobias Alt-Veit in his conference publication [9] and performed the neural experiments. The modified U-net backbone we use is based on an implementation by Dr. Pascal Peter for [168]. In addition, I performed part of the experiments for the model-driven baselines. Dr. Tobias Alt-Veit and Prof. Joachim Weickert provided valuable discussion and guidance throughout the preparation of the manuscript.

For the follow-up paper [201], the key ideas and experiments involving neural networks are my own. Dr. Pascal Peter was involved in scientific discussions and helped refine the manuscript. Niklas Kämper implemented the CUDA versions of the model-driven baselines. Prof. Joachim Weickert provided valuable discussion and guidance throughout the preparation of the manuscript.

6.1 Introduction

The classical inpainting problem [22, 61, 83, 147] deals with input images that have been partially corrupted and aims at reconstructing these missing areas. However, inpainting can be also useful when the whole image is known. For inpainting-based image compression [2, 17, 28, 32, 52, 71, 73, 107, 129, 171, 172, 198, 255, 269], the encoder stores only a small percentage of known data from which the decoder restores the discarded remainder of the image with inpainting. Some approaches [2, 17, 32, 52, 73, 269] such as the pioneering work of Carlsson [32] limit the choice of known data to edge locations. Following the diffusion-based codec of Galić et al. [71, 72], many later approaches [94, 171, 172, 198] rely on careful optimisation of the placement of known data in the image domain without the restriction to semantic image features. Inpainting with PDEs [32] has been able to outperform state-of-the-

art codecs: Already simple homogeneous diffusion [98] can compress depth-maps or flow fields better than HEVC [219] with suitably selected known data [94, 104, 105]. The problem of choosing the right positions of mask pixels, the so-called inpainting mask, is also vital for other applications such as denoising [5] or adaptive sampling [46]. In addition to this *spatial optimisation*, compression also benefits from *tonal optimisation*: The values of the known pixels can be adjusted to optimise the reconstruction quality as well.

However, even for a simple inpainting operator, spatial and tonal optimisation constitute challenging problems. This sparked a plethora of non-neural approaches [20, 26, 38, 40, 47, 50, 90, 91, 92, 94, 105, 115, 143, 145, 156, 159, 169, 171, 172, 198]. We systematically review those in Section 6.3. Among these methods, most require many inpaintings per iteration, which tend to be computationally expensive or rely on sophisticated implementations for acceleration. For instance, probabilistic methods for spatial optimisation [90, 143] yield high quality masks, but come with a high computational cost. Theoretical optimality results are rare, but have been derived from shape optimisation [20] for homogeneous diffusion inpainting. This allows near instantaneous spatial optimisation without the need for a single inpainting. However, so far, existing discrete implementations of this concept do not realise the full potential of the theoretical results from the continuous setting.

With our deep data optimisation for homogeneous diffusion inpainting, we aim for the best of both worlds. We train neural networks that can optimise both mask positions and values without the need for a single inpainting. These progress from simple models for small images and fixed mask densities to flexible strategies capable of 4K resolution and arbitrary densities. During training, we leverage new hybrid concepts that combine model-based inpainting with deep learning.

6.2 Chapter Contributions

Our contributions are separated into two stages: We begin by proposing the first deep learning framework for inpainting with homogeneous diffusion in 6.4. It is the first neural network approach that allows tonal optimisation in addition to the selection of spatial positions. In order to integrate model-based inpainting relying on PDEs into a deep learning pipeline, we propose the *surrogate solver*, which approximates diffusion-based inpainting with a neural network. This approach allows for straightforward backpropagation of gradients and a simple implementation. Integrated into suitable scaffolding, it enables us to train spatial optimisation networks that generate inpainting masks and tonal networks that output optimised known pixel values for a given mask.

The resulting mask network marries the quality of probabilistic approaches [143] with the computational efficiency of instantaneous spatial optimisation [20]. Similarly, our neural tonal optimisation consistently offers real-time performance at good quality. Compared to model-based approaches, its speed does not depend on the amount of known data in the inpainting mask. In addition, our networks do not require any parameter tuning after training, making them attractive for practical applications.

Our second contribution in Chapter 6.5 further extend the spatial optimisation capabilities to high-resolution images and arbitrary mask densities without retraining. We achieve this through first coarse-to-fine approach for neural mask generation. To this end, we partition the image into patches, and generate masks for each. Mask pixel budgets are assigned to each patch using an optimality result of Belhachmi et al. [20]. This constitutes the first transfer of their findings to the discrete setting which does not involve dithering, a step that usually leads to substantial quality losses. Our approach matches the quality of widely used stochastic mask optimisation strategies on 4K images of size 3840×2160, while being up to four orders of magnitude faster.

In addition, we improve the performance on images of all sizes by replacing the surrogate solver with a model-based inpainting directly inside the neural network. This yields an overall more transparent and reliable architecture. Additionally, it greatly reduces the number of trainable weights and speeds up the training.

6.3 Related Work

In the following, we discuss prior work for spatial and tonal optimisation, as well as related deep learning approaches.

6.3.1 Spatial Optimisation

Finding good positions for sparse known pixels constitutes a challenging optimisation problem that has sparked significant research activities. In the following, we mostly focus on approaches for diffusion-based inpainting, but there are also more broadly related works, for instance the free knot problem for spline interpolation. For instance, Schütze and Schwetlick [204] have proposed a data selection algorithm for the 2-D setting which can also be applied to images. Model-based methods for diffusion inpainting can be organised in four categories.

1. *Analytic Approaches.* From the theory of shape optimisation, Belhachmi et al. [20] derived optimality statements in the continuous setting. In practice, these can be approximated by dithering the Laplacian magnitude of the input

image. This approach does not require inpainting to find the mask pixels and is therefore very fast. However, the dithering yields only an imperfect approximation with limited quality [90, 143].

2. *Nonsmooth Optimisation Strategies.* Combining concepts from optimal control with elaborate strategies such as primal-dual solvers, multiple works [26, 38, 91, 156, 159] leverage nonsmooth optimisation for the selection of mask positions. These produce results of high quality, but are difficult to adapt to different inpainting operators. Moreover, they do not allow to specify the target amount of mask points a priori. For applications in compression, the non-binary masks need to be binarised, which reduces quality and requires tonal optimisation [92] for good results.

3. *Sparsification Methods.* Mainberger et al. [143] have proposed *probabilistic sparsification (PS)* to tackle the combinatorial complexity of spatial optimisation. They start with a full mask and iteratively remove candidate pixels. Among those candidates the algorithm discards a fraction of pixels with the smallest inpainting error permanently, while returning the remainder to the mask. This process is repeated until the desired percentage of mask points, the target density, is achieved. Besides good quality, this approach can easily be adapted to many different inpainting operators, including inpainting with PDEs [90, 143] or interpolation on triangulations [50]. This flexibility and quality comes at the cost of many inpainting operations. Nonetheless, sparsification is popular and widely used due to its advantages and its simplicity.

4. *Densification Approaches.* For applications such as compression or denoising, low densities are required. In such cases it can make sense to start with an empty mask and fill it successively instead of using sparsification. Such strategies [40, 47, 110, 115] share the benefits of simplicity, good quality, and broad applicability with sparsification. They have already been successfully used for diffusion-based [40, 47, 110] and exemplar-based [115] inpainting operators. For compression, densification also has been applied to data structures such as subdivision trees instead of individual pixels [53, 71, 171, 198]. A heavily parallelised domain decomposition approach relying on Delaunay triangulation by Kämper et al. [109, 110] offers state-of-the art performance. In addition, Jost et al. [103] have extended densification to general linear features besides image pixels. Their method also allows to store patch averages or derivative features among others.

5. *Relocation Methods.* Greedy optimisation strategies can get stuck in local minima. To escape from them, *nonlocal pixel exchange* [143] tests if moving some

randomly selected mask pixels into the unknown regions leads to a better reconstruction. Successful moves are kept, while unsuccessful ones are reverted. Given sufficient time, this method can produce very good masks. Related strategies have also been used for interpolation on triangulations [145]. They are usually applied as a postprocessing tool.

6. *Neural Methods.* Works such as [46, 168] learn the inpainting operator along with a mask generation network. As this results in an opaque model, we focus on well-understood homogeneous diffusion inpainting. For it, Alt et al. [9] have has presented first results which we build upon.

Note that the approach from Category 1 is the only one to require no inpaintings or complex solvers. Unfortunately, this near instantaneous spatial optimisation yields notably worse results in terms of quality than the methods from Categories 2–4. While Category 5 can improve any mask further, it is slow for large images. The deep learning framework we proposed here aims to achieve the best of both worlds: Fast spatial optimisation without the need for any inpaintings while producing results of a quality comparable to Categories 2–5.

6.3.2 Tonal Optimisation

So far, we have discussed methods that focus on finding optimal positions at which the original image data is kept. However, in a data optimisation scenario, we are not confined to selecting the location, but can also alter the value of mask pixels. This tonal optimisation introduces errors at mask pixels if those lead to a more accurate reconstruction in larger missing areas. Also for tonal optimisation, one can distinguish several categories:

1. *Least Squares Approaches.* For spatially fixed mask pixels, tonal optimisation leads to a least squares problem. The resulting linear system of equations is given by the normal equations. It has as many unknowns as mask pixels. The system matrix is a quadratic, dense matrix that is symmetric and positive definite [143].

   To solve it numerically, various algorithms can be applied. Direct methods include Cholesky, LU, and QR factorisations, while conjugate gradients and the LSQR algorithm constitute suitable iterative approaches [25]. Other iterative methods that have been used for tonal optimisation are the L-BFGS algorithm [38] and a gradient descent with cyclically varying step sizes [90]. All of these approaches suffer from the fact that they require to store the full matrix, which can become prohibitive for masks with too many pixels.

A potential remedy of this memory restriction consists of subsequently computing a so-called inpainting echo in a mask pixel [143]. It describes the influence of the mask pixel on the final inpainting result and can be used to adjust the grey or colour value accordingly. Doing this in random order for all mask pixels can be interpreted as a randomised Gauss–Seidel or SOR iteration step. If one does not store all inpainting echoes but computes them again in each iteration step, one achieves low memory requirements at the expense of a long runtime.

Discrete Green's functions offer another way to decompose the inpainting problem into pixel-wise contributions [95]. From this dictionary, the inpainting result can be assembled with simple linear superposition. Hoffmann [93] have used this property to derive an alternative least squares formulation for tonal optimisation which can be solved efficiently with a Cholesky solver. Even though its solution is equivalent to the direct least squares approach, it benefits from speed-ups for low amounts of mask pixels which are represented by only a few entries from the Green's function dictionary.

A recent alternative goes back to Chizhov and Weickert [40] and was further extended by Kämper et al. [109, 110]. It uses nested solvers and it is both efficient w.r.t. memory and runtime.

2. *Nonsmooth Optimisation Methods.* Hoeltgen and Weickert [92] have shown that thresholded non-binary spatial mask optimisation [26, 38, 90, 156, 159] is equivalent to a combined selection of binary masks and a tonal optimisation. Thus, the previously discussed nonsmooth strategies also indirectly perform tonal optimisation. However, this is inherently coupled to a spatial optimisation with the advantages and drawbacks described in the previous section.

3. *Localisation Approaches.* Since the influence of a single mask pixel mainly affects its local neighbourhood, tonal optimisation can be sped up by localisation. Strategies exist for localised operators such as Shepard interpolation with truncated Gaussians [169], 1-D linear interpolation [170], or smoothed particle hydrodynamics [47]. Other approaches limit the influence artificially by subdivision trees [172] or segmentation [94, 105].

4. *Quantisation-based Strategies.* All compression codecs rely on quantisation, the coarse discretisation of the colour domain. It can be beneficial to directly take quantisation into account during tonal optimisation instead of applying it in postprocessing. Thereby, one replaces the continuous optimisation problem by a discrete one. To this end, Schmaltz et al. [198] proposed a simple

strategy that visits pixels in random order and changes their values if increasing or decreasing the quantisation level yields a better results. Peter et al. [171] instead augment the Gauss-Seidel strategy with echoes [143] with a projection to the quantised grey levels. For interpolation on triangulations, Marwood et al. [145] use a stochastic approach that randomly assigns different quantisation levels in combination with spatial optimisation.

In addition to tonal optimisation itself, there are additional related strategies. Galić et al. [71] proposed an early predecessor that modified tonal values to avoid singularities in PDE-based inpainting. To avoid visually unpleasant singularities at mask pixels, Schmaltz et al. [198] use interpolation swapping: After the initial inpainting, they remove disks around the known data and use the more reliable reconstruction for a second inpainting.

The tonal category 1 is restricted to linear diffusion operators, including homogeneous diffusion. Category 2 marks the indirect tonal optimisation performed by nonsmooth spatial methods and categories 3 and 4 are mainly relevant for practical applications in compression. We aim at providing a neural network alternative to Category 1 methods for homogeneous diffusion inpainting. As for spatial inpainting, our goal is to propose a deep optimisation approach that offers high speed at good quality.

6.3.3 Relations to Deep Learning Approaches

To our best knowledge, deep learning approaches for sparse data optimisation are still very rare and so far, only spatial optimisation has been covered at all. Dai et al. [46] have proposed a deep learning method for adaptive sampling that trains an inpainting and an optimisation network separately. Joint training for spatial optimisation and inpainting with Wasserstein GANs was introduced by Peter [168]. Both approaches differ significantly from the current one, since they aim at learning both a spatial optimisation CNN and the inpainting operator. In contrast, we optimise known data for model-based diffusion inpainting with a surrogate solver for homogeneous diffusion inpainting. Moreover, our deep data selection is the first to consider both spatial and tonal optimisation.

In addition, a plethora of deep inpainting methods exist (e.g. [136, 165, 235, 239, 258, 259, 263]). A full review is beyond the scope of work, because these approaches do not consider any form of data optimisation. Since the selection of known data is decisive for the quality of inpainting-based compression, the current lack of research in this direction is the primary reason why deep inpainting has not played a role in this area, yet.

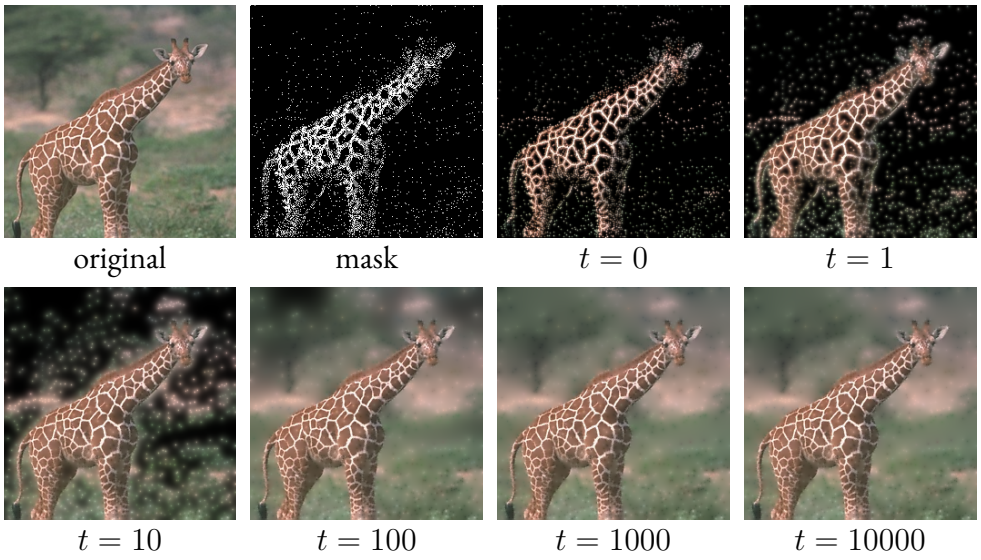


Figure 6.1: **Image Evolution of Homogeneous Diffusion Inpainting.** This figure shows a reconstruction of image 130014 from the BSDS500 database [15] cropped to size 256×256. White mask pixels indicate a total of 10% known data. At time $t = 0$, we assign the original pixel values to known areas and initialise the unknown regions with zero (black). For $t \to \infty$, diffusion propagates the known values and yields the inpainted image as the steady state.

6.4 Deep Spatial and Tonal Optimisation for Homogeneous Diffusion Inpainting

Diffusion-based Inpainting and Data Optimisation

Consider a grey value image $f : \Omega \to \mathbb{R}$ that is only known on the inpainting mask, a subset $K \subset \Omega$ of the rectangular image domain $\Omega \subset \mathbb{R}^2$. Diffusion-based inpainting [32, 249] reconstructs the missing areas $\Omega \setminus K$ by propagating the information of the fixed known pixels from K over the diffusion time t. The inpainted image is the steady state $t \to \infty$ of this evolution. Fig. 6.1 illustrates such a propagation over time. For our inpainting purposes, we are only interested in the steady state and not the intermediate steps of the evolution.

There are sophisticated anisotropic diffusion approaches [71, 106, 172, 198, 249] that adapt the amount of propagation in different directions to the image structure and can achieve results of very good quality even if the dataset K is not highly optimised. However, in the following, we consider simple homogeneous diffusion [98]

for inpainting. It is parameter-free and can achieve surprisingly high quality for a well-optimised dataset. In this case, the inpainted image u fulfils the inpainting equation

$$(1-c)\Delta u - c(u-f) = 0\,, \tag{6.1}$$

which arises as the steady state if one inpaints with the homogeneous diffusion equation $\partial_t u = \Delta u$. Here, $\Delta u = \partial_{xx} u + \partial_{yy} u$ denotes the Laplacian and c is a binary confidence function with $c(\boldsymbol{x}) = 1$ for known data in K and $c(\boldsymbol{x}) = 0$ otherwise. At the image boundaries $\partial\Omega$ we impose reflecting boundary conditions. Note that it is also possible to use non-binary confidence values [92], which we will do in Section 6.4.1. Since homogeneous diffusion is a linear operator, colour inpainting is implemented by channel-wise processing.

In practice, we implement this method on a discrete input image $\boldsymbol{f} \in \mathbb{R}^{n_x n_y}$ with resolution $n_x \times n_y$. Discretising Eq. 6.1 with finite differences leads to a linear system of equations. Then, reconstructing the image $\boldsymbol{u} \in \mathbb{R}^{n_x n_y}$ is achieved with a suitable numerical solver.

The discrete problem of mask optimisation for homogeneous diffusion inpainting consists in finding the binary mask $\boldsymbol{c} \in \{0,1\}^{n_x n_y}$ with a user-specified target density d such that $\|\boldsymbol{c}\|_1/(n_x n_y) = d$ where $\|\cdot\|_1$ denotes the 1-norm. This density can be seen as a budget that specifies the percentage of image pixels that should be contained in the final mask.

For comparisons, we consider the *analytic approach* of Belhachmi et al. [20]. It is based on results from the theory of shape optimisation that demonstrate that mask pixels should be placed at locations of large Laplace magnitude. In the discrete setting, they use a Floyd-Steinberg dithering [68] of the Laplace magnitude. This leads to an imperfect, but very fast approximation of the theoretical optimum. This algorithm is a representative for simple approaches that do not require any inpaintings to determine the optimised mask.

As a prototype for better performing mask optimisation algorithms, we consider the widely used *probabilistic sparsification* of Mainberger et al. [143]. It yields better results than the analytic approach by taking the discrete nature into account directly and greedily removing pixels that are not important for the reconstruction. It starts with a full inpainting mask. In each iteration, it removes a fraction p of candidate pixels from the mask. After an inpainting with the new mask, it analyses the local inpainting error: Candidate pixels which have a high local inpainting error are hard to reconstruct and should thus not be removed. Therefore, the algorithm adds back the fraction q of candidates with the largest errors. The iterations are repeated until the target density d is reached.

Further improvements can be achieved with the *nonlocal pixel exchange* [143]. It is designed to escape from potential local minima by moving a set of p candidate lo-

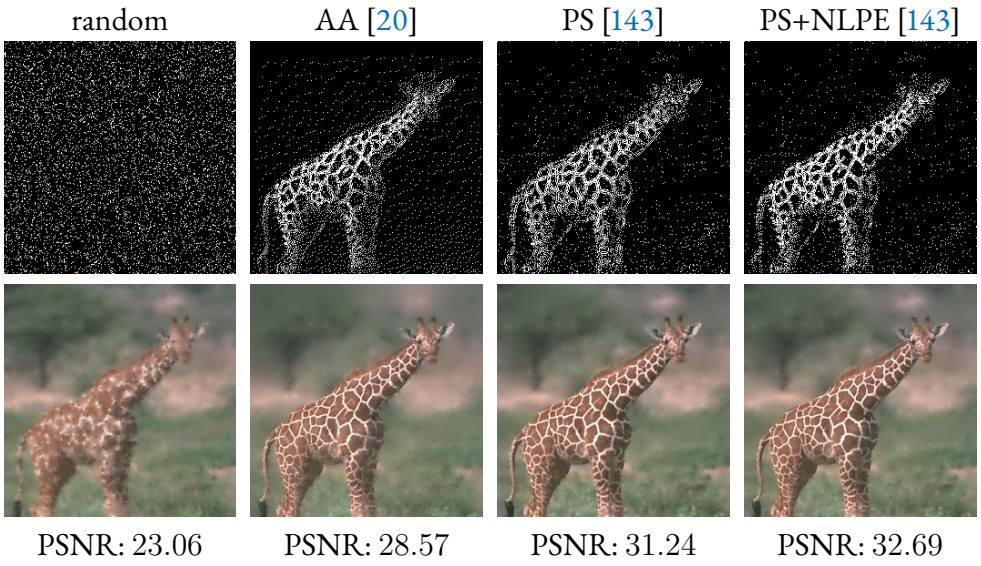


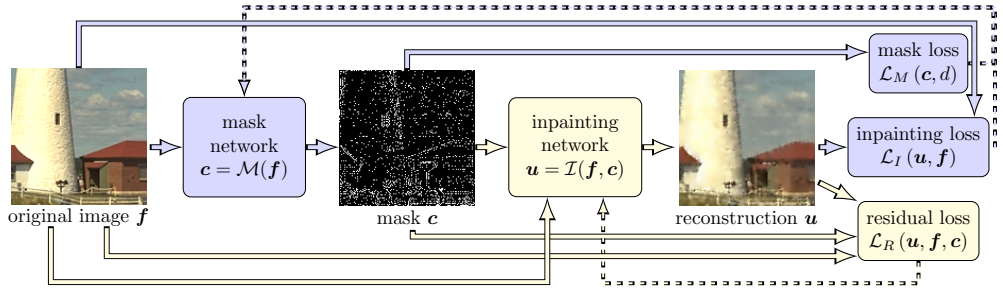
Figure 6.2: **Spatial Optimisation Techniques.** For reference, we consider a uniformly random mask with 10% known data and the corresponding reconstruction of image 130014 from Fig. 6.1 with homogeneous diffusion inpainting. The analytic approach [20] (AA) already yields a significant improvement over the random mask. Probabilistic sparsification (PS) and non-local pixel exchange (NLPE) [143] refine the background as well as the fur patterns of the giraffe. A gain of more than 9dB PSNR illustrates the vital importance of spatial optimisation for homogeneous diffusion inpainting.

cations from the inpainting mask to locations in the unknown image areas. If this positional exchange improves the overall inpainting, it is maintained, otherwise it is reverted. While this guarantees that mask quality cannot deteriorate, each step requires an inpainting and therefore, convergence tends to be slow.

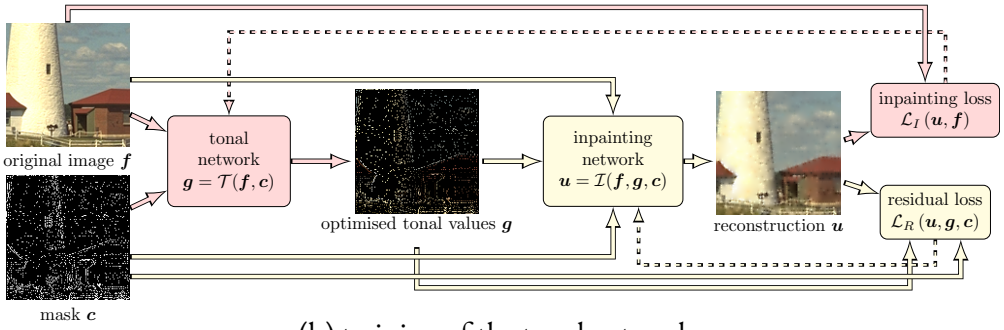
In Fig. 6.2, a comparison of the three aforementioned spatial optimisation techniques with a uniformly random mask highlights their significant impact. Carefully optimised known data are integral for good inpainting results.

Since we consider homogeneous diffusion and do not require quantisation, we use a least squares approach for tonal optimisation. Due to the similar quality of the tonal methods from Section 6.3, we choose the Green's formulation by Hoffmann et al. [93] equipped with a Cholesky solver. It offers good quality at fairly low computational cost, in particular for very sparse masks.

In the following sections we introduce a deep learning approach that does not require inpaintings during spatial or tonal optimisation and approximates the quality of probabilistic methods and model-based tonal optimisation.



(a) training of the mask network



(b) training of the tonal network

Figure 6.3: **Structure of Our Deep Tonal Optimisation Framework**. The surrogate inpainting network and its associated residual loss are marked in yellow, the mask network and loss in blue, and the tonal network in red. Forward passes between the networks are indicated by solid arrows, while dashed arrows represent backpropagation.

6.4.1 Spatial and Tonal Optimisation with Surrogate Inpainting

In this section, we describe the three types of networks that act as the building blocks for our neural data optimisation framework. The centrepiece required for our different pipelines is the *surrogate inpainting network*. It approximates inpainting with homogeneous diffusion by minimising the residual of the inpainting equation. We only use it during training. Its sole purpose is to act as a fast approximate solver for the inpainting problem that is still differentiable and allows backpropagation.

For the data optimisation, we consider a *mask network* for spatial optimisation and a *tonal network* for optimisation of the pixel values. Each of them is trained together with a separate surrogate inpainting network. Both data optimisation networks minimise the inpainting error w.r.t. the reconstruction by the respective surrogate solver.

In addition, the mask network requires a separate loss to approximate the intended mask density d. The macro architecture of our spatial approach with can be found in Fig. 6.3(a).

For the tonal setting in Fig. 6.3(b), we have a similar overall setup. However, here the binary masks are already part of the training dataset. In practice, we use our tonal network to generate these inputs, but also other sources such as model-based spatial optimisation approaches or even randomly generated masks could be used instead. Note that here, the optimised mask values are fed into the surrogate solver instead of the original ones.

All three types of networks use a similar U-net structure [185] that we discuss in more detail in Section 6.4.1. In the following sections on the individual networks, we only discuss deviations from this standard U-net architecture.

Deploying our networks for practical applications comes down to first applying the mask network to the input image. The resulting mask is then optionally fed into the tonal network together with the original. This yields the complete known data for homogeneous diffusion inpainting. The surrogate solver is never used in an evaluation scenario. Instead, we use model-based inpainting.

The Surrogate Inpainting Network

To train our mask and tonal networks, we require backpropagation from inpainting results. For instance, this could be achieved by translating a classical discrete implementation of a diffusion process into a neural network [11], which results in a sequence of ResNet [86] blocks. However, this might require very deep networks to reach the steady state of the diffusion process since the number of ResNet blocks is tied to the diffusion time in such a scenario. Instead, we propose an alternative that approximates inpainting results more efficiently by also having access to the ground truth.

The *surrogate inpainting network* $\mathcal{I}$ takes known data specified in terms of the locations in a binary or non-binary mask $\boldsymbol{c}$ and pixel values $\boldsymbol{g}$ as an input. Note that these known values do not necessarily need to coincide with the corresponding data in the original $\boldsymbol{f}$. In addition, it has access to the full known image $\boldsymbol{f}$. This network will only be used during training, and for evaluation, a model-based solver is responsible for the inpainting. Therefore, having access to the unknown pixels in $\Omega \setminus K$ eases the networks task and does not compromise the validity of data optimisation in any way.

The reconstruction $\boldsymbol{u} = \mathcal{I}(\boldsymbol{f}, \boldsymbol{g}, \boldsymbol{c})$ should solve the discrete inpainting equation

$$(\boldsymbol{I} - \boldsymbol{C})\boldsymbol{A}\boldsymbol{u} - \boldsymbol{C}(\boldsymbol{u} - \boldsymbol{g}) = \boldsymbol{0}, \tag{6.2}$$

which is a discretised version of Eq. (6.1). The finite difference discretisation of the Laplacian is represented by the matrix $\boldsymbol{A} \in \mathbb{R}^{n_x n_y \times n_x n_y}$ and $\boldsymbol{C} \in [0,1]^{n_x n_y \times n_x n_y}$ is a diagonal matrix containing the mask entries.

Since the network aims at simulating a numerical solver for Eq. (6.2), we follow the ideas of Alt et al. [11] and define a corresponding *residual loss*

$$\mathcal{L}_R(\boldsymbol{u}, \boldsymbol{g}, \boldsymbol{c}) = \frac{1}{n_x n_y} \|(\boldsymbol{I} - \boldsymbol{C})\boldsymbol{A}\boldsymbol{u} - \boldsymbol{C}(\boldsymbol{u} - \boldsymbol{g})\|_2^2 \,. \tag{6.3}$$

Here $\|\cdot\|_2$ denotes the Euclidean norm. Note that the inpainting network is explicitly *not* trained to minimise any reconstruction loss w.r.t. the original $\boldsymbol{f}$. The residual loss only makes sure that the networks produces a good approximation of homogeneous diffusion inpainting given the mask $\boldsymbol{c}$ and the pixel values $\boldsymbol{g}$. It follows similar principles as deep energy approaches [77]. This ensures that the surrogate solver's access to the full original image does not skew the data optimisation.

The Mask Network

Given the original image $\boldsymbol{f}$, our *mask network* $\mathcal{M}$ outputs positional data in terms of the mask $\boldsymbol{c} = \mathcal{M}(\boldsymbol{f})$ with a density d.

Non-binary Mask Networks

Our network outputs non-binary masks with values in $[0,1]$. Our goal is to optimise $\boldsymbol{c}$ for the best possible inpainting result. Therefore, our network is equipped with an *inpainting loss* that measures the deviation of the reconstruction $\boldsymbol{u}$ from the original $\boldsymbol{f}$ in terms of

$$\mathcal{L}_I(\boldsymbol{u}, \boldsymbol{f}) = \frac{1}{n_x n_y} \|\boldsymbol{u} - \boldsymbol{f}\|_2^2 \,. \tag{6.4}$$

While this loss establishes a connection between mask positions and reconstruction quality, it does not address the density. To this end, we apply a sigmoid activation at the last layer of our mask U-net, which limits the non-binary mask outputs to $[0,1]$. If the preliminary mask $\hat{\boldsymbol{c}}$ exceeds the target density d, we rescale it according to

$$\boldsymbol{c} = \frac{d\hat{\boldsymbol{c}}}{\frac{\|\hat{\boldsymbol{c}}\|_1}{n_x n_y} + \varepsilon} \,. \tag{6.5}$$

With $\varepsilon = 10^{-5}$ we avoid rounding issues for very low estimated mask densities and potential division by zero.

During training, our network passes on the non-binary confidence values. Values close to 1 indicate that the mask network sees this position as highly important, and a value close to 0 marks unimportant positions. For practical applications, however, we still require binary masks. These can be extracted with a simple postprocessing: Interpreting the confidence values as a probability, we perform a weighted coin flip for each confidence value.

Our experiments show that this non-binary mask optimisation creates a challenging energy landscape. During the training process, the mask network can get stuck in local minima that assign equal confidence to every mask pixel. Combined with the coin flip, this can lead to a uniform random mask. As a remedy, we propose an additional *mask loss* $\mathcal{L}_M$ that acts as a regulariser by penalising the inverse variance

$$\mathcal{L}_M(\boldsymbol{c}) = \alpha\left(\sigma_{\boldsymbol{c}}^2 + \varepsilon\right)^{-1} \tag{6.6}$$

As in Eq. (6.5), ε avoids division by zero. The regularisation parameter α balances the influence of the mask loss with the inpainting loss. Not only does this discourage flat masks with equal confidence in every pixel, but it also encourages confidence values close to 0 and 1. This yields the additional benefit of a closer approximation of binary masks during training.

Binary Mask Networks

Recently, strategies for deep data optimisation of neural network-based inpainting have been proposed that also allow direct output of binary masks [168]. This constitutes a challenge since the binarisation of real input values is a non-differentiable operation. However, end-to-end approaches that also learn the inpainting benefit from this binarisation, since the training of the inpainting network tends to be biased by a non-binary mask input. This leads to worse results during deployment of the inpainting network.

For our own strategy, we investigate two different alternatives for direct binarisation and evaluate their performance in Section 6.4.2.

Strategy 1: Quantisation First, we directly adopt the strategy of Peter [168]: We interpret binarisation of $x \in \mathbb{R}$ by hard rounding $x \mapsto \lfloor c + 0.5 \rfloor$ as very coarse quantisation. Theis et al. [224] have shown that simply approximating the derivative by 1 yields very good results among more sophisticated alternatives.

For this strategy, the variance-based regularisation from Eq. (6.6) is not necessary. However, the enforcement of the target density via rescaling from the non-binary

approach also does not work in this case. Therefore, we define the mask loss directly as the deviation from the target density d according to

$$\mathcal{L}_M(\boldsymbol{c}) = \left| \frac{\|\boldsymbol{c}\|_1}{n_x n_y} - d \right| . \tag{6.7}$$

Since the mask contains only binary values, the 1-norm $\|\cdot\|_1$ yields the number of mask points and thus the mask loss measures the deviation from the target density d. While the non-binary strategy does not require a density loss, we found in our experiments that it can have a stabilising effect on training if added to the regulariser loss from Eq. (6.6).

Strategy 2: Coin Flip Instead of quantisation, we can also modify our non-binary approach to output binary masks. We keep the regularisation mask loss and rescaling from Section 6.4.1, yielding a non-binary confidence mask. However, during training, we directly add the coin flip binarisation. This can be seen as an alternative quantisation approach instead of the rounding operation in Strategy 1. We apply the same synthetic gradient as in the first binary mask approach.

In Section 6.4.2 we evaluate the binary and non-binary alternatives for mask generation in an ablation study.

The Tonal Network

Finally, our tonal network takes both the original image $\boldsymbol{f}$ and a mask $\boldsymbol{c}$ as an input. The mask can either originate from the mask network or an external source.

Fortunately, we do not require binarisation layers, since the input masks are already binary. Furthermore, the mask density is already fixed. Therefore, the tonal network uses the U-net described in Section 6.4.1 without further need for modifications. It feeds the optimised pixel values $\boldsymbol{g} = \mathcal{T}(\boldsymbol{f}, \boldsymbol{c})$ into the inpainting loss from Eq. (6.4).

The residual network is trained with the residual loss w.r.t. the optimised known data $\mathcal{L}_R(\boldsymbol{u}, \boldsymbol{g}, \boldsymbol{c})$ as well. While this works well, we have found in our experiments that the training of the surrogate solver can be stabilised by also minimising the residual $\mathcal{L}_R(\boldsymbol{u}, \boldsymbol{f}, \boldsymbol{c})$ w.r.t. the original known data. This provides a fixed reference point for the residual solver, since in contrast to $\boldsymbol{g}$, the known data from $\boldsymbol{f}$ is not influenced by the training progress of the tonal network. This prevents the training of the residual solver from getting trapped in local minima.

Network Architecture

For all three networks, we use a U-net [185] architecture, since U-nets implement the core principles of multigrid solvers for PDE-based inpainting [11]. This makes them

a perfect fit for the surrogate solver. U-nets and multigrid have in common that they operate on multiple scales, first restricting the image in multiple stages down to the coarsest scale and then prolongating it again to the finest scale. We follow this general structure in Fig. 6.4(a), and offer further information on basic U-nets in Section 3.1.3.

We also rely on modifications to the standard U-net approach that were first used for inpainting by Vašata et al. [230]. They replace traditional convolutional layers by multiple parallel dilated convolutions with dilation factors 0, 2, and 5 followed by ELU activations. As shown in Fig. 6.4(b), the results are concatenated to a joint output. This so-called multiscale context aggregation was originally designed by Yu and Koltun [261] to increase the receptive field for segmentation. We discuss its benefits for our application in Section 6.4.2 with an ablation study.

For restriction, we also use context aggregation [261] with 5×5 dilated convolutions followed by a 2×2 max pooling. The corresponding prolongation uses the same structure, but with 5×5 transposed convolutions and 2×2 upsampling. Two context aggregation blocks without any upsampling or max pooling perform post-processing on the coarsest scale. The final hard sigmoid activation limits the results to the original image range $[0, 1]$. Only in the case of our binary mask networks, this is followed by a quantisation or coin flip binarisation layer. As commonly the case in multiscale architectures, the number of channels increases for coarser scales. It ranges from 64 to 256 (see Fig. 6.4(a) for details), which is half of the channel bandwidth used by Vašata et al. [230]. In Section 6.4.2 we have verified that such smaller networks suffice for our task.

6.4.2 Experimental Evaluation

After an overview of the technical details of our evaluation in Section 6.4.2, we justify our design decisions for the networks with an ablation study in Section 6.4.2. We compare with model-based approaches for spatial optimisation in Section 6.4.2 and with tonal optimisation methods in Section 6.4.2. In both cases, we assess reconstruction quality and speed.

Experimental Setup

Unless stated otherwise, all networks rely on the modified U-net architecture from Section 6.4.1 with ≈ 2.9 million parameters per network.

All of our networks have been trained on an Intel Xeon E5-2689 v4 CPU (2 cores), together with an Nvidia Pascal P100 16GB GPU. For training, we use a subset of 100,000 images randomly sampled from ImageNet [51] by Dai et al. [46] and the corresponding validation dataset containing 1,000 images. We use centre crops to reduce the size of the images, thus speeding up the training process. For model selec-

Figure 6.4: **Modified U-Net Architecture.** (a) The original inputs are subsampled by max pooling three times, pass through the bottleneck and are then prolongated by upsampling again to the finest scale. Exchange of information between scales is implemented by skip connections. Two post processing blocks with transposed convolutions conclude the pipeline. The numbers below each context aggregation block indicate the number of channels. (b) In a context aggregation block, three parallel dilated convolution increase the receptive field of the filter. All results are concatenated together.

tion we crop to 64×64, while the remainder of the experiments are performed on size 128×128. All networks were with the Adam optimiser [119] and a learning rate of $5 \cdot 10^{-5}$. We used 50 epochs for the spatial experiments, and 100 for tonal experiments. For evaluation, we used an AMD Ryzen 7 5800X CPU equipped with an Nvidia RTX 3090 24GB GPU. We performed model selection based on the lowest achieved inpainting error on the validation set. Our test set is based on all 500 images of the BSDS500 database [15]. These were centre cropped to size 128×128 in order to fit the size of the training data. The cropping also speeds up the model-based competitors and thus allows us to compare with them on a larger variety of images. We measure qualitative results with the peak signal-to-noise ratio (PSNR).

We compare with three spatial optimisation methods. The analytic approach by Belhachmi et al. [20] (AA) acts as a representative of very fast spatial optimisation. It is implemented with Floyd-Steinberg dithering [68] of the Laplace magnitude. Probabilistic sparsification (PS) in combination with a non-local pixel exchange (NLPE) provides qualitative benchmarks. These methods have been implemented with a conjugate gradient solver, ensuring convergence up to a relative residual of 10^{-6} for the diffusion inpainting. NLPE is run for 5 so-called cycles, each consisting of $\|\boldsymbol{c}\|_1$ iterations.

Ablation Study

In the following, we first evaluate different architectures and design principles, to select the best among those for the comparison with model-based approaches.

Network Architecture Compared to the standard U-net architecture that was used in [9], the modified U-net from Section 6.4.1 benefits from the context aggregation and more sophisticated postprocessing layers after upsampling to the finest scale. In [9], Alt et al. used sequential 3×3 convolutions on each scale. Therefore, propagation of information over larger distances works mainly via downsampling to coarse scales and upsampling. On each individual scale, the receptive field of the simple convolutions is relatively small. In contrast, the context aggregation allows our network to perceive larger regions of the image on each individual scale. Our evaluation in Table 6.1(a) contrasts these modifications with the standard U-net using a similar total amount of weights. The modifications yield up to 2.3 dB improvement w.r.t. PSNR, especially on challenging very sparse masks.

We also evaluated other modifications to the U-net structure such as gated convolutions, but the context aggregation yielded the best combination of good qualitative performance and stability during training.

In Table 6.1(b), we compare the full size U-net proposed by Vašata et al. [230] with our leaner version from Section 6.4.1 on 128×128 color images. The large U-net

density	PSNR (dB) 1%	5%	10%
non-binary [9]	19.40	25.45	28.34
our non-binary	**21.72**	**25.92**	**29.06**
coinflip	18.61	24.99	22.58
binary	20.08	24.00	26.04

(a) binary vs. non-binary

density	PSNR (dB) 1%	5%	10%
small	21.10	25.48	28.58
large	21.04	25.64	28.63

(b) small vs. large Masknet

Table 6.1: **Ablation Experiments.** All experiments have been conducted on 64×64 grey value centre crops from BSDS500 [15] with mask densities 1%, 5%, and 10%. (a) The non-binary masks outperform both binary options qualitatively over the full range of mask densities. In addition, our modified network architecture and training methodology outperforms the earlier non-binary mask network [9]. The coinflip variant showed instabilities during training for high densities and did thus not yield satisfying results for 10%. (b) Reducing the number of channels in the modified U-net by a factor 2 does not deteriorate the quality.

uses twice the amount of channels in relation to Fig. 6.4(a) in all but the last two postprocessing layers. This results in $\approx$ 11.5 million parameters, compared to our significantly lower $\approx$ 2.9 million. The larger network does not yield a qualitative advantage over a wide range of densities. The PSNR results of the large network only deviate marginally from those of the small masknet in Table 6.1(b). However, it increases training times from 43 min to 93 min per epoch. Therefore, we use our lean nets instead.

Non-binary vs. Binary Masks In Section 6.4.1 we have proposed three possible output options for our mask networks: non-binary masks, binary masks based on quantisation, and binary masks produced by a coin flip. For full deep learning based approaches [168], the binarisation during training is a key component of their architecture.

Surprisingly, our ablation study in Table 6.1(a) paints a different picture: The non-binary mask network clearly outperforms both binary options. This results from a key difference in our method compared to full deep learning approaches. Using a non-binary mask while simultaneously training an inpainting network introduces a bias. This deteriorates inpainting quality during testing [46]. However, our surrogate solver is only deployed during training and is not coupled directly to an inpainting loss. It merely approximates diffusion-based inpainting. During testing, we use a model-based implementation of homogeneous diffusion inpainting.

Therefore, we benefit from a non-binary mask network that does not rely on synthetic gradients for binarisation layers. Consequentially, we use the non-binary variant for our comparisons with model-based data optimisation.

Spatial Optimisation

We evaluate our networks on the BSDS500 greyscale image database for mask densities of up to 20% in Fig. 6.7(a). Our mask network not only consistently outperforms both the analytic approach [20] (AA) and probabilistic sparsification [143] (PS), but very closely approximates the quality of PS+NLPE.

The same ranking also applies in the case of the full colour version of BSDS500 in Fig. 6.7(b). Thus, our mask network rivals the best model-based approach in the comparison. Visually, it yields similar results as the probabilistic methods in Fig. 6.5 and Fig. 6.6. Especially for low densities, there is a large quality gap between the analytical approach and all other competitors.

Even though our mask net offers a similar quality as PS+NLPE, it requires significantly less computational time since it does not rely on any inpaintings during inference. On the CPU, it accelerates mask computation by up to a factor 3,500 and even up to a factor 140,000 on the GPU in Fig. 6.9(a). Only the analytic approach is faster with ≈ 0.2 ms. However, there the speed comes at the cost of a significantly diminished quality. Without compromising on quality, our mask net is also real-time capable with 1.4 ms on GPU and 55 ms on CPU.

Thus, our mask network reaches our goal of providing an easy to use, parameter-free spatial optimisation which approximates the quality of stochastic methods at a computational cost close to the instantaneous analytic approach.

Tonal Optimisation

In Fig. 6.8(a) we compare our tonal network with the Green's function approach of Hoffmann [93] on the masks obtained from our mask network. Especially for sparse known data, our deep tonal optimisation reaches a similar quality as the model-based approach. Only above 15%, the improvements over the unoptimised data from the mask net decline.

Our results in Fig. 6.5 show that our network approach also remains competitive to PS+NLPE when adding tonal optimisation. Here we apply the tonal network for our own deep learning method and the Green's function optimiser for all model-based competitors.

As for spatial optimisation, our tonal network offers a viable alternative for time critical applications. Fig. 6.9(b) shows that the computational cost of the Green's function approach grows significantly with the number of mask values that need to

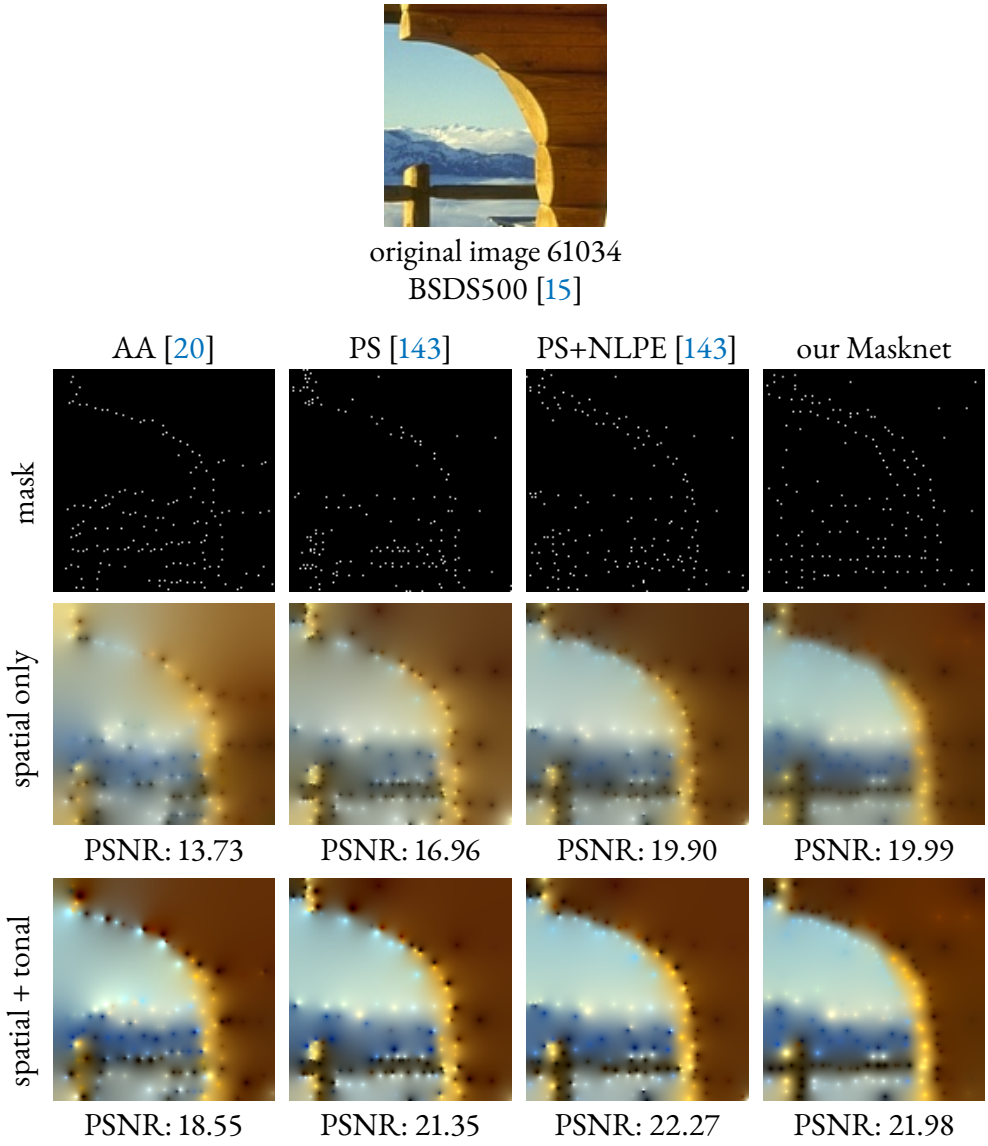


Figure 6.5: **Visual Comparison for 1% Mask Density.** Visually, probabilistic sparsification (PS) in combination with non-local pixel exchange (NLPE) [143], and our network-based approach significantly outperform the analytic approach (AA) [20]. For such sparse masks, the visual impact of tonal optimisation is apparent.

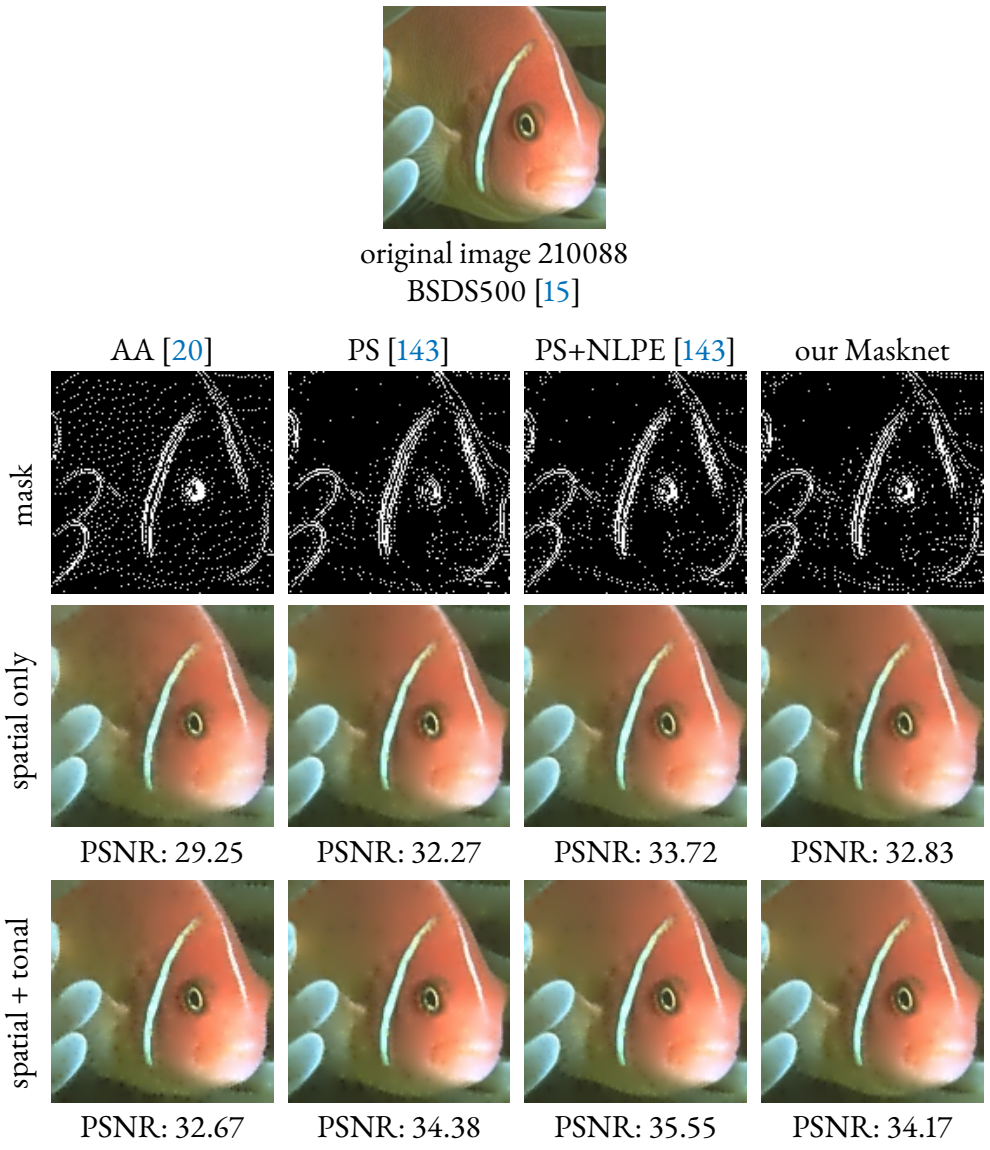
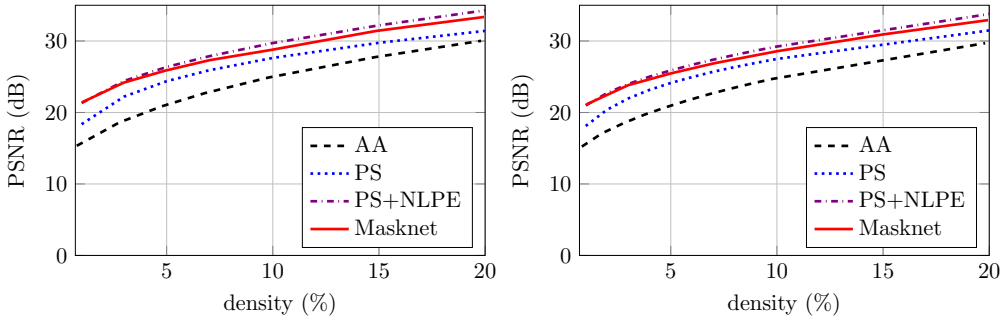
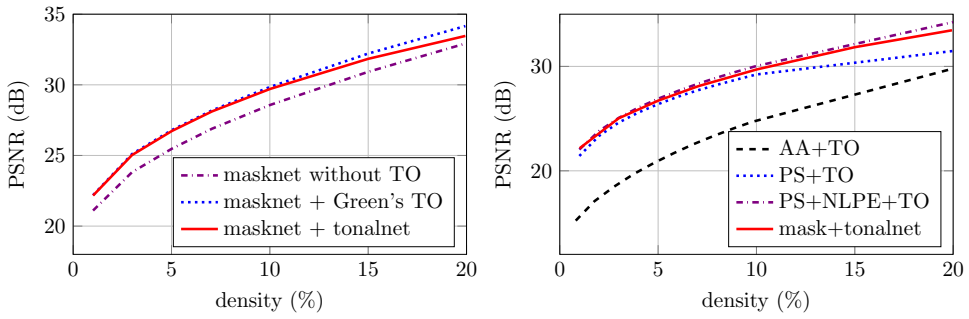


Figure 6.6: **Visual Comparison for 10% Mask Density.** Also at high density, our mask network yields results that are comparable to the probabilistic approaches PS and PS+NLPE [143]. The visual gap towards the analytic approach (AA) [20] is smaller, but still noticable.



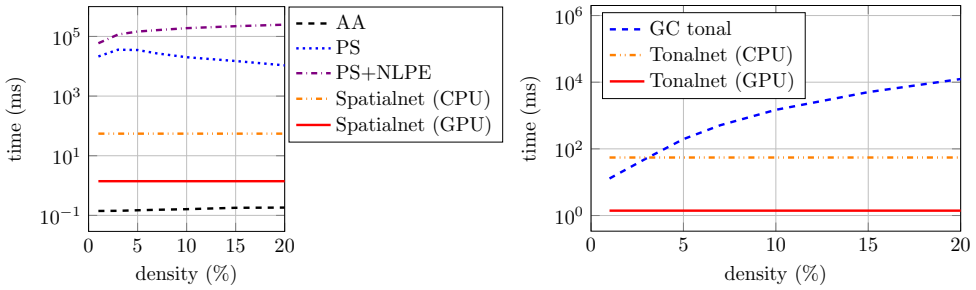
(a) BSDS500 grey (b) BSDS500 colour

Figure 6.7: **Spatial Optimisation.** (a) On grey level images, our network consistently outperforms the analytic spatial optimisation (AA) [20] and probabilistic sparsification (PS) [143]. Especially for lower densities, Masknet results rival the quality of PS with non-local pixel exchange (NLPE) [143] as postprocessing. (b) For colour images, our mask network also closely approximates the quality of PS+NLPE for the whole range from 1% to 20%.



(a) neural vs. model-based (b) full optimisation comparison

Figure 6.8: **Tonal Optimisation.** (a) We compare our tonal network with the Green's function approach [93] on masks from our mask network. Our tonal optimisation (TO) reaches a comparable quality up to 15% known data. (b) In a comparison that combines the spatially optimised mask with tonal optimisation, our full network approach yields competitive results to PS+NLPE combined with tonal optimisation. All model-based approaches use the Green's function approach [93] for tonal optimisation.



(a) runtime spatial (b) runtime tonal

Figure 6.9: **Runtime Comparison with Logarithmic Time Axis.** (a) The analytic approach (AA) [20] is the fastest method, followed closely by our Masknet on the GPU and CPU. These three methods all have a constant speed independently of mask density. The speed of PS and PS+NLPE [143] is density dependent. Overall, our methods are consistently faster by several orders of magnitude compared to probabilistic approaches. (b) The situation for tonal optimisation is comparable. The Green's function-based solver [93] becomes increasingly slower with rising mask density. Only for very low densities its speed is comparable to our tonal network on the CPU. The networks have constant runtime independent of density and are faster by 1 to 5 orders of magnitude.

be optimised. In contrast, the computational time of the tonal network is independent of the mask density. For densities larger than 5%, speed-ups by multiple orders of magnitude can be achieved with our mask net.

Thus, a combination of our spatial and tonal networks is a viable option for real-time applications that does not require to sacrifice quality for speed.

6.5 Efficient Neural Generation of 4K Masks for Homogeneous Diffusion Inpainting

Our mask optimisation framework from Section 6.4 offers a good combination of quality and speed for greyscale images of sizes up to 256×256 and colour images up to 128×128. However, it is inflexible: The mask network can only generate masks for the density and resolution it was trained for. Furthermore, a naïve extension to high-resolution images requires a prohibitive amounts of compute power during training: At a resolution of 3840×2160, 4K colour images have around 25 million values, two to three orders of magnitude more than previously considered.

We solve this with a coarse-to-fine approach, which divides the image into patches. Each patch is then processed with an improved version of the framework introduced

in Section 6.4. Most notably, we replace the surrogate solver with a model-based inpainting directly inside the deep learning model.

Model-Based Mask Optimisation

A mask optimisation strategy that does not require any inpaintings has been proposed by Belhachmi et al. [20]. They show that for optimal masks, the local density should increase with the Laplacian magnitude. In Section 6.5.2 we use this result to predict a suitable density for image regions. However, the application of these results to discrete images relies on dithering. The commonly used Floyd-Steinberg dithering [68] is fast and simple, but suffers from a directional bias and generates masks of relatively low quality.

A method which produces better masks but remains simple is *probabilistic sparsification* [143]. It starts with a full mask and gradually removes pixels until the target density d is reached. In every iteration, a fraction p of mask pixels is removed. Then, an inpainting is computed, and the fraction q which resulted in the largest loss of quality is added back.

We also consider *nonlocal pixel exchange* [143] as a post-processing method. In each step, it moves a fraction p of mask pixels into the unknown image area. It then inpaints with the new mask to check for improvements: Exchanges which increase inpainting quality are kept, while unsuccessful ones are reverted. This is repeated for k cycles with $\|\boldsymbol{c}\|_1$ iterations each. Nonlocal pixel exchange can help greedy approaches like probabilistic sparsification escape from poor local minima at the cost of significant runtime.

Both probablisitic sparsification and nonlocal pixel exchange require an inpainting for each iteration, with the other operations being quick in comparison. As such, their runtime is primarily determined by the number and the speed of the inpainting operations. A naïve CPU-based inpainter using a conjugate gradient solver takes around 3.5 seconds per 4K colour image on a contemporary PC. At this speed, 5 cycles of nonlocal pixel exchange for a mask with 5% known data would take around 12 weeks. To enable meaningful comparisons, we have developed implementations of probablistic sparsification and nonlocal pixel exchange that use the very fast GPU-based inpainting of Kämper et al. [111]. With it, an iteration of either method requires only ≈ 6 milliseconds, a speedup of two orders of magnitude. To the best of our knowledge, our implementations are the fastest available.

6.5.1 Neural Mask Generation

We follow the basic network architecture from Peter et al. [174] with a mask generator network and an inpainting approximator. The latter is used to train the mask

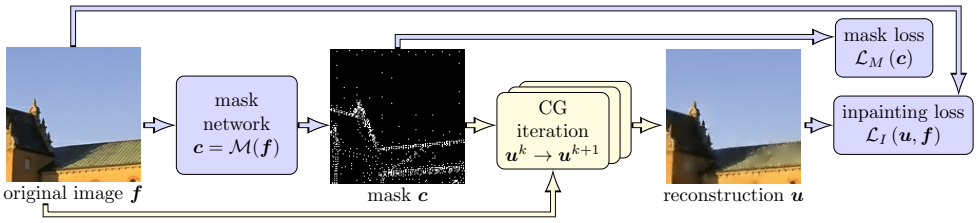


Figure 6.10: **Our Mask Optimisation Architecture.** The mask network and its losses are coloured in blue, our newly introduced sequence of CG iterations is given in yellow.

network by evaluating the reconstruction quality and is discarded during inference. An overview of our architecture can be found in Figure 6.10. While we retain the mask network as is, we completely replace the original inpainting approximator.

Mask Network The mask network receives the original image $\boldsymbol{f}$ and generates a mask $\boldsymbol{c} = \mathcal{M}(\boldsymbol{f})$. Its output is restricted to $[0, 1]$ by applying a sigmoid activation function. Additionally, the network outputs are rescaled if they exceed the desired density d. The goal of the network is to produce masks such that the corresponding inpainted image $\boldsymbol{u}$ is close to the original $\boldsymbol{f}$. To this end, we minimise their MSE $\mathcal{L}_I(\boldsymbol{u}, \boldsymbol{f}) = \frac{1}{n_x n_y} \|\boldsymbol{u} - \boldsymbol{f}\|_2^2$. In addition, we require a mechanism to encourage the generation of binary masks. To this end, we penalise the inverse variance of the mask using $\mathcal{L}_M(\boldsymbol{c}) = \alpha(\sigma_{\boldsymbol{c}}^2 + \varepsilon)^{-1}$ where ε is a small numerical constant to avoid division by 0, and α balances variance loss $\mathcal{L}_M$ and inpainting loss $\mathcal{L}_I$.

Inpainting Approximator In order to train the mask network with an inpainting loss $\mathcal{L}_I$, we need to approximate the inpainting process inside the network. The approach from Section 6.4 achieves this with a surrogate inpainting network which receives the original $\boldsymbol{f}$ and mask $\boldsymbol{c}$, and is trained to find a reconstruction $\boldsymbol{u}$ which solves the discrete version of the inpainting equation (6.1):

$$(\boldsymbol{I} - \boldsymbol{C})\boldsymbol{A}\boldsymbol{u} - \boldsymbol{C}(\boldsymbol{u} - \boldsymbol{f}) = \boldsymbol{0} \tag{6.8}$$

Here, the matrix $\boldsymbol{C} = \operatorname{diag}(\boldsymbol{c})$ contains the mask entries on the diagonal, and $\boldsymbol{A}$ applies a finite difference discretisation of the Laplacian Δ with reflecting boundary conditions. This approach significantly increases the total number of weights and adds complexity to the architecture. Furthermore, it decreases interpretability as the surrogate is, apart from its loss function, a black box.

We propose to replace it by a sufficient number of iterations of a successful numerical solver. As homogeneous diffusion leads to a linear system of equations with

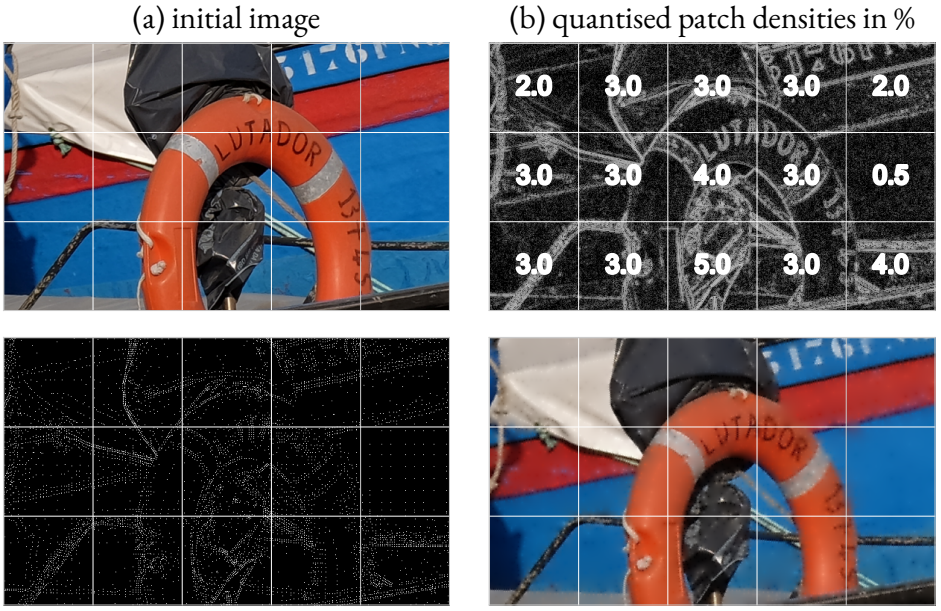


Figure 6.11: **Stages of the Mask Generation Process.** (a) Initial image with subdivision into patches. (b) Quantised patch densities, and Laplacian magnitude with logarithmic dynamic compression for better visibility in the background. (c) Binary mask with 3% known data. (d) Inpainted result.

a symmetric system matrix, we can use the conjugate gradient (CG) method [252]. It offers convergence guarantees while remaining simple and efficient. As each iteration is differentiable, we can backpropagate through the solver to train the mask network. By introducing this well-understood numerical solver, we have reduced the total number of weights by half compared to [174] while increasing interpretability. Section 6.5.3 confirms that this improves inpainting approximation quality greatly and the quality of generated masks slightly.

Mask Generation in Practice As the generated masks are not guaranteed to be binary, we need to binarise them. To this end, Peter et al. [174] perform weighted coinflips at each mask pixel and then choose the best-performing mask out of 30 attempts. We found that rounding leads to a comparable quality for our masks. It is less time intensive and involves no randomness.

6.5.2 Coarse-To-Fine Approach for Mask Generation

Simply partitioning a large image into patches and generating masks with equal density for each leads to suboptimal results: Textured regions receive too few, while homogeneous areas receive too many mask pixels. As such, we require a fast and simple mechanism that estimates suitable densities for each patch while taking the whole image into account.

To this end, we estimate a good patch density using the average Laplacian magnitude per patch, similar to the approach by Belhachmi et al. [20]. They have shown that for optimal continuous masks, the local mask density should increase with the Laplacian magnitude. The discrete approximation of this result through dithering requires significant compromises. We, however, aggregate the Laplacian magnitude over an image patch to estimate the optimal density and thus avoid dithering completely. To the best of our knowledge, we are the first to apply the optimality result this way. This motivates the following algorithm, which is also visualised in Figure 6.11:

1. Compute the Laplacian magnitude of the luma channel for every pixel.
2. Rescale it such that its global mean matches the target density.
3. Compute the target patch densities as the mean of the rescaled Laplacian magnitude per patch. Section 6.5.3 confirms that these target densities correlate well with high-quality masks.
4. Quantise the patch densities to values for which pre-trained mask networks are available and generate masks for every patch.
5. Assemble the mask patches into a mask for the whole image.

As an additional benefit, this methodology allows to generate masks with arbitrary densities. This is achieved by selecting densities per patch out of the available ones such that their mean approximates the desired average well.

6.5.3 Experiments

After mentioning the technical details in Section 6.5.3, we show the improvements made by introducing a CG solver into the network in Section 6.5.3. Additionally, we provide empirical evidence for our patch density estimation in Section 6.5.3, and compare the mask generation performance for 4K images in Section 6.5.3.

Figure 6.12: Our 12 test images of size 3840×2160. Photos by J. Weickert.

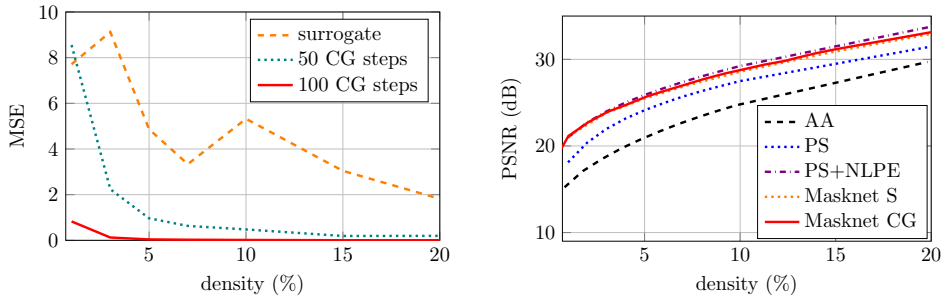
Experimental Setup

For our mask generator, we use the same small multiscale context aggregation network [230] with ≈ 2.9 million parameters and settings as in [174] to facilitate direct comparisons. It is based on a U-Net architecture with four scales and uses blocks of parallel dilated convolutions with different dilation rates. All mask networks are trained with the losses described in Section 6.5.1. The mask variance loss weight is set to $\alpha = 0.01$. The surrogate network in Section 6.5.3 shares the mask network architecture and uses the squared residual of the discrete inpainting equation (6.8) as its loss. All networks are trained for 100 epochs with a batch size of 8 and the Adam optimiser [119] with a learning rate of $5 \cdot 10^{-5}$. Performance is evaluated on an AMD Ryzen 7 5800X CPU and an Nvidia RTX 3090 GPU. For comparisons against [174], we trained on a subset of 100,000 images from ImageNet [51] sampled by Dai et al. [46]. Tests are performed on the full BSDS500 dataset [15]. We used 128×128 centre crops for both. The networks used for 4K inpainting were trained on 100,000 random patches of size 120×120 from the high-resolution image dataset Div2K [6]. Tests are performed on 12 representative 4K images photographed by one of the authors; see Figure 6.12. Our selection of pre-trained networks is optimised for target densities $\leq 10\%$, as those are practically relevant for compression. The set contains networks for different densities in the range from 0.5% to 80%. For 1%-15%, we use increments of 1%. For 15%-25% we increment by 2%, and by 5% for densities between 25% and 50%. Finally, in the range of 50%-80% we have steps of 10%. Reference inpaintings are computed using a CG solver which is stopped after a relative decrease of the Euclidean norm of the residual of 10^{-6}.

For our comparisons against model-driven approaches, we choose the analytic approach (AA) by Belhachmi et al. [20], and probabilistic sparsification (PS) alone or combined with nonlocal pixel exchange (PS+NLPE). PS uses candidate fractions $p = 0.3$ and $q = 0.005$. NLPE runs for 5 cycles. The other NLPE parameters were optimised individually for the different target resolutions.

Quality of Inpainting Approximations

We measure the quality of different inpainting approximators by comparing against inpaintings with our reference solver. To avoid a bias against the surrogate inpainting networks, we test on binarised masks generated by their corresponding mask networks. In Figure 6.13(a), we see that 100 CG iterations are a better approximator than the surrogate network across all densities. Additionally, they allow for faster training: On image batches, one forward and backward pass takes only 1.9 ms per image, while the surrogate requires 5.6 ms. Nevertheless, the quality of the trained mask network is only slightly improved by using CG as an inpainting approximator as can be seen



(a) Inpainting Approximation Quality (b) Mask Network Quality

Figure 6.13: **Comparison of Different Inpainting Approximators.** (a) Distance between the inpainting approximations and a converged inpainting. The MSEs decrease in a nonmonotone fashion due to the varying quality of different mask and inpainting networks. (b) Comparison of mask networks trained with either a surrogate inpainter (Masknet S) or 100 CG iterations (Masknet CG). Even though 100 CG iterations are a much better inpainting approximator, the quality of the mask network trained with them is only slightly improved. Both methods are better than the analytic approach (AA) and probablistic sparsification (PS), and are very close to PS with added nonlocal pixel exchange (PS+NLPE).

in Figure 6.13(b). This makes sense as the reconstruction error is about 3000 times larger than the approximation error for 100 CG steps and all densities, making small deviations insignificant.

Justification of Mask Pixel Distribution

In Figure 6.14 we show that the rescaled Laplacian magnitude is a good predictor of optimal mask densities at a patch level. There we compare against the patch densities of a well optimised mask generated with the approach from Chizhov and Weickert [40]. The relationship between the patch densities is almost linear, and the mean absolute error between densities is only 0.5%. While dithering the rescaled Laplacian magnitude leads to low-quality masks, its aggregation over a patch avoids this and produces a good coarse scale density estimate.

High-Resolution Mask Generation

The tests on 4K images in Figure 6.15(a) show that our masknet trained with 100 CG iteration is superior to AA and PS for all densities. In addition, we are even able to outperform PS+NLPE on densities smaller than 8%. This range is practically relevant for inpainting-based compression.

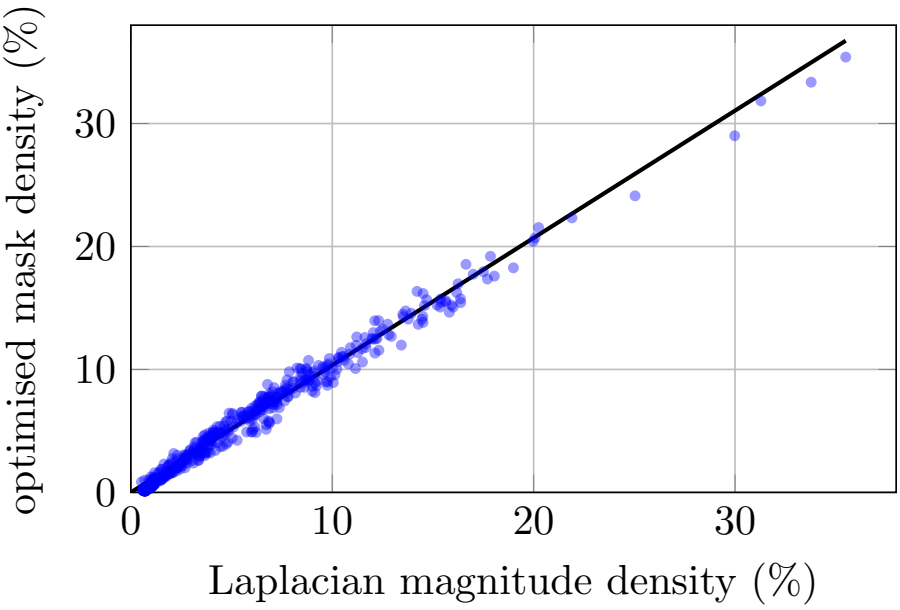
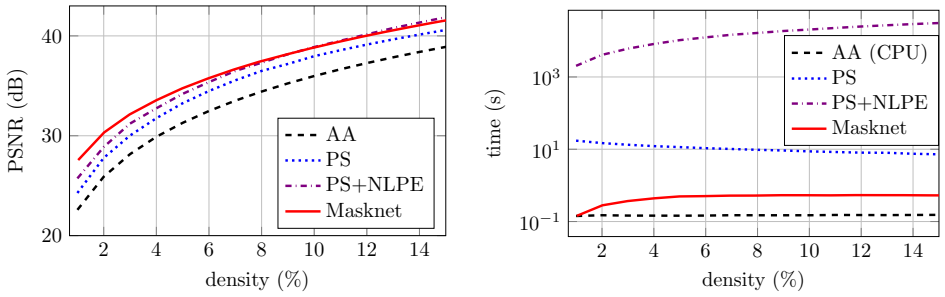


Figure 6.14: **Patch Density Correlation between Laplacian Magnitude and High Quality Mask.** Comparison of the mean rescaled Laplacian magnitude for each patch of the image *lofsdalen* against patch densities of a high quality mask, both with a total density of 5%. A correlation coefficient of 0.994 confirms the strong connection.



(a) quality (b) runtime

Figure 6.15: **Spatial Optimisation.** Our masknet outperforms the faster analytic approach, but also the slower PS across all densities. It even beats the significantly slower PS+NLPE for densities $\leq 8\%$. Time measurements exclude input/output operations.

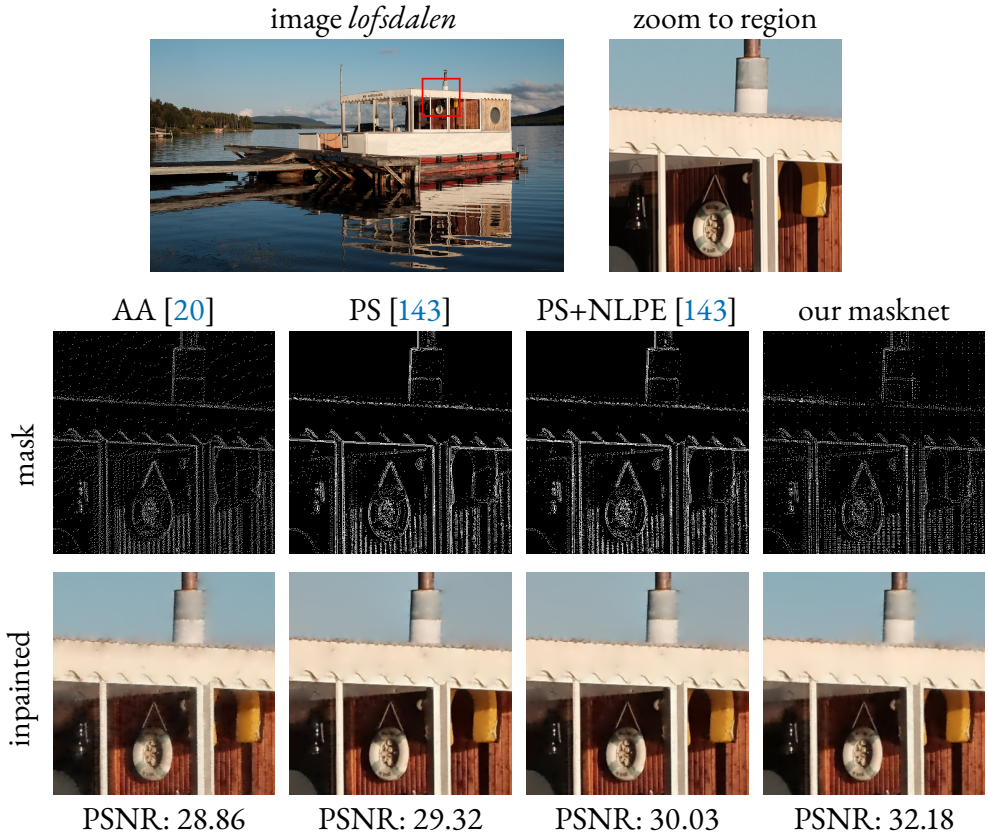


Figure 6.16: **Visual Comparison for 4% Mask Density on *lofsdalen*.** PSNRs are for the whole image. Notice the discoloured sky for PS and PS+NLPE. The full images are available in the supplementary material.

In Figure 6.15(b) we compare the speed of the different methods. The runtime of PS+NLPE scales with the required inpaintings, taking between 30 minutes and 8.5 hours depending on the density. In contrast, our neural approach takes only about 0.6 seconds per image. It is even quicker for the lowest densities where fewer of the pre-trained mask networks are active and the batches per net are larger. Our masknet is also more than 10 times faster than the qualitatively worse PS. The analytic approach requires no inpaintings, and even a CPU-based implementation is significantly faster than our neural method. Still, its quality is inferior by a large margin. In Figure 6.16, the most noticeable differences are slightly blurry edges for AA, and a discoloured sky for PS and PS+NLPE.

6.6 Chapter Conclusions

Our data optimisation approach in Section 6.4 merges classical inpainting with partial differential equations and deep learning with a surrogate solver. This allows us to select both position and values of known data for homogeneous diffusion inpainting that minimise the reconstruction error.

With this strategy we rival the results of probabilistic sparsification with postprocessing by non-local pixel exchange and tonal optimisation in terms of quality. Simultaneously, we are reaching the near instantaneous speed of the qualitatively inferior analytic approach.

We further refine this process with our coarse-to-fine approach. It is orders of magnitude faster than qualitatively similar approaches on 4K images. This shows that the estimation of patch densities using the optimality result by Belhachmi et al. [20] works well, and our mask networks are able to outperform dithering. Performance of the neural approach relative to PS+NLPE even grew with the increase in resolution. Our experiments suggest that the coarse-to-fine approach produces local problems that are easier and faster to solve in high quality. This is in line with transform-based codecs like JPEG [166], which also use a block structure to enable parallelisation and reduce complexity.

In the future, we plan to integrate our framework into image compression codecs. Time-consuming spatial and tonal optimisation still present a bottleneck in this area. This holds true especially for practical applications with high demand for computational efficiency, such as video coding. While real-time decoding is already possible with diffusion [14, 122, 173], the data selection during encoding will benefit from our deep optimisation.

We believe that this chapter illustrates well what neuroexplicit methods stand for. Our algorithms learn from data, but incorporate model-driven solvers at the same time. In addition, our loss functions contains both the MSE as a quality measure and the squared residual to ensure adherence to our explicit constrains. This interplay of aspects of both worlds is, to us, the core of neuroexplicit research.

Our approach with the model-driven solver inside the network has also shown that neural networks can learn to interact with model-driven components. Through the training procedure, the mask network learned which structures in an image are hard to reconstruct for homogeneous diffusion, and how to distribute the mask pixels accordingly. Just as humans have constructed heuristics for this task, so has the neural network. In comparison to probabilistic methods, we have thus traded some measure of transparency for speed and quality.

7 Conclusions and Outlook

7.1 Conclusions

This thesis demonstrates that neuroexplicit models are always a trade-off. Introducing large amounts of weights or parameters into a model always reduces interpretability, at least on a local level. A human expert is usually able to comprehend the meaning and impact of a handful of model parameters, but somewhere on the road to the billions of parameters used today, it becomes impossible for any human to truly understand them.

Somewhere along the way, understanding shifts from individual values to larger building blocks. In Chapter 5, we represented discrete images and continuous sound fields as CNNs and neural fields respectively. As such, we still understand their role in the larger task, as well as their inputs, outputs, and training objective. However, the individual values of our trained weights hold no human-interpretable meaning on their own any more.

This also symbolises that just being able to explicitly describe a model is not meaningful on its own. A trained neural network can be stated explicitly as a sequence of various matrix and point operations. However, this typically does not allow a human to glean any insight. In contrast, the individual steps of numerical algorithms like CG are both explicit and interpretable to human experts.

However, as can be seen both in this thesis and the overwhelming success of neural networks in the last decade, introducing non-interpretable components can be a price worth paying. Of course, by building neural architectures capable of discovering connections and representations beyond human comprehension, one receives just that: Networks which can exceed the capabilities of human-designed methods.

Therefore, the design of neuroexplicit methods comes down to finding the best trade-offs: Making the smallest part of the overall algorithm non-interpretable while reaping the largest possible speed and quality benefits. This thesis demonstrates multiple examples of this, each with small, deliberately chosen neural building blocks which lead to demonstrable benefits.

In Chapter 4, we unify a range of discretisations for anisotropic diffusion in a single framework and derive stability bounds. Before this work, time step size limits were only experimentally confirmed. By connecting the discretisation to the build-

ing block of residual networks, we obtain a simple yet efficient implementation that benefits from deep learning frameworks while retaining theoretical guarantees.

Chapter 5 establishes that solving PDEs and variational models with tools from deep learning can serve as a viable alternative to purely model-driven solvers. By combining deep priors with a variational energy loss and principled numerical formulations, our solver for Euler's elastica inpainting matches the quality of highly specialised state-of-the-art methods while remaining broadly applicable. Moreover, our extension of physics-informed neural networks for simultaneous magnitude and phase prediction in magnitude distribution reconstruction demonstrates the flexibility of these techniques. Our approach remains interpretable while exploiting optimisation capabilities inherent to deep learning frameworks.

In Chapter 6, we address the relatively unexplored field of neural mask optimisation. Combining both data-driven and model-driven losses into a single architecture allows us to benefit from both. Our extension based on domain decomposition with theoretical backing enables efficient parallelisation. Within each subproblem, neural solvers are guided by model-based objectives, producing interpretable results. Finally, our approach does not even require any annotated image data and is able to learn in an unsupervised fashion. Our work demonstrates that neuroexplicit approaches offer a competitive alternative to probablistic methods. Furthermore, we overcome the drawbacks of early works, which were limited to fixed mask densities and resolutions.

Observing the applications in this thesis, we can identify some common patterns where neuroexplicit solutions can help in otherwise model-driven problems. Clearly, when there is a simple, reliable, fast, and high-quality model-driven solution available, then there is no need to introduce neural networks. In such a situation, they can only make things worse. However, plenty of model-driven approaches still fall short in at least one of those categories. Anisotropic diffusion in Chapter 4 could, in addition to existing acceleration strategies, still benefit from a further speed-up using a simple but fast neural implementation. Existing approaches for Euler's elastica in Section 5.4 were largely slow, highly involved, or of mixed quality. Our neural solver offered competitive quality while remaining simple. Even the well-explored problem of mask learning could still benefit in terms of speed and quality for large images through a data-driven approach in Chapter 6.

While not every problem benefits from the indiscriminate addition of neural networks, the domains studied here exemplify a broad class where carefully chosen neural ideas are a worthwhile trade-off, enhancing performance while retaining large parts of the interpretability and transparency of model-driven methods. The puzzle pieces provided in this thesis contribute to realising the potential of neuroexplicit models, in particular those involving well-defined numerical models.

7.2 Outlook

Based on our arguments in the previous section, one might think that more research on neuroexplicit models, particularly those which are very light on neural components, would be conducted. However, most research involving deep learning happens on one extreme of the possible trade-offs: maximal performance, with very little interpretability remaining. In areas where there is sufficient data to train those models, plenty of neural publications even laud their performance gains over model-driven solutions, while not mentioning the lack of transparency and interpretability of their networks.

Furthermore, a lot of the research that values understanding works on post-hoc explanations of large models, probing neural black boxes to extract some human-interpretable meaning. While this work has shed light on some of the inner workings of these highly sophisticated models, there is also a lot to gain by starting from the other direction: Designing models which are interpretable right from the start. The research in this thesis and beyond shows that the symbiosis of neural and model-driven methods can achieve more than either of those on their own. We hope in particular that our successful examples of mathematical model with well-contained neural black-boxes will serve as inspiration for further contributions to this field. Together with this general outlook, we also see specific opportunities for the applications we considered.

The principle of directional splitting presented in Chapter 4 extends beyond anisotropic diffusion and can be applied to other anisotropic processes such as mean curvature motion [245]. The strategy is not limited to 2-D grids and extends naturally to hexagonal grids or 3-D processes [241]. Challenges could arise, however, in the spectral analysis of nonsymmetric system matrices. Furthermore, the connection between numerical schemes and neural architectures suggests that many PDE-based methods, particularly those expressible in terms of convolutions and pointwise operations, can benefit from the highly optimised implementations in deep learning frameworks. While customised CUDA implementations may still achieve superior performance, the simplicity and accessibility of these translations make them compelling for rapid prototyping.

For Chapter 5, the application of neural solvers to poorly conditioned PDEs and variational models represents only a first step. Many relevant problems in practice are poorly conditioned, and moving beyond model-driven solvers can allow for further progress. Specifically for our work on elastica, we hope that further advances in discretisations may mitigate checkerboard artefacts and lessen the reliance on deep priors. In the case of sound field magnitude inpainting, our work did not require any datasets. To move even further into the neuroexplicit domain, one could also train on

a set of different frequencies and room configurations to improve quality and enable wider applicability without retraining for every instance.

For Chapter 6, scaling beyond patch-based decomposition is a natural next step. By training fully convolutional architectures on an appropriately rescaled version of the Laplacian magnitude, mask generation could be adapted to arbitrary target densities and image sizes. Future work may also extend the framework to more advanced inpainting operators by either learning operator approximations or embedding suitable solvers directly into the optimisation pipeline. Directions toward compression are equally promising: combining mask optimisation with concepts from neural coding, where expected code length is incorporated into loss functions, could enable further inpainting-based compression codecs.

A Contributions and Publications

A.1 Further Contributions

Apart from our journal paper on neural mask optimisation [174] in Chapter 6.4, only first-author publications are discussed in this thesis. Here, we shortly discuss other works to which we contributed.

- Our work on connecting anisotropic diffusion stencils to ResNets [202] was preceded by multiple other papers which connected neural networks and numerical algorithms for PDEs [10, 11]. In these, we analyse how numerical schemes for nonlinear diffusion, wavelet shrinkage, and variational methods all connect to ResNet blocks. Furthermore, we discover how diffusivities and activations functions relate to each other, and inspire novel activation functions this way. Lastly, we also translate notions of stability to the neural realm.
- PDEs and variational methods often offer invariance under translation and rotation by design. This is typically rooted in the underlying model assumptions, which are then translated into a mathematical model. While CNNs are shift invariant by design, they are not rotation invariant. Through the introduction of coupling activation functions [12] we are able to enforce rotation invariance while still preserving the directional filtering capabilities of the network.

A.2 List of Publications

Journal Publications

- T. Alt, **K. Schrader**, M. Augustin, P. Peter, and J. Weickert: "Connections between numerical algorithms for PDEs and neural networks", *Journal of Mathematical Imaging and Vision*, Vol. 65, 185–208, June 2022
- T. Alt, **K. Schrader**, J. Weickert, P. Peter, and M. Augustin: "Designing rotationally invariant neural networks from PDEs and variational methods", *Research in the Mathematical Sciences*, Vol. 9, No. 3, Article 52, Sept. 2022

- P. Peter, **K. Schrader**, T. Alt, and J. Weickert: "Deep spatial and tonal data optimisation for homogeneous diffusion inpainting", *Pattern Analysis and Applications*, Vol. 26, No. 4, 1585–1600, 2023

Conference Papers

- T. Alt, P. Peter, J. Weickert, and **K. Schrader**: "Translating numerical concepts for PDEs into neural architectures", In A. Elmoataz, J. Fadili, Y. Quéau, J. Rabin, and L. Simon (Eds.): *Scale Space and Variational Methods in Computer Vision*, Lecture Notes in Computer Science, Vol. 12679, 294–306, Springer, Cham, 2021

- **K. Schrader**, T. Alt, J. Weickert, and M. Ertel: "CNN-based Euler's elastica inpainting with deep energy and deep image prior", *Proc. 10th European Workshop on Visual Information Processing*, Lisbon, Portugal, Sept. 2022

- **K. Schrader**, P. Peter, N. Kämper, and J. Weickert: "Efficient neural generation of 4K masks for homogeneous diffusion inpainting.", In L. Calatroni, M. Donatelli, S. Morigi, M. Prato, and M. Santavesaria (Eds.): *Scale Space and Variational Methods in Computer Vision*, Lecture Notes in Computer Science, Vol. 14009, 16–28, Springer, Cham, 2023

- **K. Schrader**, J. Weickert, and M. Krause: "Anisotropic diffusion stencils: from simple derivations over stability estimates to ResNet implementations", To appear in K. Burnecki, J. Szwabiński, and M. Teuerle (Ed.): *Progress in Industrial Mathematics at ECMI 2023*, Springer, Cham, 2026

- **K. Schrader**, S. Koyama, T. Nakamura, and M. Pezzoli: "Phase-Retrieval-Based Physics-Informed Neural Networks For Acoustic Magnitude Field Reconstruction", To appear in *Proc. 2026 IEEE International Conference on Acoustics, Speech, and Signal Processing*, Barcelona, Spain, 2026.

B Bibliography

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, et al.: *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*, Software available from tensorflow.org, 2015 (cited on pp. 9, 16).

[2] T. Acar and M. Gökmen: "Image coding using weak membrane model of images", In A. K. Katsaggelos (Ed.): *Visual Communications and Image Processing '94*, Proceedings of SPIE, Vol. 2308, 1221–1230, SPIE Press, Bellingham, 1994 (cited on p. 75).

[3] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, et al.: *GPT-4 technical report*, arXiv:2303.08774 [cs.CL], 2023 (cited on pp. 1, 22).

[4] D. Ackermann, F. Brinkmann, F. Zotter, M. Kob, and S. Weinzierl: "Comparative evaluation of interpolation methods for the directivity of musical instruments", *EURASIP Journal on Audio, Speech, and Music Processing*, Article No. 36, 2021 (cited on p. 62).

[5] R. D. Adam, P. Peter, and J. Weickert: "Denoising by inpainting", In F. Lauze, Y. Dong, and A. B. Dahl (Eds.): *Scale Space and Variational Methods in Computer Vision*, Lecture Notes in Computer Science, Vol. 10302, 121–132, Springer, Cham, 2017 (cited on p. 76).

[6] E. Agustsson and R. Timofte: "NTIRE 2017 challenge on single image super-resolution: dataset and study", *Proc. 2017 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, Vol. 1, 1122–1131, Honolulu, HI, July 2017 (cited on p. 104).

[7] S. Alemohammad, J. Casco-Rodriguez, L. Luzi, A. I. Humayun, H. Babaei, et al.: "Self-consuming generative models go MAD", *Proc. 12th International Conference on Learning Representations*, Vienna, Austria, May 2024 (cited on p. 22).

[8] J. B. Allen and D. A. Berkley: "Image method for efficiently simulating small-room acoustics", *Journal of the Acoustical Society of America*, Vol. 65, No. 4, 943–950, 1979 (cited on p. 68).

[9] T. Alt, P. Peter, and J. Weickert: "Learning sparse masks for diffusion-based image inpainting", In A. J. Pinho, P. Georgieva, L. F. Teixeira, and J. A. Sánchez (Eds.): *Pattern Recognition and Image Analysis*, Lecture Notes in Computer Science, Vol. 13256, 528–239, Springer, Cham, 2022 (cited on pp. 75, 79, 92, 93).

[10] T. Alt, P. Peter, J. Weickert, and K. Schrader: "Translating numerical concepts for PDEs into neural architectures", In A. Elmoataz, J. Fadili, Y. Quéau, J. Rabin, and L. Simon (Eds.): *Scale Space and Variational Methods in Computer Vision*, Lecture Notes in Computer Science, Vol. 12679, 294–306, Springer, Cham, 2021 (cited on pp. 2, 113).

[11] T. Alt, K. Schrader, M. Augustin, P. Peter, and J. Weickert: "Connections between numerical algorithms for PDEs and neural networks", *Journal of Mathematical Imaging and Vision*, Vol. 65, 185–208, June 2022 (cited on pp. 2, 34, 41, 42, 43, 44, 86, 87, 89, 113).

[12] T. Alt, K. Schrader, J. Weickert, P. Peter, and M. Augustin: "Designing rotationally invariant neural networks from PDEs and variational methods", *Research in the Mathematical Sciences*, Vol. 9, No. 3, Article 52, Sept. 2022 (cited on pp. 34, 35, 48, 113).

[13] F. Andreu, C. Ballester, V. Caselles, and J. M. Mazón: "Minimizing total variation flow", *Differential and Integral Equations*, Vol. 14, No. 3, 321–360, Mar. 2001 (cited on p. 11).

[14] S. Andris, P. Peter, R. M. K. Mohideen, J. Weickert, and S. Hoffmann: "Inpainting-based video compression in FullHD", In A. Elmoataz, J. Fadili, Y. Quéau, J. Rabin, and L. Simon (Eds.): *Scale Space and Variational Methods in Computer Vision*, Lecture Notes in Computer Science, Vol. 12679, 425–436, Springer, Cham, 2021 (cited on p. 108).

[15] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik: "Contour detection and hierarchical image segmentation", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 33, No. 5, 898–916, Aug. 2011 (cited on pp. 82, 92, 93, 95, 96, 104).

[16] D. Bahdanau, K. Cho, and Y. Bengio: "Neural machine translation by jointly learning to align and translate", *Proc. 2nd International Conference on Learning Representations*, Y. Bengio and Y. LeCun (Ed.), Banff, Canada, Apr. 2014 (cited on p. 22).

[17] V. Bastani, M. Helfroush, and K. Kasiri: "Image compression based on spatial redundancy removal and image inpainting", *Journal of Zhejiang University – Science C (Computers & Electronics)*, Vol. 11, No. 2, 92–100, 2010 (cited on p. 75).

[18] A. Bauer, S. Trapp, M. Stenger, R. Leppich, S. Kounev, et al.: *Comprehensive Exploration of Synthetic Data Generation: A Survey*, arXiv:2401.02524 [cs.LG], Jan. 2024 (cited on p. 23).

[19] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind: "Automatic differentiation in machine learning: a survey", *Journal of Machine Learning Research*, Vol. 18, No. 153, 1–43, 2017 (cited on pp. 15, 16, 63, 65).

[20] Z. Belhachmi, D. Bucur, B. Burgeth, and J. Weickert: "How to choose interpolation data in images", *SIAM Journal on Applied Mathematics*, Vol. 70, No. 1, 333–352, 2009 (cited on pp. 76, 77, 83, 84, 92, 94, 95, 96, 97, 98, 99, 102, 104, 107, 108).

[21] Y. Bengio, P. Simard, and P. Frasconi: "Learning long-term dependencies with gradient descent is difficult", *IEEE Transactions on Neural Networks*, Vol. 5, No. 2, 157–166, Mar. 1994 (cited on p. 21).

[22] M. Bertalmío, G. Sapiro, V. Caselles, and C. Ballester: "Image inpainting", *Proc. SIGGRAPH 2000*, 417–424, New Orleans, LI, July 2000 (cited on pp. 48, 75).

[23] M. Bertero, T. A. Poggio, and V. Torre: "Ill-posed problems in early vision", *Proceedings of the IEEE*, Vol. 76, No. 8, 869–889, Aug. 1988 (cited on p. 12).

[24] D. Bertoin, J. Bolte, S. Gerchinovitz, and E. Pauwels: "Numerical influence of ReLU'(0) on backpropagation", *Proc. 35th International Conference on Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan (Ed.), Vol. 34, Advances in Neural Information Processing Systems, 468–479, Nov. 2021 (cited on p. 17).

[25] Å. Björck: *Numerical methods for least squares problems*, SIAM, Philadelphia, 1996 (cited on p. 79).

[26] S. Bonettini, I. Loris, F. Porta, M. Prato, and S. Rebegoldi: "On the convergence of a linesearch based proximal-gradient method for nonconvex optimization", *Inverse Problems*, Vol. 33, No. 5, Article no. 055005, Mar. 2017 (cited on pp. 76, 78, 80).

[27] D. Breunig: *Does the Bitter Lesson Have Limits?*, https://www.dbreunig.com/2025/08/01/does-the-bitter-lesson-have-limits.html, Accessed September 20, 2025, Aug. 2025 (cited on pp. 22, 23).

[28] M. Breuß, L. Hoeltgen, and G. Radow: "Towards PDE-based video compression with optimal masks prolongated by optic flow", *Journal of Mathematical Imaging and Vision*, Vol. 63, No. 2, 144–156, July 2021 (cited on p. 75).

[29] C. Brito-Loeza and K. Chen: "Fast numerical algorithms for Euler's elastica inpainting model", *International Journal of Modern Mathematics*, Vol. 5, No. 2, 157–182, 2010 (cited on p. 48).

[30] J. Brokman, A. Giloni, O. Hofman, R. Vainshtein, H. Kojima, et al.: "Identifying memorization of diffusion models through p-laplace analysis", In T. Bubba, R. Gaburro, S. Gazzola, K. Papafitsoros, M. Pereyra, et al. (Eds.): *Scale Space and Variational Methods in Computer Vision*, Lecture Notes in Computer Science, Vol. 15667, 295–307, Springer, Cham, 2025 (cited on p. 1).

[31] J. Brokman, A. Giloni, O. Hofman, R. Vainshtein, H. Kojima, et al.: "Manifold induced biases for zero-shot and few-shot detection of generated images", *Proc. 2025 International Conference on Learning Representations*, Singapore, Apr. 2025 (cited on p. 1).

[32] S. Carlsson: "Sketch based coding of grey level images", *Signal Processing*, Vol. 15, 57–83, 1988 (cited on pp. 3, 75, 82).

[33] F. Catté, P.-L. Lions, J.-M. Morel, and T. Coll: "Image selective smoothing and edge detection by nonlinear diffusion", *SIAM Journal on Numerical Analysis*, Vol. 32, 1895–1909, 1992 (cited on p. 24).

[34] A. Chambolle and T. Pock: "Total roto-translational variation", *Numerische Mathematik*, Vol. 142, 611–666, Mar. 2019 (cited on pp. 48, 58, 60).

[35] P. Charbonnier, L. Blanc-Féraud, G. Aubert, and M. Barlaud: "Two deterministic half-quadratic regularization algorithms for computed imaging", *Proc. 1994 IEEE International Conference on Image Processing*, Vol. 2, 168–172, Austin, TX, Nov. 1994 (cited on p. 24).

[36] L. Chen, P. Wu, K. Chitta, B. Jaeger, A. Geiger, et al.: "End-to-end autonomous driving: challenges and frontiers", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 46, No. 12, 10164–10183, Dec. 2024 (cited on p. 22).

[37] X. Chen, X. Luo, Y. Zaho, S. Zhang, G. Wang, et al.: *Learning Euler's elastica model for medical image segmentation*, arXiv:2011.00526 [eess.IV], Nov. 2020 (cited on p. 49).

[38] Y. Chen, R. Ranftl, and T. Pock: "A bi-level view of inpainting-based image compression", *Proc. 19th Computer Vision Winter Workshop*, 19–26, Křtiny, Czech Republic, Feb. 2014 (cited on pp. 76, 78, 79, 80).

[39] M. Cheon, S.-J. Yoon, B. Kang, and J. Lee: "Perceptual image quality assessment with transformers", *Proc. 2021 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 433–442, Nashville, TN, June 2021 (cited on p. 10).

[40] V. Chizhov and J. Weickert: "Efficient data optimisation for harmonic inpainting with finite elements", In N. Tsapatsoulis, A. Panayides, T. Theo, A. Lanitis, and C. Pattichis (Eds.): *Computer Analysis of Images and Patterns. Part 2*, Lecture Notes in Computer Science, Vol. 13053, 432–441, Springer, Cham, 2021 (cited on pp. 76, 78, 80, 105).

[41] M. Cobos, F. Antonacci, A. Alexandridis, A. Mouchtaris, and B. Lee: "A survey of sound source localization methods in wireless acoustic sensor networks", *Wireless Communications and Mobile Computing*, No. 1, Article no. 3956282, 2017 (cited on p. 62).

[42] D. Colton and R. Kress: *Inverse Acoustic and Electromagnetic Scattering Theory*, Applied Mathematical Sciences, Vol. 93, Springer, Cham, 2019 (cited on p. 63).

[43] C. Cortes and V. Vapnik: "Support-vector networks", *Machine Learning*, Vol. 28, 273–297, Sept. 1995 (cited on p. 11).

[44] M. R. Costa-jussà, J. Cross, O. Çelebi, M. Elbayad, K. Heafield, et al.: "Scaling neural machine translation to 200 languages", *Nature*, Vol. 630, 841–46, June 2024 (cited on p. 1).

[45] G. Cybenko: "Approximation by superpositions of a sigmoidal function", *Mathematics of Control, Signals and Systems*, Vol. 2, 303–314, 1989 (cited on p. 17).

[46] Q. Dai, H. Chopp, E. Pouyet, O. Cossairt, M. Walton, et al.: "Adaptive image sampling using deep learning and its application on X-Ray fluorescence image reconstruction", *IEEE Transactions on Multimedia*, Vol. 22, No. 10, 2564–2578, Dec. 2019 (cited on pp. 76, 79, 81, 90, 93, 104).

[47] V. Daropoulos, M. Augustin, and J. Weickert: "Sparse inpainting with smoothed particle hydrodynamics", *SIAM Journal on Applied Mathematics*, Vol. 14, No. 4, 1669–1704, Nov. 2021 (cited on pp. 76, 78, 80).

[48] I. Daubechies, R. DeVore, S. Foucart, B. Hanin, and G. Petrova: "Nonlinear approximation and (deep) ReLU networks", *Constructive Approximation*, Vol. 55, 127–172, Apr. 2021 (cited on p. 17).

[49] Y. N. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, et al.: "Identifying and attacking the saddle point problem in high-dimensional non-convex optimization", *Proc. 27nd International Conference on Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger (Ed.), Vol. 2, Advances in Neural Information Processing Systems, 2933–2941, Montréal, Canada, Dec. 2014 (cited on p. 14).

[50] L. Demaret, N. Dyn, and A. Iske: "Image compression by linear splines over adaptive triangulations", *Signal Processing*, Vol. 86, No. 7, 1604–1616, 2006 (cited on pp. 76, 78).

[51] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, et al.: "Imagenet: A large-scale hierarchical image database", *Proc. 2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 248–255, Miami, FL, Aug. 2009 (cited on pp. 90, 104).

[52] U. Y. Desai, M. M. Mizuki, I. Masaki, and B. K. P. Horn: *Edge and mean based image compression*, Technical Report No. 1584 (A.I. Memo), Artificial Intelligence Lab., Massachusetts Institute of Technology, Cambridge, MA, Nov. 1996 (cited on p. 75).

[53] R. Distasi, M. Nappi, and S. Vitulano: "Image compression by B-tree triangular coding", *IEEE Transactions on Communications*, Vol. 45, No. 9, 1095–1100, Sept. 1997 (cited on p. 78).

[54] S. Dittmer, T. Kluth, P. Maass, and D. Otero Baguer: "Regularization by architecture: a deep prior approach for inverse problems", *Journal of Mathematical Imaging and Vision*, Vol. 62, No. 3, 456–470, 2020 (cited on pp. 46, 49, 52).

[55] S. Dong and N. Ni: "A method for representing periodic functions and enforcing exactly periodic boundary conditions with deep neural networks", *Journal of Computational Physics*, Vol. 435, Article no. 110242, June 2021 (cited on p. 48).

[56] J. Duchi, E. Hazan, and Y. Singer: "Adaptive subgradient methods for online learning and stochastic optimization", *Journal of Machine Learning Research*, Vol. 12, 2121–2159, July 2011 (cited on p. 15).

[57] W. E: "A proposal on machine learning via dynamical systems", *Communications in Mathematics and Statistics*, Vol. 5, 1–11, Mar. 2017 (cited on p. 47).

[58] W. E, J. Han, and A. Jentzen: "Algorithms for solving high dimensional PDEs: from nonlinear Monte Carlo to machine learning", *Nonlinearity*, Vol. 35, No. 1, 278–310, Dec. 2021 (cited on p. 47).

[59] W. E, J. Han, and A. Jentzen: "Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations", *Communications in Mathematics and Statistics*, Vol. 5, 349–380, Nov. 2017 (cited on p. 47).

[60] W. E and B. Yu: "The deep Ritz method: a deep learning-based numerical algorithm for solving variational problems", *Communications in Mathematics and Statistics*, Vol. 6, 1–12, Feb. 2018 (cited on p. 47).

[61] A. A. Efros and T. Leung: "Texture synthesis by non-parametric sampling", *Proc. Seventh International Conference on Computer Vision*, Vol. 2, 1033–1038, Kerkyra, Greece, Sept. 1999 (cited on pp. 48, 75).

[62] R. Eldan and O. Shamir: "The power of depth for feedforward neural networks", *Proc. 29th Conference on Learning Theory*, V. Feldman, A. Rakhlin, and O. Shamir (Ed.), 907–940, New York, NY, June 2016 (cited on p. 17).

[63] M. Elgendy: *Deep Learning for Vision Systems*, Manning, Shelter Island, NY, 2020 (cited on pp. 1, 11).

[64] N. B. Erichson, M. Muehlebach, and M. W. Mahoney: "Physics-informed autoencoders for Lyapunov-stable fluid flow prediction", *Proc. NeurIPS 2019 Machine Learning and the Physical Sciences*, Vancouver, Canada, 2019 (cited on p. 47).

[65] M. S. Eshaghi, C. Anitescu, M. Thombre, Y. Wang, X. Zhuang, et al.: "Variational physics-informed neural operator (VINO) for solving partial differential equations", *Computer Methods in Applied Mechanics and Engineering*, Vol. 437, Article no. 117785, Mar. 2025 (cited on p. 47).

[66] L. Euler: *Methodus inveniendi lineas curvas maximi minimive proprietate gaudentes, sive solutio problematis isoperimetrici latissimo sensu accepti*, Marc-Michel Bousquet et Cie., Lausanne, 1744 (cited on p. 48).

[67] A. Fick: "Ueber Diffusion", *Annalen der Physik*, Vol. 170, No. 1, 59–86, Apr. 1855 (cited on p. 23).

[68] R. W. Floyd and L. Steinberg: "An adaptive algorithm for spatial grey scale", *Proceedings of the Society of Information Display*, Vol. 17, 75–77, 1976 (cited on pp. 83, 92, 99).

[69] J. Fourier: *Théorie analytique de la chaleur*, Didot, Paris, 1822 (cited on p. 23).

[70] K. Fukushima: "A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position", *Biological Cybernetics*, Vol. 36, 193–202, 1980 (cited on pp. 16, 18).

[71] I. Galić, J. Weickert, M. Welk, A. Bruhn, A. Belyaev, et al.: "Image compression with anisotropic diffusion", *Journal of Mathematical Imaging and Vision*, Vol. 31, No. 2–3, 255–269, July 2008 (cited on pp. 11, 75, 78, 81, 82).

[72] I. Galić, J. Weickert, M. Welk, A. Bruhn, A. Belyaev, et al.: "Towards PDE-based image compression", In N. Paragios, O. Faugeras, T. Chan, and C. Schnörr (Eds.): *Variational, Geometric and Level-Set Methods in Computer Vision*, Lecture Notes in Computer Science, Vol. 3752, 37–48, Springer, Berlin, 2005 (cited on p. 75).

[73] J. Gautier, O. Le Meur, and C. Guillemot: "Efficient depth map compression based on lossless edge coding and diffusion", *Proc. 2012 Picture Coding Symposium*, 81–84, Kraków, Poland, May 2012 (cited on p. 75).

[74] S. Gerschgorin: "Fehlerabschätzung für das Differenzenverfahren zur Lösung Partieller Differentialgleichungen", *Zeitschrift für Angewandte Mathematik und Mechanik*, Vol. 10, 373–382, 1930 (cited on p. 7).

[75] G. Gilboa, N. A. Sochen, and Y. Y. Zeevi: "Forward-and-backward diffusion processes for adaptive image enhancement and denoising", *IEEE Transactions on Image Processing*, Vol. 11, No. 7, 689–703, Nov. 2002 (cited on p. 24).

[76] X. Glorot and Y. Bengio: "Understanding the difficulty of training deep feedforward neural networks", *Proc. 13th International Conference on Artificial Intelligence and Statistics*, 249–256, Sardinia, Italy, May 2010 (cited on p. 17).

[77] A. Golts, D. Freedman, and M. Elad: "Deep energy: task driven training of deep neural networks", *IEEE Journal of Selected Topics in Signal Processing*, Vol. 15, No. 2, 324–338, Feb. 2021 (cited on pp. 46, 47, 48, 49, 52, 87).

[78] I. J. Goodfellow, Y. Bengio, and A. Courville: *Deep Learning*, MIT Press, Cambridge, MA, 2016 (cited on pp. 1, 11, 13, 48).

[79] I. J. Goodfellow, J. Shlens, and C. Szegedy: "Explaining and harnessing adversarial examples", *Proc. 3rd International Conference on Learning Representations*, Y. Bengio and Y. LeCun (Ed.), San Diego, CA, May 2015 (cited on p. 1).

[80] Google DeepMind: *Veo 3 Technical Report*, Retrieved September 20, 2025 from `https://storage.googleapis.com/deepmind-media/veo/Veo-3-Tech-Report.pdf`, 2025 (cited on p. 22).

[81] A. Griewank: “On automatic differentiation”, In M. Iri and K. Tanabe (Eds.): *Mathematical Programming: Recent Developments and Applications*, Mathematics and its Applications, Vol. 6, 83–108, Kluwer, Amsterdam, 1989 (cited on p. 15).

[82] K. Gruizenga: *PlotNeuralNet: A Python package for generating neural network architecture diagrams*, Retrieved September 22, 2025 from https://github.com/kgruiz/PlotNeuralNet, 2025 (cited on p. 20).

[83] C. Guillemot and O. Le Meur: “Image inpainting: overview and recent advances”, *IEEE Signal Processing Magazine*, Vol. 31, No. 1, 127–144, 2014 (cited on pp. 48, 75).

[84] D. Hafner, P. Ochs, J. Weickert, M. Reißel, and S. Grewenig: “FSI schemes: fast semi-iterative solvers for PDEs and optimisation methods”, In B. Rosenhahn and B. Andres (Eds.): *Pattern Recognition*, Lecture Notes in Computer Science, Vol. 9796, 91–102, Springer, Cham, 2016 (cited on p. 44).

[85] J. Han, A. Jentzen, and W. E: “Solving high-dimensional partial differential equations using deep learning”, *Proceedings of the National Academy of Sciences*, Vol. 115, No. 34, 8505–8510, Aug. 2018 (cited on p. 47).

[86] K. He, X. Zhang, S. Ren, and J. Sun: “Deep residual learning for image recognition”, *Proc. 2016 IEEE Conference on Computer Vision and Pattern Recognition*, 770–778, Las Vegas, NV, June 2016 (cited on pp. 21, 34, 41, 86).

[87] K. He, X. Zhang, S. Ren, and J. Sun: “Identity mappings in deep residual networks”, In B. Leibe, J. Matas, N. Sebe, and M. Welling (Eds.): *Computer Vision – ECCV 2016*, Lecture Notes in Computer Science, Vol. 9912, 630–645, Springer, Cham, 2016 (cited on p. 21).

[88] G. Hinton: *Neural networks for machine learning: lecture 6*, Coursera Lecture, Retrieved August 29, 2025 from http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf, 2012 (cited on p. 15).

[89] P. Hitzler and M. K. Sarker: *Neuro-symbolic artificial intelligence: The state of the art*, IOS press, Amsterdam, Netherlands, 2022 (cited on p. 2).

[90] L. Hoeltgen, M. Mainberger, S. Hoffmann, J. Weickert, C. H. Tang, et al.: “Optimising spatial and tonal data for PDE-based inpainting”, In M. Bergounioux, G. Peyré, C. Schnörr, J.-P. Caillau, and T. Haberkorn (Eds.): *Variational Methods in Imaging and Geometric Control*, Radon Series on Computational and Applied Mathematics, Vol. 18, 35–83, De Gruyter, Berlin, 2017 (cited on pp. 76, 78, 79, 80).

[91] L. Hoeltgen, S. Setzer, and J. Weickert: "An optimal control approach to find sparse data for Laplace interpolation", In A. Heyden, F. Kahl, C. Olsson, M. Oskarsson, and X.-C. Tai (Eds.): *Energy Minimisation Methods in Computer Vision and Pattern Recognition*, Lecture Notes in Computer Science, Vol. 8081, 151–164, Springer, Berlin, 2013 (cited on pp. 76, 78).

[92] L. Hoeltgen and J. Weickert: "Why does non-binary mask optimisation work for diffusion-based image compression?", In X.-C. Tai, E. Bae, T. F. Chan, S. Y. Leung, and M. Lysaker (Eds.): *Energy Minimisation Methods in Computer Vision and Pattern Recognition*, Lecture Notes in Computer Science, Vol. 8932, 85–98, Springer, Berlin, 2015 (cited on pp. 76, 78, 80, 83).

[93] S. Hoffmann: "Competitive Image Compression with Linear PDEs", PhD thesis, Department of Computer Science, Saarland University, Saarbrücken, Germany, 2017 (cited on pp. 80, 84, 94, 97, 98).

[94] S. Hoffmann, M. Mainberger, J. Weickert, and M. Puhl: "Compression of depth maps with segment-based homogeneous diffusion", In A. Kuijper, K. Bredies, T. Pock, and H. Bischof (Eds.): *Scale Space and Variational Methods in Computer Vision*, Lecture Notes in Computer Science, Vol. 7893, 319–330, Springer, Berlin, 2013 (cited on pp. 75, 76, 80).

[95] S. Hoffmann, G. Plonka, and J. Weickert: "Discrete Green's functions for harmonic and biharmonic inpainting with sparse atoms", In X.-C. Tai, E. Bae, T. F. Chan, and M. Lysaker (Eds.): *Energy Minimization Methods in Computer Vision and Pattern Recognition*, Lecture Notes in Computer Science, Vol. 8932, 169–182, Springer, Berlin, 2015 (cited on p. 80).

[96] M. Holschneider, R. Kronland-Martinet, J. Morlet, and P. Tchamitchian: "A real-time algorithm for signal analysis with the help of the wavelet transform", In J. M. Combes, A. Grossman, and P. Tchamitchian (Eds.): *Wavelets: Time-Frequency Methods and Phase Space*, 286–297, Springer, Berlin, 1989 (cited on p. 18).

[97] T. Iijima: "Basic theory of pattern observation", In: *Papers of Technical Group on Automata and Automatic Control*, In Japanese: IECE, Japan, 1959 (cited on p. 23).

[98] T. Iijima: "Basic theory on normalization of pattern (in case of typical one-dimensional pattern)", *Bulletin of the Electrotechnical Laboratory*, Vol. 26, In Japanese, 368–388, Jan. 1962 (cited on pp. 11, 23, 76, 82).

[99] T. Iijima: "Theory of pattern recognition", *Electronics and Communications in Japan*, In English, 123–134, Nov. 1963 (cited on p. 23).

[100] Y. Ito, T. Nakamura, S. Koyama, and H. Saruwatari: "Head-related transfer function interpolation from spatially sparse measurements using autoencoder with source position conditioning", *Proc. 17th International Workshop on Acoustic Signal Enhancement*, Bamberg, Germany, Sept. 2022 (cited on p. 63).

[101] A. D. Jagtap, E. Kharazmi, and G. E. Karniadakis: "Conservative physics-informed neural networks on discrete domains for conservation laws: applications to forward and inverse problems", *Computer Methods in Applied Mechanics and Engineering*, Vol. 365, Article no. 113028, June 2020 (cited on p. 47).

[102] A. Jagtap and G. Karniadakis: "Extended physics-informed neural networks (XPINNs): a generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations", *Communications in Computational Physics*, Vol. 28, 2002–2041, 2020 (cited on p. 47).

[103] F. Jost, V. Chizhov, and J. Weickert: "Optimising different feature types for inpainting-based image representations", *Proc. 2023 IEEE International Conference on Acoustics, Speech, and Signal Processing*, Rhodes, Greece, June 2023 (cited on p. 78).

[104] F. Jost, P. Peter, and J. Weickert: "Compressing flow fields with edge-aware homogeneous diffusion inpainting", *Proc. 2020 International Conference on Acoustics, Speech, and Signal Processing*, 2198–2202, Barcelona, Spain, May 2020 (cited on p. 76).

[105] F. Jost, P. Peter, and J. Weickert: "Compressing piecewise smooth images with the Mumford–Shah cartoon model", *Proc. 28th European Signal Processing Conference*, 511–515, Amsterdam, Netherlands, Jan. 2021 (cited on pp. 76, 80).

[106] I. Jumakulyyev and T. Schultz: "Fourth-order anisotropic diffusion for inpainting and image compression", In E. Özarslan, T. Schultz, E. Zhang, and A. Fuster (Eds.): *Anisotropy Across Fields and Scales*, Mathematics and Visualization, 99–124, Springer, Cham, 2021 (cited on pp. 25, 82).

[107] I. Jumakulyyev and T. Schultz: "Lossless PDE-based compression of 3D medical images", In A. Elmoataz, J. Fadili, Y. Quéau, J. Rabin, and L. Simon (Eds.): *Scale Space and Variational Methods in Computer Vision*, Lecture Notes in Computer Science, Vol. 12679, 450–462, Springer, Cham, 2021 (cited on p. 75).

[108] A. T. Kalai, O. Nachum, S. S. Vempala, and E. Zhang: "Why language models hallucinate" Sept. 2025 (cited on p. 1).

[109] N. Kämper, V. Chizhov, and J. Weickert: *Efficient Parallel Algorithms for Inpainting-Based Representations of 4K Images - Part I: Homogeneous Diffusion Inpainting*, arXiv:2401.06744 [eess.IV], Jan. 2024 (cited on pp. 78, 80).

[110] N. Kämper, V. Chizhov, and J. Weickert: "Efficient parallel data optimization for homogeneous diffusion inpainting of 4k images", *SIAM Journal on Imaging Sciences*, Vol. 18, No. 1, 701–734, Mar. 2025 (cited on pp. 78, 80).

[111] N. Kämper and J. Weickert: "Domain decomposition algorithms for real-time homogeneous diffusion inpainting in 4K", *Proc. 2022 IEEE International Conference on Acoustics, Speech and Signal Processing*, 1680–1684, Singapore, May 2022 (cited on pp. 24, 99).

[112] S. H. Kang, X.-C. Tai, and W. Zhu: "Survey of fast algorithms for Euler's elastica-based image segmentation", In R. Kimmel and X.-C. Tai (Eds.): *Processing, Analyzing and Learning of Images, Shapes, and Forms: Part 2*, Handbook of Numerical Analysis, Vol. 20, chap. 13, 533–552, Elsevier, Amsterdam, 2019 (cited on p. 48).

[113] G. Kanizsa: *Organization in Vision: Essays on Gestalt Perception*, Praeger, New York, 1979 (cited on p. 50).

[114] G. E. Karniadakis, I. G. Kevrekidis, P. P. L. Lu, S. Wang, and L. Yang: "Physics-informed machine learning", *Nature Reviews Physics*, Vol. 3, No. 6, 422–440, June 2021 (cited on pp. 2, 46, 62, 63).

[115] L. Karos, P. Bheed, P. Peter, and J. Weickert: "Optimising data for exemplar-based inpainting", In J. Blanc-Talon, D. Helbert, W. Philips, D. Popescu, and P. Scheunders (Eds.): *Advanced Concepts for Intelligent Vision Systems*, Lecture Notes in Computer Science, Vol. 11182, 547–558, Springer, Cham, 2018 (cited on pp. 76, 78).

[116] M. A. Khan, H. E. Sayed, S. Malik, T. Zia, J. Khan, et al.: "Level-5 autonomous driving—are we there yet? a review of research literature", *ACM Computing Surverys*, Vol. 55, No. 2, Article no. 27, Jan. 2022 (cited on p. 1).

[117] E. Kharazmi, Z. Zhang, and G. E. Karniadakis: "Hp-VPINNs: variational physics-informed neural networks with domain decomposition", *Computer Methods in Applied Mechanics and Engineering*, Vol. 374, Article no. 113547, Feb. 2021 (cited on p. 47).

[118] S. Kichenassamy: "The Perona–Malik paradox", *SIAM Journal on Applied Mathematics*, Vol. 57, 1343–1372, 1997 (cited on p. 24).

[119] D. P. Kingma and J. Ba: "Adam: A method for stochastic optimization", *Proc. 3rd International Conference on Learning Representations*, San Diego, CA, May 2015 (cited on pp. 15, 55, 92, 104).

[120] J. J. Koenderink: "The structure of images", *Biological Cybernetics*, Vol. 50, 363–370, 1984 (cited on p. 23).

[121] K. Kontolati, S. Goswami, G. E. Karniadakis, and M. D. Shields: "Using goal-driven deep learning models to understand sensory cortex", *Nature Communications*, Vol. 15, No. 1, Article no. 5101, June 2024 (cited on p. 47).

[122] H. Köstler, M. Stürmer, C. Freundl, and U. Rüde: *PDE based video compression in real time*, Technical Report No. 07-11, Lehrstuhl für Informatik 10, Univ. Erlangen–Nürnberg, Germany, 2007 (cited on p. 108).

[123] S. Koyama and L. Daudet: "Sparse representation of a spatial sound field in a reverberant environment", *IEEE Journal of Selected Topics in Signal Processing*, Vol. 13, No. 1, 172–184, 2019 (cited on p. 62).

[124] S. Koyama and K. Ishizuka: "Learning magnitude distribution of sound fields via conditioned autoencoder", *Proc. 11th Forum Acusticum*, European Acoustics Association, Málaga, Spain, June 2025 (cited on pp. 31, 62, 63).

[125] S. Koyama, J. G. C. Ribeiro, T. Nakamura, N. Ueno, and M. Pezzoli: "Physics-informed machine learning for sound field estimation: fundamentals, state of the art, and challenges", *IEEE Signal Processing Magazine*, Vol. 41, No. 6, 60–71, 2025 (cited on pp. 30, 31, 62, 63).

[126] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum: "Human-level concept learning through probabilistic program induction", *Science*, Vol. 350, 1332–1338, 2015 (cited on p. 22).

[127] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner: "Gradient-based learning applied to document recognition", *Proceedings of the IEEE*, Vol. 86, No. 11, 2278–2324, Nov. 1998 (cited on pp. 16, 18).

[128] X. Li, J. Deng, J. Wu, S. Zhang, W. Li, et al.: "Physical informed neural networks with soft and hard boundary constraints for solving advection-diffusion equations using Fourier expansions", *Computers & Mathematics with Applications*, Vol. 159, No. 3, 60–75, Apr. 2024 (cited on p. 47).

[129] Y. Li, M. Sjostrom, U. Jennehag, and R. Olsson: "A scalable coding approach for high quality depth image compression", *Proc. 3DTV-Conference: The True Vision – Capture, Transmission and Display of 3D Video*, Zurich, Switzerland, Oct. 2012 (cited on p. 75).

[130] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharaya, et al.: "Fourier neural operator for parametric partial differential equations", *Proc. 9th International Conference on Learning Representations*, Vienna, Austria, May 2021 (cited on p. 47).

[131] Z. Li, H. Zheng, N. Kovachki, D. Jin, H. Chen, et al.: "Physics-informed neural operator for learning partial differential equations", *ACM/IMS Journal of Data Science*, Vol. 1, No. 3, Article no. 9, May 2024 (cited on p. 47).

[132] T. Lin, H. Xue, L. Wang, B. Huang, and H. Zha: "Supervised learning via Euler's elastica models", *Journal of Machine Learning Research*, Vol. 16, No. 1, 1532–4435, Jan. 2015 (cited on p. 49).

[133] S. Linnainmaa: "The representation of the cumulative rounding error of an algorithm as a Taylor expansion of the local rounding errors", MA thesis, University of Helsinki, Finland, 1970 (cited on p. 15).

[134] G. Linscott, G.-C. Pascutto, and LeelaChessZero Development Team: *Leela Chess Zero v0.32: Open Source Neural Network Chess Engine*, 2025 (cited on p. 23).

[135] T. Linzen: "How can we accelerate progress towards human-like linguistic generalization?", *Proc. 58th Annual Meeting of the Association for Computational Linguistics*, D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault (Ed.), 5210–5217, Association for Computational Linguistics, July 2020 (cited on p. 22).

[136] H. Liu, B. Jiang, Y. Xiao, and C. Yang: "Coherent semantic attention for image inpainting", *Proc. 2019 IEEE/CVF International Conference on Computer Vision*, 4170–4179, Seoul, Korea, Oct. 2019 (cited on p. 81).

[137] F. Lluís, P. Martínez-Nuevo, M. B. Møller, and S. E. Shepstone: "Sound field reconstruction in rooms: inpainting meets super-resolution", *Journal of the Acoustical Society of America*, Vol. 148, No. 2, 649–659, Aug. 2020 (cited on pp. 62, 63).

[138] I. Loshchilov and F. Hutter: "Decoupled weight decay regularization", *Proc. 7th International Conference on Learning Representations*, New Orleans, LA, May 2019 (cited on p. 68).

[139] L. Lu, P. Jin, G. Pang, Z. Zhang, and G. E. Karniadakis: "Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators", *Nature Machine Intelligence*, Vol. 3, 218–229, Mar. 2021 (cited on p. 47).

[140] S. M. Lundberg and S.-I. Lee: "A unified approach to interpreting model predictions", *Proc. 31st International Conference on Neural Information Processing Systems*, I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, et al. (Ed.), Vol. 30, Advances in Neural Information Processing Systems, 4768–4777, Long Beach, CA, Dec. 2017 (cited on p. 1).

[141] A. Luo, Y. Du, M. J. Tarr, J. B. Tenenbaum, A. Torralba, et al.: "Learning neural acoustic fields", *Proc. 36th International Conference on Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, et al. (Ed.), Vol. 35, Advances in Neural Information Processing Systems, 3165–3177, Oct. 2022 (cited on pp. 62, 63).

[142] K. Luo, J. Zhao, Y. Wang, J. Li, J. Wen, et al.: "Physics-informed neural networks for PDE problems: a comprehensive review.", *Artificial Intelligence Review*, Vol. 58, Article no. 323, July 2025 (cited on p. 47).

[143] M. Mainberger, S. Hoffmann, J. Weickert, C. H. Tang, D. Johannsen, et al.: "Optimising spatial and tonal data for homogeneous diffusion inpainting", In A. M. Bruckstein, B. ter Haar Romeny, A. M. Bronstein, and M. M. Bronstein (Eds.): *Scale Space and Variational Methods in Computer Vision*, Lecture Notes in Computer Science, Vol. 6667, 26–37, Springer, Berlin, 2012 (cited on pp. 76, 77, 78, 79, 80, 81, 83, 84, 94, 95, 96, 97, 98, 99, 107).

[144] G. Marcus and E. Davis: *Rebooting AI: Building Artificial Intelligence We Can Trust*, Pantheon Books, 2019 (cited on p. 22).

[145] D. Marwood, P. Massimino, M. Covell, and S. Baluja: "Representing images in 200 bytes: compression via triangulation", *Proc 2018 IEEE International Conference on Image Processing*, 405–409, Athens, Greece, Oct. 2018 (cited on pp. 76, 79, 81).

[146] N. Maslej, L. Fattorini, R. Perrault, Y. Gil, V. Parli, et al.: *The AI index 2025 annual report*, Technical Report, Accessed September 20, 2025, Stanford University, Stanford, CA, Apr. 2025 (cited on p. 22).

[147] S. Masnou and J.-M. Morel: "Level lines based disocclusion", *Proc. 1998 IEEE International Conference on Image Processing*, Vol. 3, 259–263, Chicago, IL, Oct. 1998 (cited on pp. 48, 75).

[148] J. D. Maynard, E. G. Williams, and Y. Lee: "Nearfield acoustic holography: I. Theory of generalized holography and the development of NAH", *Journal of the Acoustical Society of America*, Vol. 78, No. 4, 1395–1413, 1985 (cited on p. 62).

[149] C. Molnar: *Interpretable Machine Learning, A Guide for Making Black Box Models Explainable*, 3rd Edition, https://christophm.github.io/interpretable-ml-book/, 2025 (cited on p. 1).

[150] K. W. Morton and L. M. Mayers: *Numerical Solution of Partial Differential Equations*, Second, Cambridge University Press, Cambridge, UK, 2005 (cited on p. 41).

[151] B. Moseley, A. Markham, and T. Nissen-Meyer: "Finite basis physics-informed neural networks (FBPINNs): a scalable domain decomposition approach for solving differential equations", *Advances in Computational Mathematics*, Vol. 49, No. 4, Article no. 62, July 2023 (cited on p. 47).

[152] P. Mrázek and M. Navara: "Consistent positive directional splitting of anisotropic diffusion", *Proc. Sixth Computer Vision Winter Workshop*, B. Likar (Ed.), 37–48, Bled, Slovenia, Feb. 2001 (cited on p. 34).

[153] S. Müller, J. Weickert, and N. Graf: "Wilms' tumor in childhood: can pattern recognition help for classification?", In Y. Zheng, B. M. Williams, and K. Chen (Eds.): *Medical Image Understanding and Analysis – MIUA 2020*, Communications in Computer and Information Science, Vol. 1065, 38–47, Springer, Cham, 2020 (cited on p. 22).

[154] D. Mumford: "Elastica and computer vision", In C. L. Bajaj (Ed.): *Algebraic Geometry and its Applications*, Vol. 5681, chap. 31, 491–506, Springer, New York, 1994 (cited on pp. 3, 48).

[155] K. P. Murphy: *Machine Learning: A Probabilistic Perspective*, MIT Press, 2012 (cited on p. 70).

[156] R. Nahme: "Inertial Proximal Algorithms in Diffusion-based Image Compression", MA thesis, Dept. of Mathematics, University of Göttingen, Germany, 2015 (cited on pp. 76, 78, 80).

[157] V. Nair and G. E. Hinton: "Rectified linear units improve restricted Boltzmann machines", *Proc. 27th International Conference on Machine Learning*, 807–814, Haifa, Israel, June 2010 (cited on p. 17).

[158] J. Nocedal and S. J. Wright: *Numerical Optimization*, Springer, New York, 2006 (cited on p. 15).

[159] P. Ochs, Y. Chen, T. Brox, and T. Pock: "IPiano: inertial proximal algorithm for nonconvex optimization", *SIAM Journal on Imaging Sciences*, Vol. 7, 1388–1419, 2014 (cited on pp. 76, 78, 80).

[160] G. S. Ohm: *Die galvanische Kette, mathematisch bearbeitet*, T. H. Riemann, Berlin, 1827 (cited on p. 23).

[161] M. Olivieri, X. Karakonstantis, M. Pezzoli, F. Antonacci, A. Sarti, et al.: "Physics-informed neural network for volumetric sound field reconstruction of speech signals", *EURASIP Journal on Audio, Speech, and Music Processing*, Article No. 42, Sept. 2024 (cited on pp. 62, 63).

[162] S. J. Pan and Q. Yang: "A survey on transfer learning", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 22, No. 10, 1345–1359, Oct. 2010 (cited on p. 23).

[163] M. Park and B. Rafaely: "Sound-field analysis by plane-wave decomposition using spherical microphone array", *Journal of the Acoustical Society of America*, Vol. 118, No. 5, 3094–3103, 2005 (cited on p. 62).

[164] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, et al.: "PyTorch: an imperative style, high-performance deep learning library", *Proc. 33rd International Conference on Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, et al. (Ed.), Vol. 32, Advances in Neural Information Processing Systems, 8026–8037, Vancouver, Canada, Dec. 2019 (cited on pp. 9, 16).

[165] D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, and A. A. Efros: "Context encoders: feature learning by inpainting", *Proc. 2016 IEEE Conference on Computer Vision and Pattern Recognition*, 2536–2544, Las Vegas, NV, June 2016 (cited on pp. 48, 49, 81).

[166] W. B. Pennebaker and J. L. Mitchell: *JPEG: Still Image Data Compression Standard*, Springer, New York, 1992 (cited on p. 108).

[167] P. Perona and J. Malik: "Scale space and edge detection using anisotropic diffusion", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 12, 629–639, July 1990 (cited on pp. 11, 24, 35).

[168] P. Peter: "A Wasserstein GAN for joint learning of inpainting and its spatial optimisation", In H. Wang, W. Lin, P. Manoranjan, G. Xiao, K. Chan, et al. (Eds.): *Image and Video Technology*, Lecture Notes in Computer Science, Vol. 13763, 132–145, Springer, Cham, 2023 (cited on pp. 75, 79, 81, 88, 93).

[169] P. Peter: "Fast inpainting-based compression: combining Shepard interpolation with joint inpainting and prediction", *Proc 2019 IEEE International Conference on Image Processing*, 3557–3561, Taipei, Taiwan, Sept. 2019 (cited on pp. 76, 80).

[170] P. Peter, J. Contelly, and J. Weickert: "Compressing audio signals with inpainting-based sparsification", In J. Lellmann, M. Burger, and J. Modersitzki (Eds.): *Scale Space and Variational Methods in Computer Vision*, Lecture Notes in Computer Science, Vol. 11603, 92–103, Springer, Cham, 2019 (cited on p. 80).

[171] P. Peter, S. Hoffmann, F. Nedwed, L. Hoeltgen, and J. Weickert: "Evaluating the true potential of diffusion-based inpainting in a compression context", *Signal Processing: Image Communication*, Vol. 46, 40–53, Aug. 2016 (cited on pp. 75, 76, 78, 81).

[172] P. Peter, L. Kaufhold, and J. Weickert: "Turning diffusion-based image colorization into efficient color compression", *IEEE Transactions on Image Processing*, Vol. 26, No. 2, 860–869, Feb. 2017 (cited on pp. 25, 75, 76, 80, 82).

[173] P. Peter, C. Schmaltz, N. Mach, M. Mainberger, and J. Weickert: "Beyond pure quality: progressive mode, region of interest coding and real time video decoding in PDE-based image compression", *Journal of Visual Communication and Image Representation*, Vol. 31, 256–265, Aug. 2015 (cited on p. 108).

[174] P. Peter, K. Schrader, T. Alt, and J. Weickert: "Deep spatial and tonal data optimisation for homogeneous diffusion inpainting", *Pattern Analysis and Applications*, Vol. 26, No. 4, 1585–1600, 2023 (cited on pp. 75, 99, 101, 104, 113).

[175] D. Podell, Z. English, K. Lacey, A. Blattmann, T. Dockhorn, et al.: "SDXL: improving latent diffusion models for high-resolution image synthesis", *Proc. 12th International Conference on Learning Representations*, Vienna, Austria, May 2024 (cited on p. 1).

[176] M. A. Poletti: "Three-dimensional surround sound systems based on spherical harmonics", *Journal of the Audio Engineering Society*, Vol. 53, No. 11, 1004–1025, 2005 (cited on p. 62).

[177] M. Raissi, P. Perdikaris, and G. E. Karniadakis: *Physics Informed Deep Learning (Part I): Data-driven Solutions of Nonlinear Partial Differential Equations*, arXiv:1711.10561v1 [cs.AI], Nov. 2017 (cited on p. 46).

[178] M. Raissi, P. Perdikaris, and G. E. Karniadakis: "Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations", *Journal of Computational Physics*, Vol. 378, 686–707, Feb. 2019 (cited on pp. 3, 46, 47, 62, 63, 64).

[179] J. G. C. Ribeiro, S. Koyama, R. Horiuchi, and H. Saruwatari: “Sound field estimation based on physics-constrained kernel interpolation adapted to environment”, *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, Vol. 32, 4369–4383, 2024 (cited on pp. 62, 63).

[180] M. T. Ribeiro, S. Singh, and C. Guestrin: “"Why should I trust you?": explaining the predictions of any classifier”, *Proc. 22st ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, B. Krishnapuram, M. Shah, A. J. Smola, C. C. Aggarwal, D. Shen, et al. (Ed.), 1135–1144, ACM, New York, NY, Aug. 2016 (cited on p. 1).

[181] T. Ringholm, J. Lazić, and C.-B. Schönlieb: “Variational image regularization with Euler’s elastica using a discrete gradient scheme”, *SIAM Journal on Imaging Sciences*, Vol. 11, No. 4, 2665–2691, 2018 (cited on p. 48).

[182] H. Robbins and S. Monro: “A stochastic approximation method”, *Annals of Mathematical Statistics*, Vol. 22, No. 3, 400–407, 1951 (cited on p. 14).

[183] R. T. Rockafellar: *Convex Analysis*, Princeton University Press, Princeton, 1970 (cited on p. 9).

[184] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer: “High-resolution image synthesis with latent diffusion models”, *Proc. 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10684–10695, New Orleans, LA, June 2022 (cited on pp. 1, 45).

[185] O. Ronneberger, P. Fischer, and T. Brox: “U-net: convolutional networks for biomedical image segmentation”, In N. Navab, J. Hornegger, W. Wells, and A. Frangi (Eds.): *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, Lecture Notes in Computer Science, Vol. 9351, 234–241, Springer, Cham, 2015 (cited on pp. 19, 20, 53, 86, 89).

[186] F. Rosenblatt: *Principles of Neurodynamics*, Spartan, New York, 1962 (cited on p. 16).

[187] F. Rosenblatt: “The perceptron: a probabilistic model for information storage and organization in the brain”, *Psychological Review*, Vol. 65, No. 6, 386–408, 1958 (cited on p. 16).

[188] F. Rousseau, L. Drumetz, and R. Fablet: “Residual networks as flows of diffeomorphisms”, *Journal of Mathematical Imaging and Vision*, Vol. 62, 365–375, Apr. 2020 (cited on p. 34).

[189] A. Roussos and P. Maragos: “Tensor-based image diffusions derived from generalizations of the total variation and Beltrami functionals”, *Proc. 17th IEEE International Conference on Image Processing*, 4141–4144, Hong Kong, Sept. 2010 (cited on p. 25).

[190] L. I. Rudin, S. Osher, and E. Fatemi: "Nonlinear total variation based noise removal algorithms", *Physica D*, Vol. 60, No. 1–4, 259–268, Nov. 1992 (cited on p. 50).

[191] D. E. Rumelhart, G. E. Hinton, and R. J. Williams: "Learning representations by back-propagating errors", *Nature*, Vol. 323, No. 9, 533–536, Oct. 1986 (cited on pp. 16, 52).

[192] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, et al.: "ImageNet large scale visual recognition challenge", *International Journal of Computer Vision*, Vol. 115, 211–252, Apr. 2015 (cited on p. 21).

[193] L. Ruthotto and E. Haber: "Deep neural networks motivated by partial differential equations", *Journal of Mathematical Imaging and Vision*, Vol. 62, 352–364, Apr. 2020 (cited on pp. 34, 43).

[194] J. C. A. Sánchez, L. Comanducci, M. Pezzoli, and F. Antonacci: "Towards HRTF personalization using denoising diffusion models", *Proc. 2025 IEEE International Conference on Acoustics, Speech and Signal Processing*, Hyderabad, India, May 2025 (cited on p. 63).

[195] K. Schaefer and J. Weickert: "Stabilised inverse flowline evolution for anisotropic image sharpening", *Proc. 10th European Workshop on Visual Information Processing*, Lisbon, Portugal, Sept. 2022 (cited on p. 41).

[196] H. Scharr, M. J. Black, and H. W. Haussecker: "Image statistics and anisotropic diffusion", *Proc. Ninth International Conferance on Computer Vision*, Vol. 2, 840–847, Nice, France, 2003 (cited on p. 25).

[197] R. Scheibler, E. Bezzam, and I. Dokmanić: "Pyroomacoustics: a Python package for audio room simulation and array processing algorithms", *Proc. 2018 IEEE International Conference on Acoustics, Speech and Signal Processing*, 351–355, Calgary, Canada, Apr. 2018 (cited on p. 68).

[198] C. Schmaltz, P. Peter, M. Mainberger, F. Ebel, J. Weickert, et al.: "Understanding, optimising, and extending data compression with anisotropic diffusion", *International Journal of Computer Vision*, Vol. 108, No. 3, 222–240, July 2014 (cited on pp. 59, 75, 76, 78, 80, 81, 82).

[199] C. Schmaltz, J. Weickert, and A. Bruhn: "Beating the quality of JPEG 2000 with anisotropic diffusion", In J. Denzler, G. Notni, and H. Süße (Eds.): *Pattern Recognition*, Lecture Notes in Computer Science, Vol. 5748, 452–461, Springer, Berlin, 2009 (cited on p. 11).

[200] K. Schrader, T. Alt, J. Weickert, and M. Ertel: "CNN-based Euler's elastica inpainting with deep energy and deep image prior", *Proc. 10th European Workshop on Visual Information Processing*, Lisbon, Portugal, Sept. 2022 (cited on p. 45).

[201] K. Schrader, P. Peter, N. Kämper, and J. Weickert: "Efficient neural generation of 4K masks for homogeneous diffusion inpainting.", In L. Calatroni, M. Donatelli, S. Morigi, M. Prato, and M. Santavesaria (Eds.): *Scale Space and Variational Methods in Computer Vision*, Lecture Notes in Computer Science, Vol. 14009, 16–28, Springer, Cham, 2023 (cited on p. 75).

[202] K. Schrader, J. Weickert, and M. Krause: "Anisotropic diffusion stencils: from simple derivations over stability estimates to ResNet implementations", In H. Neunzert (Ed.): *Progress in Industrial Mathematics at ECMI 2023*, to appear: Springer, Cham, 2024 (cited on pp. 33, 113).

[203] C. Schuhmann, R. Beaumont, R. Vencu, C. Gordon, R. Wightman, et al.: "LAION-5B: an open large-scale dataset for training next generation image-text models", *Proc. 36th International Conference on Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, et al. (Ed.), Vol. 35, Advances in Neural Information Processing Systems, 25278–25294, Dec. 2022 (cited on p. 22).

[204] T. Schütze and H. Schwetlick: "Bivariate free knot splines", *BIT Numerical Mathematics*, Vol. 43, No. 1, 153–178, Mar. 2003 (cited on p. 77).

[205] E. Shelhamer, J. Long, and T. Darrell: "Fully convolutional networks for semantic segmentation", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 39, No. 4, 640–651, Apr. 2017 (cited on p. 19).

[206] J. Shen and T. F. Chan: "Mathematical models for local nontexture inpaintings", *SIAM Journal on Applied Mathematics*, Vol. 62, No. 3, 1019–1043, 2002 (cited on p. 11).

[207] J. Shen, S. H. Kang, and T. F. Chan: "Euler's elastica and curvature-based inpainting", *SIAM Journal on Applied Mathematics*, Vol. 63, No. 2, 564–592, 2002 (cited on p. 50).

[208] I. Shumailov, Z. Shumaylov, Y. Zhao, Y. Gal, N. Papernot, et al.: *The curse of recursion: training on generated data makes models forget*, arXiv:2305.17493 [cs.LG], May 2023 (cited on p. 22).

[209] I. Shumailov, Z. Shumaylov, Y. Zhao, N. Papernot, R. Anderson, et al.: "AI models collapse when trained on recursively generated data", *Nature*, Vol. 631, 755–759, July 2024 (cited on p. 22).

[210] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, et al.: "Mastering the game of Go without human knowledge", *Nature*, Vol. 550, 354–359, Oct. 2017 (cited on p. 22).

[211] J. Sirignano and K. Spiliopoulos: "DGM: a deep learning algorithm for solving partial differential equations", *Journal of Computational Physics*, Vol. 375, 1339–1364, 2018 (cited on p. 47).

[212] V. Sitzmann, J. Martel, A. Bergman, D. Lindell, and G. Wetzstein: "Implicit neural representations with periodic activation functions", *Proc. 34th International Conference on Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin (Ed.), Vol. 33, Advances in Neural Information Processing Systems, 7462–7473, 2020 (cited on p. 63).

[213] J. Snell, K. Swersky, and R. Zemel: "Prototypical networks for few-shot learning", *Proc. 31st International Conference on Neural Information Processing Systems*, I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, et al. (Ed.), Vol. 30, Advances in Neural Information Processing Systems, 4080–4090, Long Beach, CA, Dec. 2017 (cited on p. 23).

[214] Y. Song, T. Wang, P. Cai, S. K. Mondal, and J. P. Sahoo: "A comprehensive survey of few-shot learning: evolution, applications, challenges, and opportunities", *ACM Computing Surverys*, Vol. 55, No. 13s, 1–40, Dec. 2023 (cited on p. 23).

[215] B. Speelpenning: "Compiling fast partial derivatives of functions given by algorithms", PhD thesis, University of Illinois, Illinois, 1980 (cited on p. 15).

[216] Stockfish Development Team: *Stockfish 12*, Retrieved September 21, 2025 from https://stockfishchess.org/, 2020 (cited on p. 23).

[217] K. Su, M. Chen, and E. Shlizerman: "INRAS: implicit neural representation for audio scenes", *Proc. 36th International Conference on Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, et al. (Ed.), Vol. 35, Advances in Neural Information Processing Systems, 8144–8158, Dec. 2022 (cited on p. 63).

[218] N. Sukumar and A. Srivastava: "Exact imposition of boundary conditions with distance functions in physics-informed deep neural networks", *Computer Methods in Applied Mechanics and Engineering*, Vol. 389, Article no. 114333, Feb. 2022 (cited on p. 48).

[219] G. J. Sullivan, J. R. Ohm, W. J. Han, and T. Wiegand: "Overview of the high efficiency video coding (HEVC) standard", *IEEE Transactions on Circuits, Systems and Video Technology*, Vol. 22, No. 12, 1649–1668, Sept. 2012 (cited on p. 76).

[220] S. Sun, Z. Cao, H. Zhu, and J. Zhao: "A survey of optimization methods from a machine learning perspective", *IEEE Transactions on Cybernetics*, Vol. 50, 3668–3681, 2019 (cited on p. 15).

[221] R. S. Sutton: *The Bitter Lesson*, Retrieved September 20, 2025 from http://www.incompleteideas.net/IncIdeas/BitterLesson.html, Mar. 2019 (cited on p. 21).

[222] X. Tai, J. Hahn, and G. Chung: "A fast algorithm for Euler's elastica model using augmented Lagrangian method", *SIAM Journal on Imaging Sciences*, Vol. 4, No. 1, 313–344, Mar. 2011 (cited on pp. 48, 58, 59).

[223] M. Tancik, P. Srinivasan, B. Mildenhall, S. Fridovich-Keil, N. Raghavan, et al.: "Fourier features let networks learn high frequency functions in low dimensional domains", *Proc. 34nd International Conference on Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin (Ed.), Vol. 33, Advances in Neural Information Processing Systems, 7537–7547, Virtual, Dec. 2020 (cited on pp. 47, 66).

[224] L. Theis, W. Shi, A. Cunningham, and F. Huszár: "Lossy image compression with compressive autoencoders", *Proc. 5th International Conference on Learning Representations*, Toulon, France, Apr. 2016 (cited on p. 88).

[225] *Top Chess Engine Championship (TCEC)*, Retrieved September 21, 2025 from https://tcec-chess.com, 2010 (cited on p. 23).

[226] D. Tschumperlé and R. Deriche: "Vector-valued image regularization with PDEs: a common framework for different applications", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 27, No. 4, 506–516, Apr. 2005 (cited on p. 25).

[227] N. Ueno and S. Koyama: "Sound field estimation: theories and applications", *Foundations and Trends in Signal Processing*, Vol. 19, No. 1, 1–98, Feb. 2025 (cited on pp. 30, 31, 63).

[228] N. Ueno, S. Koyama, and H. Saruwatari: "Directionally weighted wave field estimation exploiting prior information on source direction", *IEEE Transactions on Signal Processing*, Vol. 69, 2383–2395, 2021 (cited on pp. 62, 63).

[229] D. Ulyanov, A. Vedaldi, and V. Lempitsky: "Deep image prior", *Proc. 2018 IEEE Conference on Computer Vision and Pattern Recognition*, 9446–9454, Salt Lake City, UT, June 2018 (cited on pp. 3, 49, 52, 53, 70).

[230] D. Vašata, T. Halama, and M. Friedjungová: "Image inpainting using Wasserstein generative adversarial imputation network", In I. Farkaš, P. Masulli, S. Otte, and S. Wermter (Eds.): *Artificial Neural Networks and Machine Learning – ICANN 2021*, Lecture Notes in Computer Science, Vol. 12892, 575–586, Springer, Cham, 2021 (cited on pp. 90, 92, 104).

[231] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, et al.: "Attention is all you need", Advances in Neural Information Processing Systems, Vol. 30, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, et al. (Ed.), 6000–6010, Dec. 2017 (cited on p. 22).

[232] P. Villalobos, A. Ho, J. Sevilla, T. Besiroglu, L. Heim, et al.: "Will we run out of data? limits of LLM scaling based on human-generated data", *Proc. 41st International Conference on Machine Learning*, R. Salakhutdinov, Z. Kolter, K. Heller, A. Weller, N. Oliver, et al. (Ed.), Vol. 235, Proceedings of Machine Learning Research, 49523–49544, Vienna, Austria, July 2024 (cited on p. 22).

[233] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra: "Matching networks for one shot learning", *Proc. 30th International Conference on Neural Information Processing Systems*, D. D. Lee, M. Sugiyama, U. von Luxburg, I. Guyon, and R. Garnett (Ed.), Vol. 29, Advances in Neural Information Processing Systems, 3637–3645, Barcelona, Spain, Dec. 2016 (cited on p. 23).

[234] H. Wang, T. Li, Z. Zhuang, T. Chen, H. Liang, et al.: "Early stopping for deep image prior", *Transactions on Machine Learning Research* Dec. 2023 (cited on p. 49).

[235] N. Wang, Y. Zhang, and L. Zhang: "Dynamic selection network for image inpainting", *IEEE Transactions on Image Processing*, Vol. 30, 1784–1798, Jan. 2021 (cited on p. 81).

[236] S. Wang, S. Sankaran, H. Wang, and P. Perdikaris: *An expert's guide to training physics-informed neural networks*, arXiv:2308.08468 [cs.LG], Aug. 2023 (cited on p. 47).

[237] S. Wang, Y. Teng, and P. Perdikaris: "Understanding and mitigating gradient flow pathologies in physics-informed neural networks", *SIAM Journal on Scientific Computing*, Vol. 43, No. 5, A3055–A3081, Sept. 2021 (cited on p. 47).

[238] S. Wang, X. Yu, and P. Perdikaris: "When and why PINNs fail to train: a neural tangent kernel perspective", *Journal of Computational Physics*, Vol. 449, Article no. 110768, Jan. 2022 (cited on p. 47).

[239] W. Wang, J. Zhang, L. Niu, H. Ling, X. Yang, et al.: "Parallel multi-resolution fusion network for image inpainting", *Proc. 2021 IEEE/CVF International Conference on Computer Vision*, 14559–14568, Oct. 2021 (cited on p. 81).

[240] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli: "Image quality assessment: from error visibility to structural similarity", *IEEE Transactions on Image Processing*, Vol. 13, 600–612, Apr. 2004 (cited on p. 10).

[241] J. Weickert: *δ-Stencil for Anisotropic Diffusion on a Hexagonal Grid*, Sept. 2025 (cited on p. 111).

[242] J. Weickert: *Anisotropic Diffusion in Image Processing*, Teubner, Stuttgart, 1998 (cited on pp. 2, 11, 17, 26, 29, 33, 34, 35, 36, 41, 43).

[243] J. Weickert: "Coherence-enhancing diffusion filtering", *International Journal of Computer Vision*, Vol. 31, No. 2/3, 111–127, Apr. 1999 (cited on p. 25).

[244] J. Weickert: "Mathematische Bildverarbeitung mit Ideen aus der Natur", *Mitteilungen der DMV*, Vol. 20, 80–92, 2012 (cited on p. 60).

[245] J. Weickert: *The δ-Stencil Family for MCM*, Lecture, Retrieved September 19, 2025 from https://www.mia.uni-saarland.de/Teaching/DIC21/dic21-23.pdf, Jan. 2022 (cited on p. 111).

[246] J. Weickert: "Theoretical foundations of anisotropic diffusion in image processing", *Computing Supplement*, Vol. 11, 221–236, 1996 (cited on p. 25).

[247] J. Weickert and T. Brox: "Diffusion and regularization of vector- and matrix-valued images", In M. Z. Nashed and O. Scherzer (Eds.): *Inverse Problems, Image Analysis, and Medical Imaging*, Contemporary Mathematics, Vol. 313, 251–268, AMS, Providence, 2002 (cited on p. 25).

[248] J. Weickert, S. Grewenig, C. Schroers, and A. Bruhn: "Cyclic schemes for PDE-based image analysis", *International Journal of Computer Vision*, Vol. 118, No. 3, 275–299, July 2016 (cited on p. 44).

[249] J. Weickert and M. Welk: "Tensor field interpolation with PDEs", In J. Weickert and H. Hagen (Eds.): *Visualization and Processing of Tensor Fields*, 315–325, Springer, Berlin, 2006 (cited on p. 82).

[250] J. Weickert, M. Welk, and M. Wickert: "L^2-stable nonstandard finite differences for anisotropic diffusion", In A. Kuijper, K. Bredies, T. Pock, and H. Bischof (Eds.): *Scale Space and Variational Methods in Computer Vision*, Lecture Notes in Computer Science, Vol. 7893, 390–391, Springer, Berlin, 2013 (cited on pp. 2, 29, 34, 37, 38, 44).

[251] M. Welk: "Diffusion, pre-smoothing and gradient descent", In A. Elmoataz, J. Fadili, Y. Quéau, J. Rabin, and L. Simon (Eds.): *Scale Space and Variational Methods in Computer Vision*, Lecture Notes in Computer Science, Vol. 12679, 78–90, Springer, Cham, 2021 (cited on p. 35).

[252] H. Wendland: *Numerical Linear Algebra: An Introduction*, Cambridge University Press, Cambridge, UK, 2018 (cited on pp. 7, 38, 101).

[253] R. Wengert: "A simple automatic derivative evaluation program", *Communications of the ACM*, Vol. 7, No. 8, 463–464, 1964 (cited on p. 15).

[254] A. P. Witkin: "Scale-space filtering", *Proc. Eighth International Joint Conference on Artificial Intelligence*, Vol. 2, 945–951, Karlsruhe, Germany, Aug. 1983 (cited on p. 23).

[255] Y. Wu, H. Zhang, Y. Sun, and H. Guo: "Two image compression schemes based on image inpainting", *Proc. 2009 International Joint Conference on Computational Sciences and Optimization*, 816–820, IEEE Press, Sanya, China, Apr. 2009 (cited on p. 75).

[256] Z. Xiang, W. Peng, X. Liu, and W. Yao: "Self-adaptive loss balanced physics-informed neural networks", *Neurocomputing*, Vol. 496, 11–34, C 2022 (cited on p. 65).

[257] B. Xie and T. Zhang: "The audibility of spectral detail of head-related transfer functions at high frequency", *Acta Acustica united with Acustica*, Vol. 96, 328–339, 2010 (cited on p. 66).

[258] J. Xie, L. Xu, and E. Chen: "Image denoising and inpainting with deep neural networks", *Proc. 26th International Conference on Neural Information Processing Systems*, P. L. Bartlett, F. C. N. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (Ed.), Vol. 25, Advances in Neural Information Processing Systems, 350–358, Lake Tahoe, NV, Dec. 2012 (cited on p. 81).

[259] C. Yang, X. Lu, Z. Lin, E. Shechtman, O. Wang, et al.: "High-resolution image inpainting using multi-scale neural patch synthesis", *Proc. 2017 IEEE Conference on Computer Vision and Pattern Recognition*, 6721–6729, Honolulu, HI, July 2017 (cited on p. 81).

[260] D. Yao, J. Zhao, Y. Liang, Y. Wang, J. Gu, et al.: "Perceptually enhanced spectral distance metric for head-related transfer function quality prediction", *Journal of the Acoustical Society of America*, Vol. 156, No. 6, 4133–4152, Dec. 2024 (cited on p. 31).

[261] F. Yu and V. Koltun: "Multi-scale context aggregation by dilated convolutions", *Proc. 4th International Conference on Learning Representations*, San Juan, Puerto Rico, May 2016 (cited on pp. 18, 48, 58, 90).

[262] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, et al.: "Free-form image inpainting with gated convolution", *Proc. 2019 International Conference on Computer Vision (ICCV)*, Vol. 1, 4470–4479, IEEE Computer Society Press, Seoul, Korea, Oct. 2019 (cited on pp. 48, 49, 53, 55).

[263] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, et al.: "Generative image inpainting with contextual attention", *Proc. 2018 IEEE Conference on Computer Vision and Pattern Recognition*, 5505–5514, Salt Lake City, UT, June 2018 (cited on p. 81).

[264] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus: "Deconvolutional networks", *Proc. 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2528–2535, San Francisco, CA, June 2010 (cited on p. 19).

[265] A. Zhang, Z. C. Lipton, M. Li, and A. J. Smola: *Dive into Deep Learning*, Cambridge University Press, Cambridge, 2023 (cited on pp. 1, 11).

[266] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang: "The unreasonable effectiveness of deep features as a perceptual metric", *Proc. 2018 IEEE Conference on Computer Vision and Pattern Recognition*, 586–595, Salt Lake City, UT, June 2018 (cited on p. 10).

[267] S. Zhang, L. Yao, A. Sun, and Y. Tay: "Deep learning based recommender system: a survey and new perspectives", *ACM Computing Surverys*, Vol. 52, No. 1, 1–38, Feb. 2019 (cited on p. 1).

[268] Y. Zhang, Y. Wang, and Z. Duan: "HRTF field: unifying measured HRTF magnitude representation with neural fields", *Proc. 2023 IEEE International Conference on Acoustics, Speech and Signal Processing*, Rhodes Island, Greece, June 2023 (cited on p. 63).

[269] C. Zhao and M. Du: "Image compression based on PDEs", *Proc. 2011 International Conference of Computer Science and Network Technology*, 1768–1771, IEEE Press, Harbin, China, Dec. 2011 (cited on p. 75).

[270] F. Zhuang, Z. Qi, K. Duan, D. Xi, Y. Zhu, et al.: "A comprehensive survey on transfer learning", *Proceedings of the IEEE*, Vol. 109, No. 1, 43–76, 2021 (cited on p. 23).

C Glossary

AA . . . . . Analytic Approach

AdaGrad . . . Adaptive Gradient Optimiser

Adam . . . . . Adaptive Moment Estimation

ADMM . . . . Alternating Direction Method of Multipliers

AI . . . . . Artificial Intelligence

BSDE . . . . . Backward Stochastic Differential Equation

BSDS500 . . . Berkeley Segmentation Database

CG . . . . . Conjugate Gradients

CNN . . . . . Convolutional Neural Network

CPU . . . . . Central Processing Unit

CUDA . . . . Compute Unified Device Architecture

dB . . . . . Decibels

EED . . . . . Edge-Enhancing Anisotropic Diffusion

GPU . . . . . Graphics Processing Unit

HEVC . . . . . High Efficiency Video Coding

HRTF . . . . . Head-Related Transfer Function

IVP . . . . . Initial Value Problem

JPEG . . . . . Joint Photographic Experts Group

LCZero . . . . Leela Chess Zero

LSD . . . . . Logarithmic Spectral Distance

MAE . . . . . Mean Absolute Error

MLP . . . . . Multilayer Perceptron

MNIST . . . . Modified National Institute of Standards and Technology

MSE . . . . . Mean Square Error

NF . . . . . Neural Field

NLPE . . . . . Non-local Pixel Exchange

NODE . . . . Neural Ordinary Differential Equation

ODE . . . . . Ordinary Differential Equation

PDE . . . . . Partial Differential Equation

PINN . . . . . Physics-Informed Neural Network

PINO . . . . . Physics-Informed Neural Operator

PS . . . . . Probablistic Sparsification

PSNR . . . . . Peak Signal-to-noise Ration

ReLU . . . . . Rectified Linear Unit

ResNet . . . . Residual Network

RFF . . . . . Random Fourier Features

RGB . . . . . Red-green-blue Colour Space

RMSProp . . Root Mean Square Propagation

TO . . . . . Tonal Optimisation

TV . . . . . Total Variation

VPINN . . . . Variational Physics-Informed Neural Network

XAI . . . . . Explainable Artificial Intelligence

D List of Symbols

$|\cdot|$ . . . . . Absolute value

$*$ . . . . . Convolution operator

$\lfloor \cdot \rfloor$ . . . . . Floor function

$\boldsymbol{\nabla}$ . . . . . Gradient operator

$\widehat{\cdot}$ . . . . . Fourier transform

$\|\cdot\|_p$ . . . . . p-norm or vector or matrix

$\|$ . . . . . Parallel vectors

$\perp$ . . . . . Orthogonal/Perpendicular vectors

$\boldsymbol{A}$ . . . . . System matrix

$b, \boldsymbol{b}$ . . . . . Bias in neural networks

c . . . . . Speed of sound

$\boldsymbol{C}$ . . . . . Binary inpainting mask

$\boldsymbol{D}$ . . . . . Diffusion tensor of anisotropic diffusion

d . . . . . Target mask density

$\boldsymbol{D}_h^{+/-}$ . . . . . First order forward/backward difference with grid size h

div . . . . . Divergence operator

E . . . . . Energy functional

f . . . . . Continuous input image, noisy/masked

$\boldsymbol{f}$ . . . . . Discrete input image, noisy/masked

$g(\cdot)$ . . . . . Diffusivity function

h . . . . . Grid size of images or signals

$\boldsymbol{I}$ . . . . . Identity matrix

i . . . . . Imaginary unit

Id . . . . . Identity function

K . . . . . Inpainting mask

k . . . . . Time step in iterative schemes, wave number

$\boldsymbol{k}, \boldsymbol{K}$ . . . . . Discrete convolution kernel

K_σ . . . . . Gaussian with zero mean and standard deviation σ

$\mathcal{L}$ . . . . . Loss function

$\mathcal{N}$ . . . . . Neural network

$\boldsymbol{n}$ . . . . . Outer normal vector at image boundary $\partial\Omega$

$\mathcal{O}$ . . . . . Bachmann-Landau notation

t . . . . . Time

u . . . . . Continuous evolving or output image

$\boldsymbol{u}$ . . . . . Discrete evolving or output image

$\boldsymbol{v}$ . . . . . Eigenvector

$\boldsymbol{w}, \boldsymbol{W}$ . . . . . Weight in a neural network

$\boldsymbol{x}$ . . . . . Spatial coordiante

∂_t . . . . . Partial derivative w.r.t. t

$\partial_{\boldsymbol{n}}$ . . . . . Directional derivative in direction $\boldsymbol{n}$

$\partial\Omega$ . . . . . Boundary of a domain

Δ . . . . . Laplace operator

ε . . . . . Small regularisation constant

κ . . . . . Curvature

λ . . . . . Contrast parameter for diffusivities, eigenvalues

Ω . . . . . Image or signal domain

ω . . . . . Angular frequency

Φ . . . . . Flux function of isotropic diffusion

$\rho(\cdot)$ . . . . . Spectral radius of matrix

σ . . . . . Standard deviation, or activation function

τ . . . . . Time step size

$\boldsymbol{\theta}$ . . . . . Trainable parameters of a neural network

E List of Figures