Universität Bremen

ZeTeM

Fachbereich 3
Mathematik und Informatik

Regularization Theory for Special Neural Network Architectures Applied on Inverse Problems

PhD thesis

Clemens Arndt

Date of Colloquium: 9 January 2025

Reviewers:
Prof. Dr. Dr. h.c. Peter Maaß,
Professor Simon Arridge

Zentrum für Industriemathematik

Abstract

This thesis is about deep learning methods for solving ill-posed inverse problems which make use of special neural network architectures in order to obtain theoretical guarantees about regularization properties. In the first part, we provide an overview of the classical theory of linear inverse problems and various deep learning approaches for solving them. We discuss which theoretical properties are particularly desirable for reconstruction methods (as existence, uniqueness, stability, and convergence of solutions) and which technical conditions are required for different methods to achieve them. Besides, we describe different ways for neural networks to gain prior knowledge about the solutions of inverse problems, e.g., via end-to-end learning or regularization by architecture.

The second part of the thesis, consisting of six publications, is about specific methods which combine rigorous theoretical regularization guarantees with high practical reconstruction performance. First, we prove regularization properties for a deep image prior approach with a LISTA architecture. This is done by exploiting the structural similarity between the network architecture and classical iterative reconstruction methods, resulting in the analytic deep prior formulation. For analytic deep prior, we then establish an equivalence to variational regularization and thus obtain the desired properties. In addition, we report on the application of different deep learning based methods in two international challenges about solving real-world inverse problems. The results particularly demonstrate the importance of high-quality training data so that the methods can learn extensive prior knowledge about the solutions. End-to-end learning emerged as the most successful strategy in both challenges. Furthermore, we introduce a new deep learning based reconstruction method using invertible residual networks (iResNets). The inherent properties of this type of architecture allow for proving theoretical regularization guarantees as well as analyzing what exactly the networks learn in different training strategies about the training data and the forward operator. Moreover, we show the practical performance of iResNets for solving inverse problems in numerical experiments.

Zusammenfassung

Diese Dissertation behandelt Deep Learning Methoden zum Lösen schlecht gestellter inverser Probleme unter Verwendung spezieller Netzwerk-Architekturen, um theoretische Garantien bezüglich der Regularisierungseigenschaften zu erhalten. Der erste Teil bietet eine Übersicht über die klassische Theorie linearer inverser Probleme und verschiedene Deep Learning Methoden zum Lösen dieser. Es wird erörtert, welche theoretischen Eigenschaften für Rekonstruktionsverfahren besonders wünschenswert sind (wie Existenz, Eindeutigkeit, Stabilität und Konvergenz von Lösungen) und welche technischen Bedingungen für die verschiedenen Methoden notwendig sind, um diese zu erreichen. Außerdem werden verschiedene Möglichkeiten beschrieben, wie neuronale Netze wichtiges Vorwissen über die Lösungen inverser Probleme erwerben können, zum Beispiel mittels „end-to-end learning“ oder „regularization by architecture“.

Der zweite Teil der Arbeit, bestehend aus sechs Publikationen, behandelt konkrete Methoden, die strikte theoretische Regularisierungsgarantien mit hoher Effektivität in der praktischen Anwendung vereinen. Zunächst werden Regularisierungseigenschaften für einen „deep image prior“ Ansatz mit „LISTA“-Architektur nachgewiesen. Dafür wird die strukturelle Ähnlichkeit zwischen der Netzwerk-Architektur und klassischen iterativen Rekonstruktionsverfahren ausgenutzt und man erhält die „analytic deep prior“ Formulierung. Für analytic deep prior wird eine Äquivalenz zu variationellen Regularisierungsverfahren hergeleitet, um auf diese Weise die gewünschten Eigenschaften zu erhalten. Des Weiteren wird vom Einsatz verschiedener Deep Learning Methoden in zwei internationalen Wettbewerben zum Lösen realer inverse Probleme berichtet. Die Resultate zeigen insbesondere die Bedeutung von Trainingsdaten mit hoher Qualität für die Methoden, um Vorwissen über die Lösungen zu lernen. In beiden Wettbewerben hat sich „end-to-end learning“ als die erfolgreichste Strategie herausgestellt. Darüber hinaus führen wir eine neue auf invertierbaren „residual networks“ (iResNets) basierende Rekonstruktionsmethode ein. Die intrinsischen Eigenschaften dieser Architektur erlauben zum einen den Nachweis von theoretischen Regularisierungsgarantien. Zum anderen ist es möglich zu untersuchen, was die Netze bei verschiedenen Trainingsstrategien über die Trainingsdaten und den Vorwärtsoperator lernen. Außerdem wird die praktische Effektivität von iResNets in numerischen Experimenten gezeigt.

Acknowledgments

First of all, I would like to thank my supervisor Peter Maaß, who shared his enthusiasm for inverse problems with me and gave me a lot of valuable feedback during my PhD.

In addition, I would like to thank Simon Arridge for reviewing this thesis and sharing ideas over the last few years, especially during my visit to London in 2022.

Besides, I want to thank Alexander Denker, Nick Heilenkötter, Sören Dittmer, Meira Iske, Tobias Kluth, and Judith Nickel for the motivating and effective collaboration and the creative exchange in our regular meetings about iResNets.

Furthermore, I want to thank the whole team of the working group Industrial Mathematics for offering a pleasant working environment and always providing help with both mathematical and organizational issues.

Finally, I would like to thank my family and especially my wife Felicitas for always supporting me.

Part I
Background

Contents

Chapter 1

Introduction

My interest in inverse problems, the task of determining the cause for a certain observation, was sparked by the very specific challenges of this field. Due to their inherent properties, solving these problems is particularly difficult in various ways. More precisely, inverse problems may have multiple potential solutions (as there might exist different causes for the same observation) such that determining the correct one is only possible if suitable additional information about the solution is considered. One example from everyday life is the restoration of blurred images where the subject is no longer recognizable. We might need to know that there is, e.g., a human face on it in order to be able to reconstruct a clean image. In addition, the process of recovering the clean image might be unstable, i.e., small deviations in the blurred image may lead to big deviations in the reconstruction. Because of these challenges, inverse problems are often referred to as ill-posed in mathematics.

Further applications where inverse problems occur are, e.g., medical imaging and non-destructive testing. Computed tomography (CT) makes it possible to reconstruct the inside of a human body by sending radiation through it and measuring the dose on the other side. Similar procedures also exist for technical components in order to find internal defects without having to dismantle the component. These applications are often safety-critical such that reconstruction errors must be avoided as far as possible. Therefore, a strong theoretical foundation is required to guarantee a predictable behavior of the reconstruction algorithms. For this purpose, the mathematical theory of regularization has been developed and has been the source of a wide range of classical (handcrafted and model-based) reconstruction methods.

However, the currently best-performing reconstruction methods for inverse problems in most applications are data-driven and based on deep learning. For these approaches, one has to collect (a lot of) data related to the problem in advance, e.g., samples of solutions. Then, a neural network is employed to extract useful information from this data (e.g., that there is a human face on the blurred image) in order to "learn" how the inverse problem can be solved. The success of deep learning in practice has often been demonstrated, for example in the Helsinki Deblur Challenge 2021 [10] and the Helsinki Tomography Challenge 2022 [9].

Despite this high performance, data-driven methods sometimes show unpredictable behavior. Guaranteeing certain properties of deep learning based reconstruction schemes, e.g., stability, is usually far more challenging than for classical regularization methods, and improving the theoretical understanding of deep learning methods is still an active field of research. Two promising approaches in this direction are the combination of classical algorithms with data-driven methods and the usage of special neural network architectures which provide possibilities for analytical investigations. In this thesis, we particularly investigate the analytic deep prior [7] approach and invertible neural networks (iResNets) [8, 11, 12]. Analytic deep prior is a variant of the deep image prior approach [64], where the neural network is replaced by a classical reconstruction method, which opens the way for applying the classical regularization theory. The architecture of iResNets is designed in a way that allows for a stable inversion and therefore provides a provably stable reconstruction scheme for inverse problems. However, the analytical benefits of these approaches are based on certain restrictions on the network architecture. Thus, the crucial challenge is to find an appropriate compromise between high performance in practical applications and strong theoretical guarantees.

1.1 Structure of the Thesis

Part I of the thesis starts with an overview of the classical theory for linear inverse problems in Chapter 2. We begin with the characteristic challenges which result from ill-posedness. Thereafter, the concept of regularization is introduced. A special focus is laid on the theoretical guarantees (as existence, uniqueness, stability, and convergence of solutions) that regularization schemes can provide. Spectral and variational regularization methods are explained in more detail. The chapter concludes with a Bayesian view on variational regularization and iterative methods for solving variational minimization problems.

Chapter 3 describes how deep learning methods can be used for solving inverse problems. After a brief introduction to neural networks, we discuss which theoretical guarantees for learning-based algorithms would be desirable in the context of inverse problems. It follows an overview of common deep learning approaches for inverse problems. There we highlight again the theoretical guarantees which can be obtained via the different strategies.

We then move on to special network architectures with provable regularization properties, the main topic of this thesis, in Chapter 4. There, we provide the motivation for the research of this dissertation, focussing on both theoretical and practical aspects. In particular, we summarize the results and the connections of the six research papers which constitute Part II of the thesis.

1.2 Included Research Papers and Contributions

This cumulative thesis is a synthesis of the following six papers that emerged during my time as a PhD student. For each of the papers, my particular contributions to the research are specified below. Throughout the thesis, citations of my own work are indicated in red, while all other literature references are marked in blue.

- **Regularization theory of the analytic deep prior approach** by C. Arndt, published in *Inverse Problems* [7].
  The paper is based on the results of my master's thesis. The theoretical parts are completely revised with enhanced proofs of the equivalence result (Lemma 3.1 and 3.2), a new result about the relation between the regularization parameters of analytic deep prior (ADP) and Tikhonov (Lemma 3.6), clarified relations between ADP and Ivanov methods (Remark 3.7 and 3.9), and a generalized framework for the regularization theory of ADP-β (Section 3.4). The numerics section is completely new.

- **Invertible residual networks in the context of regularization theory for linear inverse problems** by C. Arndt, A. Denker, S. Dittmer, N. Heilenkötter, M. Iske, T. Kluth, P. Maass, and J. Nickel, published in *Inverse Problems* [8].
  Most of the ideas for the content of the paper were developed together in group meetings. I made major contributions to all theoretical aspects of the paper (Section 2, 3, and 4). For a large majority of the results, I did a substantial amount of work. In particular, I did the proofs of Lemma 2.1, Lemma 3.2, Lemma 4.1, and Lemma 4.4 and made major contributions to the proofs of Theorem 3.1, Lemma 3.3, Lemma 4.2, Lemma 4.3, and Lemma 4.5. Besides, I did the graphical illustrations of the filter functions in Section 4 and the computation of the filter function (4.22) (Appendix A.5).

- **Bayesian view on the training of invertible residual networks for solving linear inverse problems** by C. Arndt, S. Dittmer, N. Heilenkötter, M. Iske, T. Kluth, and J. Nickel, published in *Inverse Problems* [11].
  I made major contributions and did substantial work for a majority of the theoretical aspects of the paper (Section 2, 3, and 4). In particular, I developed (and proved) the main results about the expectation values of approximation training (Theorem 3.1) and reconstruction training (Lemma 4.1 and Lemma 4.2). Furthermore, I did the proofs of Lemma 2.1 and Lemma 3.1. Besides, I arranged the major part of the problem setting (Section 2) and made major contributions to the ideas and interpretations of some of the numerical experiments (Figure 5 and Figure 9).

- **Invertible ResNets for Inverse Imaging Problems: Competitive Performance with Provable Regularization Properties** by C. Arndt, and J. Nickel, submitted for publication in *SIAM Journal on Imaging Sciences* [12].

Judith Nickel and I contributed equally to the paper, the basic ideas were developed together. I am particularly responsible for the content of Section 3, Section 4.1, and Section 4.3.1. Regarding the numerical experiments, we contributed equally to the procedure of improving the iResNet architecture by testing different ideas and variants. I particularly did the training of the small iResNet architecture, the implementation of the Lipschitz constraint and the deep equilibrium model, and the experiments about local ill-posedness with directional derivatives (Section 4.3.1). Besides, the basic evaluations and investigations of the networks (Section 4.2 and the first part of Section 4.3) were done together.

- **In focus - hybrid deep learning approaches to the HDC2021 challenge** by C. Arndt, A. Denker, J. Nickel, J. Leuschner, M. Schmidt, and G. Rigaud, published in *Inverse Problems and Imaging* [10].
  The main work was due to Alexander Denker. Judith Nickel and I contributed equally to two of our four reconstruction approaches, the "Baseline U-Net" (Section 4.1) and the "Educated Deep Image Prior" (Section 4.2) and the corresponding numerical results in Section 5.

- **Modelbased deep learning approaches to the Helsinki Tomography Challenge 2022** by C. Arndt, A. Denker, S. Dittmer, J. Leuschner, J. Nickel, and M. Schmidt, published in *Applied Mathematics for Modern Challenges* [9].
  The main work was due to Alexander Denker. I implemented one of the data generation methods (the third one of Section 2.2) and the initial classical reconstruction for the "edge inpainting" method. Further, Judith Nickel and I contributed equally to the inpainting U-Net and the segmentation U-Net of this method (Section 3.2 and corresponding numerical results in Section 4).

Chapter 2

Classical Theory of Linear Inverse Problems

Both in science and in everyday life, we observe the principle of *causes* and *effects*. Computing the effect that is induced by a certain cause is often called the *direct problem*. In this thesis, we are particularly interested in situations where the process which connects causes and effects is known to a degree that enables us to solve the direct problem mathematically. However, our focus is on the opposite problem, which is inverse to the causality. That is why the task of determining the cause of a certain effect is called *inverse problem*.

There are two phenomena which are particularly characteristic for inverse problems. First, we consider the fact that effects which occur in the real world must be measured, which is rarely a totally exact procedure. Thus, the measurement data is assumed to be corrupted by a certain amount of noise. Second, two completely different causes can sometimes lead to very similar effects. This will be discussed in detail in the next section.

To tackle inverse problems mathematically, we model the measurement process in a functional analytical setting. The causes and effects are represented as elements x and y from Hilbert spaces[1] X and Y, respectively. Between causes and effects, we assume the linear dependence

$$Ax = y, \tag{2.0.1}$$

with $A \in L(X, Y)$ being called the *forward operator*. While the forward operators of general inverse problems might be nonlinear, there exist a lot of applications (e.g., in imaging and image processing) where the linearity assumption is reasonable and allows for a stronger theory. The inverse problem consists in determining an unknown cause $x^\dagger \in X$ (called *ground truth*) from the operator equation (2.0.1) for given y and A. However, we consider the case that there is no access to clean measurements $y^\dagger = Ax^\dagger$ but only to noisy data $y^\delta \in Y$ with $\|y^\delta - y^\dagger\| \leqslant \delta$ for a certain noise level $\delta > 0$. Thus, we do not expect to perfectly recover $x^\dagger$ but are looking for an approximate solution instead.

[1] There also exists a more general (but weaker) theory with Banach spaces, e.g., [90].

2.1 The Special Difficulties of Inverse Problems

A "hard" mathematical problem is often associated with the need for complicated algorithms or extensive computations. But independent on the effort which is required for solving a problem, there is the fundamental question of whether expedient universal solution algorithms could exist at all. This question is particularly crucial in the context of inverse problems. To distinguish between so-called *well-posed* and *ill-posed* problems, the following definition has been established and is used by the majority of works [85, 84, 56, 44, 90, 60].

Definition 2.1.1 (well-posedness by Hadamard)**.** *An inverse problem $Ax = y$ is called well-posed in the sense of Hadamard if, for all $y \in Y$,*

- *there exists a solution $x \in X$,*
- *the solution x is uniquely determined,*
- *the solution x depends continuously on y.*

If one of these conditions is violated, the problem is called ill-posed.

When considering this definition in the context of the setting introduced at the beginning of this chapter, we observe that inverse problems are likely to violate any of the three conditions. Due to the noise on y^δ, which might not be contained in the range $\mathcal{R}(A)$ of the forward operator, the existence of a solution may become unclear. Besides, A might have a non-trivial nullspace $\mathcal{N}(A)$ such that solutions are not unique. However, the third condition is actually the most critical, which will become clear after explaining how the first two issues can be overcome.

For $y^\delta \in \mathcal{R}(A) \oplus \mathcal{R}(A)^\perp$ (which is at least a dense subspace of Y) one can project the data onto $\mathcal{R}(A)$. The inverse problem with the projected data is then equivalent to the minimizing problem $\min_{x\in X} \|Ax - y^\delta\|^2$ and also equivalent to solving the *normal equation* $A^*Ax = A^*y^\delta$ [85, Satz 2.1.1]. We thus look for a solution x which is most compatible with the data y^δ.

Addressing the non-uniqueness, we can choose a certain element from the set of all solutions. In a Hilbert space, the most common choice is the *minimum-norm solution*

$$\arg\min_{x\in X} \|x\|^2 \quad \text{s.t. } A^*Ax = A^*y, \tag{2.1.1}$$

which is unique due to the strict convexity of the norm. The mapping of $y \in \mathcal{R}(A) \oplus \mathcal{R}(A)^\perp$ to the minimum-norm solution in X is called the *Moore-Penrose inverse* of A [85, Definition 2.1.5] and denoted by $A^\dagger$.

The third condition of Definition 2.1.1 is called stability. A lack of stability is critical due to the presence of noise in the data y^δ. If $A^\dagger$ is discontinuous, $A^\dagger y^\delta$ and $A^\dagger y^\dagger$ might be totally different from each other, even in the case of a very small noise level δ. The solution $A^\dagger y^\delta$ would be totally unfeasible in this case.

Whether $A^\dagger$ is continuous or not depends on the forward operator A and its range. By [85, Satz 2.1.8], [60, Theorem 1.5.7], the Moore-Penrose inverse is continuous if and only if $\mathcal{R}(A)$ is closed. Additionally, in this case, $\mathcal{R}(A) \oplus \mathcal{R}(A)^\perp = Y$ holds, thus, $A^\dagger$ is defined on the whole data space. All issues are therefore reduced to the question of whether $\mathcal{R}(A)$ is closed. This leads to the following alternative definition of well-posedness [85, 90, 56].

Definition 2.1.2 (well-posedness by Nashed)**.** *An inverse problem $Ax = y$ is called well-posed in the sense of Nashed if the range $\mathcal{R}(A) \subset Y$ of the forward operator is closed. Otherwise, the problem is called ill-posed.*

This definition provides an explicit criterion for whether a given inverse problem is ill-posed and the Moore-Penrose inverse is inappropriate due to instability. For a stable solution of the inverse problem, we then need a different reconstruction algorithm $T\colon Y \to X$ which is continuous. However, in addition to stability, we also require T to produce low reconstruction errors. Unfortunately, the dilemma with ill-posed problems is that a reconstruction algorithm cannot be arbitrarily accurate and stable at the same time, even in the absence of noise. As the following lemma shows, there always exist two different ground truth elements $x_0^\dagger, x_1^\dagger \in \mathcal{N}(A)^\perp \subset X$ such that T produces significant reconstruction errors for at least one of them.

Lemma 2.1.3. *Let $T\colon Y \to X$ be a continuous reconstruction algorithm for an ill-posed (Nashed) inverse problem $Ax = y$. Then, for arbitrary $x_0^\dagger \in \mathcal{N}(A)^\perp$ and $C, \varepsilon > 0$ there exists $x_1^\dagger \in \mathcal{N}(A)^\perp$ such that*

$$\|x_1^\dagger - x_0^\dagger\| = C \qquad \wedge \qquad \|T(Ax_1^\dagger) - x_1^\dagger\| + \|T(Ax_0^\dagger) - x_0^\dagger\| \geqslant C - \varepsilon. \tag{2.1.2}$$

Proof. Since T is continuous, for any $\varepsilon > 0$ there exists $\delta > 0$ such that

$$\forall y \in B_\delta(Ax_0^\dagger)\colon \qquad \|T(Ax_0^\dagger) - T(y)\| \leqslant \varepsilon. \tag{2.1.3}$$

And since the problem is ill-posed, for any $C > 0$ there exists $x_1^\dagger \in \mathcal{N}(A)^\perp$ such that

$$\|x_1^\dagger - x_0^\dagger\| = C \qquad \wedge \qquad \|Ax_1^\dagger - Ax_2^\dagger\| < \delta, \tag{2.1.4}$$

otherwise, the Moore-Penrose inverse $A^\dagger$ would be continuous. It follows

$$\begin{aligned} C &= \|x_1^\dagger - x_0^\dagger\| \\ &\leqslant \|x_1^\dagger - T(Ax_1^\dagger)\| + \|T(Ax_1^\dagger) - T(Ax_0^\dagger)\| + \|T(Ax_0^\dagger) - x_0^\dagger\| \\ &\leqslant \|x_1^\dagger - T(Ax_1^\dagger)\| + \varepsilon + \|T(Ax_0^\dagger) - x_0^\dagger\|. \end{aligned} \tag{2.1.5}$$

Finally, we only have to subtract ε. □

Ill-posed problems in the sense of Nashed only exist in infinite dimensional spaces, because in finite dimensions, the range of continuous mappings is always closed [90, Remark 3.6]. This means, that in finite dimensions, the Moore-Penrose inverse is well-defined

and continuous on the whole space Y. In other words, a stable inversion of a finite dimensional A is theoretically always possible. So, one could think that all challenges can be overcome by discretizing the inverse problem for the computational solution. But the discretization of ill-posed operator equations typically leads to *ill-conditioned* linear systems.

Ill-conditioned problems are characterized by a Moore-Penrose inverse whose operator norm $\|A^\dagger\|$ is finite but still very large. This becomes apparent when the continuity of $A^\dagger$ is used to estimate a reconstruction error

$$\|A^\dagger y^\delta - x^\dagger\| \leqslant \|A^\dagger\| \cdot \|y^\delta - y^\dagger\| \leqslant \|A^\dagger\| \cdot \delta. \tag{2.1.6}$$

Due to limited numerical precision or limited measuring accuracy in reality, δ cannot be forced to become arbitrarily small in general. Thus, from a practical point of view, $A^\dagger$ might behave almost like a discontinuous operator if $\|A^\dagger\|$ is too large. It therefore seems as if the solution x does not depend continuously on y, as required in the well-posedness definition of Hadamard. Accordingly, it is reasonable to treat ill-conditioned problems like ill-posed problems. Using the Moore-Penrose inverse for reconstruction is not sufficient then. Instead, *regularization* methods are necessary.

2.2 The Idea of Regularization

Summarizing the results of the last section, ill-posedness implies that the data y^δ does not provide enough information to determine an appropriate reconstruction of the ground truth $x^\dagger$. Accordingly, additional information about $x^\dagger$ is needed. Such information is also called *prior knowledge* about the ground truth and it includes specific properties a potential solution is expected to fulfill. This is typically very application-specific and ranges from simple conditions to highly abstract features.

The fundamental idea of regularization is to use prior knowledge to transform the ill-posed inverse problem into a well-posed auxiliary problem, which is to be solved instead. How this is done precisely is again highly application-specific. This is also the reason why there is such a wide variety of reconstruction approaches (Section 2.3, 2.4 and 3.3).

According to Hadamard's definition 2.1.1, the regularized problem should guarantee existence, uniqueness, and stability of solutions. In addition, we want those solutions to be good approximations to the actual ground truth. However, due to the noise on the data y^δ, a perfect accordance cannot be expected. Instead, we demand the regularized solutions to converge to the ground truth if the noise level δ vanishes. This motivates the mathematical definition of a regularization scheme [85, Definition 3.1.1], [44, Definition 3.1], [90, Definition 3.20], [84, Definition 4.1], [56, Definition 4.1].

Definition 2.2.1 (convergent regularization scheme)**.** *Let $T_\alpha \colon Y \to X$ be a family of continuous operators and $\hat{\alpha} \colon \mathbb{R}_{>0} \times Y \to \mathbb{R}_{>0}$ an arbitrary mapping such that for any*

$y \in \mathcal{R}(A)$ *it holds*

$$\sup \left\{ \|T_{\hat{\alpha}(\delta,y^\delta)}(y^\delta) - A^\dagger y\| \,\middle|\, y^\delta \in Y \colon \|y^\delta - y\| \leqslant \delta \right\} \to 0 \quad \textit{for} \quad \delta \to 0. \tag{2.2.1}$$

Then, $(T_\alpha, \hat{\alpha})$ *is called a convergent regularization scheme.*

In this definition, T_α is the actual reconstruction algorithm, and α is called *regularization parameter*. As the mapping $\hat{a}$ suggests, α can be chosen depending on the noise level δ and the data y^δ to control how accurately T_α approximates the Moore-Penrose inverse $A^\dagger$. This is important to find the right balance between accuracy and stability of reconstructions (Lemma 2.1.3). Together, the reconstruction algorithm T_α and an appropriate parameter choice rule $\hat{\alpha}$ may form a convergent regularization scheme.

Remark 2.2.2. Usually, the dependency of the operator T_α on $\alpha > 0$ is defined in a way that larger values of α correspond to a higher amount of regularization. Accordingly, a parameter choice rule $\hat{\alpha}$ usually fulfills $\hat{\alpha}(\delta, y^\delta) \to 0$ for $\delta \to 0$. This is different in the case of iResNets [8], where the regularization parameter $L < 1$ is a Lipschitz constant and a hyperparameter of the architecture. In this context, one chooses $L \to 1$ for $\delta \to 0$.

In practice, the term regularization is not always used in the strict sense of Definition 2.2.1. Even in the classical theory, there exist reconstruction approaches which do not fulfill all the required properties of a regularization. For example, the stability condition is often replaced with a weaker form of continuity. Besides, for some approaches the convergence $T_\alpha(y^\delta) \to A^\dagger y^\dagger$ only holds w.r.t. specific distance measures which are weaker than the norm of X. This is explained in more detail in Section 2.4.

However, in practice, even technical continuity might sometimes be insufficient for a robust reconstruction. We recall that for finite dimensional problems, $A^\dagger$ is indeed continuous but inappropriate for ill-conditioned problems (Section 2.1). What is really needed is a controllable amount of stability which can be adapted to the requirements of the situation. We took this aspect in particular into account with iResNets [8].

2.3 Spectral Regularization

The spectral theory from functional analysis provides tools for a deeper understanding of ill-posed inverse problems and also a framework for a class of linear regularization methods. The key is to decompose elements x and y with orthonormal bases (ONBs) of the Hilbert spaces X and Y. In this context, (infinite dimensional) compact forward operators A are particularly interesting because they always lead to ill-posedness. This is a consequence of the spectral theorem [110, Satz VI.3.6], which introduces the singular value decomposition (SVD).

Theorem 2.3.1 (singular value decomposition)**.** *Let* $A \colon X \to Y$ *be a compact, linear operator. Then, there exists a singular value decomposition* $(u_j, v_j, \sigma_j)_{j \in N}$*, where* $N \subset \mathbb{N}$*, and the following conditions hold:*

- $(u_j) \subset Y$ *is an ONB of* $\overline{\mathcal{R}(A)}$,
- $(v_j) \subset X$ *is an ONB of* $\mathcal{N}(A)^\perp$,
- $\sigma_j > 0$ *and* $\sigma_j \to 0$ *if* $|N| = \infty$,
- $Av_j = \sigma_j u_j$, $A^* u_j = \sigma_j v_j$, *and*

$$Ax = \sum_{j \in N} \sigma_j \langle x, v_j \rangle u_j. \tag{2.3.1}$$

Proof. We define $K = A^*A$, which is a compact and self-adjoint operator since compactness of A implies compactness of A^* [4, Satz 10.6]. Note that $\mathcal{N}(K) = \mathcal{N}(A)$ holds.

The spectral theorem for compact normal operators [4, Satz 10.12] provides an ONB $(v_j) \subset X$ of $\mathcal{N}(K)^\perp$ and a sequence $(\lambda_j) \subset \mathbb{C}\backslash\{0\}$, $j \in N \subset \mathbb{N}$ with $\lambda_j \to 0$ (or $|N|$ is finite) such that

$$Kx = \sum_{j \in N} \lambda_j \langle x, v_j \rangle v_j. \tag{2.3.2}$$

Due to $\lambda_j = \langle \lambda_j v_j, v_j \rangle = \langle Kv_j, v_j \rangle = \langle Av_j, Av_j \rangle \geqslant 0$, we can deduce that the values λ_j are real and positive. Thus, we can define $\sigma_j = \sqrt{\lambda_j}$ and $u_j = \frac{1}{\sigma_j} Av_j$. It directly follows $A^* u_j = \frac{1}{\sigma_j} Kv_j = \sigma_j v_j$ and due to

$$\langle u_i, u_j \rangle = \frac{1}{\sigma_i \sigma_j} \langle Av_i, Av_j \rangle = \frac{1}{\sigma_i \sigma_j} \langle Kv_i, v_j \rangle = \frac{\lambda_i}{\sigma_i \sigma_j} \langle v_i, v_j \rangle = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{if } i \neq j, \end{cases} \tag{2.3.3}$$

(u_j) is an orthonormal system.

Finally, it holds

$$Ax = A\left(\sum_{j \in N} \langle x, v_j \rangle v_j \right) = \sum_{j \in N} \langle x, v_j \rangle Av_j = \sum_{j \in N} \sigma_j \langle x, v_j \rangle u_j. \tag{2.3.4}$$

Besides, this implies that (u_j) is an ONB of $\overline{\mathcal{R}(A)}$. □

In the following, we assume $N = \mathbb{N}$ which holds if $\mathcal{N}(A)$ and $\mathcal{R}(A)$ are infinite dimensional. By using the SVD, the Moore-Penrose inverse of A can be explicitly expressed as

$$A^\dagger y = \sum_{j=1}^{\infty} \sigma_j^{-1} \langle y, u_j \rangle v_j, \tag{2.3.5}$$

where $y \in \mathcal{R}(A) \oplus \mathcal{R}(A)^\perp$ [85, Satz 2.3.9]. The inversion is basically reduced to one-dimensional operations on the components of y w.r.t. the singular vectors (u_j). This formulation also provides the reason why (infinite dimensional) compact forward operators A lead to ill-posed inverse problems. According to Theorem 2.3.1, the singular values (σ_j) are a sequence converging to zero. Thus, it holds $\sigma_j^{-1} \to \infty$, which implies that the Moore-Penrose inverse is unbounded.

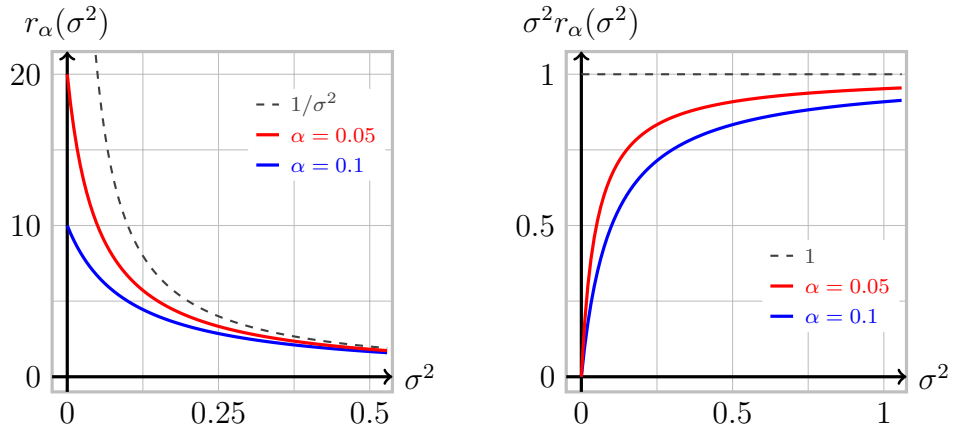


Figure 2.1: Illustration of the filter function $r_\alpha(\sigma^2) = \frac{1}{\sigma^2+\alpha}$, which is known as Tikhonov regularization [44, Chapter 5]. The left plot shows that r_α is bounded (for fixed α) and approximates $1/\sigma^2$ for $\alpha \to 0$. The right plot depicts $\sigma^2 r_\alpha(\sigma^2)$ to illustrate the damping of the singular values, which is particularly strong for the small singular values.

Remark 2.3.2. Common examples of compact operators on infinite dimensional spaces are the integration operator

$$A_1 \colon C([a,b]) \to C([a,b]), \quad (A_1 x)(t) = \int_a^t x(s)\,\mathrm{d}s \tag{2.3.6}$$

and blurring operators (convolutions)

$$A_2 \colon L^2(\Omega) \to L^2(\Omega), \quad A_2 x = k * x_{\text{pad}}, \quad x_{\text{pad}}(\omega) = \begin{cases} x(\omega) & \text{on } \Omega, \\ 0 & \text{elsewhere,} \end{cases} \tag{2.3.7}$$

with $k \in C_0^1(\mathbb{R}^n)$ and $\Omega \subset \mathbb{R}^n$ being an open and bounded set with Lipschitz boundary.

The integration operator is compact since $A_1 x$ is a continuously differentiable function and $C^1([a,b])$ is compactly embedded in $C([a,b])$ [4, Satz 8.6]. The output function of the blurring operator $A_2 x$ is also continuously differentiable due to the differentiable blurring kernel k and it holds $\partial_i(k * x_{\text{pad}}) = (\partial_i k) * x_{\text{pad}}$ [24, Theorem 3.13]. Thus, A_2 is a bounded operator into the Sobolev space $H^1(\Omega)$, which is compactly embedded into $L^2(\Omega)$ [4, Satz 8.9]. Note that this is not the case for unbounded sets $\Omega \subset \mathbb{R}^n$.

The idea of spectral regularization is to modify (2.3.5) in order to obtain stability. For this purpose, we introduce filter functions $r_\alpha \colon [0, \|A\|^2] \to \mathbb{R}$ which damp the influence of the small singular values. Based on r_α, filter-based reconstruction methods [85, Section 3.3] are defined as

$$T_\alpha(y) = \sum_{j=1}^{\infty} r_\alpha(\sigma_j^2)\sigma_j \langle y, u_j \rangle v_j. \tag{2.3.8}$$

A comparison of (2.3.5) and (2.3.8) shows that for $r_\alpha(\lambda) = \frac{1}{\lambda}$, we would get $T_\alpha = A^\dagger$. An example of a filter function is depicted in Figure 2.1.

Remark 2.3.3. There also exist alternative definitions of filter-based reconstruction methods, e.g.,

$$\tilde{T}_\alpha(y) = \sum_{j=1}^{\infty} \tilde{r}_\alpha(\sigma_j)\sigma_j^{-1}\langle y, u_j\rangle v_j. \tag{2.3.9}$$

in [56, Section 4.1.1]. In this case $\tilde{r}_\alpha \equiv 1$ corresponds to $\tilde{T}_\alpha = A^\dagger$. Apart from that, both definitions are equivalent.

Whether the reconstruction method T_α fulfills the required regularization properties of Definition 2.2.1 depends on the choice of the filter function r_α. The following theorem specifies the conditions r_α has to fulfill and provides a guideline for choosing α.

Theorem 2.3.4. *Let $r_\alpha\colon [0, \|A\|^2] \to \mathbb{R}$ be a family of piecewise continuous[2], bounded functions which fulfill*

$$\begin{aligned} &(i)\ \forall \lambda \in \left(0, \|A\|^2\right]: && \lim_{\alpha\to 0} r_\alpha(\lambda) = \frac{1}{\lambda}, \\ &(ii)\ \exists C \geqslant 0\colon \forall \lambda \in \left[0, \|A\|^2\right], \forall \alpha > 0\colon && \lambda |r_\alpha(\lambda)| \leqslant C. \end{aligned} \tag{2.3.10}$$

If $\alpha = \alpha(\delta)$ is chosen such that

$$\alpha(\delta) \to 0 \quad \text{and} \quad \delta \cdot \sqrt{\|r_{\alpha(\delta)}\|_\infty} \to 0 \tag{2.3.11}$$

hold for $\delta \to 0$, then $(T_\alpha, \alpha(\delta))$ is a convergent regularization scheme.

Proof. First, we show that T_α is a continuous operator. For arbitrary $y \in Y$ it holds

$$\begin{aligned} \|T_\alpha(y)\|^2 &= \sum_{j=1}^{\infty} |r_\alpha(\sigma_j^2)\sigma_j\langle y, u_j\rangle|^2 \leqslant \sum_{j=1}^{\infty} \sigma_j^2 |r_\alpha(\sigma_j^2)| \cdot |r_\alpha(\sigma_j^2)| \cdot |\langle y, u_j\rangle|^2 \\ &\leqslant C \cdot \|r_\alpha\|_\infty \cdot \|y\|^2. \end{aligned} \tag{2.3.12}$$

To prove the convergence property, we consider $y^\delta, y \in Y$ with $\|y^\delta - y\| \leqslant \delta$ and split up the reconstruction error

$$\|T_\alpha(y^\delta) - A^\dagger y\| \leqslant \|T_\alpha(y^\delta) - T_\alpha(y)\| + \|T_\alpha(y) - A^\dagger y\| \tag{2.3.13}$$

into a data error and an approximation error. For the data error, we use the linearity of T_α and the continuity estimation from above and obtain

$$\|T_\alpha(y^\delta) - T_\alpha(y)\| \leqslant \sqrt{C \cdot \|r_\alpha\|_\infty} \cdot \|y^\delta - y\| \leqslant \sqrt{C} \cdot \delta \cdot \sqrt{\|r_\alpha\|_\infty}. \tag{2.3.14}$$

This converges to zero for $\delta \to 0$ with the proposed parameter choice $\alpha = \alpha(\delta)$. The approximation error $\|T_\alpha(y) - A^\dagger y\|$ also converges to zero for $\alpha \to 0$ [85, Satz 3.3.1], thus, $(T_\alpha, \alpha(\delta))$ is a convergent regularization scheme. □

[2] At all discontinuities, the left-handed and right-handed limits must exist.

According to this theorem, it is possible to design custom filter functions depending on the application in order to obtain a convergent regularization scheme. However, the SVD is not known for arbitrary forward operators A, and computing it can be quite extensive, in particular for high dimensional problems. That is why other algorithms have mainly prevailed in practice.

2.4 Variational Regularization

Elements $x \in X$ with a high *data fidelity* (i.e., it holds $\|Ax - y^\delta\| \leqslant \delta$) can be considered as potential solutions of the inverse problem $Ax = y$. Variational reconstruction methods aim at finding those elements by formulating the task as a minimization problem

$$\min_{x \in X} \frac{1}{2}\|Ax - y^\delta\|^2. \tag{2.4.1}$$

To obtain a solution, analytical or numerical optimization methods can be applied (Section 2.6). However, in case of ill-posedness, additional regularization is needed to guarantee the existence of solutions, as the following remark shows.

Remark 2.4.1. For ill-posed inverse problems in the sense of Nashed (Definition 2.1.2), the minimum problem (2.4.1) usually does not admit a solution. More precisely, there only exists a solution if the projection of y^δ onto $\overline{\mathcal{R}(A)}$ is even contained in $\mathcal{R}(A)$.

This can be seen by decomposing $y^\delta = y_1^\delta + y_2^\delta$ into $y_1^\delta \in \overline{\mathcal{R}(A)}$ and $y_2^\delta \in \mathcal{R}(A)^\perp$. Then, it holds

$$\|Ax - y^\delta\|^2 = \|Ax - y_1^\delta\|^2 + \|y_2^\delta\|^2. \tag{2.4.2}$$

By considering a sequence $(x_k) \subset X$ with $Ax_k \to y_1^\delta$, it follows $\inf_x \|Ax - y_1^\delta\| = 0$. For $y_1^\delta \in \overline{\mathcal{R}(A)} \backslash \mathcal{R}(A)$, this infimum cannot be attained by any x.

To regularize the problem, we introduce a so-called *penalty term* $R\colon X \to \mathbb{R} \cup \{\infty\}$. This functional can encode prior knowledge which we might have about the ground truth $x^\dagger$, and penalize unwanted features in x with high values of $R(x)$. A solution of the inverse problem is thus expected to have small values w.r.t. both (2.4.1) and R. By bringing this together, we define a variational regularization method

$$\min_{x \in X} \frac{1}{2}\|Ax - y^\delta\|^2 + \alpha R(x), \tag{2.4.3}$$

where the parameter $\alpha > 0$ balances between data fidelity and penalty term. It is also possible to consider other data discrepancy terms of a more general form $F(Ax, y^\delta)$ with $F\colon Y \times Y \to \mathbb{R}$ instead of the squared norm, as done in [21], for example.

One of the most basic penalty terms is $R(x) = \frac{1}{2}\|x\|^2$, known as Tikhonov regularization [44, Chapter 5]. Due to its differentiability, it allows for computing the first-order optimality condition of (2.4.3) as

$$(A^*A + \alpha \mathrm{Id})x = A^* y^\delta. \tag{2.4.4}$$

This can be interpreted as a regularized normal equation (Section 2.1). Another commonly used penalty term is $R(x) = \sum_j |\langle x, b_j \rangle|$, i.e., the ℓ^1-norm of the components of x w.r.t. an ONB $(b_j) \subset X$. Using this in (2.4.3) enforces the solution x to be sparse w.r.t. (b_j) [21, Section 3.2], i.e., only finitely many of the components are non-zero. For the task of image denoising, the so-called total variation (TV) regularization has been established [89], which is based on the penalty term $R(x) = \|\nabla x\|_1$. It enforces the edges ∇x of an image x to be sparsely distributed, and it is one of the most frequently used penalty terms for imaging problems.

Whether the minimizing problem (2.4.3) admits a solution depends on the properties of R. In order to define a convergent regularization scheme in the sense of Definition 2.2.1, we also have to verify the uniqueness, stability, and convergence of solutions. These properties and the required conditions on R are covered in the remainder of this section.

Existence and Uniqueness

We start with the definitions of four fundamental properties for functionals. The first one, convexity, is graded into three levels.

Definition 2.4.2 (convex)**.** *A functional $R\colon X \to \mathbb{R} \cup \{\infty\}$ is called*

- *convex if for all $x_1, x_2 \in X$ and $\lambda \in [0,1]$ it holds*

$$R\big(\lambda x_1 + (1-\lambda)x_2\big) \leqslant \lambda R(x_1) + (1-\lambda)R(x_2), \tag{2.4.5}$$

- *strictly convex if for all $x_1, x_2 \in X$ s.t. $x_1 \neq x_2$ and for all $\lambda \in (0,1)$ it holds*

$$R\big(\lambda x_1 + (1-\lambda)x_2\big) < \lambda R(x_1) + (1-\lambda)R(x_2), \tag{2.4.6}$$

- *m-strongly convex if there exists $m > 0$ such that $R - \frac{m}{2}\|\cdot\|^2$ is convex.*

Definition 2.4.3 (coercive)**.** *A functional $R\colon X \to \mathbb{R} \cup \{\infty\}$ is called coercive if $\|x\| \to \infty$ implies $R(x) \to \infty$.*

Definition 2.4.4 (weakly lower semicontinuous)**.** *A functional $R\colon X \to \mathbb{R} \cup \{\infty\}$ is called weakly lower semicontinuous if for $x_k \rightharpoonup x$ it holds $R(x) \leqslant \liminf_{k\to\infty} R(x_k)$.*

Definition 2.4.5 (proper)**.** *A functional $R\colon X \to \mathbb{R} \cup \{\infty\}$ is called proper if there exists at least one $x \in X$ such that $R(x) < \infty$.*

Using these properties, one can prove a theorem for the existence and uniqueness of solutions of variational regularization schemes [57, Theorem 3.1], [24, Theorem 6.17].

Theorem 2.4.6 (existence and uniqueness)**.** *If R is proper, weakly lower semicontinuous, coercive, and bounded from below, there exists a solution of (2.4.3). If R is additionally strictly convex, the minimizer of (2.4.3) is unique.*

Proof. Since R is bounded from below, the infimum of $J(x) = \frac{1}{2}\|Ax - y^\delta\|^2 + \alpha R(x)$ is a real number. Thus, there exists a minimizing sequence $(x_k) \subset X$ such that $J(x_k)$ converges to this infimum. Then, $J(x_k)$ will in particular be bounded. Due to the coercivity of R, J is also coercive which implies that (x_k) itself is bounded.

Since X is a Hilbert space, there exists a weakly convergent subsequence (x_{k_l}) with $x_{k_l} \rightharpoonup \hat{x}$ [4, Satz 6.10]. By using the weak lower semicontinuity of R and $\frac{1}{2}\|A \cdot - y^\delta\|^2$, we obtain

$$J(\hat{x}) \leqslant \liminf_{l \to \infty} J(x_{k_l}) = \inf_{x \in X} J(x). \tag{2.4.7}$$

Thus, $\hat{x}$ is a minimizer of F.

If R, and thus J, is strictly convex, then for two different elements $x_1, x_2 \in X$ with $J(x_1) = J(x_2)$, the element $\frac{1}{2}(x_1 + x_2)$ has an even lower value of J. That is why the minimizer of J is unique in this case. □

We note that the ℓ^1-penalty term and the TV functional do not fulfill all of the required assumptions. Both of them are only convex but not strictly convex and TV is only coercive if, for example, the boundary values or the average value of x is kept fixed. However, these deficiencies can be easily overcome, e.g., by adding $\varepsilon\|\cdot\|^2$ for a small constant $\varepsilon > 0$.

Stability

For convex functionals, there exists a generalization of the concept of derivatives. The so-called *subdifferential* [86, Section 23] of R at $x \in X$ is defined as the set

$$\partial R(x) = \{v \in X \,|\, \forall z \in X \colon R(z) \geqslant R(x) + \langle v, z - x\rangle\}. \tag{2.4.8}$$

Some of its properties are analogous to the classical gradient. In case of classical differentiability, it holds $\partial R(x) = \{\nabla R(x)\}$. Besides, if x is a minimizer of the functional, the zero-element is contained in the subdifferential. This will also be used in the proof of the stability theorem 2.4.7.

With the assumptions that were sufficient for the existence of solutions of (2.4.3) (Theorem 2.4.6), one only obtains weak stability results [57, Theorem 3.2]. That means, a convergent sequence of data $y_k^\delta \to y^\delta$ does not imply the convergence of the corresponding sequence of solutions (x_k). Instead, only the weak limits of subsequences (x_{k_l}) can be guaranteed to be minimizers w.r.t. y^δ.

For an actual stability guarantee, a strongly convex penalty term is needed. This allows even for an explicit estimation of the Lipschitz constant of the reconstruction method $(y^\delta \mapsto x)$. As the following theorem shows, the desired amount of stability can then be controlled via the regularization parameter α.

Theorem 2.4.7 (stability)**.** *Let R be m-strongly convex, proper, weakly lower semicontinuous, coercive, and bounded from below. Then, the solutions of (2.4.3) depend Lipschitz continuously on y^δ. In particular, for $y_1^\delta, y_2^\delta \in Y$ and the corresponding solutions $x_1, x_2 \in X$ of (2.4.3) it holds $\|x_1 - x_2\| \leqslant \frac{1}{2\sqrt{m\alpha}} \cdot \|y_1^\delta - y_2^\delta\|$.*

Proof. First, we write $R = \tilde{R} + \frac{m}{2}\|\cdot\|^2$ as the sum of a convex functional $\tilde{R}$ and the squared norm. By Theorem 2.4.6, there exist unique solutions x_i (for $i = 1, 2$) of

$$\min_{x \in X} \frac{1}{2}\|Ax - y_i^\delta\|^2 + \alpha\tilde{R}(x) + \frac{\alpha m}{2}\|x\|^2. \tag{2.4.9}$$

According to the first-order optimality condition, it holds

$$0 \in (A^*A + m\alpha\mathrm{Id})x_i - A^*y_i^\delta + \alpha\partial\tilde{R}(x_i). \tag{2.4.10}$$

Thus, we can choose $v_i \in \partial\tilde{R}(x_i)$ from the subdifferential such that

$$(A^*A + m\alpha\mathrm{Id})x_i + \alpha v_i = A^*y_i^\delta. \tag{2.4.11}$$

We can now subtract this equation for $i = 2$ from $i = 1$ and obtain

$$(A^*A + m\alpha\mathrm{Id})(x_1 - x_2) + \alpha(v_1 - v_2) = A^*(y_1^\delta - y_2^\delta). \tag{2.4.12}$$

Taking the inner product of both sides of the equation with $x_1 - x_2$ leads to

$$\begin{aligned} &\langle A^*A(x_1 - x_2), x_1 - x_2\rangle + \langle m\alpha(x_1 - x_2), x_1 - x_2\rangle + \langle \alpha(v_1 - v_2), x_1 - x_2\rangle \\ &= \langle A^*(y_1^\delta - y_2^\delta), x_1 - x_2\rangle. \end{aligned} \tag{2.4.13}$$

Basic transformations result in

$$\|A(x_1 - x_2)\|^2 + m\alpha\|x_1 - x_2\|^2 + \alpha\langle v_1 - v_2, x_1 - x_2\rangle = \langle y_1^\delta - y_2^\delta, A(x_1 - x_2)\rangle. \tag{2.4.14}$$

On the right hand side, one can apply the inequalities of Cauchy-Schwartz and Young, which imply

$$\begin{aligned} &\langle y_1^\delta - y_2^\delta, A(x_1 - x_2)\rangle \\ &\leqslant \|y_1^\delta - y_2^\delta\| \cdot \|A(x_1 - x_2)\| = \frac{1}{\sqrt{2}}\|y_1^\delta - y_2^\delta\| \cdot \sqrt{2}\|A(x_1 - x_2)\| \\ &\leqslant \frac{1}{4}\|y_1^\delta - y_2^\delta\|^2 + \|A(x_1 - x_2)\|^2. \end{aligned} \tag{2.4.15}$$

Combining this with (2.4.14) leads to

$$m\alpha\|x_1 - x_2\|^2 + \alpha\langle v_1 - v_2, x_1 - x_2\rangle \leqslant \frac{1}{4}\|y_1^\delta - y_2^\delta\|^2. \tag{2.4.16}$$

Due to the monotonicity of subdifferentials [19, Section 1], it holds $\langle v_1 - v_2, x_1 - x_2\rangle \geqslant 0$ and we end up with

$$\|x_1 - x_2\|^2 \leqslant \frac{1}{4m\alpha}\|y_1^\delta - y_2^\delta\|^2, \tag{2.4.17}$$

which is the desired stability estimate. □

Convergence

In the following, we study the convergence of solutions of (2.4.3) for vanishing noise level δ. To highlight the dependency on δ and the regularization parameter α, we denote those solutions by x_α^δ. For the ground truth $x^\dagger$, we generalize the term of a minimum-norm solution (2.1.1) to an R-minimizing solution

$$\arg\min_{x \in X} R(x) \quad \text{s.t. } Ax = y^\dagger. \tag{2.4.18}$$

If R is strictly convex, the minimizer is uniquely determined.

Exploiting the assumptions that were sufficient for the existence of solutions of (2.4.3) (Theorem 2.4.6), it is possible to derive a result about weak convergence of subsequences of (x_α^δ) to R-minimizing solutions [57, Theorem 3.5]. This can be improved by employing two modifications. The first one is a so-called *source condition*

$$\exists w \in Y: \quad A^* w \in \partial R(x^\dagger), \tag{2.4.19}$$

which is assumed for the ground truth. Second, it is beneficial to measure the convergence w.r.t. the *Bregman distance*

$$D_\xi(x, z) = R(x) - R(z) - \langle \xi, x - z \rangle, \qquad \text{for } \xi \in \partial R(z), \tag{2.4.20}$$

which is based on the idea of subdifferentials. Then it is even possible to obtain a convergence rate, i.e., $D_\xi(x_\alpha^\delta, x^\dagger) \leqslant C \cdot \delta$ [26, Theorem 2] with some constant $C > 0$. For this result, the regularization parameter α has to be chosen proportionally to δ.

In the following, we present a result about convergence w.r.t. the norm of X, which relies on strong convexity of R. It does not require a source condition for $x^\dagger$, but makes no assertion about the speed of convergence.

Theorem 2.4.8 (convergence)**.** *Let R be m-strongly convex, proper, weakly lower semi-continuous, coercive and bounded from below, $x^\dagger \in X$ be an R-minimizing solution of $Ax = y^\dagger$, consider $y^\delta \in Y$ with $\|y^\delta - y^\dagger\| \leqslant \delta$, and let x_α^δ be the solution of (2.4.3). Then, with α chosen proportionally to δ, it holds $\|x_\alpha^\delta - x^\dagger\| \to 0$ for $\delta \to 0$.*

Proof. Let T_α be the operator which maps y^δ to the unique solution (Theorem 2.4.6) of (2.4.3), i.e., $T_\alpha(y^\delta) = x_\alpha^\delta$. Then, we can decompose $\|x_\alpha^\delta - x^\dagger\| \leqslant \|T_\alpha(y^\delta) - T_\alpha(y^\dagger)\| + \|T_\alpha(y^\dagger) - x^\dagger\|$ into a data error and an approximation error.

By Theorem 2.4.7, the data error can be estimated from above by

$$\|T_\alpha(y^\delta) - T_\alpha(y^\dagger)\| \leqslant \frac{1}{2\sqrt{m\alpha}} \cdot \|y^\delta - y^\dagger\| \leqslant \frac{\delta}{2\sqrt{m\alpha}}. \tag{2.4.21}$$

Due to choosing α proportional to δ, this term converges to zero for $\delta \to 0$.

It remains to prove $\|T_\alpha(y^\dagger) - x^\dagger\| \to 0$ for $\alpha \to 0$. Due to the minimizing property of

$T_\alpha(y^\dagger)$, it holds

$$\frac{1}{2}\|AT_\alpha(y^\dagger) - y^\dagger\|^2 + \alpha R\left(T_\alpha(y^\dagger)\right) \leqslant \frac{1}{2}\|Ax^\dagger - y^\dagger\|^2 + \alpha R(x^\dagger) = \alpha R(x^\dagger). \tag{2.4.22}$$

This implies $R\left(T_\alpha(y^\dagger)\right) \leqslant R(x^\dagger)$ and

$$\frac{1}{2}\|AT_\alpha(y^\dagger) - y^\dagger\|^2 \leqslant \alpha R(x^\dagger) \to 0 \qquad \text{for } \alpha \to 0. \tag{2.4.23}$$

Note that both properties still hold true if we choose an arbitrary subsequence $\left(T_{\alpha_k}(y^\dagger)\right)$, which will later be important. Due to the coercivity of R, any such subsequence is bounded and there exists a weakly convergent subsubsequence $\left(T_{\alpha_{k_l}}(y^\dagger)\right)$ [4, Theorem 6.10], i.e., $T_{\alpha_{k_l}}(y^\dagger) \rightharpoonup \hat{x}$. Using weak lower semicontinuity, we obtain

$$R(\hat{x}) \leqslant \liminf_{l\to\infty} R\big(T_{\alpha_{k_l}}(y^\dagger)\big) \leqslant R(x^\dagger), \tag{2.4.24}$$

$$\|A\hat{x} - y^\dagger\| \leqslant \liminf_{l\to\infty} \big\|AT_{\alpha_{k_l}}(y^\dagger) - y^\dagger\big\| = 0. \tag{2.4.25}$$

Since $x^\dagger$ is the unique R-minimizing solution of $Ax = y^\dagger$, it follows $\hat{x} = x^\dagger$. If any subsequence $\left(T_{\alpha_k}(y^\dagger)\right)$ has a subsubsequence which converges weakly to $x^\dagger$, then the whole sequence must already fulfill $T_\alpha(y^\dagger) \rightharpoonup x^\dagger$.

Besides, $R\left(T_\alpha(y^\dagger)\right) \leqslant R(x^\dagger)$ implies $R\left(T_\alpha(y^\dagger)\right) \to R(x^\dagger)$. Due to strong convexity, we can write $R = \tilde{R} + \frac{m}{2}\|\cdot\|^2$ with $\tilde{R}$ being convex and also weakly lower semicontinuous. Thus, it holds

$$\begin{aligned}
&\frac{m}{2}\|x^\dagger\|^2 \leqslant \liminf \frac{m}{2}\|T_\alpha(y^\dagger)\|^2 \leqslant \limsup \frac{m}{2}\|T_\alpha(y^\dagger)\|^2 \\
&\leqslant \limsup R\left(T_\alpha(y^\dagger)\right) + \limsup -\tilde{R}\left(T_\alpha(y^\dagger)\right) \\
&= \limsup R\left(T_\alpha(y^\dagger)\right) - \liminf \tilde{R}\left(T_\alpha(y^\dagger)\right) \leqslant R(x^\dagger) - \tilde{R}(x^\dagger) = \frac{m}{2}\|x^\dagger\|^2,
\end{aligned} \tag{2.4.26}$$

which leads to $\|T_\alpha(y^\dagger)\| \to \|x^\dagger\|$. Since X is a Hilbert space, $\|T_\alpha(y^\dagger)\| \to \|x^\dagger\|$ together with $T_\alpha(y^\dagger) \rightharpoonup x^\dagger$ is equivalent to $T_\alpha(y^\dagger) \to x^\dagger$. □

2.5 Bayesian View on Variational Regularization

In addition to the functional analytic view from the previous sections, inverse problems can also be analyzed from the statistical side, using random variables and probability distributions. To do so, we consider the spaces X and Y to be finite dimensional, which enables us to easily work with probability density functions (pdf). Detailed introductions into this type of theory for inverse problems (called *Bayesian* theory) are provided by [61, 100, 36, 13].

Since the ground truth $x^\dagger$ is unknown, it can be modeled as the realization of a random variable $\mathcal{X} \sim p_\mathcal{X}$. The pdf $p_\mathcal{X}\colon X \to [0,\infty)$ represents the so-called *prior distribution*. This is related to the prior knowledge which we might have about $x^\dagger$, but typically we

cannot expect to know the pdf explicitly. Similarly, the additive noise on y^δ is also modeled as a random variable $\mathcal{E} \sim p_\mathcal{E}$. Thus, y^δ is a realization of a random variable $\mathcal{Y} = A\mathcal{X} + \mathcal{E}$. Exploiting this dependency between the three random variables, the conditional pdf of $\mathcal{Y}$ given $\mathcal{X}$ can be computed as $p_{\mathcal{Y}|\mathcal{X}}(y^\delta|x) = p_\mathcal{E}(y^\delta - Ax)$.

The inverse problem, however, consists in finding $\mathcal{X}$ given $\mathcal{Y}$, which is why we are particularly interested in the conditional pdf $p_{\mathcal{X}|\mathcal{Y}}$. It represents the probability distribution of all possible solutions of the inverse problem and is thus called *posterior distribution*. The dependency between prior and posterior distribution is given by Bayes' theorem [61, Theorem 3.1]

$$p_{\mathcal{X}|\mathcal{Y}}(x|y^\delta) = \frac{p_{\mathcal{Y}|\mathcal{X}}(y^\delta|x) \cdot p_\mathcal{X}(x)}{p_\mathcal{Y}(y^\delta)}. \tag{2.5.1}$$

Thus, to solve the inverse problem one needs to determine $p_\mathcal{X}$ and $p_\mathcal{E}$.

The element x for which $p_{\mathcal{X}|\mathcal{Y}}(x|y^\delta)$ is maximal is called the maximum a posteriori (MAP) estimate for a solution of the inverse problem. If we assume the noise distribution to be Gaussian, i.e.

$$p_\mathcal{E}(\eta) = \frac{1}{(2\pi)^{\frac{n}{2}}\delta^n} \cdot \exp\left(-\frac{1}{2\delta^2}\|\eta\|^2\right), \tag{2.5.2}$$

we can find an interesting relation between Bayesian theory and variational regularization (Section 2.4). The MAP estimate is the solution of

$$\max_{x \in X} p_{\mathcal{X}|\mathcal{Y}}(x|y^\delta). \tag{2.5.3}$$

We can now use the log function and Bayes' theorem (2.5.1) to deduce that (2.5.3) is equivalent to

$$\min_{x \in X} -\log\left(p_{\mathcal{X}|\mathcal{Y}}(x|y^\delta)\right) \tag{2.5.4}$$

$$\Leftrightarrow \quad \min_{x \in X} -\log\left(\frac{p_{\mathcal{Y}|\mathcal{X}}(y^\delta|x) \cdot p_\mathcal{X}(x)}{p_\mathcal{Y}(y^\delta)}\right) \tag{2.5.5}$$

$$\Leftrightarrow \quad \min_{x \in X} -\log\left(p_{\mathcal{Y}|\mathcal{X}}(y^\delta|x)\right) - \log\left(p_\mathcal{X}(x)\right) + \log\left(p_\mathcal{Y}(y^\delta)\right) \tag{2.5.6}$$

$$\Leftrightarrow \quad \min_{x \in X} -\log\left(p_\mathcal{E}(y^\delta - Ax)\right) - \log\left(p_\mathcal{X}(x)\right) \tag{2.5.7}$$

$$\Leftrightarrow \quad \min_{x \in X} \log\left((2\pi)^{\frac{n}{2}}\delta^n\right) + \frac{1}{2\delta^2}\|y^\delta - Ax\|^2 - \log\left(p_\mathcal{X}(x)\right) \tag{2.5.8}$$

$$\Leftrightarrow \quad \min_{x \in X} \frac{1}{2}\|Ax - y^\delta\|^2 - \delta^2 \log\left(p_\mathcal{X}(x)\right). \tag{2.5.9}$$

Note that the minimizing problem (2.5.9) corresponds to a variational regularization scheme (2.4.3) with $-\log \circ\, p_\mathcal{X}$ being the penalty term. In other words, variational regularization equals MAP estimation and the penalty functional R encodes the prior distribution. The regularization parameter α equals the squared noise level δ^2, i.e., the variance of $p_\mathcal{E}$. Besides, the equations (2.5.7) to (2.5.9) show that the data discrepancy term $\frac{1}{2}\|Ax - y^\delta\|^2$ is the optimal choice for gaussian noise. For other types of noise, one can choose other data discrepancy terms according to (2.5.7).

Remark 2.5.1. The Bayesian view demonstrates that data fidelity and prior knowledge can be modeled independently of each other. The first only depends on the noise distribution, the second only on the prior distribution. That is why many reconstruction methods, in particular the deep learning approaches from Section 3.3.2 and 3.3.3, are separating these tasks too. This leads to interpretability and flexibility.

However, in the end, data discrepancy term and penalty term have to complement each other to work well together. Parts of $x^\dagger$ which are well-determined solely from the data y^δ do not necessarily have to be regularized by a penalty term. Thus, sometimes it can be easier to learn the complete reconstruction process at once (Section 3.3.1).

We also analyzed iResNets from the Bayesian point of view. There, we found a relation to the mean of the posterior distribution, i.e., the conditional expectation $\mathbb{E}(\mathcal{X}|\mathcal{Y})$ [11].

2.6 Solving Variational Minimization Problems

Variational reconstruction methods with appropriate penalty terms are convergent regularization schemes (Section 2.4), but they define the regularized solution only implicitly via a minimization problem. Thus, an additional algorithm is needed for an explicit computation, or approximation, of the minimizer. Most of the time, iterative numerical methods are used.

The data fidelity term $\frac{1}{2}\|Ax-y^\delta\|^2$ is differentiable with $A^*(Ax-y^\delta)$ being the gradient. If the penalty term R is also differentiable, a gradient descent algorithm

$$x^{k+1} = x^k - \lambda \left(A^*(Ax^k - y^\delta) + \alpha \nabla R(x^k)\right) \tag{2.6.1}$$

with a step size $\lambda > 0$ can be used to approximate the minimizer of (2.4.3). For this to work reliably, convexity of R is crucial since that prevents the algorithm from converging to a local minimum.

The ℓ^1-penalty term $R(x) = \sum_j |\langle x, b_j\rangle|$, however, is not differentiable. For this functional the iterative shrinkage-thresholding algorithm (ISTA) [37]

$$x^{k+1} = S_{\lambda\alpha}\left(x^k - \lambda A^*(Ax^k - y^\delta)\right). \tag{2.6.2}$$

has been developed. The soft-thresholding function $S_{\lambda\alpha}(x_j) = \operatorname{sgn}(x_j)\max\{0, |x_j| - \lambda\alpha\}$ is applied on each component w.r.t. the ONB $(b_j) \subset X$, and therefore causes a decrease of the ℓ^1-functional. Using the so-called *proximal mapping* [81, Section 1.1]

$$\operatorname{prox}_R(z) = \operatorname*{arg\,min}_{x\in X} \frac{1}{2}\|x - z\|^2 + R(x), \tag{2.6.3}$$

this algorithm can also be generalized to other penalty terms R. This is known as the proximal gradient method [81, Section 4.2]

$$x^{k+1} = \operatorname{prox}_{\lambda\alpha R}\left(x^k - \lambda A^*(Ax^k - y^\delta)\right). \tag{2.6.4}$$

Another well-known approach is the so-called primal-dual algorithm [28]. It was developed for minimization problems of the form

$$\min_{x \in X} F(Kx) + G(x), \tag{2.6.5}$$

where $K \colon X \to Z$ is a linear operator mapping to a Hilbert space Z, and $F \colon Z \to [0, \infty]$ and $G \colon X \to [0, \infty]$ are proper, convex and lower semicontinuous functions. By using the convex conjugate F^* of F, one can introduce a dual variable $z \in Z$ and reformulate the problem as

$$\min_{x \in X} \max_{z \in Z} \langle Kx, z \rangle + G(x) - F^*(z). \tag{2.6.6}$$

Based on this, one can compute proximal gradient steps w.r.t. z and x. These are the main components of the final algorithm

$$\begin{aligned} z^{k+1} &= \operatorname{prox}_{\sigma F^*}(z^k + \sigma K \bar{x}^k), \\ x^{k+1} &= \operatorname{prox}_{\tau G}(x^k - \tau K z^{k+1}), \\ \bar{x}^{k+1} &= x^{k+1} + \rho(x^{k+1} - x^k), \end{aligned} \tag{2.6.7}$$

where σ, τ and ρ are step size parameters. More details and convergence theory are covered in [28].

There are two possibilities to apply this algorithm for variational regularization methods. First, for $K = A$, $F(z) = \frac{1}{2}\|z - y^\delta\|^2$ and $G = \alpha R$, the primal variable x is optimized in the solution space X and the dual variable z in the data space Y. This is particularly well-suited for problems where X and Y are fundamentally different (e.g., CT reconstruction). The learned primal-dual architecture [3] is based on this idea (Section 3.3.1). Second, the algorithm can be applied for specific penalty terms like $\mathrm{TV}(x) = \|\nabla x\|_1$. By choosing $K = \nabla$, $F(z) = \alpha\|z\|_1$ and $G(x) = \frac{1}{2}\|Ax - y^\delta\|^2$, the TV functional can be decomposed. This way, the complicated proximal mapping of TV is avoided.

A different idea of variable splitting is used for the alternating direction method of multipliers (ADMM) [23]. There, (2.4.3) is reformulated into a two-variable optimization problem with side constraint

$$\min_{x, z \in X} \frac{1}{2}\|Ax - y^\delta\|^2 + \alpha R(z), \qquad \text{s.t. } x = z. \tag{2.6.8}$$

By using the augmented Lagrangian one can derive the iterations

$$\begin{aligned} x^{k+1} &= \arg\min_x \frac{1}{2}\|Ax - y^\delta\|^2 + \frac{\rho}{2}\|x - z^k + u^k\|^2, \\ z^{k+1} &= \operatorname{prox}_{\frac{\alpha}{\rho}R}(x^{k+1} + u^k), \\ u^{k+1} &= u^k + x^{k+1} - z^{k+1}, \end{aligned} \tag{2.6.9}$$

where $\rho > 0$ is a step size parameter and u^k is the dual variable. More details and convergence theory can be found in [23]. Due to the splitting of the objective functional

into two parts which are easier to optimize (w.r.t. x and z, respectively), few iterations are often sufficient to come at least in a neighborhood of the exact solution. This is ideal in applications where computations have to be fast, but inexact minimization is sufficient, e.g., plug-and-play methods (Section 3.3.2).

Remark 2.6.1. There are many deep learning approaches for inverse problems which are based on one of the presented algorithms. In unrolled architectures (Section 3.3.1), the layers of the network are inspired by the structure of the iterations. For plug-and-play methods (Section 3.3.2), the proximal mappings are replaced with learned denoisers. And learned penalty terms (Section 3.3.3) need a minimization algorithm for their application.

In practice, iterative algorithms can only be executed with a finite number of steps. In case of fast convergence and enough time for computations, the resulting error is neglectable. However, terminating the iteration significantly before reaching the limit is also a type of regularization. This strategy is called *early stopping*. Performing gradient descent with early stopping just for the data discrepancy (2.4.1) without further penalty term is known as the Landweber method, which is indeed a convergent regularization scheme [53]. Early stopping is also frequently applied in deep learning algorithms, in particular with deep image prior (Section 3.3.4). We also considered this type of regularization for the analytic deep prior approach [7].

Chapter 3

Deep Learning for Inverse Problems

Machine learning is the term for algorithms which teach themselves how to solve a specific task. To do so, one typically needs (a lot of) data which is related to the task. These data are then processed by the algorithm in order to adjust itself in a way that the task can be solved afterward even with new data. *Deep learning* is a subfield of machine learning which relies on (deep) neural networks (Section 3.1).

In imaging and image processing applications, deep learning is particularly successful and often outperforms classical (purely model-based, not data-driven) methods [69], [10, 9]. Some fundamental reasons for the success of deep learning are the ability of neural networks to precisely extract features and information from big amounts of high-dimensional data and a great variability of network architectures [63].

In the context of inverse problems, deep learning is particularly desirable for obtaining prior knowledge about the ground truth. If the solution of an inverse problem should be, e.g., a natural-looking image, one has to encode this knowledge into a regularization method (Section 2.2). Doing this by hand can sometimes be hard or even intractable. In other words, it is hard to define mathematically what a natural-looking image is. This is the point where deep learning can help out. Given a (sufficiently large) dataset of ground truth samples, a neural network can extract useful information from the prior distribution (Section 2.5) to form a reconstruction method.

For all deep learning based reconstruction methods considered in this thesis, the Hilbert spaces X and Y of ground truth and data (Chapter 2) need to be discretized for the numerical application of the algorithms[1]. Accordingly, the numerically obtained reconstructions are finite dimensional. In order to still interpret them as (discretized) solutions of the original inverse problem $Ax = y$, and not just as solutions of a discretized problem, it is beneficial to consider the numerical reconstruction algorithm as a discrete version of an infinite dimensional approach acting on X and Y. Therefore, it is important that all theoretical results for reconstruction algorithms are independent of the discretization or even hold in the infinite dimensional setting. We thus do not use an extra notation for the discretized spaces, but define all reconstruction approaches directly on X and Y. Besides, the practical challenges of ill-posedness exist both in the original as well as in

[1] There also exist approaches capable of modeling infinite dimensional operators, e.g., DeepONets [68]

the discretized setting, as discussed in Section 2.1.

The next section will briefly cover the basics of convolutional neural networks and their learning process. After this, we will discuss the challenges of obtaining theoretical guarantees for deep learning approaches and properties which are desired in the context of inverse problems. For this purpose, the regularization properties from the classical theory (Section 2.2 and 2.4) will be of particular interest. Finally, we give an overview of the most common deep learning approaches for solving inverse problems.

3.1 Neural Networks

A feedforward neural network [47, Chapter 6] is a parametric function $\varphi_\theta \colon \mathbb{R}^{d_{\text{in}}} \to \mathbb{R}^{d_{\text{out}}}$ which is almost everywhere differentiable w.r.t. both the input $u \in \mathbb{R}^{d_{\text{in}}}$ and the parameters $\theta \in \mathbb{R}^{d_{\text{param}}}$. Typically, it is composed of several so-called *layers* $\varphi_{\theta_k,k}$, i.e.,

$$\varphi_\theta = \varphi_{\theta_K,K} \circ \ldots \circ \varphi_{\theta_1,1}. \tag{3.1.1}$$

More general structures are also possible. The specific design of φ_θ is called the *architecture* of the network.

Each layer is usually of a mathematically simple form, such that there are explicit formulas for the derivatives $D_{u_k}\varphi_{\theta_k,k}(u_k)$ and $D_{\theta_k}\varphi_{\theta_k,k}(u_k)$ w.r.t. input and parameters of the layer. Using the chain rule, it is thus possible to compute the derivative of the whole network, w.r.t. all parameters θ and the input u, automatically. This procedure is called *backpropagation* [55, Section 5].

For imaging applications, convolutional neural networks (CNNs) [47, Chapter 9] with convolutional layers are the most widespread type of architecture (Figure 3.1). The parameters θ of a convolutional layer are a weight tensor, which is applied as a linear filter on the input image, and a bias vector, which is added to the result. This affine linear transformation is often followed by a nonlinear activation function, which is applied pointwise.

Convolutions have two crucial properties which are beneficial for deep learning. First, they require relatively few parameters while being highly expressive. In other words, convolutions define a valuable and diverse class of transformations in a parameter-efficient way. For even more expressivity, CNNs can have multiple channels, which increases the dimensionality of the processed data. Second, convolutions are equivariant w.r.t. translations of the input, i.e., for a translated input the corresponding output is translated in the same way but stays the same apart from that. This property is quite useful in many image processing applications.

For learning how to solve a specific task with φ_θ, an algorithm for adjusting the parameters θ is required. We refer to this process as the *training* of the neural network. The most basic form is the so-called *supervised* training, which can be applied if a dataset of input-output-pairs (u_i, v_i) is available. During training, the network has to learn to

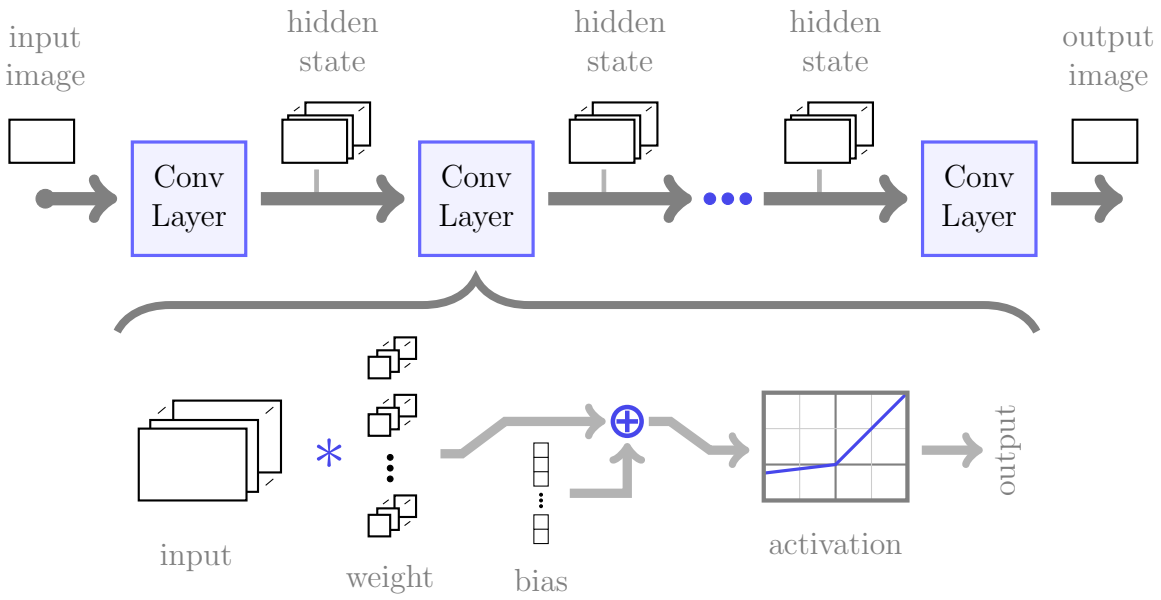


Figure 3.1: Typical architecture of a CNN. The network is composed of several convolutional layers. Each layer consists of a multi-channel convolution with a weight tensor followed by the addition of a bias vector and a pointwise nonlinear activation function. Weight and bias are learnable parameters of the network.

map each u_i to the corresponding target v_i (e.g., $\varphi_\theta(y_i^\delta) \overset{!}{\approx} x_i^\dagger$ for $u_i = y_i^\delta$ and $v_i = x_i^\dagger$). For this purpose a differentiable *loss function* $\mathcal{L}\colon \mathbb{R}^{d_{\text{out}}} \times \mathbb{R}^{d_{\text{out}}} \to \mathbb{R}$ is defined, which models the distance (or the error, respectively) between the output of the network and the desired target.

A large value of $\mathcal{L}(\varphi_\theta(u_i), v_i)$ means that the network makes large errors in the attempt to solve the task. Accordingly, the objective of the training is to find parameters θ such that the loss function is minimized on a given set of pairs $(u_i, v_i)_{i=1,\dots,N}$, the so-called *training data*. Mathematically, the training of φ_θ is an optimization process for solving

$$\min_{\theta \in \mathbb{R}^{d_{\text{param}}}} \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(\varphi_\theta(u_i), v_i) \tag{3.1.2}$$

as good as possible.

A variety of algorithms for solving such minimization problems numerically has been presented in Section 2.6. For (3.1.2), gradient descent is a suitable choice since the gradient $\nabla_\theta \mathcal{L}(\varphi_\theta(u_i), v_i)$ can be automatically computed via backpropagation. Since the number of training samples N can be very large in general, one typically chooses a smaller subset of pairs (u_i, v_i) (called *batch*) for each step. This procedure reduces the computational costs of the individual gradient steps and is called *stochastic gradient descent* [55, Section 4].

Tasks without a clear input-target-relation (e.g., learning the prior distribution from a dataset of ground truth samples $(x_i^\dagger)$) have to be learned by using *unsupervised* training. That means instead of $\mathcal{L}$, a task-specific objective function, which may depend on the network output and possible further arguments, is used. Due to the large variety of possible applications, the strategies for modeling the training objective are highly individual.

3.2 Guarantees for Deep Learning Approaches

Task-specific properties (e.g. reconstruction performance, stability) which a neural network is expected to fulfill after training are often only empirically verified. That means the network is evaluated on a test dataset, but uncertainty about the behavior on new data remains. In the context of inverse problems, this is a theoretical drawback compared to classical methods, which have provable regularization properties (Chapter 2).

There are certain types of reconstruction errors which occur particularly often. Three such errors are emphasized in [49]. The first one consists of so-called hallucinations, which means that a reconstruction may contain realistic-looking artifacts that are not present in the ground truth. Second, reconstructions may be instable w.r.t. the data y^δ as a direct consequence of ill-posedness. The third issue is an unpredictable performance on (new) data with slightly different features.

Another weakness found in neural networks is the existence of so-called adversarial examples [101]. Originally discovered in classification tasks, they denote input images with small (almost invisible) perturbations that cause misclassification. Transferred to inverse problems, this means that there may exist a small additive noise on the data y^δ such that the reconstruction with a neural network fails severely.

All these observations illustrate that theoretical guarantees ensuring the reliability of deep learning based reconstruction methods are highly desirable. Naturally, the four regularization properties from the classical theory about existence, uniqueness, stability, and convergence of solutions (Section 2.2) would be appropriate conditions. However, dependent on the concrete approach, it can be quite challenging to obtain those properties for deep learning methods. In addition, there are further aspects which are desirable in the context of deep learning for inverse problems, which we will discuss in the following.

With straightforward approaches, where a neural network directly outputs the reconstruction (Section 3.3.1), the question of existence and uniqueness is trivial. For other approaches, e.g., where neural networks are integrated into iterative methods (Section 3.3.2 and 3.3.3), these properties often remain unclear.

As elaborated above, stability is of particular importance. In the context of deep learning this is often called *robustness* w.r.t. perturbations of the input. Unfortunately, it is a quite difficult property to obtain. According to Lemma 2.1.3, reconstruction methods cannot be arbitrarily accurate and stable at the same time. Since neural networks are often great in learning highly accurate solutions, robustness can be their weakness. There are certain techniques for increasing the robustness during training, e.g., CLIP (cheap Lipschitz training) [25] or stability training [113], but they don't provide theoretical guarantees. Approaches which are provably stable often rely on Lipschitz constraints for certain network parts, e.g., iResNets [8] and plug-and-play methods (Section 3.3.2). However, this typically causes a significant reduction of the expressivity of the neural network.

The classical convergence property is to make sure that the regularized reconstruction

is close to the ground truth. However, due to its technical formulation, it is often of less practical relevance. This is because, in many applications, the noise level δ is assumed to be fixed. Low reconstruction errors for this specific noise level are then more in demand than the theoretical convergence for $\delta \to 0$. Besides, most deep learning approaches do not have a regularization parameter α, which can be chosen after the training to change the amount of regularization. Nevertheless, deep learning methods with convergence guarantees do exist [77]. For iResNets, we also derived a convergence theorem [8, Theorem 3.1].

A further desirable aspect is the property of *data consistency*, i.e., the reconstruction $\hat{x}$ has to fulfill $\|A\hat{x} - y^\delta\| \approx \|y^\dagger - y^\delta\|$. This particularly addresses the previously described phenomenon of hallucination. Typically, a reconstruction produced by a neural network has a realistic appearance, but, dependent on the approach, it might be unclear whether it is actually a valid solution w.r.t. the data y^δ. To obtain a guarantee for data consistency it is beneficial to combine classical and deep learning approaches.

In addition to rigorous guarantees, *interpretability* of reconstruction approaches is of high relevance for practical application. If the way of computing a reconstruction is traceable and comprehensible, the underlying algorithm is probably more reliable than a black-box model.

In principle, high reconstruction performances and theoretical regularization properties do not exclude each other. On the contrary, according to [52] any appropriate reconstruction method must exhibit some regularization properties otherwise it fails in practice. However, these properties are often not explicitly defined, but only a consequence of the training process, and therefore cannot be guaranteed. To overcome these challenges, I investigated regularization properties for special neural network architectures (Chapter 4). For doing so, a compromise between the expressivity of the architectures and possibilities for theoretical analysis is necessary.

3.3 Overview of Common Approaches

The possibilities of applying deep learning to inverse problems are highly diverse. Solution approaches may differ in various aspects, e.g., in terms of the underlying architecture, the role of the neural network in the reconstruction process, the data needed for training, obtainable guarantees, and possible applications. Many successful methods rely on the combination of deep learning with ideas from classical regularization [13].

For the following overview, the approaches are sorted into five categories. We are focussing primarily on general methods which are applicable to a wide range of inverse problems, in particular in the fields of imaging and image processing, and we try to cover the most common ones. Beyond that, there also exist a lot of specialized approaches for very specific inverse problems, e.g., for parameter identification in partial differential equations [78], which we will not cover. This overview is therefore not complete. Besides, we note that some approaches cannot be uniquely identified with one of the five categories.

Accordingly, the classification of methods is partly subjective.

Special emphasis is placed on the achievable guarantees for the approaches within the framework of regularization theory. In each of the five categories, we also point out the aspects which are related to iResNets and analytic deep prior.

3.3.1 End-to-End Learning

The most straightforward way to solve inverse problems with neural networks is to train them to map the data y^δ directly to the ground truth $x^\dagger$. This strategy is often called end-to-end learning. To do so, a training dataset of pairs $(x_i^\dagger, y_i^\delta)$ is needed, which is then employed in a supervised training scheme

$$\min_\theta \frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(\varphi_\theta(y_i^\delta), x_i^\dagger), \tag{3.3.1}$$

for a network architecture $\varphi_\theta \colon Y \to X$ and a suitable loss function $\mathcal{L}$.

Remark 3.3.1. The training objective (3.3.1) can also be analyzed from the Bayesian point of view (Section 2.5). For this purpose $x_i^\dagger$, y_i^δ are interpreted as realizations of random variables $\mathcal{X}$, $\mathcal{Y}$. In case of a large number N of independent training samples, the average of the loss function over i in (3.3.1) approximates the expectation value $\mathbb{E}_{\mathcal{X},\mathcal{Y}} \mathcal{L}(\varphi_\theta(\mathcal{Y}), \mathcal{X})$.

A common choice for the loss function is $\mathcal{L}(z, x) = \|z - x\|_X^2$. Then, the solution of $\min_\varphi \mathbb{E}_{\mathcal{X},\mathcal{Y}} \mathcal{L}(\varphi(\mathcal{Y}), \mathcal{X})$ is the posterior expectation $\varphi(y^\delta) = \mathbb{E}(\mathcal{X}|\mathcal{Y})(y^\delta)$ [2, Proposition 2]. In other words, end-to-end learning searches for a network φ_θ which maps y^δ to the conditional expectation value of the posterior distribution $p_{\mathcal{X}|\mathcal{Y}}$.

Thus, an end-to-end trained model is likely to be an accurate solver for the inverse problem, if there is enough training data and the model architecture is well-chosen. For iResNets, which are also trained according to (3.3.1) but with a side constraint that guarantees invertibility, we derived a similar result in [11, Lemma 4.2].

Still, with finitely many training samples there are settings in which naive end-to-end learning fails due to a high variance of the training result [71]. This can happen in particular with severely ill-posed problems and a low noise level δ, where the posterior expectation $\mathbb{E}(\mathcal{X}|\mathcal{Y})$ is itself (nearly) unstable and thus hard to learn.

In principle any network architecture $\varphi_\theta \colon Y \to X$ could be used for the training objective (3.3.1). In this section, the focus is on two very common approaches, namely *unrolled architectures* and *learned post-processing*. Both of them provide ways to integrate the forward operator A into the reconstruction process.

Unrolled Architectures. The idea of algorithm unrolling [74] (sometimes also called unfolding) starts with choosing a classical iterative algorithm $x^{k+1} = f_{\text{iter}}(x^k)$ for solving the inverse problem (Section 2.6). Then, f_{iter} is implemented as a neural network layer f_θ, i.e., some parts of f_{iter} are parametrized and made learnable. Finally, one chooses a

fixed number of iterations K (e.g., $K = 10$) and concatenates K layers to an architecture $\varphi_\theta = f_\theta \circ ... \circ f_\theta$.

The origin of algorithm unrolling for inverse problems is the so-called LISTA (learned ISTA) architecture [50], which is based on the ISTA algorithm (Section 2.6)

$$x^{k+1} = S_{\lambda\alpha}\left((\mathrm{Id} - \lambda A^* A)x^k + \lambda A^* y^\delta\right). \tag{3.3.2}$$

One then replaces the linear operations $\mathrm{Id} - \lambda A^* A$ and λA^* by learnable matrices W_1 and W_2 and implements the soft-thresholding function $S_{\lambda\alpha}$ as an activation function. The resulting layer is

$$f_\theta(x^k, y^\delta) = S_\gamma(W_1 x^k + W_2 y^\delta) \tag{3.3.3}$$

with the learned parameters $\theta = (W_1, W_2, \gamma)$, which are shared across all layers. For the theory of analytic deep prior [7] in particular, the LISTA architecture plays a fundamental role (Section 4.1).

The idea of LISTA can be further generalized in different ways. Using a different activation function corresponds to changing the proximal mapping [99]. The most commonly used activation functions are in fact proximal mappings with respect to convex functionals [32]. Besides, algorithm unrolling can also be applied to ADMM, as done in [111] for compressive sensing in magnetic resonance imaging, to primal-dual methods, as done in [109], or to time steps of diffusion equations, as done in [31].

For the Helsinki Deblur Challenge [10], we used the so-called *learned gradient descent* (LGD) [1] (among others), which comes with several crucial advantages. In contrast to LISTA, for LGD the proximal mapping is modeled as a subnetwork and the affine linear part $A^*(Ax^k - y^\delta)$ is left original. This way, the architecture has built-in knowledge of the true forward operator. Besides, it is very natural to make the proximal mapping a learnable function, because this is the part which is responsible for the regularization and should thus highly depend on the (training) data. One further improvement is the use of so-called memory channels (additional channels making the architecture wider and more expressive). These are motivated by accelerated algorithms which make use of several of the previous iterates to compute the next iterates.

When the spaces X and Y are fundamentally different (e.g., for CT problems), the *Learned Primal-Dual* (LPD) algorithm [3] is particularly well-suited. The scheme

$$y^{k+1} = f_{\theta_k^{\mathrm{d}}}(y^k,\, Ax^k,\, y^\delta), \tag{3.3.4}$$

$$x^{k+1} = g_{\theta_k^{\mathrm{p}}}(x^k,\, A^* y^{k+1}), \tag{3.3.5}$$

based on the primal-dual algorithm (Section 2.6), makes use of two different subnetworks. Since the dual network $f_{\theta_k^{\mathrm{d}}}$ acts on Y and the primal network $g_{\theta_k^{\mathrm{p}}}$ acts on X, the scheme is able to learn features in both spaces. LPD also makes use of memory channels (as LGD) and the weights θ_k^{p}, θ_k^{d} are individual for each iteration $k = 1, ..., K$, which further increases the expressivity of the network. This type of architecture has frequently pro-

duced outstanding results, e.g., for low-dose and sparse-angle CT reconstruction [65], and we used it very successfully for the Helsinki Tomography Challenge [9].

Remark 3.3.2. Unrolled architectures have a long list of fundamental advantages. The possibility of incorporating the forward operator A and a simple but effective training setup often results in a high reconstruction quality. To achieve this, unrolled architectures typically only require relatively few parameters [74]. Besides, the similarity to classical iterative algorithms promotes the interpretability of the networks.

However, rigorous theoretical guarantees (like stability) are rare, since they require hard restrictions on the architectures. We found iResNets as an adequate compromise between stability and expressivity [12]. Furthermore, some works developed strategies to enforce the subnetworks of unrolled architectures to behave like proximal mappings. In [27], equivariant architectures are used to encode symmetries of the prior distribution. A recent work [34] trains the subnetworks to mimic the proximal mapping of a simultaneously learned penalty term. Besides, [54] uses an LPD architecture for learning a model correction of an incorrect forward operator A.

Learned Post-Processing. Classical algorithms are strong in terms of theoretical guarantees but they show weaknesses when having to reconstruct natural-looking images. CNNs, however, are particularly well-suited for imaging tasks like denoising or removing artifacts. Thus, using a neural network to enhance the output of a classical reconstruction algorithm is another promising way to combine traditional and deep learning methods. Such approaches are sometimes distinguished from end-to-end learning since the neural network $f_\theta \colon X \to X$ does not get the data y^δ as input but is only used for post-processing. But if one considers the classical algorithm $T \colon Y \to X$ to be part of the whole reconstruction scheme $\varphi_\theta = f_\theta \circ T$, the training (3.3.1) is equivalent.

Post-processing methods are often applied for medical imaging, e.g., CT reconstruction. Two typical examples are [29], where an autoencoder network is used to enhance FBP (filtered back projection) reconstructions of low-dose CT measurements (i.e., y^δ contains more noise than usual) and [112], where ART (algebraic reconstruction technique) reconstructions from few-view CT measurements (i.e., y^δ contains measurements of less angles than usual) are enhanced by a small CNN. Further post-processing methods for CT are listed and evaluated in [65]. For PET (positron emission tomography), another medical imaging technique, [80] provides an extensive overview. The most commonly used architectures are U-Nets [88]. For the Helsinki Tomography Challenge [9], we also used a post-processing method (among others).

To obtain theoretical regularization guarantees, [91] introduced deep nullspace learning. This approach clearly separates the task of the classical reconstruction method and the post-processing network, by restricting the latter to only act on the nullspace of A. Thus, the classical method reconstructs the parts of x which are visible in the measurements y^δ and the network adds the components from the nullspace using prior information. This way, data fidelity is guaranteed. For stability and convergence, the post-processing

network must be Lipschitz continuous, a similar condition is fundamental in the context of iResNets [8].

3.3.2 Plug-and-Play

In many classical iterative approaches for solving inverse problems, a proximal mapping is responsible for the actual regularization (see Section 2.6). By recalling the definition

$$\operatorname{prox}_{\alpha R}(z) = \operatorname*{arg\,min}_{x \in X} \frac{1}{2}\|x - z\|^2 + \alpha R(x), \tag{3.3.6}$$

we observe that the proximal mapping corresponds to a variational regularization scheme for a denoising problem, i.e., an inverse problem with $A = \mathrm{Id}$. So-called *plug-and-play* methods make use of this observation by replacing the proximal mapping in an iterative scheme with general denoising algorithms.

The idea was first proposed in [107] for the ADMM algorithm (2.6.9), which becomes

$$\begin{aligned} x^{k+1} &= \operatorname*{arg\,min}_{x} \frac{1}{2}\|Ax - y^\delta\|^2 + \frac{\rho}{2}\|x - z^k + u^k\|^2, \\ z^{k+1} &= D_\theta(x^{k+1} + u^k), \\ u^{k+1} &= u^k + x^{k+1} - z^{k+1} \end{aligned} \tag{3.3.7}$$

if a denoiser $D_\theta \colon X \to X$ replaces the proximal mapping. The same idea is also possible for other iterative algorithms, e.g., proximal gradient descent

$$x^{k+1} = D_\theta(x^k - \lambda A^*(Ax^k - y^\delta)). \tag{3.3.8}$$

The denoiser D_θ can, e.g., be modeled as a neural network [30]. To train it, only a set of clean data samples $\{x_i^\dagger\}$ is necessary, the data y^δ is only needed after training in the iterative scheme. This way, plug-and-play algorithms are a clever combination of modeling (the forward operator) and learning (the regularization). The main motivation to use ADMM is that only a few iterations, and thus applications of D_θ, are often already sufficient (Section 2.6). However, convergence of the plug-and-play iteration can only be guaranteed under specific conditions on D_θ.

Remark 3.3.3. For the plug-and-play variant (3.3.8), which is based on proximal gradient descent, theoretical guarantees can be quite easily derived. If the denoiser D_θ is contractive (i.e., $\operatorname{Lip}(D_\theta) < 1$) and the step size fulfills $\lambda \leqslant 2\|A\|^2$, the iteration in (3.3.8) converges to a unique fixed point (by the fixed-point theorem of Banach [110, Section IV.7]). Besides, this fixed point depends Lipschitz continuously on y^δ [42, Corollary 23].

Thus, with a contractive denoiser, one obtains existence, uniqueness, and stability of solutions. Interestingly, for iResNets the exact same properties are also based on the contractivity of specific network parts [8].

Many works have addressed the problem of obtaining a contractive or at least a non-

expansive denoiser. In [105], it is proposed to project the weights of convolutional layers to a certain bounded set to make the convolution nonexpansive. This projection step can be included in the normal training routine to obtain a nonexpansive CNN. For convolutional iResNets, we used a similar strategy [12]. A drawback of this approach is that the expressivity of the network gets restricted too. A different approach uses the fact that solution trajectories of ordinary differential equations (ODEs) will not deviate from each other if the right hand side of the ODE is the negative of a monotone operator [93]. By modeling this operator with a monotone neural network one obtains a learnable nonexpansive mapping via the ODE. A possibility to put fewer restrictions on the network architecture is applying a penalty functional on the gradient of the network during training [82]. However, the gradient can only be evaluated on finitely many points, thus, there is no guarantee that the resulting network is overall nonexpansive.

While plug-and-play methods are based on algorithms for variational minimization, for a general denoiser D_θ there might exist no corresponding functional which is minimized. The so-called regularization by denoising (RED) approach [87, 83] addresses this lack of interpretability by defining a penalty term $R(x) = \langle x, x - D_\theta(x) \rangle$. Under some conditions on D_θ, the gradient of R fulfills $\nabla R(x) = x - D_\theta(x)$ and can thus be used in any gradient-based method for minimizing

$$\frac{1}{2}\|Ax - y^\delta\|^2 + \alpha R(x). \tag{3.3.9}$$

Alternatively, it is also possible to first model a penalty term R_θ with a neural network architecture and then define $D_\theta = \mathrm{Id} - \nabla R_\theta$ to be a so-called gradient step denoiser [59]. In this setting, ∇R_θ being Lipschitz continuous is sufficient for at least obtaining some weak results about convergence, e.g., $\|x^{k+1} - x^k\| \to 0$, but no rigorous regularization properties [58, 104].

Obtaining strong theoretical guarantees while using really powerful denoisers remains an open problem. Besides, plug-and-play algorithms can be quite slow in applications because evaluating the denoiser is necessary in each iteration. However, due to the separation into a data fidelity part and a regularization part, the approach exhibits high flexibility and interpretability.

3.3.3 Learned Penalty Terms

Variational regularization methods (Section 2.4) are highly interpretable and provide strong theoretical guarantees. For a successful application in practice the choice of the penalty functional, encoding the prior knowledge about the ground truth data, is crucial. This task can be solved with deep learning by modeling the penalty term $R_\theta \colon X \to \mathbb{R}$ as a neural network. It has to be trained to output small values for clean ground truth samples $x^\dagger$ and large values for any kind of disturbed inputs. After training, R_θ can be

applied in a variational scheme like

$$\min_{x \in X} \frac{1}{2} \|Ax - y^\delta\|^2 + \alpha R_\theta(x) \tag{3.3.10}$$

to solve inverse problems.

Remark 3.3.4. A learned penalty term is a very close connection between deep learning and the classical regularization theory. However, to also obtain the theoretical guarantees from Section 2.4, R_θ has to fulfill the assumptions of Theorem 2.4.6, 2.4.7 and 2.4.8. Most common neural network architectures are already continuous, thus, the crucial property is strong convexity of R_θ. If this is fulfilled, then R_θ is also weakly lower semicontinuous, coercive, and bounded from below. With these properties, the learned regularization method exhibits theoretical guarantees about the existence, uniqueness, stability, and convergence of solutions.

There are several different strategies for the training of R_θ. The so-called adversarial regularizer [70] is trained to penalize reconstructions of a pseudo-inverse $A^\dagger$, which usually contain artifacts due to insufficient regularization. In each training step, one randomly chooses a ground truth sample $x_i^\dagger$ and a data sample y_j^δ and, additionally, one random point ξ on the line between $x_i^\dagger$ and $A^\dagger y_j^\delta$. Then, one gradient step w.r.t.

$$R_\theta(x_i) - R_\theta(A^\dagger y_j^\delta) + \mu \cdot (\|\nabla R_\theta(\xi)\| - 1)^2 \tag{3.3.11}$$

is performed. The last term ensures that R_θ has a Lipschitz constant of approximately one. Thus, the magnitude of the gradient of R_θ is technically fixed and the training focuses more on optimizing the direction of the gradient.

To obtain provable regularization properties, [76] introduced the adversarial convex regularizer (ACR), which uses an input-convex neural network for R_θ. This way, it is possible to learn a strongly convex penalty functional that guarantees existence, uniqueness, stability, and convergence of solutions (see Remark 3.3.4). However, due to the architecture constraints, the ACR shows a slightly worse reconstruction performance than its nonconvex variant in numerical experiments.

It is also possible to combine an adversarial regularizer R_θ with end-to-end training [75]. An unrolled architecture $\varphi_{\tilde{\theta}}$ is trained to map y^δ to the minimizer of the variational formulation (3.3.10), while R_θ is adversarially trained in the same time to distinguish between real ground truth data $x^\dagger$ and network outputs $\varphi_{\tilde{\theta}}(y^\delta)$. This way, the high interpretability of variational regularization can be combined with the fast evaluation of end-to-end trained architectures.

Another idea is called NETT (network Tikhonov) [66], where the goal is to obtain a coercive penalty term, which is sufficient for some weak regularization properties. For this purpose, $R_\theta = \Psi \circ E_\theta$ is defined as the concatenation of a predefined coercive functional $\Psi \colon Z \to \mathbb{R}$ and a neural network $E_\theta \colon X \to Z$ with some latent space Z. As the adversarial regularizer, R_θ is also trained using clean ground truth samples $x_i^\dagger$ and reconstructions of

a pseudo-inverse $A^\dagger y_j^\delta$. One possible sufficient condition to ensure coercivity of R_θ is to use a network E_θ which is stably invertible, e.g., an iResNet [8].

Alternatively, one can also define $R_\theta(x) = \Psi(E_\theta(x)) + \frac{\gamma}{2}\|x - D_\theta(E_\theta(x))\|^2$, which is called augmented NETT [79]. Here, E_θ and D_θ act as encoder and decoder network. With a coercive functional Ψ and a Lipschitz continuous D_θ (with arbitrary Lipschitz constant), R_θ is guaranteed to be coercive.

A different possible strategy for training R_θ is called bilevel optimization, which is, e.g., applied in [43]. This approach requires paired data $(x_i^\dagger, y_i^\delta)$ and aims to solve

$$\begin{aligned} &\min_\theta \frac{1}{N}\sum_{i=1}^N \|x_{\theta,i}^\delta - x_i^\dagger\|^2 \\ &\text{s.t. } x_{\theta,i}^\delta = \operatorname*{arg\,min}_{x\in X} \frac{1}{2}\|Ax - y_i^\delta\|^2 + R_\theta(x). \end{aligned} \tag{3.3.12}$$

However, one major difficulty is that the dependence of $x_{\theta,i}^\delta$ on the network parameters θ is only implicitly defined via a minimization problem. A similar bilevel optimization problem is faced in the analytic deep prior approach [7].

As plug-and-play methods (Section 3.3.2), learned penalty terms also have the potential to provide strong theoretical guarantees, high interpretability, and flexibility w.r.t. changes in the forward operator A or noise level δ. However, their application can be quite slow since (3.3.10) has to be solved iteratively, and theoretical guarantees are only valid if appropriate restrictions are put on R_θ.

3.3.4 Learning Without Training Data

So far, we discussed deep learning algorithms which learn how to regularize an inverse problem from a given training data set. Such approaches are successful if the underlying neural network is able to extract useful information from the training data. However, [64] observed that, even without training, convolutional network architectures encode prior knowledge about natural-looking images pretty well. This section is about reconstruction methods which solve inverse problems using neural networks for regularization but no kind of training data (in particular no ground truth samples). Instead, the regularization abilities of such approaches are purely due to inherent properties the network architectures.

The so-called deep image prior (DIP) [64] approach can be formulated as

$$\min_\theta \frac{1}{2}\|A\varphi_\theta(z) + y^\delta\|^2, \tag{3.3.13}$$

where $\varphi_\theta \colon Z \to X$ is a CNN and z is a random input from a latent space Z. We note that (3.3.13) looks quite similar to variational regularization (Section 2.4) but with no explicit penalty term. Instead, the regularization of DIP is based on the observation that CNNs can approximate clean natural images faster than noisy images. Thus, (3.3.13) is usually

solved iteratively with early stopping (Section 2.6) to avoid overfitting. DIP is trained for each data sample y^δ separately. After optimization, the solution is $x = \varphi_\theta(z)$. However, it is therefore quite slow in application.

An easy adaption of DIP is adding an additional penalty term $R\colon X \to \mathbb{R} \cup \{\infty\}$, i.e.

$$\min_\theta \frac{1}{2}\|A\varphi_\theta(z) + y^\delta\|^2 + \alpha R(\varphi_\theta(z)). \tag{3.3.14}$$

Since DIP is mainly applied to imaging problems, TV regularization is particularly suitable [67]. In CT reconstruction, this approach showed a performance between data-driven deep learning methods and classical reconstruction methods (TV and FBP) [65]. Besides, [72] proposed to combine DIP with the RED penalty term (Section 3.3.2). As [15] showed, it is also possible to obtain analogous theoretical regularization properties as with variational regularization, dependent on R. However, the parameter α is typically chosen quite small in this setting since the architecture φ_θ itself contributes significantly to the regularization. Thus, the level of stability which can be guaranteed might be rather low (Theorem 2.4.7).

Another way for obtaining regularization properties is the analytic deep prior [41] approach. There, φ_θ is considered to have a LISTA architecture (Section 3.3.1) in order to exploit the parallels between unrolled architectures and iterative methods for solving variational minimization problems. I developed a complete regularization theory for this approach in [7]. More detailed information is provided in Section 4.1.

To increase the speed of the DIP evaluation, [18] proposed to pretrain the network φ_θ before solving (3.3.13), called educated DIP. More precisely, φ_θ is trained end-to-end (Section 3.3.1) on a synthetic dataset $(x_i^\dagger, y_i^\delta)$ to avoid the need for real data. Thus, instead of a random input z, the data y^δ is now fed into the network. This leads to a significant increase of speed and a slight performance boost in comparison to standard DIP. Alternatively, the approach of performing DIP after an end-to-end training can be interpreted as a guarantee for data consistency of φ_θ. We made use of this in the Helsinki Deblur Challenge [10].

A further difficulty in the application of DIP is to apply early stopping at the right moment. To address this, [94] analyzed the so-called spectral bias of convolutional layers, i.e., the property that low-frequency parts of an image are fitted faster than high-frequency parts. This spectral bias can be controlled by putting a Lipschitz constraint on the convolutions, similar to the strategy in iResNets [8]. By doing so, the overfitting can be delayed or even avoided, such that the right moment for stopping becomes less critical.

The ability of a CNN φ_θ to approximate certain images $x \in X$ better than others can also be encoded as a penalty functional R on X [51]. In other words, $R(x)$ is defined in a way that it has small values for images x that can be generated easily by φ_θ (i.e., with bounded weights and small inputs of all network layers) and larger values for images that are harder to generate. By proving weak lower semicontinuity and coercivity of R, some weak regularization properties are obtained.

3.3.5 Generative Models

From the Bayesian point of view (see Section 2.5) there exists a whole (conditional) distribution $p_{\mathcal{X}|\mathcal{Y}}$ of possible solutions x for the inverse problem $Ax = y$. So instead of just computing one possible solution, one could try to recover more information about the posterior distribution. Access to $p_{\mathcal{X}|\mathcal{Y}}$ (or an approximation of it) would be useful for several purposes, e.g., sampling from the posterior, computing Bayesian estimators like the posterior mean value, and computing uncertainties of the reconstruction via the variance of the posterior distribution. As in Section 2.5, we consider X and Y to be finite dimensional for simplicity.

For learning $p_{\mathcal{X}|\mathcal{Y}}$, so-called *generative models* can be used. Such approaches are able to produce new samples that are similar to the data on which they were trained (e.g., natural-looking images). In other words, they learn to sample from the probability distribution of the training data. Some models are also able to estimate the probability of a given sample w.r.t. this probability distribution.

Remark 3.3.5. There are two main ways to solve inverse problems using generative models. First, there is the "unconditional" approach, where the model is trained to learn the prior distribution. This can be done unsupervised since only samples x from $p_{\mathcal{X}}$ are needed. After learning the prior, the generative model is combined with a data discrepancy term (e.g., $\|Ax - y^\delta\|^2$) to sample from the posterior distribution. How this can be done exactly depends on the particular model one uses. Second, there is the "conditional" approach, where the generative model is directly trained to learn the posterior distribution. This is typically achieved via supervised training. The generator gets y^δ as input and then produces samples x conditioned on y^δ.

The field of generative modeling is very broad and there are many different deep learning approaches. This section can only give a small insight, we therefore focus on a few popular methods. In the following, the concepts of normalizing flows, generative adversarial networks, and score-based diffusion are briefly presented.

A normalizing flow [103, 102] is an invertible mapping $\varphi\colon X \to X$ between the target distribution (with unknown pdf $p\colon X \to \mathbb{R}$) and a simple baseline distribution (with known pdf $q\colon X \to \mathbb{R}$, usually Gaussian). If φ maps an element $x \sim p$ to $\varphi(x) \sim q$, the pdf of the target distribution can be computed using the change of variables formula

$$p(x) = q(\varphi(x)) \cdot |\det(D\varphi(x))|. \tag{3.3.15}$$

This result can be used to learn the unknown distribution of a given training dataset $\{x_i\}$. By making φ learnable, i.e., using a neural network φ_θ, the pdf $p = p_\theta$ becomes learnable. Then, φ_θ can be trained to maximize the log-likelihood

$$\max_\theta \sum_i \log(q(\varphi_\theta(x_i)) + \log(|\det(D\varphi_\theta(x_i))|) \tag{3.3.16}$$

of the data x_i w.r.t. p_θ. After training, the inverse φ_θ^{-1} can be applied to samples $z \sim q$ from the baseline distribution to generate new samples $\varphi_\theta^{-1}(z) \sim p_\theta$. Thus, invertible neural networks are needed for this purpose. A very common type of invertible architectures are so-called coupling layers, introduced in [40]. Alternatively, iResNets [8] can also be employed.

Normalizing flows can be used in an unconditional or a conditional approach (see Remark 3.3.5) for solving inverse problems. In [14], a normalizing flow φ_θ is trained on a dataset of ground truth samples $\{x_i^\dagger\}$ to learn the prior distribution. After training, one solves the variational problem

$$\min_z \|A\varphi_\theta^{-1}(z) - y^\delta\|^2 + \alpha\|z\|^2 \tag{3.3.17}$$

for given y^δ, and $\varphi_\theta^{-1}(z)$ parametrizes the solution of the inverse problem. However, for high-dimensional imaging problems, mapping the full prior distribution to a simple baseline distribution q can be difficult. To overcome this, [5] proposed to train the normalizing flow only on small image patches. Another advantage of this is that only a few training images $x_i^\dagger$ are needed because every image contains a lot of different patches.

If paired data $\{y_i^\delta, x_i^\dagger\}$ are given, one can directly learn the posterior distribution with a conditional normalizing flow [106]. For this purpose, φ_θ gets y^δ as an additional input, i.e., $\varphi_\theta(x, y^\delta) = z$ and $x = \varphi_\theta(\cdot, y^\delta)^{-1}(z)$. It is also possible to transform the conditional input y^δ first. This is proposed by [38] for CT reconstruction, where the normalizing flow is conditioned on the FBP reconstruction of the data.

The idea of generative adversarial nets (GANs) [48] is to learn a generator $G\colon Z \to X$ which receives random inputs $z \sim q$ (q being a simple baseline distribution), and whose outputs $G(z)$ can hardly be distinguished from the target distribution p. For this purpose, a second model $D\colon X \to [0,1]$ is introduced whose task is to estimate the probability that a given sample x is not generated by G but a true sample from p. These two models are trained simultaneously but compete with each other. This is expressed in the min-max-problem

$$\min_G \max_D \mathbb{E}_{x\sim p}\big(\log(D(x))\big) + \mathbb{E}_{z\sim q}\big(\log(1 - D(G(z))\big), \tag{3.3.18}$$

where D "tries to distinguish" between x and $G(z)$ while G "tries to fool" D. In practice, the samples x are drawn from a training dataset $\{x_i\}$, and z is randomly sampled from q during training. An overview of theory, possible applications, and architectures for GANs can be found in [33].

To solve inverse problems, an unconditional GAN can be trained to learn the prior distribution, and the generator G is then used to parametrize the solution as $G(z)$ [22, 92]. This way, the solution is restricted to the range of G and it is possible to further regularize with a penalty on z. GANs can also be made conditional by making both G and D additionally dependent on a further variable y [73]. If paired training data $\{x_i^\dagger, y_i^\delta\}$ are given, one can thus learn the posterior distribution with a conditional GAN [35].

In recent years, score-based diffusion models (SDMs) have received a lot of attention

for achieving state-of-the-art performance in image synthesis [39]. Their idea is to consider probability distributions p_t for $t \in [0, T]$ which arise from the target distribution $p = p_0$ being gradually blurred with a diffusion process. Since diffusion has the effect of equalizing differences, the possibly complicated initial distribution p_0 becomes smoother with growing t until finally, p_T is assumed to be approximately a Gaussian distribution. Then, by inverting the diffusion process, one obtains a generator for the target distribution p [96]. Mathematically, the diffusion process can be modeled with a stochastic differential equation (SDE) and there also exists an SDE which models the reverse diffusion [98, 6].

However, one important part of the reverse SDE is the so-called score function

$$\nabla_x \log(p_t(x)). \tag{3.3.19}$$

This means, for inverting the diffusion process one needs specific knowledge about the distributions p_t. Since the score function is typically unknown, one has to train a neural network $s_\theta(x, t)$ to approximate $\nabla_x \log(p_t(x))$ via the so-called denoising score matching technique [108]. The trained network and a solver for the reverse SDE then form a generative model for sampling from p.

To solve inverse problems with unconditionally trained SDMs, one has to learn the score function of the prior distribution first. After this, there are several approaches to integrate the conditional dependence on y^δ into the reverse diffusion process to be able to sample from the posterior distribution [97, 45, 17, 95]. Alternatively, one can directly use the learned score function s_θ in a gradient descent iteration corresponding to a variational formulation, even with some theoretical properties about the convergence of the iterates [46]. If paired training data $\{x_i^\dagger, y_i^\delta\}$ is given, s_θ can be conditioned on y^δ in order to directly learn the conditional score function of the posterior distribution [16].

Remark 3.3.6. In the Bayesian setting, an inverse problem is called well-posed if the properties existence, uniqueness, and stability (Section 2.2) hold for the whole posterior distribution $p_{\mathcal{X}|\mathcal{Y}}$ [77]. In contrast to Definition 2.1.1 (Hadamard), this means that Bayesian well-posedness does not only depend on the forward operator A but also on the prior distribution p_X. Thus, the desired theoretical guarantees for Bayesian reconstruction algorithms are different and of a rather statistical nature. For example, in [62] a learned iterative algorithm creates a Markov chain which provably converges to some stationary distribution which is close to the true posterior distribution.

Chapter 4

Regularization Theory for Special Architectures

The aim of this chapter is to introduce my own work regarding deep learning methods with regularization guarantees and to relate it to the current state of research. I particularly studied the regularization properties of two specific approaches, i.e., analytic deep prior and iResNets. The intention behind this research is to increase the trustability and interpretability of neural networks and to enhance their understanding from a theoretical point of view. Besides, I also focused on the practical application of deep learning approaches in order to gain more insights into the reasons for the success of neural networks in solving inverse problems. This has been a crucial part of the two Helsinki deep learning challenges.

However, attempting to develop deep learning methods with state-of-the-art reconstruction performance and provable regularization properties at the same time typically results in a conflict of interest. On the one hand, large and expressive architectures are needed, preferably unrestricted in their abilities to learn. On the other hand, small architectures with additional assumptions, conditions, restrictions, or a special structure provide more and easier possibilities for theoretical analyses.

The research of my PhD is about finding an appropriate compromise with the best of both worlds. As the overview in Section 3.3 shows, there already exist a lot of different strategies attempting to do so. But most approaches that are not severely restricting the expressivity of the architecture do not achieve strong theoretical guarantees like stability (in the sense of Lipschitz continuity).

Analytic deep prior provides a possible explanation for the inherent regularization abilities of neural networks. With iResNets, we introduced a powerful architecture with provable regularization properties. In the following, we describe both approaches and the findings of the Helsinki challenges in more detail and explain their relations to each other (in a narratively meaningful order). We use the notation of Chapter 2 for the inverse problem setting.

4.1 Analytic Deep Prior

Among all deep learning methods for solving inverse problems, deep image prior (DIP, Section 3.3.4) takes on a special role since it relies on a neural network, but it does not make use of any training data. The authors of [41] described this phenomenon with the apposite term *regularization by architecture*. Naturally, it raises the question of how a neural network actually regularizes if no prior information (Section 2.2) has been learned from data. Most explanations from the literature of this observation are based on the approximation capabilities of CNNs, as elaborated in Section 3.3.4. Alternatively, it is possible to consider architectures with a special structure which allows for a deeper theoretical analysis. We thus consider DIP with a LISTA (Section 3.3.1) architecture. The reason for this choice is that LISTA is based on an iterative algorithm (Section 2.6) for solving variational minimization problems, which provides a connection to the classical regularization theory.

For analytic deep prior (ADP), the crucial idea is to interpret the output of the LISTA network as the solution of a variational regularization scheme (Section 2.4). The details of establishing and exploiting this connection are described in [41], where ADP was introduced first, and also in [7]. Essentially, the neural network of DIP is replaced by a variational approach, resulting in the formulation

$$\begin{aligned} &\min_{B \in L(X,Y)} \frac{1}{2}\|Ax(B) - y^\delta\|^2 \\ \text{s.t.} \quad &x(B) = \operatorname*{arg\,min}_{x \in X} \frac{1}{2}\|Bx - y^\delta\|^2 + \alpha R(x) \end{aligned} \tag{4.1.1}$$

for some penalty term $R\colon X \to \mathbb{R} \cup \{\infty\}$.

Instead of an abstract neural network parameter θ, we have now a linear operator $B \in L(X,Y)$ which parametrizes the solution $x(B)$ of ADP. This way, we deviate quite a bit from the original DIP formulation (3.3.13) but we managed to transform the theoretical idea of regularization by architecture into a concrete variational regularization scheme. ADP can thus be seen as a demonstration of how classical regularization theory can be used to explain specific phenomena of deep learning.

Using an additional constraint on the SVD (Theorem 2.3.1) of B, some first theoretical results for ADP were achieved in [41]. In [7], we managed to abolish this constraint and derived an equivalence of the bilevel optimization problem (4.1.1) to

$$\begin{aligned} &\min_{x \in X} \frac{1}{2}\|Ax - y^\delta\|^2 \\ \text{s.t.} \quad &\exists v \in \partial R(x)\colon \quad \alpha\langle v, x\rangle \leqslant \frac{\|y^\delta\|^2}{4}. \end{aligned} \tag{4.1.2}$$

This way, we obtained a simpler convex optimization problem, where the solution x is no longer dependent on the operator B. Based on this, a complete regularization theory with existence, stability, and convergence of solutions can be developed for ADP.

Besides, [7] also introduces the variant ADP-β, addressing the fact that DIP mostly uses an early stopping strategy (Section 2.6). For this purpose, (4.1.1) is modified with an additional penalty term on the operator B. By proving the existence, stability, and convergence of solutions also for ADP-β, we provide a regularization theory for this case as well.

4.2 The Deblurring and Tomography Challenges

In 2021 and 2022 the Finnish Inverse Problems Society[1] organized the *Helsinki Deblur Challenge* (HDC2021) and the *Helsinki Tomography Challenge* (HTC2022). Both challenges were about solving ill-posed inverse problems with real physical measurements. Scientists and research groups from all over the world could participate and develop reconstruction algorithms for the given tasks. For this purpose, the organizers also provided some samples of ground truth and measurements as training data. In the end, all submitted methods were evaluated on new data that was unknown to the participants. Regarding the ill-posedness, the tasks were posed with several levels of difficulty in order to create a ranking of the participating algorithms and to determine a winner.

The subject of the HDC2021 was the inversion of a blurring caused by a camera which is out of focus. Each of the pictures taken displayed three lines of letters, which were increasingly difficult to recognize, dependent on the extent of the blurring. This was divided into 20 levels of difficulty, examples can be seen in Figure 4.1. For rating the reconstruction quality, a character recognition software had to identify as many as possible letters on the deblurred images correctly.

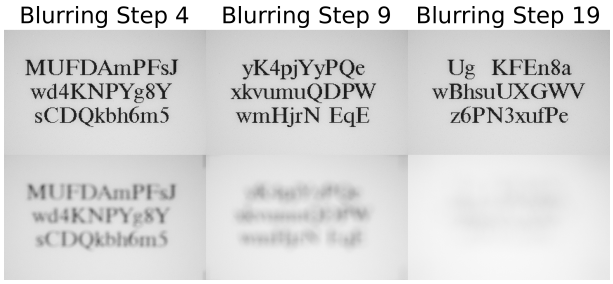


Figure 4.1: Examples of data pairs from the HDC2021. All pictures are taken with a real camera, which was correctly focused for the ground truth images (top row) and moved gradually out of focus for the blurred images (bottom row).

The HTC2022 was about CT reconstruction of circular acrylic disks from limited angle measurements (Figure 4.2). All disks were of the same size but had a different number of holes with variable shapes inside. This inner structure had to be reconstructed. In the first of seven difficulty levels, only measurements from a 90° rotation inside the scanner were provided. This view was reduced to 30° for the last level, making the problem increasingly ill-posed.

[1] https://www.fips.fi/

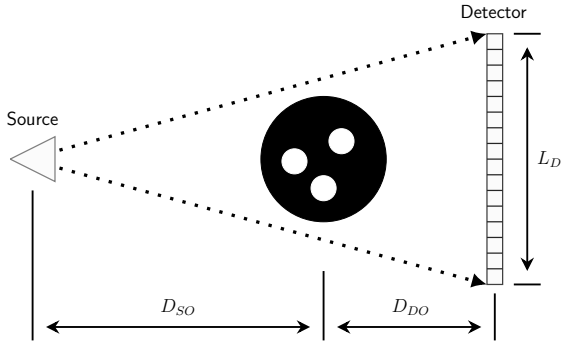


Figure 4.2: Measurement setting of the HTC2022. The goal is to reconstruct the inner structure of circular acrylic disks from limited angle measurements of a fan-beam CT scanner. Adapted from https://fips.fi/HTC2022.php.

For both of the challenges, we submitted several algorithms, respectively. Our main approaches were end-to-end trained unrolled neural networks (Section 3.3.1) which incorporated the forward operator of the problem. In addition, we tested models with slightly more theoretical foundation. We deployed an educated DIP (Section 3.3.4) as postprocessing for an end-to-end trained CNN in order to guarantee data fidelity, in the HDC2021. For the HTC2022, we used a hybrid approach consisting of an initial classical reconstruction followed by a deep learning based inpainting of missing details which could not be reconstructed from the limited angle measurements. More details are described in [10] and [9]. We achieved the fourth place in the HDC2021 and the second place in the HTC2022.

While theoretical guarantees for the submitted algorithms were not required at all, some ideas from regularization theory still proved to be highly beneficial. Prior knowledge (Section 2.2) about the challenge data emerged as the most important issue. For high performance in the more difficult levels, the models needed to learn the features (readable letters and disc-shaped objects with holes) of the ground truth data precisely. Thus, a pure DIP approach turned out to be suboptimal, because its intrinsic regularization is more suitable for generic image textures than for very specific structures such as letters and discs. Since only a few samples of ground truth were provided for training, using synthetic data was a very advantageous strategy. Besides, a certain robustness of the algorithms was required because, as announced in advance, the evaluation data differed slightly from the training data (e.g., numbers in addition to the letters).

The challenges were open for solution algorithms of all types. Unsurprisingly, deep learning methods showed a clearly superior performance in comparison to classical approaches. In both challenges, the first places went to end-to-end trained architectures. For methods with theoretical guarantees, however, it was hard to compete.

4.3 Invertible Residual Networks

Since end-to-end learning has shown a remarkable performance in the two deep learning challenges (Section 4.2), it would be highly desirable to find such an approach with theoretical regularization guarantees. Our natural idea for addressing inverse problems is to employ invertible neural networks φ_θ. So far, these kinds of architectures have been applied, e.g., as normalizing flows for generative modeling (Section 3.3.5). We want to use them for approximating the forward operator A with φ_θ and solving the inverse problem with φ_θ^{-1}. As our architecture, we choose invertible residual networks (iResNets) [20] due to their specific properties (as explained in the following).

The idea of a residual network

$$\varphi_\theta = \mathrm{Id} - f_\theta \tag{4.3.1}$$

is to learn deviations from the identity mapping. On its own, the identity is of course invertible. If the deviation f_θ is restricted to be sufficiently small, the invertibility is preserved. Precisely, we need f_θ to be contractive, i.e., $\mathrm{Lip}(f_\theta) \leqslant L$ for a Lipschitz constant $L < 1$. Under this condition, we can compute the inverse $x = \varphi_\theta^{-1}(z)$ with the fixed point iteration

$$x^{k+1} = z + f_\theta(x^k), \tag{4.3.2}$$

which converges due to the fixed-point theorem of Banach [110, Section IV.7]. Further, we can estimate the Lipschitz constant of the inverse network as $\mathrm{Lip}(\varphi_\theta^{-1}) \leqslant \frac{1}{1-L}$. More technical details are explained in [8, 11].

Our first paper [8] is about how to guarantee regularization properties (Section 2.2) for iResNets. Existence and uniqueness of solutions are trivial for end-to-end trained networks, which directly output the reconstruction. Stability is implied by the Lipschitz continuity of φ_θ^{-1}. As with DIP and analytic deep prior, we thus have a regularization by architecture (Section 4.1). The convergence property, however, depends on the success of the training. In analogy to minimum-norm solutions in Section 2.1 or R-minimizing solutions in Section 2.4, for iResNets, the solutions x are required to fulfill a so-called local approximation property, i.e., $\varphi_\theta(x)$ approximates Ax for $L \to 1$. Accordingly, the convergence of solutions is related to the data consistency of the trained network. The Lipschitz constant L can be interpreted as a regularization parameter. Similar to the parameter α of variational regularization schemes in Theorem 2.4.7, L also controls the amount of stability of φ_θ. For specific small architectures, we showed that iResNets are even equivalent to certain variational reconstruction methods.

In our second paper [11], we analyzed iResNets from the Bayesian perspective (Section 2.5). Note that we consider X and Y to be finite dimensional for this purpose in order to simplify the use of probability density functions. However, all results apply independently of the choice of discretization, and the regularization properties from the previous paper are not affected. We particularly compare two different training strategies

for φ_θ. The first approach aims to achieve $\varphi_\theta \approx A$ via the training objective

$$\min_\theta \sum_i \|\varphi_\theta(x_i^\dagger) - y_i^\delta\|^2 \tag{4.3.3}$$

for a dataset of pairs $(x_i^\dagger, y_i^\delta)$, and is thus called approximation training. Due to the architecture constraint, φ_θ^{-1} is a regularization scheme for the inverse problem, but we showed that φ_θ learns only few information about the prior distribution of the data. More precisely, if the problem is decomposed into one-dimensional subproblems using the SVD of A (as in Section 2.3), the expected outcome φ_θ of the training always shows the same affine linear behavior, only depending on the mean of the prior distribution. This is suboptimal for a data-driven regularization, as there is more prior knowledge which could be learned. The second training strategy aims at $\varphi_\theta^{-1}(y^\delta) \approx x^\dagger$ via the objective

$$\min_\theta \sum_i \|\varphi_\theta^{-1}(y_i^\delta) - x_i^\dagger\|^2. \tag{4.3.4}$$

This approach is called reconstruction training and corresponds to common end-to-end learning (Section 3.3.1). We showed that this way of training leads to a stable approximation of the conditional expectation $\mathbb{E}(\mathcal{X}|\mathcal{Y})$ of the posterior distribution.

Finally, we demonstrate the practical performance of iResNets on real imaging inverse problems in our third paper [12]. For this purpose, we concatenate several blocks of the form (4.3.1) to a larger iResNet architecture. Furthermore, we slightly generalize the theoretical results of the previous articles in order to address nonlinear inverse problems too. For the numerical experiments, we consider a deblurring problem and a nonlinear diffusion. In comparison to other end-to-end trained deep learning methods, the iResNet emerges to be competitive but more elaborate in training. On the one hand, this demonstrates that deep learning with regularization guarantees and high reconstruction performance is possible. On the other hand, the Lipschitz constraint still causes a reduction of the expressivity of iResNets and a significantly higher computational complexity. However, in comparison to plug-and-play methods (Section 3.3.2), where contractivity is required for the full architecture in order to obtain guarantees, it is advantageous that only the subnetworks f_θ have to be constrained. Regarding interpretability, we compare the forward pass of the trained network φ_θ with the true forward operator numerically. By analyzing the disparities between those two mappings, it is possible to investigate how the network has learned to regularize the inverse problem.

In summary, iResNets provide a deep learning based reconstruction scheme for inverse problems with strong provable regularization properties. Additionally, the approach is highly interpretable due to the invertibility of the architecture and shows excellent performance in practical applications. Among the large variety of reconstruction methods for inverse problems, this combination of features is still quite rare.

Chapter 5

Conclusion and Outlook

The core of this thesis is the proof of regularization properties of certain deep learning based reconstruction methods and their successful application to solve inverse problems. With analytic deep prior we obtain a link between neural networks and classical variational regularization, providing an explanation of the phenomenon of regularization by architecture. In addition, as the results of the Helsinki deep learning challenges show, learning extensive prior information about the ground truth from suitable training data is also of great importance to achieve a problem-specific regularization. Both ideas are combined in iResNets, which can be trained via end-to-end learning and have a hyperparameter for controlling the Lipschitz constant (i.e., the stability) of the reconstruction scheme.

The results of the thesis primarily show that deep learning for inverse problems so far requires a compromise between practical performance and theoretical regularization guarantees. In order to increase the interpretability and the possibilities for theoretical analyses of an architecture, it is often necessary to restrict its expressivity. Yet, as iResNets demonstrate, reconstruction quality at almost state-of-the-art level is also possible with rigorous regularization properties. The computational complexity, however, is higher in comparison to other approaches without architectural constraints. It is therefore sometimes more practical to choose methods which are just fast and easy to implement. With regard to the nature of architectural constraints, it is noticeable that Lipschitz continuity is not only crucial for iResNets but also for other types of approaches, e.g., plug-and-play methods for guaranteeing convergence and stability. Thus, further research about effectively restricting the Lipschitz constant of neural networks would be beneficial.

Besides, it would be desirable to develop further reconstruction methods with provable regularization properties. Most approaches have specific advantages and disadvantages, which affect the type of application they are suitable for. In order for methods with regularization guarantees to become established in practice, they must catch up in terms of reconstruction performance and computational complexity.

List of Symbols

Frequently used variables:

X, Y	-	The spaces of "causes"/"ground truths" and "effects"/"data"
A	-	Forward operator $A\colon X \to Y$
$x^\dagger$, $y^\dagger$, y^δ	-	Ground truth, clean data, and noisy data
δ	-	Noise level
$A^\dagger$	-	Moore-Penrose inverse of A
T, T_α	-	A reconstruction algorithm (mapping from Y to X)
(u_k, v_k, σ_k)	-	Singular value decomposition of A
α	-	Regularization parameter
L	-	Lipschitz constant (for iResNets)
r_α	-	Filter functions for spectral regularization
R	-	Penalty term $R\colon X \to \mathbb{R} \cup \{\infty\}$ for variational regularization
$\mathcal{X}$, $\mathcal{E}$, $\mathcal{Y}$	-	Random variables representing ground truth, noise, and (noisy) data
$p_{\mathcal{X}}$	-	Probability density function of $\mathcal{X}$
φ_θ	-	Neural network with (learnable) parameter θ
$\mathcal{L}$	-	Loss function for training of neural networks

Mathematical symbols and notation:

$\lVert \cdot \rVert$	-	Standard Norm of the space of which the argument is from
$\lVert \cdot \rVert_1$	-	Norm of L^1 or ℓ^1
$x_k \to x$	-	The sequence (x_k) converges to x
$x_k \rightharpoonup x$	-	The sequence (x_k) converges weakly to x
A^*	-	Adjoint of the operator A
Id	-	Identity operator, $\mathrm{Id}x = x$
$\langle \cdot, \cdot \rangle$	-	Inner product of a Hilbert space
$\mathcal{R}(A)$	-	Range of the operator A
$\mathcal{N}(A)$	-	Nullspace of the operator A
$U^\perp$	-	Orthogonal complement of a subspace U in a Hilbert space
$\oplus$	-	Direct sum of two orthogonal subspaces
$\overline{U}$	-	Closure of a subset U in some space
$\mathrm{Lip}(\cdot)$	-	Lipschitz constant of a function or an operator
$k * x$	-	Convolution of the functions k and x
∂_i	-	Partial derivative w.r.t. the i-th component
∂R	-	Subdifferential of a convex functional R
D, D_u	-	Derivative of a function (w.r.t. a certain argument u)
∇	-	Gradient of a function or of a functional
prox_R	-	Proximal mapping of a functional R
det	-	Determinant of a matrix
$L(X, Y)$	-	Space of bounded linear operators from X to Y.
L^p	-	Lebesgue space of p-integrable functions
H^1	-	Sobolev space where the weak derivatives of first order are in L^2
C, C^1	-	Space of continuous (cont. differentiable) functions
C_0^1	-	C^1-functions with compact support
$\mathbb{E}(\cdot)$, $\mathbb{E}(\cdot\lvert\cdot)$	-	Expectation value and conditional expectation
$\mathcal{X} \sim p$	-	The random variable X is distributed according to p.

Bibliography

[1] J. Adler and O. Öktem. Solving ill-posed inverse problems using iterative deep neural networks. *Inverse Problems*, 33(12), 2017.

[2] J. Adler and O. Öktem. Deep Bayesian inversion. Preprint, arXiv:1811.05910, 11 2018.

[3] J. Adler and O. Öktem. Learned primal-dual reconstruction. *IEEE Transactions on Medical Imaging*, 37(6):1322–1332, 2018.

[4] H. Alt. *Lineare Funktionalanalysis: Eine anwendungsorientierte Einführung*. Masterclass. Springer Berlin Heidelberg, 6th edition, 2012.

[5] F. Altekrüger, A. Denker, P. Hagemann, J. Hertrich, P. Maass, and G. Steidl. PatchNR: learning from very few images by patch normalizing flow regularization. *Inverse Problems*, 39(6):064006, may 2023.

[6] B. D. Anderson. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982.

[7] C. Arndt. Regularization theory of the analytic deep prior approach. *Inverse Problems*, 38(11):115005, 2022.

[8] C. Arndt, A. Denker, S. Dittmer, N. Heilenkötter, M. Iske, T. Kluth, P. Maass, and J. Nickel. Invertible residual networks in the context of regularization theory for linear inverse problems. *Inverse Problems*, 39(12):125018, nov 2023.

[9] C. Arndt, A. Denker, S. Dittmer, J. Leuschner, J. Nickel, and M. Schmidt. Model-based deep learning approaches to the Helsinki Tomography Challenge 2022. *Applied Mathematics for Modern Challenges*, 1(2):87–104, 2023.

[10] C. Arndt, A. Denker, J. Nickel, J. Leuschner, M. Schmidt, and G. Rigaud. In focus - hybrid deep learning approaches to the HDC2021 challenge. *Inverse Problems and Imaging*, 17(5):908–924, 2023.

[11] C. Arndt, S. Dittmer, N. Heilenkötter, M. Iske, T. Kluth, and J. Nickel. Bayesian view on the training of invertible residual networks for solving linear inverse problems. *Inverse Problems*, 40(4):045021, mar 2024.

[12] C. Arndt and J. Nickel. Invertible ResNets for inverse imaging problems: Competitive performance with provable regularization properties. Preprint, arXiv:2409.13482, 09 2024.

[13] S. Arridge, P. Maass, O. Öktem, and C.-B. Schönlieb. Solving inverse problems using data-driven models. *Acta Numerica*, 28:1–174, 2019.

[14] M. Asim, M. Daniels, O. Leong, A. Ahmed, and P. Hand. Invertible generative models for inverse problems: mitigating representation error and dataset bias. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 399–409. PMLR, 13–18 Jul 2020.

[15] D. O. Baguer, J. Leuschner, and M. Schmidt. Computed tomography reconstruction using deep image prior and learned reconstruction methods. *Inverse Problems*, 36(9), 2020.

[16] L. Baldassari, A. Siahkoohi, J. Garnier, K. Solna, and M. V. de Hoop. Conditional score-based diffusion models for Bayesian inference in infinite dimensions. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 24262–24290. Curran Associates, Inc., 2023.

[17] R. Barbano, A. Denker, H. Chung, T. H. Roh, S. Arridge, P. Maass, B. Jin, and J. C. Ye. Steerable conditional diffusion for out-of-distribution adaption in imaging inverse problems. Preprint, arXiv:2308.14409, 2023.

[18] R. Barbano, J. Leuschner, M. Schmidt, A. Denker, A. Hauptmann, P. Maass, and B. Jin. An educated warm start for deep image prior-based micro CT reconstruction. *IEEE Transactions on Computational Imaging*, 8:1210–1222, 2022.

[19] H. H. Bauschke, S. M. Moffat, and X. Wang. Firmly nonexpansive mappings and maximally monotone operators: Correspondence and duality. *Set-Valued and Variational Analysis*, 20:131–153, 2012.

[20] J. Behrmann, W. Grathwohl, R. T. Chen, D. Duvenaud, and J.-H. Jacobsen. Invertible residual networks. In *International Conference on Machine Learning*, pages 573–582. PMLR, 2019.

[21] M. Benning and M. Burger. Modern regularization methods for inverse problems. *Acta Numerica*, 27:1–111, 2018.

[22] A. Bora, A. Jalal, E. Price, and A. G. Dimakis. Compressed sensing using generative models. In D. Precup and Y. W. Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 537–546. PMLR, 06–11 Aug 2017.

[23] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Mach. Learn.*, 3(1):1–122, jan 2011.

[24] K. Bredies and D. Lorenz. *Mathematical Image Processing*. Birkhäuser Cham, 1st edition, 2014.

[25] L. Bungert, R. Raab, T. Roith, L. Schwinn, and D. Tenbrinck. CLIP: Cheap Lipschitz training of neural networks. In A. Elmoataz, J. Fadili, Y. Quéau, J. Rabin, and L. Simon, editors, *Scale Space and Variational Methods in Computer Vision*, pages 307–319, Cham, 2021. Springer International Publishing.

[26] M. Burger and S. Osher. Convergence rates of convex variational regularization. *Inverse Problems*, 20(5):1411–1421, 2004.

[27] E. Celledoni, M. J. Ehrhardt, C. Etmann, B. Owren, C.-B. Schönlieb, and F. Sherry. Equivariant neural networks for inverse problems. *Inverse Problems*, 37(8):085006, 2021.

[28] A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40:120–145, 2011.

[29] H. Chen, Y. Zhang, M. K. Kalra, F. Lin, Y. Chen, P. Liao, J. Zhou, and G. Wang. Low-dose CT with a residual encoder-decoder convolutional neural network. *IEEE Transactions on Medical Imaging*, 36(12):2524–2535, 2017.

[30] W. Chen, D. Wipf, and M. Rodrigues. Deep learning for linear inverse problems using the plug-and-play priors framework. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8098–8102, 2021.

[31] Y. Chen, W. Yu, and T. Pock. On learning optimized reaction diffusion processes for effective image restoration. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5261–5269, 2015.

[32] P. L. Combettes and J.-C. Pesquet. Deep neural network structures solving variational inequalities. *Set-Valued and Variational Analysis*, 28:491–518, 2020.

[33] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath. Generative adversarial networks: An overview. *IEEE Signal Processing Magazine*, 35(1):53–65, 2018.

[34] Z.-X. Cui, Q. Zhu, J. Cheng, B. Zhang, and D. Liang. Deep unfolding as iterative regularization for imaging inverse problems. *Inverse Problems*, 40(2):025011, jan 2024.

[35] M. Damara, G. Kornhardt, and P. Jung. Solving inverse problems with conditional-GAN prior via fast network-projected gradient descent. Preprint, arXiv:2109.01105, 2021.

[36] M. Dashti and A. Stuart. The Bayesian approach to inverse problems. In R. Ghanem, D. Higdon, and H. Owhadi, editors, *Handbook of Uncertainty Quantification*, pages 311–428. Springer, Cham, 06 2017.

[37] I. Daubechies, M. Defrise, and C. De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on Pure and Applied Mathematics*, 57(11):1413–1457, 2004.

[38] A. Denker, M. Schmidt, J. Leuschner, P. Maass, and J. Behrmann. Conditional normalizing flows for low-dose computed tomography image reconstruction. In *Second workshop on Invertible Neural Networks, Normalizing Flows, and Explicit Likelihood Models (ICML)*, 2020.

[39] P. Dhariwal and A. Nichol. Diffusion models beat GANs on image synthesis. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 8780–8794. Curran Associates, Inc., 2021.

[40] L. Dinh, D. Krueger, and Y. Bengio. NICE: non-linear independent components estimation. In Y. Bengio and Y. LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings*, 2015.

[41] 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*, 62:456–470, 2020.

[42] A. Ebner and M. Haltmeier. Plug-and-play image reconstruction is a convergent regularization method. *IEEE Transactions on Image Processing*, 33:1476–1486, 2024.

[43] M. Eliasof, E. Haber, and E. Treister. DRIP: deep regularizers for inverse problems. *Inverse Problems*, 40(1):015006, 2024.

[44] H. W. Engl, M. Hanke, and A. Neubauer. *Regularization of Inverse Problems*. Springer Dordrecht, 1st edition, 2000.

[45] B. T. Feng, J. Smith, M. Rubinstein, H. Chang, K. L. Bouman, and W. T. Freeman. Score-based diffusion models as principled priors for inverse imaging. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10486–10497, 2023.

[46] P. Fernsel, Z. Kereta, and A. Denker. Convergence properties of score-based models using graduated optimisation for linear inverse problems. Preprint, arXiv:2404.18699, 2024.

[47] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[48] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[49] N. M. Gottschling, V. Antun, A. C. Hansen, and B. Adcock. The troublesome kernel – on hallucinations, no free lunches and the accuracy-stability trade-off in inverse problems. Preprint, arXiv:2001.01258, 2024.

[50] K. Gregor and Y. LeCun. Learning fast approximations of sparse coding. In *Proceedings of the 27th International Conference on Machine Learning*, ICML'10, page 399–406, Madison, WI, USA, 2010. Omnipress.

[51] A. Habring and M. Holler. A generative variational model for inverse problems in imaging. *SIAM Journal on Mathematics of Data Science*, 4(1):306–335, 2022.

[52] M. Haltmeier and L. Nguyen. Regularization of inverse problems by neural networks. In K. Chen, C.-B. Schönlieb, X.-C. Tai, and L. Younes, editors, *Handbook of Mathematical Models and Algorithms in Computer Vision and Imaging*, pages 1065–1093. Springer International Publishing, Cham, 2023.

[53] M. Hanke, A. Neubauer, and O. Scherzer. A convergence analysis of the Landweber iteration for nonlinear ill-posed problems. *Numerische Mathematik*, 72:21–37, 1995.

[54] A. Hauptmann and J. Poimala. Model-corrected learned primal-dual models for fast limited-view photoacoustic tomography. Preprint, arXiv:2304.01963, 2023.

[55] C. F. Higham and D. J. Higham. Deep learning: An introduction for applied mathematicians. *SIAM Review*, 61(4):860–891, 2019.

[56] B. Hofmann. *Mathematik inverser Probleme*. Teubner, Stuttgart, Leipzig, 1st edition, 1999.

[57] B. Hofmann, B. Kaltenbacher, C. Pöschl, and O. Scherzer. A convergence rates result for Tikhonov regularization in Banach spaces with non-smooth operators. *Inverse Problems*, 23(3):987–1010, 2007.

[58] S. Hurault, A. Chambolle, A. Leclaire, and N. Papadakis. A relaxed proximal gradient descent algorithm for convergent plug-and-play with proximal denoiser. In *Scale Space and Variational Methods in Computer Vision: 9th International Conference, SSVM 2023, Santa Margherita Di Pula, Italy, May 21–25, 2023, Proceedings*, page 379–392, Berlin, Heidelberg, 2023. Springer-Verlag.

[59] S. Hurault, A. Leclaire, and N. Papadakis. Gradient step denoiser for convergent plug-and-play. In *International Conference on Learning Representations*, 2022.

[60] V. K. Ivanov, V. V. Vasin, and V. P. Tanana. *Theory of Linear Ill-Posed Problems and its Applications*. De Gruyter, Berlin, Boston, 2002.

[61] J. Kaipio and E. Somersalo. *Statistical and computational inverse problems*, volume 160 of *Applied Mathematical Sciences*. Springer Science & Business Media, New York, 2005.

[62] R. Laumont, V. D. Bortoli, A. Almansa, J. Delon, A. Durmus, and M. Pereyra. Bayesian imaging using plug & play priors: When Langevin meets Tweedie. *SIAM Journal on Imaging Sciences*, 15(2):701–737, 2022.

[63] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521:436–444, 2015.

[64] V. Lempitsky, A. Vedaldi, and D. Ulyanov. Deep Image Prior. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9446–9454, 2018.

[65] J. Leuschner, M. Schmidt, P. S. Ganguly, V. Andriiashen, S. B. Coban, A. Denker, D. Bauer, A. Hadjifaradji, K. J. Batenburg, P. Maass, and M. van Eijnatten. Quantitative comparison of deep learning-based image reconstruction methods for low-dose and sparse-angle CT applications. *Journal of Imaging*, 7(3), 2021.

[66] H. Li, J. Schwab, S. Antholzer, and M. Haltmeier. NETT: solving inverse problems with deep neural networks. *Inverse Problems*, 36(6):065005, 2020.

[67] J. Liu, Y. Sun, X. Xu, and U. S. Kamilov. Image restoration using total variation regularized deep image prior. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7715–7719, 2019.

[68] 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*, 3:218–229, 2021.

[69] A. Lucas, M. Iliadis, R. Molina, and A. K. Katsaggelos. Using deep neural networks for inverse problems in imaging: Beyond analytical methods. *IEEE Signal Processing Magazine*, 35(1):20–36, 2018.

[70] S. Lunz, O. Öktem, and C.-B. Schönlieb. Adversarial regularizers in inverse problems. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[71] P. Maass. Deep learning for trivial inverse problem. In *Compressed Sensing and Its Applications*. Springer International Publishing, 2019.

[72] G. Mataev, P. Milanfar, and M. Elad. DeepRED: Deep image prior powered by RED. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2019.

[73] M. Mirza and S. Osindero. Conditional generative adversarial nets. Preprint, arXiv:1411.1784, 2014.

[74] V. Monga, Y. Li, and Y. C. Eldar. Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing. *IEEE Signal Processing Magazine*, 38(2):18–44, 2021.

[75] S. Mukherjee, M. Carioni, O. Öktem, and C.-B. Schönlieb. End-to-end reconstruction meets data-driven regularization for inverse problems. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 21413–21425. Curran Associates, Inc., 2021.

[76] S. Mukherjee, S. Dittmer, Z. Shumaylov, S. Lunz, O. Öktem, and C.-B. Schönlieb. Learned convex regularizers for inverse problems. Preprint, arXiv:2008.02839, 2021.

[77] S. Mukherjee, A. Hauptmann, O. Öktem, M. Pereyra, and C.-B. Schönlieb. Learned reconstruction methods with convergence guarantees: A survey of concepts and applications. *IEEE Signal Processing Magazine*, 40(1):164–182, 2023.

[78] D. Nganyu Tanyu, J. Ning, T. Freudenberg, N. Heilenkötter, A. Rademacher, U. Iben, and P. Maass. Deep learning methods for partial differential equations and related parameter identification problems. *Inverse Problems*, 39(10):103001, 2023.

[79] D. Obmann, L. Nguyen, J. Schwab, and M. Haltmeier. Augmented NETT regularization of inverse problems. *Journal of Physics Communications*, 5(10):105002, 2021.

[80] C. D. Pain, G. F. Egan, and Z. Chen. Deep learning-based image reconstruction and post-processing methods in positron emission tomography for low-dose imaging and resolution enhancement. *European Journal of Nuclear Medicine and Molecular Imaging*, 49:3098–3118, 2022.

[81] N. Parikh and S. Boyd. Proximal algorithms. *Foundations and Trends in Optimization*, 1(3):127–239, jan 2014.

[82] J.-C. Pesquet, A. Repetti, M. Terris, and Y. Wiaux. Learning maximally monotone operators for image recovery. *SIAM Journal on Imaging Sciences*, 14(3):1206–1237, 2021.

[83] E. T. Reehorst and P. Schniter. Regularization by denoising: Clarifications and new interpretations. *IEEE Transactions on Computational Imaging*, 5(1):52–67, 2019.

[84] M. Richter. *Inverse Probleme: Grundlagen, Theorie und Anwendungsbeispiele*. Springer Spektrum Berlin, Heidelberg, 1st edition, 2015.

[85] A. Rieder. *Keine Probleme mit inversen Problemen: eine Einführung in ihre stabile Lösung*. Vieweg+Teubner Verlag Wiesbaden, 1st edition, 2003.

[86] R. T. Rockafellar. *Convex Analysis*. Princeton University Press, Princeton, 1970.

[87] Y. Romano, M. Elad, and P. Milanfar. The little engine that could: Regularization by Denoising (RED). *SIAM Journal on Imaging Sciences*, 10(4):1804–1844, 2017.

[88] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional networks for biomedical image segmentation. In N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015*, volume 9351, pages 234–241, 2015.

[89] L. I. Rudin, S. Osher, and E. Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60(1):259–268, 1992.

[90] T. Schuster, B. Kaltenbacher, B. Hofmann, and K. S. Kazimierski. *Regularization Methods in Banach Spaces*. De Gruyter, Berlin, Boston, 2012.

[91] J. Schwab, S. Antholzer, and M. Haltmeier. Deep null space learning for inverse problems: convergence analysis and rates. *Inverse Problems*, 35(2), 2019.

[92] V. Shah and C. Hegde. Solving linear inverse problems using GAN priors: An algorithm with provable guarantees. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4609–4613, 2018.

[93] F. Sherry, E. Celledoni, M. J. Ehrhardt, D. Murari, B. Owren, and C.-B. Schönlieb. Designing stable neural networks using convex analysis and ODEs. *Physica D: Nonlinear Phenomena*, 463:134159, 2024.

[94] Z. Shi, P. Mettes, S. Maji, and C. G. M. Snoek. On Measuring and Controlling the Spectral Bias of the Deep Image Prior. *International Journal of Computer Vision*, 2022.

[95] I. R. Singh, A. Denker, R. Barbano, Z. Kereta, B. Jin, K. Thielemans, P. Maass, and S. Arridge. Score-based generative models for PET image reconstruction. *Machine Learning for Biomedical Imaging*, 2:547–585, 2024.

[96] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In F. Bach and D. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2256–2265, Lille, France, 07–09 Jul 2015. PMLR.

[97] Y. Song, L. Shen, L. Xing, and S. Ermon. Solving inverse problems in medical imaging with score-based generative models. In *International Conference on Learning Representations*, 2022.

[98] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

[99] P. Sprechmann, A. M. Bronstein, and G. Sapiro. Learning efficient sparse and low rank models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1821–1833, 2015.

[100] A. M. Stuart. Inverse problems: A Bayesian perspective. *Acta Numerica*, 19:451–559, 2010.

[101] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. Preprint, arXiv:1312.6199, 2014.

[102] E. G. Tabak and C. V. Turner. A family of nonparametric density estimation algorithms. *Communications on Pure and Applied Mathematics*, 66(2):145–164, 2013.

[103] E. G. Tabak and E. Vanden-Eijnden. Density estimation by dual ascent of the log-likelihood. *Communications in Mathematical Sciences*, 8(1):217 – 233, 2010.

[104] H. Y. Tan, S. Mukherjee, J. Tang, and C.-B. Schönlieb. Provably convergent plug-and-play quasi-Newton methods. *SIAM Journal on Imaging Sciences*, 17(2):785–819, 2024.

[105] M. Terris, A. Repetti, J.-C. Pesquet, and Y. Wiaux. Building firmly nonexpansive convolutional neural networks. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8658–8662, 2020.

[106] B. L. Trippe and R. E. Turner. Conditional density estimation with bayesian normalising flows. In *NeurIPS Workshop on Bayesian Deep Learning*, 2017.

[107] S. V. Venkatakrishnan, C. A. Bouman, and B. Wohlberg. Plug-and-play priors for model based reconstruction. In *2013 IEEE Global Conference on Signal and Information Processing*, pages 945–948, 2013.

[108] P. Vincent. A connection between score matching and denoising autoencoders. *Neural Computation*, 23(7):1661–1674, 2011.

[109] S. Wang, S. Fidler, and R. Urtasun. Proximal deep structured models. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[110] D. Werner. *Funktionalanalysis*. Springer Spektrum Berlin, Heidelberg, 8th edition, 2018.

[111] Y. Yang, J. Sun, H. Li, and Z. Xu. Deep ADMM-Net for compressive sensing MRI. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[112] J. Zhao, Z. Chen, L. Zhang, and X. Jin. Few-view CT reconstruction method based on deep learning. In *2016 IEEE Nuclear Science Symposium, Medical Imaging Conference and Room-Temperature Semiconductor Detector Workshop (NSS/MIC/RTSD)*, pages 1–4, 2016.

[113] S. Zheng, Y. Song, T. Leung, and I. Goodfellow. Improving the robustness of deep neural networks via stability training. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4480–4488, 2016.

Part II
Publications

OPEN ACCESS
IOP Publishing
Inverse Problems
Inverse Problems **38** (2022) 115005 (21pp)
https://doi.org/10.1088/1361-6420/ac9011

Regularization theory of the analytic deep prior approach

Clemens Arndt*

Center for Industrial Mathematics, University of Bremen, Germany

E-mail: carndt@uni-bremen.de

Received 16 March 2022, revised 22 July 2022
Accepted for publication 7 September 2022
Published 26 September 2022

Abstract

The analytic deep prior (ADP) approach was recently introduced for the theoretical analysis of deep image prior (DIP) methods with special network architectures. In this paper, we prove that ADP is in fact equivalent to classical variational Ivanov methods for solving ill-posed inverse problems. Besides, we propose a new variant which incorporates the strategy of early stopping into the ADP model. For both variants, we show how classical regularization properties (existence, stability, convergence) can be obtained under common assumptions.

Keywords: analytic deep prior, deep image prior, Tikhonov regularization, Ivanov regularization

S Supplementary material for this article is available online

(Some figures may appear in colour only in the online journal)

1. Introduction

In particular the field of image processing (e.g. denoising, deblurring) is a constant source for challenging inverse problems. The restoration of a corrupted image is typically ill-posed, so regularization techniques are needed to obtain a natural looking result. In other words, the restoration method should incorporate some prior knowledge about the appearance of natural images. However, dependent on the application it can be very difficult to give a mathematically exact definition of what natural looking images are. This makes it hard to encode such prior knowledge in a penalty term for classical variational regularization approaches (e.g. TV regularization [8, 30]).

*Author to whom any correspondence should be addressed.

However, deep learning methods with convolutional neural networks have proven to be quite successful in generating and restoring images [16, 22, 35]. One reason for that is the use of appropriate training data, but [25] shows that just the architecture of an untrained network can already serve as an image prior. The so-called deep image prior (DIP) approach consists in optimizing the weights of a neural network φ_θ to minimize the loss function

$$\frac{1}{2}\|A\varphi_\theta(z) - y^\delta\|^2 \tag{1.1}$$

for some forward operator A and noisy data y^δ (the network's input z is randomly chosen and kept fixed). Although no training data and no penalty functional is used, DIP produces remarkable results in different image processing tasks, as can be seen in [25]. Even challenging problems like sparse angle computed tomography [3] or compressive sensing [19] can be solved this way.

Developing regularization theory for deep learning methods is of high interest [2]. A very prosperous approach is to combine classical theory with deep learning (e.g. [26, 29]). The number of papers which analyze DIP from a theoretical point of view is also growing. In [18] a functional is constructed, which measures the ability of the neural network φ_θ to approximate an arbitrary image. This functional can then be used as a penalty term in a classical variational method. The authors of [32] analyze how fast a DIP network approximates the low-frequency and high-frequency components of the target image. By controlling this so-called spectral bias, overfitting is avoided. In [20] the ability to denoise images is attributed to convolutional layers, which are faster in fitting smooth images than noisy ones. The role and the choice of hyperparameters for DIP approaches is described in [34]. A Bayesian perspective is presented in [9], where DIP is interpreted as a Gaussian process.

The choice of architecture is crucial for applications of DIP. Generative neural networks are a natural choice due to their ability to reproduce natural looking images. But the authors of [13] took a LISTA-like network [17] instead to develop the so-called analytic deep prior (ADP) approach. This may not lead to a better practical performance of DIP, but it is the foundation for an interesting theory. The main aspect consists in interpreting the training of a neural network as the optimization of a Tikhonov functional. There is an analogy to [1], where the penalty term for a Tikhonov functional is optimized. But in contrast to that, the focus of [13] is on the forward operator inside the functional (see section 2).

This work summarizes deeper investigations of the ADP model. The main result (theorem 3.3) is an equivalence between the ADP approach and classical Ivanov methods [36]. Out of this follows a complete analysis of the regularization properties of ADP including the existence of solutions, stability of reconstructions and convergence towards the ground truth for vanishing noise.

In practical applications of DIP, gradient descent and early stopping is used to minimize the loss function (1.1). Thus, a global (or at least a local) minimum is not reached in general. While this fact was not considered in the theoretical derivation of ADP, we propose a new variant (called ADP-β) which incorporates the effect of early stopping into the model (section 3.2). We also analyze the regularization properties of this new approach.

In section 4 we compare different numerical ways to compute ADP and DIP (with a LISTA-like architecture) solutions of simple inverse problems.[1] We find that numerical solutions of both methods are mostly similar to each other, which is important for using the ADP theory for interpretations of DIP. But there can also be observed some interesting disparities between the

[1] Code available at https://gitlab.informatik.uni-bremen.de/carndt/analytic_deep_prior.

different numerical ways. This illustrates a crucial difference between the analytical definition of DIP as a minimization problem and the numerical implementation as a gradient descent iteration.

2. Preliminaries and methods

We consider an inverse problem based on the operator equation

$$Ax^\dagger = y^\dagger, \tag{2.1}$$

where we want to recover the unknown ground truth $x^\dagger$ as good as possible. The data $y^\dagger$ is typically not known exactly, but we have only access to noisy data y^δ.

Assumption 2.1. We make the following assumptions for the inverse problem (2.1).

- Let X, Y be Hilbert spaces and $A \in L(X, Y)$.
- There exists $x^\dagger \in X$ and for a given $\delta > 0$, it holds $\|y^\delta - y^\dagger\| \leqslant \delta$ for $y^\delta \in Y$.
- Let $R : X \to [0, \infty]$ be a convex, coercive and weakly lower semicontinuous functional with $R \not\equiv \infty$.

We recall the definition of Bregman distances, which we will use in theorem 3.12 for a convergence result, similar to the ones in [6, 21].

Definition 2.2 (Bregman distance). For a convex functional $R: X \to [0, \infty]$ with subdifferential ∂R and $\tilde{x}, x \in X$, the Bregman distance is defined as the set

$$D_R(\tilde{x}, x) = \{R(\tilde{x}) - R(x) - \langle p, \tilde{x} - x\rangle \mid p \in \partial R(x)\}. \tag{2.2}$$

The DIP approach (introduced by [25]) for the inverse problem (2.1) consists in solving

$$\min_\theta \frac{1}{2}\|A\varphi_\theta(z) - y^\delta\|^2 \tag{2.3}$$

via a gradient descent w.r.t. the parameters θ of a neural network φ_θ, as already described in the introduction. Despite the use of a neural network, DIP is a model-based approach and not data-based. To derive the ADP approach, we have to make two assumptions (see [13] for details).

The first one is choosing φ_θ to be a LISTA-like network [17], which consists of several layers of the form

$$x^{l+1} = S_{\alpha\lambda}(x^l - \lambda B^*(Bx^l - y^\delta)), \tag{2.4}$$

where $B = \theta$ is the trainable parameter. Originally, this architecture is inspired by ISTA [12], an algorithm for finding sparse solutions of the inverse problem (2.1), and $S_{\alpha\lambda}$ is the shrinkage function. More general, we can choose $S_{\alpha\lambda}$ to be the proximal mapping of a penalty functional R. Then, (2.4) equals a proximal forward–backward splitting algorithm [11, theorem 3.4] which converges to the solution of the minimization problem

$$\min_{x\in X} \frac{1}{2}\|Bx - y^\delta\|^2 + \alpha R(x). \tag{2.5}$$

The second assumption is letting the number of layers tend to infinity. This might be difficult in practice (see section 4), but it causes the output $\varphi_\theta(z)$ of the network to be a solution of (2.5). Therefore the ADP model (introduced by [13]) is defined as

$$\begin{aligned} &\min_{B\in L(X,Y)} \frac{1}{2}\|Ax(B)-y^\delta\|^2 \\ \text{s.t.} \quad &x(B) = \arg\min_{x\in X} \frac{1}{2}\|Bx-y^\delta\|^2 + \alpha R(x). \end{aligned} \tag{2.6}$$

While DIP is about optimizing the weights of a neural network, ADP is about optimizing the forward operator in a Tikhonov functional. If we add an additional regularization term for the operator B, we get the (new) ADP-β model

$$\begin{aligned} &\min_{B\in L(X,Y)} \frac{1}{2}\|Ax(B)-y^\delta\|^2 + \beta\|B-A\|^2 \\ \text{s.t.} \quad &x(B) = \arg\min_{x\in X} \frac{1}{2}\|Bx-y^\delta\|^2 + \alpha R(x). \end{aligned} \tag{2.7}$$

The reason for this modification will be explained in section 3.2.

To guarantee uniqueness of $x(B)$, the functional R should be strictly convex, but this is not always required. If we assume R even to be strongly convex, $x(B)$ depends continuously on B as the following theorem states. It will be useful for proving existence and stability results for ADP-β.

Theorem 2.3. *Let $R\colon X \to [0,\infty]$ be a strongly convex, coercive and weakly lower semi-continuous functional. Then*

$$x(B) = \arg\min_{x\in X} \frac{1}{2}\|Bx-y^\delta\|^2 + \alpha R(x) \tag{2.8}$$

depends continuously on $B \in L(X,Y)$.

The proof can be found in the appendix A.1.

3. Theoretical results

3.1. Equivalence to classical methods

DIP solutions of inverse problems are naturally restricted to be the output of a neural network. Analogously, only elements of the set

$$U_{\alpha R} = \left\{ \hat{x}\in X \;\middle|\; \exists B\in L(X,Y)\,:\, \hat{x} = \arg\min_{x\in X} \frac{1}{2}\|Bx-y^\delta\|^2 + \alpha R(x) \right\} \tag{3.1}$$

can be solutions of the ADP approach. By definition

$$\min_{x\in U_{\alpha R}} \frac{1}{2}\|Ax-y^\delta\|^2 \tag{3.2}$$

is equivalent to the original ADP problem (2.6). To get a better understanding of this minimization problem we investigate $U_{\alpha R}$. It will turn out that the set $U_{\alpha R}$ can be characterized in a

much easier way, even without using an operator $B \in L(X, Y)$. For this purpose, we formulate the following lemmas.

Lemma 3.1. *Let $R: X \to [0, \infty]$ be a convex, coercive and weakly lower semicontinuous functional and $\hat{x} \in X$, $y^\delta \in Y$, $y^\delta \neq 0$ and $\alpha > 0$ be arbitrary. If there exists $v \in \partial R(\hat{x})$ such that*

$$\alpha \langle v, \hat{x} \rangle \leqslant \frac{\|y^\delta\|^2}{4} \tag{3.3}$$

holds, then there exists a linear operator $B \in L(X, Y)$ which fulfills

$$\hat{x} = \arg\min_{x \in X} \frac{1}{2}\|Bx - y^\delta\|^2 + \alpha R(x). \tag{3.4}$$

The proof can be found in the appendix A.2.

Lemma 3.2. *Let $R: X \to [0, \infty]$ be a convex, coercive and weakly lower semicontinuous functional and $\hat{x} \in X$, $y^\delta \in Y$, $\alpha > 0$ be arbitrary. If for every $v \in \partial R(\hat{x})$*

$$\alpha \langle v, \hat{x} \rangle > \frac{\|y^\delta\|^2}{4} \tag{3.5}$$

holds, then there exists no linear operator $B \in L(X, Y)$ which fulfills

$$\hat{x} = \arg\min_{x \in X} \frac{1}{2}\|Bx - y^\delta\|^2 + \alpha R(x). \tag{3.6}$$

The proof can be found in the appendix A.3. For given $y^\delta \in Y$, $\hat{x} \in X$ and a penalty term R, these lemmas state whether there exists a linear forward operator $B: X \to Y$ such that $\hat{x}$ is the Tikhonov solution w.r.t. R of the inverse problem w.r.t. y^δ. As a consequence, we can write the ADP minimization problem with a much simpler side constraint.

Theorem 3.3. *Let assumption 2.1 hold. Then, for all $y^\delta \in Y$, $y^\delta \neq 0$, $\alpha > 0$ the formulation*

$$\begin{aligned} &\min_{x \in X} \frac{1}{2}\|Ax - y^\delta\|^2 \\ &\text{s.t.} \quad \exists\, v \in \partial R(x): \quad \alpha \langle v, x \rangle \leqslant \frac{\|y^\delta\|^2}{4} \end{aligned} \tag{3.7}$$

is equivalent to the ADP-problem (2.6).

Proof. According to lemmas 3.1 and 3.2 there exists a linear operator $B \in L(X, Y)$ such that

$$\hat{x} = \arg\min_{x \in X} \frac{1}{2}\|Bx - y^\delta\|^2 + \alpha R(x) \tag{3.8}$$

if and only if $\hat{x}$ fulfills the side constraint of (3.7). □

Remark 3.4. For the standard Tikhonov penalty term $R(x) = \frac{1}{2}\|x\|^2$, it holds $\partial R(x) = x$. In this case we get

$$\begin{aligned} &\min_{x \in X} \frac{1}{2}\|Ax - y^\delta\|^2 \\ &\text{s.t.} \quad \|x\|^2 \leqslant \frac{\|y^\delta\|^2}{4\alpha} \end{aligned} \tag{3.9}$$

as an equivalent formulation of the ADP problem (2.6). For $r = \|y^\delta\|^2/(4\alpha)$, this equals the Ivanov regularization method

$$\begin{aligned} &\min_{x\in X} \frac{1}{2}\|Ax - y^\delta\|^2 \\ &\quad \text{s.t.} \quad \|x\|^2 \leqslant r. \end{aligned} \tag{3.10}$$

As [36] shows, this method is in fact equivalent to the Tikhonov method

$$\min_{x\in X} \frac{1}{2}\|Ax - y^\delta\|^2 + \frac{\tilde{\alpha}}{2}\|x\|^2 \tag{3.11}$$

for some $\tilde{\alpha}$ dependent on y^δ and r. We note that the Tikhonov parameter $\tilde{\alpha}$ may be equal to zero and in particular it differs from the parameter α of the ADP problem (see section 3.2).

Remark 3.5. There are also cases in which the side constraint of (3.7) defines a non-convex feasible set. Then, the ADP problem is more difficult to solve. We give a simple two-dimensional example with the penalty term $R\colon \mathbb{R}^2 \to [0, \infty)$,

$$R(x_1, x_2) = \begin{cases} 3 \cdot |x_1 - 5| & \text{for } 3 \cdot |x_1 - 5| \geqslant |x_2|, \\ |x_2| & \text{for } |x_2| > 3 \cdot |x_1 - 5|. \end{cases} \tag{3.12}$$

This functional has a non-centered minimum at $(5, 0)^{\mathrm{T}}$ and the absolute value of its gradient $|\partial R(x)|$ is strongly dependent on the direction. Because of these properties, it is easy to show that the term $\langle v, x\rangle$, $v \in \partial R(x)$ in the side constraint of (3.7) is non-convex w.r.t. $x \in \mathbb{R}^2$.

3.2. Parameter choice and early stopping

By construction of the ADP model, we expect it in application to act like DIP. But in the previous section it turned out that ADP behaves in fact equivalent to classical methods like Tikhonov's. When we apply ADP to an inverse problem, the question arises whether ADP can also deliver something that is 'new' and not equivalent to a Tikhonov solution. This section presents, how the model has to be changed to produce ADP solutions that are more similar to DIP solutions. In the same time, we derive a strategy for choosing the parameter α of the ADP model.

When we compare the ADP method

$$\begin{aligned} &\min_{B\in L(X,Y)} \frac{1}{2} \parallel Ax(B) - y^\delta\|^2 \\ &\text{s.t.} \quad x(B) = \arg\min_{x\in X} \frac{1}{2} \parallel Bx - y^\delta\|^2 + \frac{\alpha_{\mathrm{ADP}}}{2} \parallel x\|^2 \end{aligned} \tag{3.13}$$

to the equivalent (see remark 3.4) Tikhonov method

$$\min_{x\in X} \frac{1}{2}\|Ax - y^\delta\|^2 + \frac{\tilde{\alpha}}{2}\|x\|^2, \tag{3.14}$$

we have to make sure not to confuse the parameters α_{ADP} and $\tilde{\alpha}$ of both models with each other. At first we state the following relation between these parameters.

Lemma 3.6. *If the solutions of* (3.13) *and* (3.14) *coincide,* $\tilde{\alpha} \leqslant \alpha_{\mathrm{ADP}}$ *holds. Equality of the parameters could only occur if* y^δ *was in the kernel of* A^* *or a singular vector of* A.

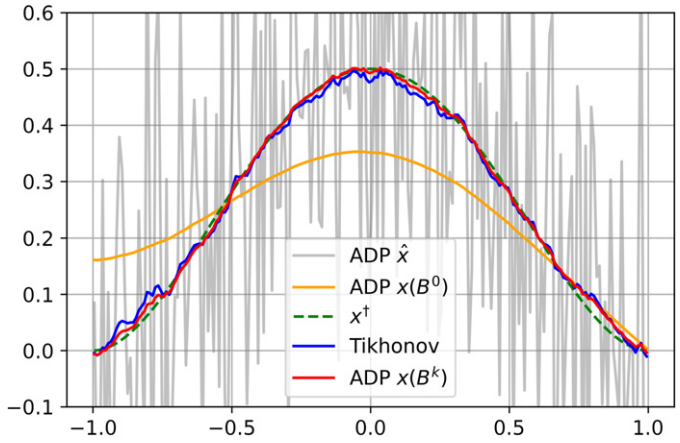


Figure 1. Comparison of ADP solutions during gradient descent to the Tikhonov method (orange: at the start of the gradient descent, red: by early stopping, gray: limit of the gradient descent). The forward operator is an integration like in (4.1) and the data y^δ is contaminated with Gaussian noise (PSNR $= 40$). The Tikhonov parameter and the stopping criterion for ADP were chosen *a posteriori* to achieve the most accurate reconstructions.

The proof can be found in the appendix A.4. In general, we can assume that $\tilde{\alpha} < \alpha_{\text{ADP}}$ holds. So in any application it makes sense to choose the ADP parameter greater than one would choose the parameter of a Tikhonov model. But independent of the parameter choice, the ADP solution will always be equivalent to a Tikhonov solution (remark 3.4). To make ADP more similar to DIP, we apply early stopping [15, section 7.8]. This strategy is often used in the application of DIP but was not considered for the ADP model yet.

For a given inverse problem, we could solve the ADP problem (3.13) with a gradient descent algorithm w.r.t. the operator B (see section 4 for details) and terminate this iteration early. Taking $B^0 = A$ as initial value leads by definition to $x(B^0)$ being equal to the Tikhonov solution w.r.t. the parameter α_{ADP}. We assume the iteration to converge successfully towards the minimizer $\hat{x}$ of (3.13). Since $\hat{x}$ is also the minimizer of (3.14), the limit of the iteration is also a Tikhonov solution but w.r.t. the parameter $\tilde{\alpha}$. Because of $\tilde{\alpha} < \alpha_{\text{ADP}}$, the starting solution $x(B^0)$ is a stronger regularized Tikhonov solution than the limit $\hat{x}$ of the iteration (see figure 1).

If we apply early stopping, we take some $x(B^k)$ in between, which in general does not equal a Tikhonov solution w.r.t. A (see figure 1). This strategy makes sense if we expect $x(B^k)$ to be a better solution than (the Tikhonov solutions) $x(B^0)$ and $\hat{x}$. That could be the case if $x(B^0)$ is slightly over-regularized and $\hat{x}$ is slightly under-regularized. Because then, the optimal regularization would lay in between.

Now, we come back to the parameter choice. If we have a criterion for estimating a suitable Tikhonov parameter α_{Tik} for a given inverse problem, we should try to choose α_{ADP} in a way that

$$\tilde{\alpha} < \alpha_{\text{Tik}} < \alpha_{\text{ADP}} \tag{3.15}$$

holds. Because then, $x(B^0)$ will be slightly over-regularized and $\hat{x}$ slightly under-regularized, as proposed.

In the example of figure 1, we see that the ADP solution $x(B^k)$, obtained with early stopping, is a better approximation for the ground truth $x^\dagger$ than the most accurate Tikhonov solution,

which corresponds to α_{Tik}. But this result is strongly dependent on the particular inverse problem. The Tikhonov method is optimal for data that is normally distributed. If the given distribution differs from that, it is theoretically possible that ADP with early stopping produces a better solution than the Tikhonov method.

Finally, we want to include the early stopping strategy directly into the ADP model to be able to investigate its effect on the regularization of inverse problems. Early stopping enforces the iterated variable to stay close to the initial value. Because of $B^0 = A$, we can expect $\|B^k - A\|$ to be small for small k. This leads to using $\|B^k - A\|$ as an additional penalty term in the ADP problem, which has a similar effect as early stopping [4, section 2.3], [33, section 4]. What comes out is the ADP-β model

$$\begin{aligned} &\min_{B\in L(X,Y)} \frac{1}{2}\|Ax(B) - y^\delta\|^2 + \beta\|B - A\|^2 \\ &\text{s.t.} \quad x(B) = \arg\min_{x\in X} \frac{1}{2}\|Bx - y^\delta\|^2 + \alpha R(x). \end{aligned} \tag{3.16}$$

3.3. Properties of ADP

The equivalence between ADP and the Ivanov method (with general convex penalty term R), shown in section 3.1, allows to obtain some regularization properties (existence, stability, convergence) for ADP. We suppose that assumption 2.1 holds. Besides the functional

$$\tilde{R}(x) = \min_{v\in\partial R(x)} \langle v, x\rangle \tag{3.17}$$

is assumed to be well-defined. Because then, the side constraint of (3.7) can be formulated as

$$\tilde{R}(x) \leqslant \frac{\|y^\delta\|^2}{4\alpha}. \tag{3.18}$$

Due to $\langle v, x\rangle = R(x) + R^*(v)$ for $v \in \partial R(x)$, where R^* denotes the convex conjugated functional, coercivity of R implies coercivity of $\tilde{R}$.

Remark 3.7 (Existence). There exists a solution of the ADP problem (2.6) if the functional $\tilde{R}$, defined in (3.17), is weakly lower semicontinuous. This follows from the equivalence theorem 3.3 and [37, theorem 2.1] about the existence of Ivanov solutions.

Uniqueness of solutions and stability w.r.t. the data y^δ is less trivial. First, the right-hand side of the side constraint (3.18) is dependent on y^δ, which is not the case for ordinary Ivanov problems. Secondly, we know from remark 3.5, that the constraint (3.18) does not always define a convex feasible set. Nevertheless, for the special case $R(x) = \frac{1}{2}\|x\|^2$ we can obtain a convenient stability result. In this case, $\tilde{R}(x) = \|x\|^2$ is a strictly convex functional. Additionally, if the given inverse problem is ill-posed, we can assume the ADP solutions to fulfill the constraint (3.18) with equality. Under these conditions, the following theorem provides stability of ADP.

Theorem 3.8 (Stability). *For $R(x) = \frac{1}{2}\|x\|^2$, let $(y_k) \subset Y$ be a sequence with $y_k \to \hat{y} \in Y$ and assume that the corresponding ADP solutions x_k, $\hat{x}$ are unique and fulfill the side constraint in (3.9) with equality. Then ADP is stable, which means $x_k \to \hat{x}$.*

The proof can be found in the appendix A.5.

To obtain a convergence result for ADP, it makes sense to use standard convergence theorems, either of the Tikhonov method [21, theorem 4.4] or of the Ivanov method [23, theorem 2.5], [31, theorem 3]. They differ especially in the source conditions they require

for the ground truth $x^\dagger$ and in the parameter choice rules. If we assume $\tilde{R}$ to be convex, by [36, theorem 2] and the equivalence theorem 3.3, the Tikhonov problem

$$\min_{x\in X} \frac{1}{2}\|Ax - y^\delta\|^2 + \tilde{\alpha}\tilde{R}(x) \tag{3.19}$$

is equivalent to the ADP formulation (3.7) for suitable chosen $\tilde{\alpha} \geqslant 0$.

Remark 3.9 (Convergence). Because of the equivalence between (3.7) and (3.19), the convergence of ADP solutions x_α^δ to $x^\dagger$ for vanishing δ w.r.t. the Bregman distance can be directly derived from Tikhonov convergence theorems. But the ADP parameter α does not coincide with the Tikhonov parameter $\tilde{\alpha}$. That's why, for ADP we do not get an explicit parameter choice rule like $\alpha \sim \delta$. Besides, a source condition for $x^\dagger$ has to be fulfilled by the functional $\tilde{R}$ (defined in (3.17)) and not by the penalty term R.

3.4. Properties of ADP-β

For proving the existence of solutions of variational regularization schemes, [21, theorem 3.1] provides a useful framework. If we want to apply this for ADP-β, it has to be ensured that $B \mapsto x(B)$ is weak-weak continuous [21, assumptions 2.1]. But unfortunately, in general this is not the case.

To obtain convenient regularization properties anyway, we restrict to $X = Y = L^2(\Omega)$ with $\Omega \subset \mathbb{R}^n$. In this setting, we consider a forward operator $A : X \to Y$ that can be parameterized by a function $f \in L^p(\Omega)$, $p \in [1, \infty)$. More precisely, we take a continuous, bilinear operator $T : L^p(\Omega) \times X \to Y$ and define

$$Ax = T(f, x). \tag{3.20}$$

The same parameterization of operators by functions is used in [5]. One typical example would be a convolutional operator $T(f, x) = f_* x$.

The crucial idea is the additional restriction $f \in W^{1,p}(\Omega)$ to take advantage of the compact embedding of Sobolev spaces $W^{1,p}(\Omega) \subset L^p(\Omega)$. A similar strategy is used in [24] for achieving weak-weak continuity of the forward operator.

We define the parameterized ADP-β approach as

$$\begin{aligned} &\min_{g\in W^{1,p}} \frac{1}{2}\|T(f, x_g) - y^\delta\|_{L^2}^2 + \beta\|f - g\|_{W^{1,p}}^2 \\ &\text{s.t.} \quad x_g = \arg\min_{x\in L^2} \frac{1}{2}\|T(g, x) - y^\delta\|_{L^2}^2 + \alpha R(x). \end{aligned} \tag{3.21}$$

In particular, this can be interpreted as a Tikhonov method for solving the nonlinear inverse problem $F(g^\dagger) = y^\dagger$ with the forward operator $F : W^{1,p}(\Omega) \to Y$, $F(g) = T(f, x_g)$.

Remark 3.10 (Existence). The forward operator F is weak-strong continuous if the penalty term R is strongly convex. This holds, because weak convergence $g_k \rightharpoonup g$ w.r.t. $W^{1,p}(\Omega)$ implies convergence by norm in $L^p(\Omega)$, by theorem 2.3 the convergence of $x_{g_k} \to x_g$ follows, and the bilinear operator T is continuous. This is more than enough to fulfill the assumptions of [21, theorem 3.1], which provides the existence of a solution of (3.21).

A weak stability result for the parameterized ADP-β method could be directly obtained from [21, theorem 3.2]. But this particular framework even allows to prove strong stability.

Theorem 3.11 (Stability). *For $p = 2$, let R be a strongly convex penalty term, $(y_k) \subset Y$ a convergent sequence with $y_k \to \hat{y}$ and $(g_k) \subset W^{1,p}(\Omega)$ the corresponding solutions of the ADP-β problem (3.21). Then, (g_k) has a convergent subsequence and the limit of each subsequence is an ADP-β solution corresponding to $\hat{y}$.*

The proof can be found in the appendix A.6.

While proving existence and stability of ADP-β-solutions required a smart parameterization and the use of compact embeddings, a convergence theorem (w.r.t. the Bregman distance) can be proven for the general formulation (2.7). Similar to classical results like [21, theorem 4.4] or [6, theorem 2], we need to assume a source condition

$$\exists\, w \in Y: \quad A^* w \in \partial R(x^\dagger). \tag{3.22}$$

The parameter β turns out to be really helpful for obtaining a convergence result.

Theorem 3.12 (Convergence). *Let assumption 2.1 hold, $x^\dagger$ be an R-minimizing solution of (2.1) which fulfills the source condition (3.22) and assume there exist ADP-β solutions $\hat{x}_\alpha^\delta$ of (2.7). If α is chosen proportional to δ, there exists $d \in D_R(\hat{x}_\alpha^\delta, x^\dagger)$ which fulfills $d = O(\delta)$.*

The proof can be found in the appendix A.7.

4. Numerical computations

Aim. We want to see whether there is a similarity between ADP and DIP also on the numerical side. The ADP approach is based on the idea of using a LISTA network in a DIP method. Usually LISTA architectures contain round about ten layers, but ADP is motivated with a network of infinite depth (see section 2). To derive the ADP model, the output of this infinite network is then replaced by the solution of a minimization problem. So the question arises, whether numerically computed ADP solutions of an inverse problem are yet similar to solutions obtained via a DIP with LISTA architecture.

In this section, we present algorithms for the computation of ADP solutions and we compare them with DIP solutions. In doing so, the focus is not on the performance of the methods (in comparison to other state-of-the-art reconstruction algorithms) but on the similarity of the different solutions.

Methods. From theorem 3.3, we know that the ADP problem is equivalent to an Ivanov problem. This creates a possibility to compute ADP solutions easily, fast and almost exactly (we call this method ADP Ivanov). In contrast to that, it is more difficult to realize a LISTA architecture with infinite depth. But there are at least two possibilities to simulate such a network.

The first idea (algorithm 1: DIP LISTA $L = \infty$) is to begin with a network φ_B of ten layers and to increase the network depth during the training process of the DIP. This is done implicitly with a simple trick. In each training step, the network's input is set to be the network's output of the previous step [13, appendix 3]. So the original input will pass through more and more layers and in each step the last ten layers are optimized (via backpropagation).

The second idea (algorithm 2: ADP IFT) is to compute $x(B)$ from (2.6) with a classical algorithm like ISTA. After that, one can compute the gradient of $x(B)$ w.r.t. B (see the proof of [13, lemma 4.1]) via the implicit function theorem (IFT). Thus, backpropagation through a big amount of layers is avoided.

Algorithm 1. DIP LISTA L $= \infty$.

initialize B_0, z_0 (e.g. $B_0 = A$, $z_0 =$ random noise);
for $k = 0, 1, \ldots$ **do**
  $z_{k+1} = \varphi_{B_k}(z_k)$;
  $\text{loss}_k = \frac{1}{2}\|A\varphi_{B_k}(z_k) - y^\delta\|^2$;
  $B_{k+1} = \text{update}(\nabla_{B_k}\text{loss}_k)$;
end
return z_k

Algorithm 2. ADP IFT.

initialize B_0 (e.g. $B_0 = A$);
for $k = 0, 1, \ldots$ **do**
  1. Calculate $x(B_k)$ with fixed point iteration
  2. IFT provides: $\nabla_{B_k} x(B_k)$;
  3. Update:
  $\text{loss}_k = \frac{1}{2}\|Ax(B_k) - y^\delta\|^2$;
  $B_{k+1} = \text{update}(\nabla_{B_k}\text{loss}_k)$;
end
return $x(B_k)$

For the standard DIP approach, we use a LISTA-like architecture of ten layers (DIP LISTA $L = 10$) and optimize the weights via backpropagation. So, in total we compare four different methods (ADP Ivanov, ADP IFT, DIP LISTA $L = \infty$, DIP LISTA, $L = 10$). Since solving the Ivanov problem results in the exact ADP solution, we use this as a reference for the other three methods (for which we do not have convergence guarantees).

In all methods we use the elastic net functional [38] $R(x) = \alpha_1\|x\|_1 + \frac{\alpha_2}{2}\|x\|^2$ as a penalty term. So there is one parameter for ℓ^1-regularization (leads to sparsity) and one parameter for ℓ^2-regularization (leads to stability and smoothness). In the LISTA-architecture, this is realized by subtracting the gradient of the ℓ^2-term before applying the activation function.

Setting. We consider two different artificial inverse problems (inversion of the integration operator and a deconvolution) on $L^2(I)$ for an interval $I \subset \mathbb{R}$. The forward operators are

$$(A_1 x)(t) = \int_0^t x(s)\,\mathrm{d}s \quad \text{and} \quad A_2 x = g * x, \tag{4.1}$$

g being a Gaussian function. Both of them lead to ill-posed inverse problems. We chose three different ground truth functions and created data by applying the forward operators and adding normally distributed random noise. This leads to six examples in total, which is enough for some basic observations. Figure 2 shows the reconstructions corresponding to the integration operator A_1. The three rows contain the three different ground truth functions and each column contains a different method. For comparison, the actual ADP solution (ADP Ivanov) and the ground truth is displayed in every plot. Since we are only interested in finding similarities and disparities between the solutions of the different methods, the choice of the regularization parameters plays a minor role. So, we took the same values α_1, α_2 for each method and simply

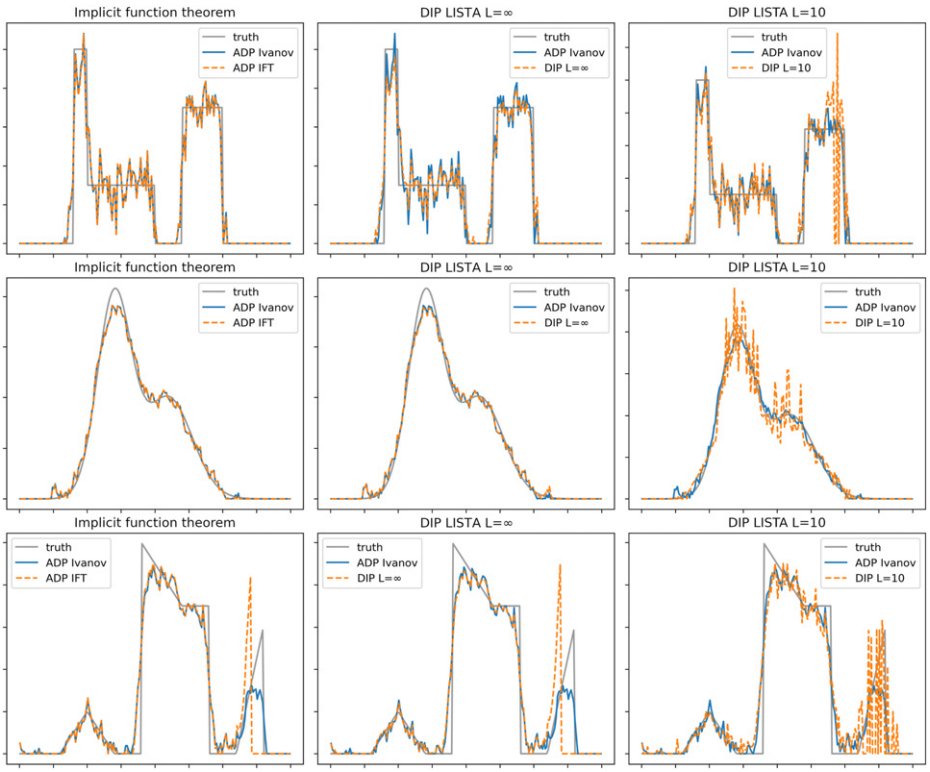


Figure 2. Computation of ADP and DIP reconstructions via the IFT, via a DIP with LISTA network ($L = \infty$ and $L = 10$) and via the equivalent Ivanov problem. The forward operator is A_1 (integration). The given data has PSNR $= 40$ due to additive Gaussian noise. The regularization parameters α_1, α_2 are chosen for each example (row) separately but are the same for each method (column).

chose them *a posteriori* for each example to minimize the L^2-error between reconstructions and ground truth. Figure 3 shows the analogous results for the deconvolution problem (forward operator A_2).

Observations. From these experiments, we can make the following observations. There is a significant difference between using $L = 10$ or $L = \infty$ layers in a LISTA network. With an infinite number of layers, the reconstructions are looking more realistic. The results of the DIP LISTA L $= \infty$ (algorithm 1) method and of the IFT method (algorithm 2) are always looking quite similar. This was expected because both of them simulate an infinitely deep LISTA network. Differences are probably due to the different ways the gradients are computed or due to slow convergence of the methods.

In most of the cases, the reconstructions of these both methods are looking quite similar to the actual ADP solution. But sometimes they contain artifacts (e.g. the peaks in figure 3, third row). It seems that there are some spots which are hard to reconstruct for the DIP methods and others are rather simple. Besides, the ADP problem (2.6) is not a convex minimization problem w.r.t. B. So there is no guarantee for the methods which do gradient descent (DIP LISTA $L = \infty$ and the IFT method) to converge towards the global minimizer. Figure 4 shows that the reconstructions of these methods are indeed dependent on the initial value B_0 of the

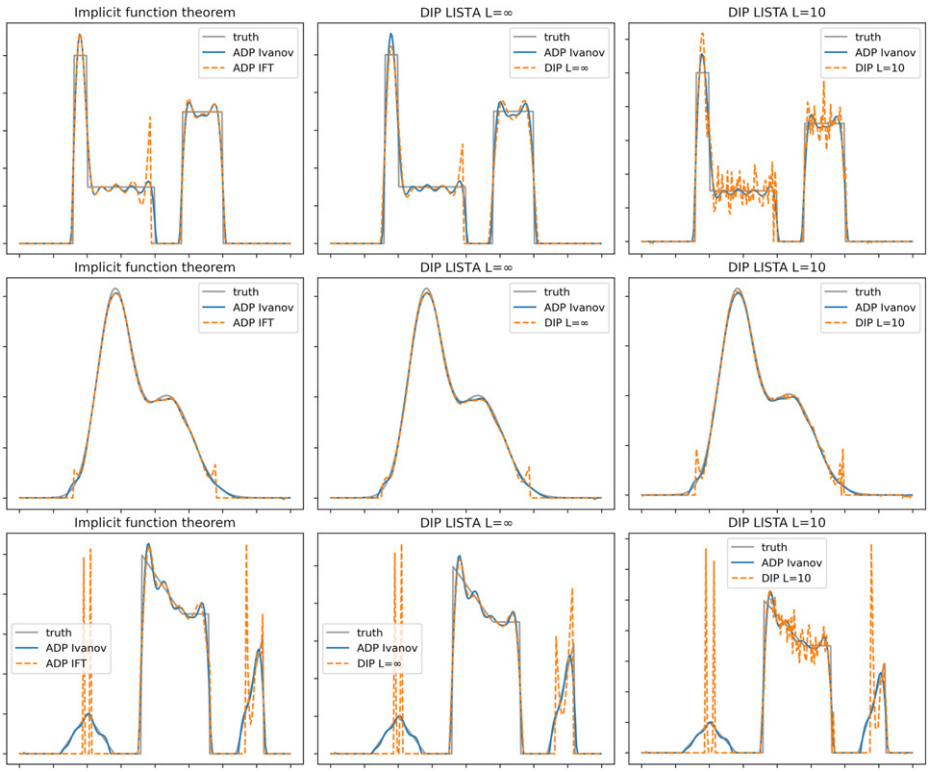


Figure 3. Computation of ADP and DIP reconstructions via the IFT, via a DIP with LISTA network ($L = \infty$ and $L = 10$) and via the equivalent Ivanov problem. The forward operator is A_2 (convolution). The given data has PSNR $= 45$ due to additive Gaussian noise. The regularization parameters α_1, α_2 are chosen for each example (row) separately but are the same for each method (column).

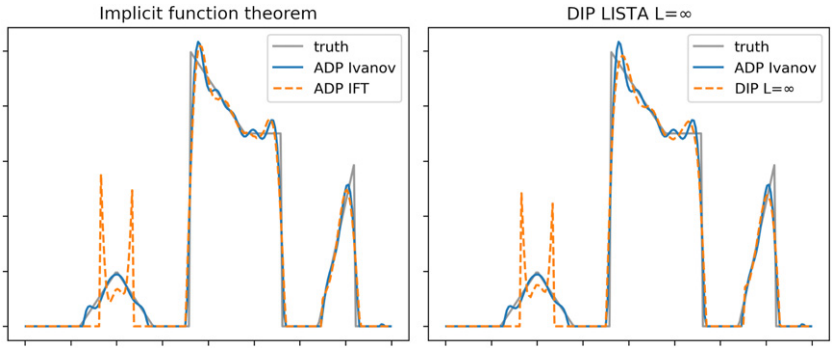


Figure 4. The same setting and methods as in figure 3 but with different initial values B_0.

algorithms. In contrast to that, the Ivanov problem from theorem 3.3 is convex (with the elastic net penalty term R). That's probably why the actual ADP solutions are the only ones which never contain strange artifacts and the only ones that are always quite good reconstructions of the ground truth.

The easiest possibility to slightly improve the reconstruction quality is to apply early stopping. In doing so, the most severe artifacts in the reconstructions can be diminished. This case corresponds to the ADP-β approach (see section 3.2), whose additional convex term $\beta\|B - A\|^2$ is a numerical advantage because it stabilizes the gradient descent for finding the minimizer. Indeed, adding the gradient of the β-term to the update in algorithm 2 (ADP IFT) can also diminish the severe artifacts. But we do not include experimental results about this, since the most interesting part is the comparison with the equivalent Ivanov problem, which does not exist for ADP-β.

The main conclusion is the numerical verification of the derivation of the ADP problem from the DIP approach. It is possible to use the theoretical analysis of the ADP problem for interpretations of the DIP approach because of the similarity between the reconstructions from the different numerical methods. However, the examples from figure 4 illustrate that DIP can be formulated as a minimization problem (2.3) but a numerical computed solution is not automatically a global minimizer of this problem. If early stopping is used, it is probably not even a local minimizer. Hence, there is a significant difference between the theoretical definition and the practical implementation of DIP.

5. Conclusion

ADP and ADP-β were introduced as methods for solving ill-posed inverse problems in a typical Hilbert space setting (assumption 2.1). Both of them are motivated by considering DIP with a LISTA-like architecture. The main result is an equivalence of ADP to the classical method of Ivanov regularization.

We have proven existence, stability and convergence results for both ADP and ADP-β. The obtained regularization properties are comparable to the ones of classical methods like Tikhonov's. In principal, these results can be transferred to DIP with LISTA-like networks. But due to non-convexity of the DIP minimization problem, numerically computed DIP solutions can differ significantly from exact ADP solutions, although they are similar in many cases. We conclude that theoretical analyses of the DIP approach should consider the whole optimization process and not only the properties of the minimizer.

One very important part is the early stopping of the DIP optimization process. In the ADP setting, we incorporated this strategy with an additional penalty term, which resulted in the ADP-β model. The effect of this regularization can be seen by comparing the convergence theorems of ADP and ADP-β. Theorem 3.12 provides a parameter choice rule ($\alpha \sim \delta$) for ADP-β, which is a big advantage over ADP.

A generalization of the ADP regularization results to DIP with general convolutional neural networks (CNNs) would be very desirable. The LISTA architecture was suitable because of its similarity to proximal splitting algorithms and the possibility to interpret the output as a solution of a variational problem. Finding similar connections for general CNNs is harder. However in [7], CNNs are used to model proximal mappings and in [28], CNNs are interpreted as algorithms for sparse coding. Besides, [10] asserts that most common activation functions are in fact proximal mappings and they establish a theory for characterizing the fixed point sets of neural networks as solutions of variational inequalities. These directions could provide ideas for possible future extensions.

Acknowledgments

I want to thank Dr. Daniel Otero Baguer, Prof. Peter Maaß, Dr. Tobias Kluth and many more colleagues from the University of Bremen and Dr. Yury Korolev from the University of Cambridge for helpful advice and feedback.

Data availability statement

All data that support the findings of this study are included within the article (and any supplementary files).

Appendix A. Proofs of theoretical results

A.1. Theorem 2.3

Continuity of $B \mapsto x(B)$.

Proof. Let $(B_k) \subset L(X, Y)$ be a sequence of operators with $B_k \to B \in L(X, Y)$. At first, we mention that the sequence $(x(B_k))$ is bounded because

$$\alpha R(x(B_k)) \leqslant \frac{1}{2}\|B_k x(B_k) - y^\delta\|^2 + \alpha R(x(B_k)) \leqslant \frac{1}{2}\|B_k x(B) - y^\delta\|^2 + \alpha R(x(B)) \tag{A.1}$$

holds. Further, we can estimate

$$\begin{aligned}
&\frac{1}{2}\|Bx(B_k) - y^\delta\|^2 + \alpha R(x(B_k)) \\
&\quad = \frac{1}{2}\|B_k x(B_k) - y^\delta + (B - B_k)\, x(B_k)\|^2 + \alpha R(x(B_k)) \\
&\quad \leqslant \frac{1}{2}(\|B_k x(B_k) - y^\delta\| + \|(B - B_k)\, x(B_k)\|)^2 + \alpha R(x(B_k)) \\
&\quad = \frac{1}{2}\|B_k x(B_k) - y^\delta\|^2 + \alpha R(x(B_k)) \\
&\qquad + \|(B - B_k)\, x(B_k)\| \cdot \left(\|B_k x(B_k) - y^\delta\| + \frac{1}{2}\|(B - B_k)\, x(B_k)\|\right).
\end{aligned} \tag{A.2}$$

Because of the boundedness of $(x(B_k))$ and the convergence $B_k \to B$, the term

$$\|(B - B_k)\, x(B_k)\| \cdot \left(\|B_k x(B_k) - y^\delta\| + \frac{1}{2}\|(B - B_k)\, x(B_k)\|\right) \tag{A.3}$$

converges to zero. For the remaining terms, we can estimate

$$\frac{1}{2}\|B_k x(B_k) - y^\delta\|^2 + \alpha R(x(B_k)) \leqslant \frac{1}{2}\|B_k x(B) - y^\delta\|^2 + \alpha R(x(B)), \tag{A.4}$$

and $\|B_k x(B) - y^\delta\|^2$ converges to $\|Bx(B) - y^\delta\|^2$. So $(x(B_k))$ is a minimizing sequence of the strongly convex functional $\frac{1}{2}\|Bx - y^\delta\|^2 + \alpha R(x)$ because $x(B)$ is the minimizer. By [27, theorem 1], the minimizing sequence converges to the minimizer $x(B)$. □

A.2. Lemma 3.1

First part of the equivalence theorem for ADP to Ivanov problems.

Proof. Let $\hat{x}, v, y^\delta, \alpha$ and R be given according to the assumptions. We have to find a linear operator B such that

$$-B^*(B\hat{x} - y^\delta) \in \alpha \partial R(\hat{x}). \tag{A.5}$$

holds. Because of $v \in \partial R(\hat{x})$, we just try to solve the equation

$$-B^*(B\hat{x} - y^\delta) = \alpha v \tag{A.6}$$

for B. If $\hat{x} = 0$, solving would be trivial. Otherwise, we can decompose v into

$$v = \mu\hat{x} + v_\perp \quad \text{s.t. } \langle v_\perp, \hat{x}\rangle = 0. \tag{A.7}$$

Accordingly it is $\mu = \langle v, \hat{x}\rangle / \|\hat{x}\|^2$. With that, we can write the equation from above as

$$B^*B\hat{x} + \alpha\mu\hat{x} + \alpha v_\perp = B^*y^\delta. \tag{A.8}$$

We consider a linear operator $B\colon X \to Y$ of the form

$$Bx = (\sigma_1\langle x, \hat{x}\rangle + \sigma_2\langle x, v_\perp\rangle) \cdot y^\delta \tag{A.9}$$

with two coefficients σ_1 and σ_2 to be determined later. Then, the adjoint operator is given by

$$B^*y = \langle y, y^\delta\rangle(\sigma_1\hat{x} + \sigma_2 v_\perp) \tag{A.10}$$

and it holds

$$B^*B\hat{x} = B^*((\sigma_1\|\hat{x}\|^2) \cdot y^\delta) = \sigma_1^2\|\hat{x}\|^2\|y^\delta\|^2\hat{x} + \sigma_1\sigma_2\|\hat{x}\|^2\|y^\delta\|^2 v_\perp, \tag{A.11}$$

$$B^*y^\delta = \sigma_1\|y^\delta\|^2\hat{x} + \sigma_2\|y^\delta\|^2 v_\perp. \tag{A.12}$$

To fulfill (A.8), we have to solve

$$\sigma_1^2\|\hat{x}\|^2\|y^\delta\|^2\hat{x} + \sigma_1\sigma_2\|\hat{x}\|^2\|y^\delta\|^2 v_\perp + \alpha\mu\hat{x} + \alpha v_\perp = \sigma_1\|y^\delta\|^2\hat{x} + \sigma_2\|y^\delta\|^2 v_\perp. \tag{A.13}$$

Because $\hat{x}$ and $v_\perp$ are orthogonal to each other, we get the two equations

$$\sigma_1^2\|\hat{x}\|^2\|y^\delta\|^2 + \alpha\mu = \sigma_1\|y^\delta\|^2, \tag{A.14}$$

$$\sigma_1\sigma_2\|\hat{x}\|^2\|y^\delta\|^2 + \alpha = \sigma_2\|y^\delta\|^2. \tag{A.15}$$

Notice that (A.15) and the coefficient σ_2 could be ignored if $v_\perp = 0$ held.

Equation (A.14) can be solved for σ_1 with a quadratic formula, which leads to

$$\sigma_1 = \frac{1}{2\|\hat{x}\|^2} \pm \sqrt{\frac{1}{4\|\hat{x}\|^4} - \frac{\alpha\mu}{\|\hat{x}\|^2\|y^\delta\|^2}}. \tag{A.16}$$

Accordingly,

$$\frac{\alpha\mu}{\|y^\delta\|^2} \leqslant \frac{1}{4\|\hat{x}\|^2} \tag{A.17}$$

must hold to get real solutions. We know from above that $\mu = \langle v, \hat{x}\rangle / \|\hat{x}\|^2$. If we insert this, we will see that this matches exactly the assumptions of the lemma.

Now, equation (A.15) has to be solved for σ_2. Excluding σ_2 leads to

$$\sigma_2(\sigma_1\|\hat{x}\|^2\|y^\delta\|^2 - \|y^\delta\|^2) + \alpha = 0. \tag{A.18}$$

If the term inside of the parenthesis does not equal zero, there will exist a solution σ_2. If the term equaled zero, equation (A.14) would lead to $\mu = 0$. But in this case, we could choose $\sigma_1 = 0$ (the quadratic formula allows two solutions), and then it is no problem to find a solution for σ_2, too.

By finding solutions for σ_1 and σ_2, we showed that the operator B defined in (A.9) solves equation (A.6). So the lemma is proved. □

A.3. Lemma 3.2

Second part of equivalence theorem for ADP to Ivanov problems.

Proof. Let $\hat{x}, y^\delta, \alpha$ and R be given according to the assumptions. Assume there exists a linear operator B such that

$$0 \in B^*(B\hat{x} - y^\delta) + \alpha\partial R(\hat{x}) \tag{A.19}$$

holds. It follows

$$v := -\frac{1}{\alpha}B^*(B\hat{x} - y^\delta) \in \partial R(\hat{x}). \tag{A.20}$$

We can calculate $\alpha\langle v, \hat{x}\rangle = -\|B\hat{x}\|^2 + \langle y^\delta, B\hat{x}\rangle$. So according to the assumptions,

$$-\|B\hat{x}\|^2 + \langle y^\delta, B\hat{x}\rangle > \frac{\|y^\delta\|^2}{4} \tag{A.21}$$

must hold. But with Young's inequality, we get

$$-\|B\hat{x}\|^2 + \langle y^\delta, B\hat{x}\rangle \leqslant -\|B\hat{x}\|^2 + \frac{1}{4}\|y^\delta\|^2 + \|B\hat{x}\|^2 = \frac{\|y^\delta\|^2}{4}. \tag{A.22}$$

Obviously, this is a contradiction. That's why such an operator B cannot exist. □

A.4. Lemma 3.6

Relation between the ADP parameter and the Tikhonov parameter of the equivalent problem.

Proof. Let $\hat{x}$ be the solution of the ADP problem (3.13). Because of the equivalence to the Tikhonov method, $\hat{x}$ is the solution of (3.14) in the same time. Besides, $x(A)$ is the Tikhonov solution w.r.t. the parameter α_{ADP} of the inverse problem. Because of the minimizing properties of $\hat{x}$ and $x(A)$,

$$\frac{1}{2}\|A\hat{x} - y^\delta\|^2 \leqslant \frac{1}{2}\|Ax(A) - y^\delta\|^2, \tag{A.23}$$

$$\frac{1}{2}\|Ax(A) - y^\delta\|^2 + \frac{\alpha_{\text{ADP}}}{2}\|x(A)\|^2 \leqslant \frac{1}{2}\|A\hat{x} - y^\delta\|^2 + \frac{\alpha_{\text{ADP}}}{2}\|\hat{x}\|^2 \tag{A.24}$$

holds. It follows $\|x(A)\|^2 \leqslant \|\hat{x}\|^2$. Both $x(A)$ and $\hat{x}$ are Tikhonov solutions of the same problem (only with different parameters). So $\tilde{\alpha} \leqslant \alpha_{\text{ADP}}$ must hold because the norm of $\hat{x}$ is greater (or equal) than the norm of $x(A)$.

Now, we assume $\tilde{\alpha} = \alpha_{\text{ADP}} > 0$. By remark 3.4, the problems

$$\min_{x\in X} \frac{1}{2}\|Ax - y^\delta\|^2 + \frac{\alpha_{\text{ADP}}}{2}\|x\|^2, \tag{A.25}$$

$$\min_{x\in X} \frac{1}{2}\|Ax - y^\delta\|^2 \quad \text{s.t. } \|x\|^2 \leqslant \frac{\|y^\delta\|^2}{4\alpha_{\text{ADP}}} \tag{A.26}$$

are equivalent. The solution $\hat{x}$ fulfills

$$(A^*A + \alpha_{\text{ADP}} \cdot \text{Id})\hat{x} = A^*y^\delta \tag{A.27}$$

and we assume $A^*y^\delta \neq 0$. Then, $\hat{x}$ must fulfill the side constraint of (A.26) with equality, otherwise $A^*A\hat{x} = A^*y^\delta$ would hold, which is a contradiction. Accordingly we get $\alpha_{\text{ADP}}\|\hat{x}\|^2 = \|y^\delta\|^2/4$ and by computing the inner product of (A.27) with $\hat{x}$, it follows

$$\|A\hat{x}\|^2 + \frac{\|y^\delta\|^2}{4} = \|A\hat{x}\|^2 + \alpha_{\text{ADP}}\|\hat{x}\|^2 = \langle A\hat{x}, y^\delta\rangle. \tag{A.28}$$

If we then apply the Cauchy–Schwarz and Young's inequality, we get

$$\langle A\hat{x}, y^\delta\rangle \leqslant \|A\hat{x}\| \cdot \|y^\delta\| \leqslant \|A\hat{x}\|^2 + \frac{\|y^\delta\|^2}{4}, \tag{A.29}$$

which means these inequalities must in fact hold as equalities. Therefore, $A\hat{x}$ and y^δ must be linear dependent (Cauchy–Schwarz) and $2\|A\hat{x}\| = \|y^\delta\|$ must hold (Young). It follows

$$A\hat{x} = \frac{1}{2}y^\delta. \tag{A.30}$$

We can plug this into (A.27) and get

$$\alpha_{\text{ADP}}\hat{x} = \frac{1}{2}A^*y^\delta. \tag{A.31}$$

Accordingly $AA^*y^\delta = \alpha_{\text{ADP}}y^\delta$ holds, so y^δ is a singular vector of A. □

A.5. Theorem 3.8

Stability of the ADP approach.

Proof. We follow some of the ideas of the proofs of [14, theorem 2.1] and [31, Theorem 2].

Let x_k and $\hat{x}$ be unique solutions of (3.9) for $y^\delta = y_k, \hat{y}$ with $y_k \to \hat{y}$. The sequence (x_k) is bounded, so there exists a weakly convergent subsequence (x_{k_l}), $x_{k_l} \rightharpoonup x_\infty$. For arbitrary $\varepsilon > 0$ and $x \in X$ with $\|x\|^2 \leqslant \|\hat{y}\|^2 \cdot (4\alpha)^{-1} - \varepsilon$, it holds

$$\|Ax_\infty - \hat{y}\| \leqslant \liminf_{l\to\infty} \|Ax_{k_l} - y_{k_l}\| \leqslant \lim_{l\to\infty} \|Ax - y_{k_l}\| = \|Ax - \hat{y}\| \tag{A.32}$$

because x_{k_l} minimizes the ADP problem w.r.t. y_{k_l} and x fulfills the side constraint for l big enough. With $\varepsilon \to 0$ and because of the uniqueness of the solutions, we obtain $x_\infty = \hat{x}$. Arguing with a subsequence of a subsequence leads to the weak convergence $x_k \rightharpoonup \hat{x}$ of the whole sequence.

According to the assumptions, it holds $\|x_k\|^2 = \|\hat{y}_k\|^2 \cdot (4\alpha)^{-1}$. So $y_k \to \hat{y}$ implies $\|x_k\| \to \|\hat{x}\|$ and together with the weak convergence, we finally obtain $x_k \to \hat{x}$. □

A.6. Theorem 3.11

Stability of the ADP-β approach.

Proof. First, we note that the sequence (g_k) is bounded in $W^{1,2}(\Omega)$. Hence, there exists at least one weakly convergent subsequence. For any subsequence with $g_k \rightharpoonup \hat{g}$, it holds

$$T(f, x_{g_k}) - y_k \;\to\; T(f, x_{\hat{g}}) - \hat{y} \tag{A.33}$$

because of the arguments from remark 3.10. For arbitrary $g \in W^{1,2}(\Omega)$,

$$\begin{aligned}
&\frac{1}{2}\|T(f, x_{\hat{g}}) - \hat{y}\|_{L^2}^2 + \beta\|\hat{g} - f\|_{W^{1,2}}^2 \\
&\quad\leqslant \liminf_{k\to\infty} \frac{1}{2}\|T(f, x_{g_k}) - y_k\|_{L^2}^2 + \beta\|g_k - f\|_{W^{1,2}}^2 \\
&\quad\leqslant \lim_{k\to\infty} \frac{1}{2}\|T(f, x_g) - y_k\|_{L^2}^2 + \beta\|g - f\|_{W^{1,2}}^2 \\
&\quad= \frac{1}{2}\|T(f, x_g) - \hat{y}\|_{L^2}^2 + \beta\|g - f\|_{W^{1,2}}^2
\end{aligned} \tag{A.34}$$

holds because of the minimizing property of g_k w.r.t. y_k. Hence, $\hat{g}$ is a minimizer of (3.21) w.r.t $\hat{y}$. If we choose $g = \hat{g}$, the first and the last line in (A.34) coincide, and we get

$$\begin{aligned}
&\lim_{k\to\infty} \frac{1}{2}\|T(f, x_{g_k}) - y_k\|_{L^2}^2 + \beta\|g_k - f\|_{W^{1,2}}^2 \\
&\quad= \frac{1}{2}\|T(f, x_{\hat{g}}) - \hat{y}\|_{L^2}^2 + \beta\|\hat{g} - f\|_{W^{1,2}}^2.
\end{aligned} \tag{A.35}$$

If follows $\lim_{k\to\infty}\|g_k - f\|_{W^{1,2}}^2 = \|\hat{g} - f\|_{W^{1,2}}^2$. Hence, (g_k) converges by norm to $\hat{g}$. □

A.7. Theorem 3.12

Convergence of the ADP-β approach.

Proof. According to (3.22), we can choose $d = R(\hat{x}_\alpha^\delta) - R(x^\dagger) - \langle A^* w, \hat{x}_\alpha^\delta - x^\dagger\rangle$ and there exists an operator $\hat{B} \in L(X, Y)$ that fulfills $\hat{x}_\alpha^\delta = x(\hat{B})$.

Because of the minimizing property of $\hat{x}_\alpha^\delta$,

$$\alpha R(\hat{x}_\alpha^\delta) \leqslant \frac{1}{2}\|\hat{B}\hat{x}_\alpha^\delta - y^\delta\|^2 + \alpha R(\hat{x}_\alpha^\delta) \leqslant \frac{1}{2}\|\hat{B}x^\dagger - y^\delta\|^2 + \alpha R(x^\dagger). \tag{A.36}$$

holds. If follows

$$\begin{aligned}
d = R(\hat{x}_\alpha^\delta) - R(x^\dagger) - \langle A^* w, \hat{x}_\alpha^\delta - x^\dagger\rangle &\leqslant \frac{1}{2\alpha}\|\hat{B}x^\dagger - y^\delta\|^2 - \langle w, A\hat{x}_\alpha^\delta - y^\dagger\rangle \\
&\leqslant \frac{1}{2\alpha}\left(\|\hat{B}x^\dagger - Ax^\dagger\| + \|y^\dagger - y^\delta\|\right)^2 + \|w\|\|A\hat{x}_\alpha^\delta - y^\dagger\| \\
&\leqslant \frac{1}{2\alpha}\left(\|x^\dagger\|\|\hat{B} - A\| + \delta\right)^2 + \|w\|\|A\hat{x}_\alpha^\delta - y^\dagger\|.
\end{aligned} \tag{A.37}$$

We will show $\|\hat{B} - A\| = O(\delta)$ and $\|A\hat{x}_\alpha^\delta - y^\dagger\| = O(\delta)$ to deduce $d = O(\delta)$ for α chosen proportional to δ. Because of the minimizing property of $\hat{B}$, we get

$$\beta \cdot \|\hat{B} - A\|^2 \leqslant \frac{1}{2}\|Ax(\hat{B}) - y^\delta\|^2 + \beta \cdot \|\hat{B} - A\|^2 \leqslant \frac{1}{2}\|Ax(A) - y^\delta\|^2. \quad \text{(A.38)}$$

From standard convergence results of the Tikhonov method [21, theorem 4.4] or [6, theorem 2], we get $\|Ax(A) - y^\delta\| = O(\delta)$. So $\|\hat{B} - A\| = O(\delta)$ holds.

Besides,

$$\|A\hat{x}_\alpha^\delta - y^\dagger\| \leqslant \|Ax(\hat{B}) - y^\delta\| + \|y^\delta - y^\dagger\| \leqslant \|Ax(A) - y^\delta\| + \delta \quad \text{(A.39)}$$

holds and we can use $\|Ax(A) - y^\delta\| = O(\delta)$ again. So $d = O(\delta)$ follows. □

ORCID iDs

Clemens Arndt https://orcid.org/0000-0001-5607-4074

References

[1] Alberti G S, De Vito E, Lassas M, Ratti L and Santacesaria M 2021 Learning the optimal Tikhonov regularizer for inverse problems *Advances in Neural Information Processing Systems* ed A Beygelzimer, Y Dauphin, P Liang and J W Vaughan (Red Hook, NY: Curran Associates, Inc.)
[2] Arridge S, Maass P, Öktem O and Schönlieb C-B 2019 Solving inverse problems using data-driven models *Acta Numer.* **28** 1–174
[3] Baguer D O, Leuschner J and Schmidt M 2020 Computed tomography reconstruction using deep image prior and learned reconstruction methods *Inverse Problems* **36** 094004
[4] Bishop C 1995 Regularization and complexity control in feed-forward networks *Int. Conf. Artificial Neural Networks ICANN'95* pp 141–8
[5] Bleyer I and Ramlau R 2013 A double regularization approach for inverse problems with noisy data and inexact operator *Inverse Problems* **29** 025004
[6] Burger M and Osher S 2004 Convergence rates of convex variational regularization *Inverse Problems* **20** 1411–21
[7] Celledoni E, Ehrhardt M J, Etmann C, Owren B, Schönlieb C-B and Sherry F 2021 Equivariant neural networks for inverse problems *Inverse Problems* **37** 085006
[8] Chambolle A and Pock T 2011 A first-order primal-dual algorithm for convex problems with applications to imaging *J. Math. Imaging Vis.* **40** 120–45
[9] Cheng Z, Gadelha M, Maji S and Sheldon D 2019 A Bayesian perspective on the deep image prior *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*
[10] Combettes P L and Pesquet J-C 2020 Deep neural network structures solving variational inequalities *Set-Valued Var. Anal.* **28** 491–518
[11] Combettes P L and Wajs V R 2005 Signal recovery by proximal forward-backward splitting *Multiscale Model. Simul.* **4** 1168–200
[12] Daubechies I, Defrise M and De Mol C 2004 An iterative thresholding algorithm for linear inverse problems with a sparsity constraint *Commun. Pure Appl. Math.* **57** 1413–57
[13] Dittmer S, Kluth T, Maass P and Otero Baguer D 2020 Regularization by architecture: a deep prior approach for inverse problems *J. Math. Imaging Vis.* **62** 456–70
[14] Engl H W, Kunisch K and Neubauer A 1989 Convergence rates for Tikhonov regularisation of non-linear ill-posed problems *Inverse Problems* **5** 523–40
[15] Goodfellow I, Bengio Y and Courville A 2016 *Deep Learning* (Cambridge, MA: MIT Press)

[16] Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A and Bengio Y 2014 Generative adversarial nets *Advances in Neural Information Processing Systems* vol 27 ed Z Ghahramani, M Welling, C Cortes, N Lawrence and K Q Weinberger (Red Hook, NY: Curran Associates, Inc.)
[17] Gregor K and LeCun Y 2010 Learning fast approximations of sparse coding *Proc. 27th Int. Conf. Int. Conf. Machine Learning, ICML'10* (Madison, WI, USA) (Omnipress) pp 399–406
[18] Habring A and Holler M 2022 A generative variational model for inverse problems in imaging *SIAM J. Math. Data Sci.* **4** 306–35
[19] Heckel R and Soltanolkotabi M 2020 Compressive sensing with un-trained neural networks: gradient descent finds a smooth approximation *Proc. 37th Int. Conf. Machine Learning, Volume 119 of Proc. Machine Learning Research PMLR* pp 4149–58
[20] Heckel R and Soltanolkotabi M 2020 Denoising and regularization via exploiting the structural bias of convolutional generators *Int. Conf. Learning Representations*
[21] Hofmann B, Kaltenbacher B, Pöschl C and Scherzer O 2007 A convergence rates result for Tikhonov regularization in Banach spaces with non-smooth operators *Inverse Problems* **23** 987–1010
[22] Jain V and Seung S 2009 Natural image denoising with convolutional networks *Advances in Neural Information Processing Systems* vol 21 ed D Koller, D Schuurmans, Y Bengio and L Bottou (Red Hook, NY: Curran Associates, Inc.)
[23] Kaltenbacher B and Klassen A 2018 On convergence and convergence rates for Ivanov and Morozov regularization and application to some parameter identification problems in elliptic PDEs *Inverse Problems* **34** 055008
[24] Kluth T, Bathke C, Jiang M and Maass P 2020 Joint super-resolution image reconstruction and parameter identification in imaging operator: analysis of bilinear operator equations, numerical solution, and application to magnetic particle imaging *Inverse Problems* **36** 124006
[25] Lempitsky V, Vedaldi A and Ulyanov D 2018 Deep image prior *IEEE/CVF Conf. Computer Vision and Pattern Recognition* pp 9446–54
[26] Li H, Schwab J, Antholzer S and Haltmeier M 2020 NETT: solving inverse problems with deep neural networks *Inverse Problems* **36** 065005
[27] Looney C G 1977 Convergence of minimizing sequences *J. Math. Anal. Appl.* **61** 835–40
[28] Papyan V, Romano Y, Sulam J and Elad M 2018 Theoretical foundations of deep learning via sparse representations: a multilayer sparse model and its connection to convolutional neural networks *IEEE Signal Process. Mag.* **35** 72–89
[29] Romano Y, Elad M and Milanfar P 2017 The little engine that could: regularization by denoising (red) *SIAM J. Imaging Sci.* **10** 1804–44
[30] Rudin L I, Osher S and Fatemi E 1992 Nonlinear total variation based noise removal algorithms *Physica* D **60** 259–68
[31] Seidman T I and Vogel C R 1989 Well posedness and convergence of some regularisation methods for non-linear ill posed problems *Inverse Problems* **5** 227–38
[32] Shi Z, Mettes P, Maji S and Snoek C G M 2022 On measuring and controlling the spectral bias of the deep image prior *Int. J. Comput. Vis.* **130** 885–908
[33] Sjoberg J and Overtraining L L 1994 regularization, and searching for minimum with application to neural networks *Int. J. Control* **62** 1391
[34] Sun Y, Zhao H and Scarlett J 2021 On architecture selection for linear inverse problems with untrained neural networks *Entropy* **23** 1481
[35] Tai Y, Yang J and Liu X 2017 Image super-resolution via deep recursive residual network *2017 IEEE Conf. Computer Vision and Pattern Recognition (CVPR)* pp 2790–8
[36] Vasin V 1970 Relationship of several variational methods for the approximate solution of ill-posed problems *Math. Notes Acad. Sci. USSR* **7** 161–5
[37] Vogel C R 1990 A constrained least squares regularization method for nonlinear III-posed problems *SIAM J. Control Optim.* **28** 34–49
[38] Zou H and Hastie T 2005 Regularization and variable selection via the elastic net *J. R. Stat. Soc.* B **67** 301–20

OPEN ACCESS
IOP Publishing
Inverse Problems
Inverse Problems **39** (2023) 125018 (37pp) https://doi.org/10.1088/1361-6420/ad0660

Invertible residual networks in the context of regularization theory for linear inverse problems

Clemens Arndt[1], **Alexander Denker**[1], **Sören Dittmer**[1,2], **Nick Heilenkötter**[1], **Meira Iske**[1], **Tobias Kluth**[1,*], **Peter Maass**[1] and **Judith Nickel**[1]

[1] Center for Industrial Mathematics, University of Bremen, 28359 Bremen, Germany
[2] Cambridge Image Analysis Group, University of Cambridge, Cambridge CB3 0WA, United Kingdom

E-mail: tkluth@math.uni-bremen.de

Received 8 June 2023; revised 26 September 2023
Accepted for publication 24 October 2023
Published 13 November 2023

Abstract

Learned inverse problem solvers exhibit remarkable performance in applications like image reconstruction tasks. These data-driven reconstruction methods often follow a two-step procedure. First, one trains the often neural network-based reconstruction scheme via a dataset. Second, one applies the scheme to new measurements to obtain reconstructions. We follow these steps but parameterize the reconstruction scheme with invertible residual networks (iResNets). We demonstrate that the invertibility enables investigating the influence of the training and architecture choices on the resulting reconstruction scheme. For example, assuming local approximation properties of the network, we show that these schemes become convergent regularizations. In addition, the investigations reveal a formal link to the linear regularization theory of linear inverse problems and provide a nonlinear spectral regularization for particular architecture classes. On the numerical side, we investigate the local approximation property of selected trained architectures and present

* Author to whom any correspondence should be addressed.

a series of experiments on the MNIST dataset that underpin and extend our theoretical findings.

Keywords: learning for inverse problems, invertible residual networks, convergent regularization, learned nonlinear spectral regularization, local approximation property

(Some figures may appear in colour only in the online journal)

1. Introduction

In inverse problems, one aims to recover underlying causes from measurements by reversing the forward measurement process. Naturally, they arise in several fields, e.g. medical imaging, non-destructive testing, and partial differential equations (PDE)-based models. One defines inverse problems via their ill-posed nature, i.e. reversing the measurement process is discontinuous, and obtaining reasonable reconstructions requires a stable reconstruction scheme. Usually, one defines the forward problem by possibly nonlinear forward operator $\mathcal{A} : X \to Y$ mapping between Banach or Hilbert spaces. Linear spectral regularizations for linear problems in Hilbert space settings were already well studied over two decades ago [17]. Also, more general nonlinear regularizations, e.g. sophisticated variational and iterative approaches, were the subject of extensive investigations for linear and nonlinear problems. We refer to the review [9] for a more general overview of this development. More recently, the linear spectral regularization approach was generalized to diagonal frames [15]. Frames provide larger flexibility by allowing advantageous representations of the data, e.g. via a set of images, instead of solely singular functions provided by the operator. Nonlinear spectral regularizations with nonlinear dependence on the measurement data were only considered in a few works, e.g. in the context of conjugate gradient methods [17, corollary 7.4].

During the last decade, these research directions have been accompanied by a highly dynamic development of learning-based methods for inverse problems (see [4] for an earlier review). Many promising methodological directions have arisen, such as end-to-end learned reconstructions [20, 36], learned postprocessing [21, 41], unrolled iteration schemes [1, 18, 19], plug-and-play priors [24, 39, 43], learned penalties in variational approaches [2, 26, 28, 31, 35], (deep) generative models [10, 13], regularization by architecture [3, 14, 42], and several more directions are developed. Besides the pure method development, an increasing number of works investigate theoretical justifications using the framework of regularization theory (see the recent survey [32] and the more detailed consideration in section 1.1).

In the present work, we follow a general supervised learning approach for a reconstruction scheme based on the concept of invertible residual networks [8]. We use this architecture to provide a nonlinear regularization scheme for which we develop the general convergence theory by exploiting a local approximation property of the underlying network. Furthermore, instead of solving the inverse problem directly, we use a training strategy that aims to approximate the forward operator. We also introduce an architecture type acting on the singular function directions, and show that it provides a data-dependent nonlinear spectral regularization which is theoretically analyzed for specific shallow architectures and it is linked to the established linear spectral regularization theory. In addition, we underpin our theoretical findings with several numerical experiments.

The manuscript is structured as follows: section 2 defines the problem setting and the theoretical framework. We then investigate general regularization properties in section 3. In section 4, we discuss specific architectures and the outcome of a training approach aiming for

approximating the forward operator. We then relate it to the classical filter-based regularization theory. Following the theoretical investigation, section 5 presents a series of numerical experiments. We conclude with a discussion and outlook in section 6.

1.1. Related work

Empirically the success of learning-based methods has been established for several applications like various imaging modalities. Theoretically, there is still a gap, e.g. regarding the convergence properties of such reconstruction schemes. Recently an increasing number of works have examined this aspect. We recommend the excellent survey article [32], illustrating regularization properties of some learned methods [32, figure 3].

One early example providing a convergent learned postprocessing approach is the deep null space network [41]. It utilizes established regularization methods and turns them into learned variants. While it replaces the concept of a minimum norm solution, it inherits convergence guarantees. [5] investigates the convergence of regularization methods trained on a finite set of training samples without access to the entire forward operator. Specifically, the authors consider the projections on the subspaces spanned by the dataset. Assumptions on the dataset ensure desired approximation properties for the convergence analysis of the regularization method. The authors then also consider the data-dependent projected problem in a variational Tikhonov-type framework and derive convergence of the regularization by assuming certain approximation properties of the projection being fulfilled for the desired solution.

Also, Tikhonov-type/variational methods and their classical regularization theory were used to obtain learned regularizations with convergence guarantees. In particular, learned convex regularizers [31, 33] and the network Tikhonov (NETT) approach [26, 35] investigated this.

Minimization of Tikhonov-type functionals is often performed by solving an equilibrium equation, e.g. obtained from first-order optimality conditions. This concept is generalized by the authors in the recent work [34] where a learned operator is introduced in the equilibrium equation replacing, for example, the component delivered by the regularizer. The authors formulate suitable assumptions on the learnable operator to guarantee convergence, and they also illustrate that residual networks fulfill the desired properties if the residual part in the network is contractive.

Ebner and Haltmeier [16] recently suggested convergence results for plug-and-play reconstructions that are conceptionally motivated by the Tikhonov theory. They guarantee convergence by assuming the required conditions on their parameterized family of denoising operators.

The deep image prior [42] proposed exploiting a network's training and architecture as a regularization to achieve better image representation. Based on this concept of unsupervised learning from one single measurement, Dittmer *et al* [14] derived an analytic deep prior framework formulated as a bilevel optimization problem. This allowed them to verify regularization properties for particular cases by exploiting relations to the classical filter theory of linear regularizations. More recently, [3] investigated the equivalence between Ivanov regularization and analytic deep prior. They investigated the inclusion of early stopping and proved regularization properties [3].

An earlier work [12] proposed a data-driven approach to a learned linear spectral regularization; more recently, [7, 22] considered convergence aspects. In [22], the authors learn a scalar for each singular function direction in a filter-based reconstruction scheme, i.e. a linear regularization scheme. Due to the assumed properties of noise and data distributions, their training outcome is equivalent to a standard Tikhonov regularization with a data- and noise-dependent linear operator included in the penalty term, which is diagonal with respect to the system of

singular functions. The authors further verify convergence results concerning the data- and noise-dependent regularization parameters.

2. Problem setting—theoretical framework

We consider linear inverse problems based on the operator equation

$$\mathcal{A}x = y, \tag{2.1}$$

where $\mathcal{A}\colon X \to Y$ is a linear and bounded operator between Hilbert spaces X and Y. For simplification, $\|\mathcal{A}\| = 1$ is assumed, which can be easily obtained by a scaling of the operator. We aim to recover the unknown ground truth $x^\dagger$ as well as possible by only having access to a noisy observation $y^\delta \in Y$ such that $\|y^\delta - \mathcal{A}x^\dagger\| \leqslant \delta$, where $\delta > 0$ is the noise level.

Instead of solving (2.1) directly, we define $A = \mathcal{A}^*\mathcal{A}$ and $z^\delta = \mathcal{A}^* y^\delta$ to get the operator equation

$$Ax = \mathcal{A}^*\mathcal{A}x = \mathcal{A}^* y = z \tag{2.2}$$

which only acts on X, i.e. we consider the normal equation with respect to $\mathcal{A}$. We propose a two-step data-based approach for solving (2.2), resp. (2.1), which we refer to as the *iResNet reconstruction approach*:

(I) Supervised training of an invertible neural network $\varphi_\theta : X \to X$ to approximate A.
(II) Using φ_θ^{-1} (or $\varphi_\theta^{-1} \circ \mathcal{A}^*$, respectively) to solve the inverse problem.

In general, the supervised training covers both cases, either training on noise-free data tuples $(x^{(i)}, Ax^{(i)})$ or on noisy tuples $(x^{(i)}, Ax^{(i)} + \eta^{(i)})$ with noise $\eta^{(i)}$ for a given dataset $\{x^{(i)}\}_{i\in\{1,\dots,N\}} \subset X$. To guarantee the invertibility, we use a residual network of the form

$$\varphi_\theta(x) = x - f_\theta(x) \tag{2.3}$$

and restrict f_θ to be Lipschitz continuous with $\mathrm{Lip}(f_\theta) \leqslant L < 1$. We refer to this architecture, which is illustrated in figure 1, as an invertible residual network (iResNet) [8]. Note that [8] considers concatenations of several of these invertible blocks while we focus on a single residual block.

The essential properties of the iResNets (like how it can be inverted) are summarized in the following lemma.

Lemma 2.1 (general properties of iResNets). *Let $\varphi_\theta, f_\theta : X \to X$, $\varphi_\theta(x) = x - f_\theta(x)$, where $\mathrm{Lip}(f_\theta) \leqslant L < 1$. Then it holds:*

(i) $$\mathrm{Lip}(\varphi_\theta) \leqslant L + 1 \tag{2.4}$$

(ii) $$\mathrm{Lip}\left(\varphi_\theta^{-1}\right) \leqslant \frac{1}{1-L} \tag{2.5}$$

(iii) *For given $z \in X$, the unique $x = \varphi_\theta^{-1}(z)$ is given by $x = \lim_{k\to\infty} x^k$ where*

$$x^{k+1} = f_\theta\left(x^k\right) + z \tag{2.6}$$

for arbitrary and given $x^0 \in X$.

Proof. The idea of the proof is taken from [8, section 2]. The first assertion follows directly from

$$\mathrm{Lip}(\varphi_\theta) \leqslant \mathrm{Lip}(Id) + \mathrm{Lip}(f_\theta). \tag{2.7}$$

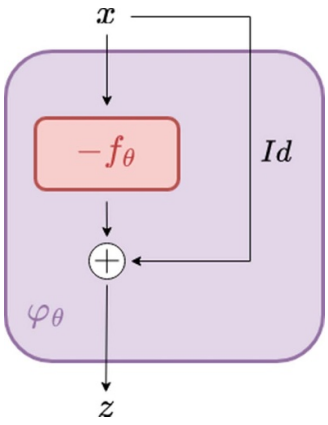


Figure 1. Illustration of the residual network architecture of φ_θ with residual function f_θ and skip connection.

Assertion (iii) follows from the fixed point theorem of Banach. Since f_θ is contractive, the iteration convergences to a unique fixed point. This fixed point x fulfills

$$z = x + f_\theta(x) = \varphi(x). \tag{2.8}$$

Finally, assertion (*ii*) follows from

$$\begin{aligned}\|\varphi_\theta(x^1) - \varphi_\theta(x^2)\| &= \|x^1 - x^2 - (f_\theta(x^1) - f_\theta(x^2))\| \\ &\geqslant \|x^1 - x^2\| - \|f_\theta(x^1) - f_\theta(x^2)\| \geqslant (1-L)\|x^1 - x^2\|\end{aligned} \tag{2.9}$$

by replacing $\varphi_\theta(x^i)$ with z^i and x^i with $\varphi_\theta^{-1}(z^i)$ for $i = 1, 2$. □

Besides, we assume the existence of a singular value decomposition (SVD) $(u_j, v_j, \sigma_j)_{j\in\mathbb{N}}$ of the operator $\mathcal{A}$, i.e. $u_j \in Y$, $v_j \in X$, $\sigma_j > 0$ such that

$$\mathcal{A}x = \sum_{j=1}^{\infty} \sigma_j \langle x, v_j\rangle u_j \tag{2.10}$$

holds[3]. All compact operators guarantee this. With respect to A, the vectors v_j are eigenvectors, and σ_j^2 are the corresponding eigenvalues. The system $\{v_j\}_j \in \mathbb{N}$ builds an orthonormal basis of $\overline{\mathcal{R}(A)} = \overline{\mathcal{R}(\mathcal{A}^*)} = \mathcal{N}(\mathcal{A})^\perp = \mathcal{N}(A)^\perp$. Due to the assumption $\|\mathcal{A}\| = 1$, it holds $\sigma_j \in (0, 1]$.

3. Regularization properties of the iResNet reconstruction

In this section, we discuss regularization properties in terms of well-definedness, stability, and convergence of the general reconstruction approach defined by the family of operators $T_L = \varphi_{\theta,L}^{-1} \circ \mathcal{A}^*$. Note that $L \in [0, 1)$ in the constraint of the Lipschitz constant of the residual function undertakes the role of the regularization parameter by $L \to 1$, as will be made apparent in this section.

[3] In case of $\dim(\mathcal{R}(\mathcal{A})) < \infty$, the number of singular values is finite.

3.1. General architecture and local approximation property

To highlight the essential dependence of the trained network on L, we write $\varphi_{\theta,L}$ and $f_{\theta,L}$ in the following. Please note that the set of possible network parameters and the underlying architecture might also depend on L. We thus consider a network parameter space $\Theta(L)$ depending on L.

Lemma 3.1 (well-definedness and stability). *For $L \in [0,1)$ and $\theta \in \Theta(L)$, let $f_{\theta,L} : X \to X$ be such that $\mathrm{Lip}(f_{\theta,L}) \leqslant L$ and $\varphi_{\theta,L}(x) = x - f_{\theta,L}(x)$. Then, the reconstruction scheme $T_L = \varphi_{\theta,L}^{-1} \circ \mathcal{A}^* : Y \to X$ is well-defined and Lipschitz continuous.*

Proof. Well-definedness follows immediately from lemma 2.1 (iii) and Lipschitz continuity from (ii). □

While well-definedness and continuity (stability) immediately follow from the construction of the reconstruction method T_L, the convergence property requires a specific approximation property, which we must guarantee during training. We do so by introducing an index function that generalizes the concept of convergence rates (see, e.g. [40]).

Definition 3.1. An index function is a mapping $\psi : \mathbb{R}_{\geqslant 0} \to \mathbb{R}_{\geqslant 0}$, which is continuous, strictly increasing and it holds $\psi(0) = 0$.

Now, we can formulate a convergence result for the reconstruction $T_L(y^\delta)$ for $\delta \to 0$ and a suitable *a priori* parameter choice $L(\delta)$.

Theorem 3.1 (convergence - local approximation property). *Let $x^\dagger \in X$ be a solution of the problem $\mathcal{A}x = y$ for $y \in R(\mathcal{A})$ and $y^\delta \in Y$ fulfill $\|y^\delta - y\| \leqslant \delta$. Furthermore, let the network parameters $\theta(L) \in \Theta(L)$ for $L \in [0,1)$ be obtained in a way that the local approximation property*

$$\|\mathcal{A}^*\mathcal{A}x^\dagger - \varphi_{\theta(L),L}\left(x^\dagger\right)\| = \mathcal{O}\left((1-L)\,\psi\left(1-L\right)\right) \quad (\text{as } L \to 1) \tag{3.1}$$

holds for some index function ψ.

If $L : (0,\infty) \to [0,1)$ fulfills

$$L(\delta) \to 1 \quad \wedge \quad \frac{\delta}{1-L(\delta)} \to 0 \qquad \text{for } \delta \to 0, \tag{3.2}$$

then for $x^\delta_{L(\delta)} := T_{L(\delta)}(y^\delta) = (\varphi^{-1}_{\theta(L(\delta)),L(\delta)} \circ \mathcal{A}^*)(y^\delta)$ it holds

$$\|x^\delta_{L(\delta)} - x^\dagger\| \to 0 \qquad \text{for } \delta \to 0. \tag{3.3}$$

Proof. For improved readability we write $\varphi_\delta := \varphi_{\theta(L(\delta)),L(\delta)}$ and $f_\delta := f_{\theta(L(\delta)),L(\delta)}$ in the remainder of the proof. Using lemma 2.1, it holds

$$\begin{aligned}
\|x^\delta_{L(\delta)} - x^\dagger\| &\leqslant \|\varphi_\delta^{-1}\left(\mathcal{A}^*y^\delta\right) - \varphi_\delta^{-1}\left(\mathcal{A}^*y\right)\| + \|\varphi_\delta^{-1}\left(\mathcal{A}^*y\right) - x^\dagger\| \\
&\leqslant \frac{\|\mathcal{A}^*\|}{1-L(\delta)}\|y^\delta - y\| + \|\varphi_\delta^{-1}\left(\mathcal{A}^*\mathcal{A}x^\dagger\right) - x^\dagger\| \\
&\leqslant \frac{\delta}{1-L(\delta)} + \|\varphi_\delta^{-1}\left(\mathcal{A}^*\mathcal{A}x^\dagger\right) - \varphi_\delta^{-1}\left(\varphi_\delta\left(x^\dagger\right)\right)\| \\
&\leqslant \frac{\delta}{1-L(\delta)} + \frac{1}{1-L(\delta)}\|\mathcal{A}^*\mathcal{A}x^\dagger - \varphi_\delta\left(x^\dagger\right)\|.
\end{aligned} \tag{3.4}$$

The assertion follows due to the assumptions (3.1) and (3.2) as $\lim_{L\to 1}\psi(1-L) = 0$. □

Remark 3.1. We can obtain a convergence rate of $\|x^{\delta}_{L(\delta)} - x^{\dagger}\| = \mathcal{O}\left(\delta^{\varepsilon/(1+\varepsilon)}\right)$ for $\varepsilon > 0$, if (3.1) is fulfilled with the index function $\psi(\lambda) = \lambda^{\varepsilon}$, i.e.

$$\|\mathcal{A}^*\mathcal{A}x^{\dagger} - \varphi_{\theta(L),L}\left(x^{\dagger}\right)\| = \mathcal{O}\left((1-L)^{1+\epsilon}\right) \tag{3.5}$$

and by choosing $1 - L(\delta) \sim \delta^{1/(1+\varepsilon)}$.

Remark 3.2. Note that the local approximation property (3.1) is a slightly stronger assumption, which is derived by exploiting the properties of the iResNet to remove the explicit dependency on the inverse within the constraint. A weaker but sufficient condition (which is implied by (3.1)) is

$$\|\varphi^{-1}_{\theta(L),L}\left(\mathcal{A}^*\mathcal{A}x^{\dagger}\right) - x^{\dagger}\| \to 0 \quad \text{as } L \to 1. \tag{3.6}$$

The local approximation property (3.1) provides a relation between the approximation capabilities of the network architecture and the regularization properties of the iResNet reconstruction scheme, which also contains a certain kind of source condition. Note that the previous statement differs from common convergence results, which are stated not locally but for a broader range of solutions, i.e. for any $y \in \mathcal{R}(\mathcal{A})$. Here, the locality is strongly linked to the approximation capabilities of the underlying network, which can imply certain limitations. Thus, of particular interest for the convergence properties is the *approximation property set*

$$S = \{x \in X \,|\, \exists \text{ index function } \psi : x \text{ fulfills } (3.1)\}. \tag{3.7}$$

There are no specific assumptions on the design of the training of the network φ_{θ} made by (3.1). But since both architecture choice and training procedure are crucial for the resulting network parameters $\theta(L)$, they also have a strong impact on the local approximation property and the structure of S. This is discussed in more detail in the context of the specific shallow architectures in section 4. In addition, we report numerical experiments regarding the local approximation property in section 5.

A simple example illustrating the conditions under which a linear network satisfies the local approximation property is given in the following remark.

Remark 3.3 (local approximation property for linear networks). Assume that the network $\varphi_{\theta,L}$ is linear and the prediction error is bounded by $\|\varphi_{\theta(L(\delta)),L(\delta)}(x^{(i)}) - \mathcal{A}^*y^{\delta,(i)}\| \leqslant \zeta(\delta)$ with $\zeta(\delta) \in \mathbb{R}$ on a dataset $(x^{(i)}, y^{\delta,(i)})_{i=1,\dots,N}$ with $\|\mathcal{A}x^{(i)} - y^{\delta,(i)}\| \leqslant \delta$. Then, for $x^{\dagger}$ given by $x^{\dagger} = \sum_{i=1}^{N} t_i x^{(i)}$ with $t_i \in \mathbb{R}$ we have

$$\begin{aligned}
\left\|\mathcal{A}^*\mathcal{A}x^{\dagger} - \varphi_{\theta(L(\delta)),L(\delta)}\left(x^{\dagger}\right)\right\| &= \left\|\mathcal{A}^*\mathcal{A}\sum_{i=1}^{N} t_i x^{(i)} - \sum_{i=1}^{N} t_i \varphi_{\theta(L(\delta)),L(\delta)}x^{(i)}\right\| \\
&\leqslant \sum_{i=1}^{N} |t_i| \, \|\mathcal{A}^*\mathcal{A}x^{(i)} - \varphi_{\theta(L(\delta)),L(\delta)}x^{(i)}\| \\
&\leqslant \sum_{i=1}^{N} |t_i| \left(\|\mathcal{A}^*\mathcal{A}x^{(i)} - \mathcal{A}^*y^{\delta,(i)}\| + \|\mathcal{A}^*y^{\delta,(i)} - \varphi_{\theta(L(\delta)),L(\delta)}x^{(i)}\|\right) \\
&\leqslant \sum_{i=1}^{N} |t_i| \, (\delta + \zeta(\delta)).
\end{aligned} \tag{3.8}$$

As as result, theorem 3.1 implies convergence of $x^{\delta}_{L(\delta)}$ to $x^{\dagger}$ if

$$\frac{\delta + \zeta(\delta)}{1 - L(\delta)} \to 0 \quad \text{for } \delta \to 0 \tag{3.9}$$

and if $L(\delta)$ satisfies the conditions of (3.2). This example shows that in the case that the reconstruction error can be bounded by $\zeta(\delta)$ satisfying (3.9), the convergence directly translates to the linear span of the training data. Consequently, for test data similar to the training data, a simple criterion, that can be numerically validated to ensure convergence to the ground truth data, is obtained.

Similar to a source condition, (3.1) also has an influence on the type of solution $x^\dagger$ one approximates in the limit $\delta \to 0$ if the operator $\mathcal{A}$ has a non-trivial kernel. For example, if $\mathcal{R}(f_{\theta(L),L}) = \mathcal{N}(\mathcal{A})^\perp$, the local approximation property enforces that $P_{\mathcal{N}(\mathcal{A})}x^\dagger = 0$, i.e. together with the approximation of $\mathcal{A}^*\mathcal{A}$ one would consider the minimum norm solution. More generally, we can formulate the following property of S.

Lemma 3.2. *Let $x_1^\dagger, x_2^\dagger \in S$ both be solutions of $\mathcal{A}x = y$ for one $y \in \mathcal{R}(\mathcal{A})$. Then it holds $x_1^\dagger = x_2^\dagger$.*

Proof. For abbreviation, we write f_L and φ_L instead of $f_{\theta(L),L}$ and $\varphi_{\theta(L),L}$. Using $\mathcal{A}x_1^\dagger = \mathcal{A}x_2^\dagger$, it holds

$$\begin{aligned}
&\|x_1^\dagger - x_2^\dagger\| \\
&\quad \leqslant \|x_1^\dagger - \mathcal{A}^*\mathcal{A}x_1^\dagger - f_L\left(x_1^\dagger\right)\| + \|\mathcal{A}^*\mathcal{A}x_1^\dagger + f_L\left(x_1^\dagger\right) - \mathcal{A}^*\mathcal{A}x_2^\dagger - f_L\left(x_2^\dagger\right)\| \\
&\qquad + \|\mathcal{A}^*\mathcal{A}x_2^\dagger + f_L\left(x_2^\dagger\right) - x_2^\dagger\| \\
&\quad = \|\varphi_L\left(x_1^\dagger\right) - \mathcal{A}^*\mathcal{A}x_1^\dagger\| + \|f_L\left(x_1^\dagger\right) - f_L\left(x_2^\dagger\right)\| + \|\mathcal{A}^*\mathcal{A}x_2^\dagger - \varphi_L\left(x_2^\dagger\right)\| \\
&\quad \leqslant \|\mathcal{A}^*\mathcal{A}x_1^\dagger - \varphi_L\left(x_1^\dagger\right)\| + L\|x_1^\dagger - x_2^\dagger\| + \|\mathcal{A}^*\mathcal{A}x_2^\dagger - \varphi_L\left(x_2^\dagger\right)\|. \qquad (3.10)
\end{aligned}$$

Subtracting $L\|x_1^\dagger - x_2^\dagger\|$, it follows

$$(1-L)\,\|x_1^\dagger - x_2^\dagger\| \leqslant \|\mathcal{A}^*\mathcal{A}x_1^\dagger - \varphi_L\left(x_1^\dagger\right)\| + \|\mathcal{A}^*\mathcal{A}x_2^\dagger - \varphi_L\left(x_2^\dagger\right)\|. \qquad (3.11)$$

Since $x_1^\dagger, x_2^\dagger$ both fulfill the local approximation property (3.1), there exists an index function ψ such that $(1-L)\|x_1^\dagger - x_2^\dagger\| = \mathcal{O}((1-L)\psi(1-L))$ must hold. This implies $\|x_1^\dagger - x_2^\dagger\| = \mathcal{O}(\psi(1-L))$, which is only possible for $x_1^\dagger = x_2^\dagger$. □

3.2. Diagonal architecture

In order to investigate relations to established and well-studied spectral regularization methods, we continue with the introduction of a particular network design, which is exploited in the remainder of the manuscript. The idea of spectral regularization is to decompose the data using the SVD $(u_j, v_j, \sigma_j)_{j\in\mathbb{N}}$ of $\mathcal{A}$ and apply filter functions to the singular values. Analogously, we consider a network architecture consisting of subnetworks $f_{\theta,j} : \mathbb{R} \to \mathbb{R}, j \in \mathbb{N}$, i.e.

$$f_\theta(x) = \sum_{j\in\mathbb{N}} f_{\theta,j}\left(\langle x, v_j\rangle\right) v_j, \qquad (3.12)$$

where all the components $\langle x, v_j\rangle$ of the data are processed separately. We refer to this architecture as *diagonal* architecture, which is also illustrated in figure 2. The naming is motivated by the similarity of the structure of f_θ to a matrix multiplication with a diagonal matrix w.r.t the basis $\{v_j\}_{j\in\mathbb{N}}$. This architecture choice is, of course, less expressive than general architectures,

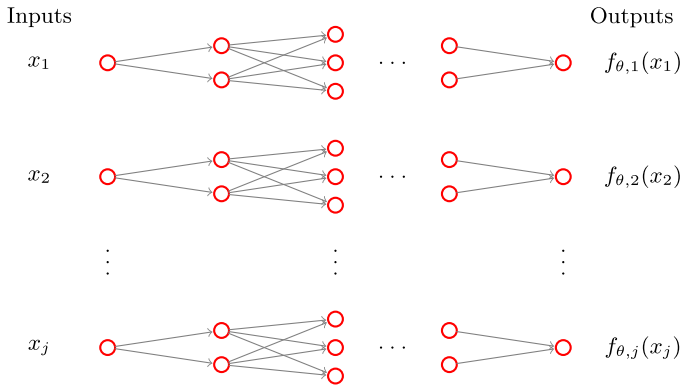


Figure 2. *Diagonal* structure of the residual function including subnetworks $f_{\theta,j}:\mathbb{R}\to\mathbb{R}$ acting on $x_j=\langle x,v_j\rangle$.

but it has the useful benefit that this way, the iResNet can be analyzed like a filter-based regularization scheme. Besides, it enables us to simplify the infinite-dimensional inverse problem $\mathcal{A}x=y$ to 1D subproblems of the form $\sigma_j\langle x,v_j\rangle=\langle y,u_j\rangle$, which will be exploited in section 4.

In case of a diagonal architecture, the Lipschitz constraint $\mathrm{Lip}(f_\theta)\leqslant L$ is fulfilled if $\mathrm{Lip}(f_{\theta,j})\leqslant L$ holds for all $j\in\mathbb{N}$, i.e. for any $x,y\in X$ it holds

$$\begin{aligned}\|f_\theta(x)-f_\theta(y)\|^2&=\sum_{j\in\mathbb{N}}|f_{\theta,j}(\langle x,v_j\rangle)-f_{\theta,j}(\langle y,v_j\rangle)|^2\\&\leqslant L^2\sum_{j\in\mathbb{N}}|\langle x-y,v_j\rangle|^2\leqslant L^2\|x-y\|^2.\end{aligned}\tag{3.13}$$

The local approximation property (3.1), in this case for $x^\dagger\in\mathcal{N}(\mathcal{A})^\perp$, implies

$$|f_{\theta(L),L,j}\left(x_j^\dagger\right)-\left(1-\sigma_j^2\right)x_j^\dagger|=\mathcal{O}\left((1-L)\,\psi\left(1-L\right)\right)\tag{3.14}$$

for any $j\in\mathbb{N}$, where $x_j^\dagger=\langle x^\dagger,v_j\rangle$. Note that for infinite-dimensional operators, the constants in $\mathcal{O}((1-L)\psi(1-L))$ must be an ℓ^2 sequence with respect to j to obtain the implication in the opposite direction and thus equivalence.

The particular aim of using the diagonal architecture is now to find a filter function that defines a spectral regularization scheme that is equivalent to $T_L=\varphi_\theta^{-1}\circ\mathcal{A}^*$. Since filter-based regularization methods are usually linear and we want to also allow for nonlinear architectures, our filter function r_L must be data-dependent. Therefore, we define $r_L\colon\mathbb{R}_+\times\mathbb{R}\to\mathbb{R}$ such that for $z\in\mathcal{R}(\mathcal{A}^*)$

$$\varphi_\theta^{-1}(z)=\sum_{j\in\mathbb{N}}r_L\left(\sigma_j^2,\langle z,v_j\rangle\right)\langle z,v_j\rangle v_j,\tag{3.15}$$

or for $y\in Y$

$$T_L(y)=\varphi_\theta^{-1}(\mathcal{A}^*y)=\sum_{j\in\mathbb{N}}r_L\left(\sigma_j^2,\sigma_j\langle y,u_j\rangle\right)\sigma_j\langle y,u_j\rangle v_j\tag{3.16}$$

holds. The first argument of r_L is the singular value (as usual), and the data-dependency comes via the second argument. This can be seen as a nonlinear extension to the established filter

theory of linear regularization methods [27, 38], where r_L would depend on σ_j^2 only. The subnetworks thus play an important role in defining the filter functions via

$$(Id - f_{\theta,L,j})^{-1}(s) = r_L\left(\sigma_j^2, s\right) s. \tag{3.17}$$

For this to be well-defined, we need to assume that all singular values $\sigma_j > 0$ have a multiplicity of one. In case of $\sigma_j = \sigma_k$ for $j \neq k$ (multiplicity greater than one) one must ensure that $f_{\theta,L,j}$ and $f_{\theta,L,k}$ also coincide. In practice, this could be realized by sharing the weights of these networks.

Remark 3.4. Some authors define filter functions in terms of the original problem with respect to the generalized inverse $\mathcal{A}^\dagger$ such that

$$\varphi_\theta^{-1}(\mathcal{A}^* y) = \sum_{j\in\mathbb{N}} F_L(\sigma_j, \langle y, u_j\rangle)\frac{1}{\sigma_j}\langle y, u_j\rangle v_j, \qquad \text{with } F_L(\sigma, s) = \sigma^2 r_L\left(\sigma^2, \sigma s\right). \tag{3.18}$$

Note that the choice of the representation, either in terms of r_L or in terms of F_L, depends on personal preference (compare, for example, [27, 38]). In the following, we represent the filter function in terms of r_L.

One simple nonlinear dependence on s will be taken into account explicitly in the following, as it becomes relevant in later investigations. In analogy to the bias in neural network architectures, we consider a linear filter framework with bias, i.e.

$$T_L(y) = \hat{b}_L + \sum_{j\in\mathbb{N}} \hat{r}_L\left(\sigma_j^2\right)\sigma_j\langle y, u_j\rangle v_j \tag{3.19}$$

where $\hat{b}_L \in X$ is a bias term and $\hat{r}_L$ is a classical filter function. With additional simple assumptions on $\hat{b}_L$, this becomes a regularization method. For the sake of completeness, we provide the following result.

Lemma 3.3 (Filter-based spectral regularization with bias). *Let $\hat{r}_L : [0, \|\mathcal{A}\|^2] \to \mathbb{R}$ be a piecewise continuous function and let $b_L \in X$ for $L \in [0,1)$. Furthermore, let*

(i) $\lim_{L\to 1} \hat{r}_L(\sigma_j^2) = \frac{1}{\sigma_j^2}$ *for any* σ_j, $j \in \mathbb{N}$,
(ii) $\exists 0 < C < \infty :\ \sigma_j^2 |\hat{r}_L(\sigma_j^2)| \leqslant C$ *for any* σ_j, $j \in \mathbb{N}$ *and* $L \in [0,1)$,
(iii) $\hat{b}_L \in X$ *for* $L \in [0,1)$ *and* $\lim_{L\to 1} \|\hat{b}_L\| = 0$

hold. Let $x^\dagger \in \mathcal{N}(\mathcal{A})^\perp$ be a solution of the problem $\mathcal{A}x = y$ for $y \in R(\mathcal{A})$, the operator $T_L : Y \to X$ be given by (3.19) and $y^\delta \in Y$ with $\|y - y^\delta\| \leqslant \delta$. In addition, let $L : (0,\infty) \to [0,1)$ be such that

$$L(\delta) \to 1 \quad \wedge \quad \delta\sqrt{\sup\left\{|\hat{r}_{L(\delta)}(\sigma_i)| \,\middle|\, i \in \mathbb{N}\right\}} \to 0 \tag{3.20}$$

as $\delta \to 0$. Then, T_L is a regularization method and for $x_{L(\delta)}^\delta := T_{L(\delta)}(y^\delta)$ it holds $\lim_{\delta\to 0} \|x_{L(\delta)}^\delta - x^\dagger\| = 0$.

Proof. By defining $\tilde{x}^{\delta}_{L(\delta)} := T_{L(\delta)}(y^{\delta}) - \hat{b}_{L(\delta)}$ (removing the bias), we get an ordinary linear regularization method (see, e.g. [38, corollary 3.3.4]). It holds

$$\|x^{\delta}_{L(\delta)} - x^{\dagger}\| \leqslant \|x^{\delta}_{L(\delta)} - \tilde{x}^{\delta}_{L(\delta)}\| + \|\tilde{x}^{\delta}_{L(\delta)} - x^{\dagger}\| = \|\hat{b}_{L(\delta)}\| + \|\tilde{x}^{\delta}_{L(\delta)} - x^{\dagger}\|. \quad (3.21)$$

Since $\|\hat{b}_{L(\delta)}\| \to 0$ by assumption and $\tilde{x}^{\delta}_{L(\delta)} \to x^{\dagger}$ by standard theory, T_L is a regularization method. □

4. Approximation training of specialized architectures

Besides the architecture choice and the available data, the loss function for network training is an important ingredient. Having regularization properties in mind, a natural choice for training the iResNet is to approximate the forward operator $A\colon X \to X$, $A = \mathcal{A}^*\mathcal{A}$, on a given training dataset $\{x^{(i)}\}_{i\in\{1,\dots,N\}} \subset X$. This is also strongly motivated by the structure of the local approximation property (3.1) to obtain convergence guarantees. The training of $\varphi_\theta = Id - f_\theta$ then consists of solving

$$\min_{\theta\in\Theta} l(\varphi_\theta, A) = \min_{\theta\in\Theta} \frac{1}{N}\sum_i \|\varphi_\theta\left(x^{(i)}\right) - Ax^{(i)}\|^2 \quad \text{s.t. Lip}\,(f_\theta) \leqslant L \qquad (4.1)$$

with $L < 1$ as a hyperparameter, which we refer to as *approximation training*. Here, we restrict ourselves to the noise-free case as the outcome of the approximation training is independent for $N \to \infty$ (assuming stochastic independence between data and noise).

In the following, we analyze to which extent φ_θ^{-1} acts as a regularized inverse of A for different simple diagonal iResNet architectures trained according to (4.1). We compute the particular network parameters θ and derive the corresponding data-dependent filter function $r_L\colon \mathbb{R}\times\mathbb{R} \to \mathbb{R}$ (see section 3.2). Additionally, we check whether the local approximation property (3.1) is fulfilled. For this purpose, the assumptions on the architecture as well as on the dataset used for the training must be specified.

4.1. One-parameter-network—Tikhonov

We begin with a very simple linear network, which only has one single scalar learnable parameter. We show that this architecture is equivalent to Tikhonov regularization if trained appropriately.

Lemma 4.1. *Let $(v_j, \sigma_j^2)_j$ be the eigenvectors and eigenvalues of A and let $\varphi_\theta = Id - f_\theta$ be an iResNet which solves (4.1) with*

(i) *$f_\theta = k(Id - A)$, where $\theta = k$ and $\Theta = \mathbb{R}$ (**architecture** assumption),*
(ii) *the training dataset $\{x^{(i)}\}_{i\in\{1,\dots,N\}}$ contains at least one $x^{(i)}$ s.t. $Ax^{(i)} \neq x^{(i)}$ (**dataset** assumption).*

Then, the solution of (4.1) is $k = L$ and (3.15) holds with

$$r_L\left(\sigma^2, s\right) = \frac{1}{L}\cdot\frac{1}{\alpha+\sigma^2}, \qquad \alpha = \frac{1-L}{L}, \qquad (4.2)$$

for any $s \in \mathbb{R}$.

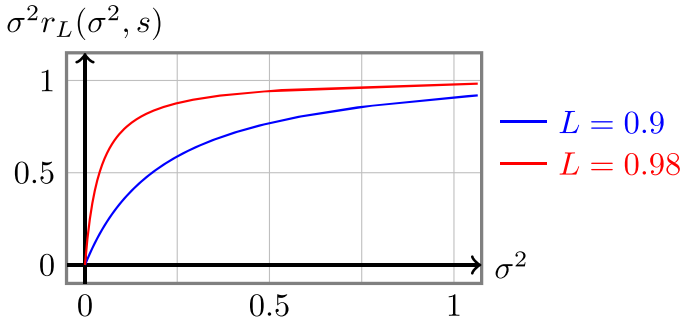


Figure 3. The plot depicts graphs of $\sigma^2 r_L(\sigma^2, s)$ as given in (4.2) to visualize the effect of the regularization, which is also known as standard Tikhonov regularization.

The proof of the lemma can be found in appendix A.1. The derived filter function corresponds to the Tikhonov method (with reference element)

$$\min_{x\in X} \|\mathcal{A}x - y^\delta\|^2 + \alpha\|x - \mathcal{A}^* y^\delta\|^2 \tag{4.3}$$

as one can see by considering the first-order optimality condition, which implies

$$x = \sum_j \frac{1+\alpha}{\alpha+\sigma_j^2}\sigma_j \langle y^\delta, u_j\rangle v_j \tag{4.4}$$

and using the fact that $\frac{1}{L} = 1+\alpha$. It is illustrated in figure 3. We plot $\sigma^2 r_L(\sigma^2, s)$ since the multiplication of the filter function with σ^2 corresponds to the concatenation $\varphi_\theta^{-1} \circ A$ which emphasizes the regularizing effect.

Remark 4.1. This particular choice of the residual function f_θ is one example where the local approximation property is not fulfilled for any $x^\dagger \in X$ except for eigenvectors corresponding to $\sigma_1^2 = 1$, i.e. $S = \operatorname{span}\{v_1\}$. But the linear nature and the specific spectral representation of the residual network allow for the verification of the weaker constraint in (3.6) being highlighted in remark 3.2 and being sufficient for convergence. For the residual function $f_{\theta(L),L} = L(Id - A)$ and $x^\dagger \in \mathcal{N}(A)^\perp$ we obtain

$$\begin{aligned}\|\varphi_{\theta(L),L}^{-1}\left(Ax^\dagger\right) - x^\dagger\|^2 &= \|\left(\left(Id - L\left(Id - A\right)\right)^{-1} A - Id\right) x^\dagger\|^2 \\ &= \sum_{j\in\mathbb{N}} \underbrace{\left(\frac{\sigma_j^2}{1 - L\left(1-\sigma_j^2\right)} - 1\right)^2}_{=\sigma_j^2 r_L\left(\sigma_j^2\right)} |\langle x^\dagger, v_j\rangle|^2,\end{aligned} \tag{4.5}$$

which converges to zero for $L \to 1$.

Alternatively, convergence can be verified by standard arguments following, for example, the line of reasoning in the proof of [27, theorem 3.3.3], where properties of the filter function $F_L(\sigma_j) = \sigma_j^2 r_L(\sigma_j^2)$ are exploited. Since F_L defines the filter function for Tikhonov regularization and is therefore known to fulfill the desired properties, we have convergence for arbitrary $x^\dagger \in \mathcal{N}(A)^\perp$.

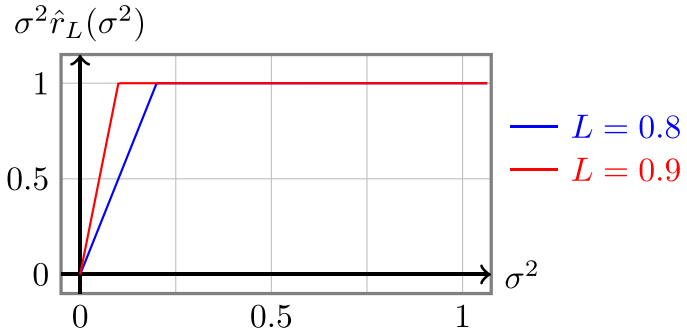


Figure 4. The plot depicts graphs of $\sigma^2\hat{r}_L(\sigma^2)$ as given in (4.8) to visualize the effect of the regularization, which is similar to the known soft TSVD regularization.

4.2. Affine linear network—squared soft TSVD

We continue with a slightly more general affine linear architecture $f_\theta(x) = Wx + b$. It can be observed that the linear operator $W : X \to X$ only depends on A and L while the bias vector $b \in X$ is data dependent.

Lemma 4.2. *Let $(v_j, \sigma_j^2)_j$ be the eigenvectors and eigenvalues of A and $\varphi_\theta = Id - f_\theta$ be an iResNet which solves (4.1) where*

(i) *$f_\theta(x) = Wx + b$, with $\theta = (W, b)$ and*

$$\Theta = \left\{ (W,b) \in L(X) \times X \,\middle|\, \exists (w_j)_{j\in\mathbb{N}}, (b_j)_{j\in\mathbb{N}} : \; W = \sum_{j\in\mathbb{N}} w_j \langle \cdot, v_j \rangle v_j, \, b = \sum_{j\in\mathbb{N}} b_j v_j \right\}, \tag{4.6}$$

(ii) *the training dataset $\{x^{(i)}\}_{i\in\{1,\dots,N\}}$ and the mean values $\mu_j = \frac{1}{N}\sum_{i=1}^N \langle x^{(i)}, v_j \rangle$ fulfill*

$$\forall j \in \mathbb{N} \colon \; \exists i \in \{1,\dots,N\} : \quad \langle x^{(i)}, v_j \rangle \neq \mu_j. \tag{4.7}$$

Then, the solution of the training problem (4.1) is $w_j = \min\{1 - \sigma_j^2, L\}$, $b_j = \max\{0, 1 - L - \sigma_j^2\}\mu_j$ and $\varphi_\theta^{-1} \circ \mathcal{A}^*$ is equivalent to T_L in (3.19) with

$$\hat{r}_L\left(\sigma^2\right) = \frac{1}{\max\{\sigma^2, 1-L\}}, \qquad \hat{b}_L = \frac{1}{1-L}\sum_{j\in\mathbb{N}} b_j v_j = \sum_{\sigma_j^2 < 1-L} \frac{1-L-\sigma_j^2}{1-L}\mu_j v_j. \tag{4.8}$$

The proof of the lemma can be found in appendix A.2. The filter function is illustrated in figure 4.

Remark 4.2. Note that a similar filter function has been found in the context of an analytic deep prior approach for regularization [14]. The authors derived the filter function $F_L(\sigma) = \sigma^2\hat{r}_L(\sigma^2)$, which is linear for small σ, and named it *soft TSVD* (soft truncated SVD). In contrast, the filter function in (4.8) is a polynomial of degree two for small σ, which we refer to as *squared soft TSVD*.

Due to the affine linear nature of the found reconstruction scheme, regularization properties can immediately be derived by verifying certain properties of the filter function $\hat{r}_L$ and the bias $\hat{b}_L$ using lemma 3.3. This is summarized in the following corollary.

Corollary 4.1. *The filter based method (3.19) with $\hat{r}_L$ and $\hat{b}_L$ given by (4.8) fulfills the properties (i)–(iii) in lemma 3.3, i.e. in the setting of lemma 3.3 it is a convergent regularization.*

The previous result provides a link to the classical theory for linear regularization operators. Besides the well-known classical theory, we further want to investigate the local approximation property (3.1) to verify whether the convergence property can also be obtained using theorem 3.1. As a first step, we derive a technical source condition for $x^\dagger$, which implies the local approximation property.

Lemma 4.3. *Let the network φ_θ and the setting be that of lemma 4.2. Assume $x^\dagger \in \mathcal{N}(A)^\perp$ and let ψ be an index function such that*

$$\exists \bar{\beta} \in (0,1] : \forall \beta \in (0,\bar{\beta}) : \sum_{j:\ \sigma_j^2 \leqslant \beta} \langle x^\dagger, v_j\rangle^2 = \mathcal{O}(\psi(\beta)^2) \tag{4.9}$$

holds. Then there exists another index function $\tilde{\psi}$ such that the local approximation property (3.1) is fulfilled for $x^\dagger$.

Proof. With the setting of lemma 4.2 and the trained parameters (W, b) of the network f_θ, it holds

$$\|\varphi_{\theta(L)}\left(x^\dagger\right) - Ax^\dagger\| = \|\left(Id - W\right)x^\dagger - b - Ax^\dagger\| \leqslant \|\left(Id - W - A\right)x^\dagger\| + \|b\|. \tag{4.10}$$

Now, we estimate the convergence order of both terms w.r.t $(1-L) \to 0$.

For the first part, we exploit the simple computations

$$\begin{aligned}
\|\left(Id - W - A\right)x^\dagger\|^2 &= \|\sum_j \left(1 - w_j - \sigma_j^2\right)\langle x^\dagger, v_j\rangle v_j\|^2 \\
&= \sum_j \left(1 - \sigma_j^2 - \min\left\{1-\sigma_j^2, L\right\}\right)^2 \langle x^\dagger, v_j\rangle^2 \\
&= \sum_j \max\left\{1 - L - \sigma_j^2, 0\right\}^2 \langle x^\dagger, v_j\rangle^2 \\
&= \sum_{\sigma_j^2 \leqslant 1-L} \left(1 - L - \sigma_j^2\right)^2 \langle x^\dagger, v_j\rangle^2 \\
&\leqslant \sum_{\sigma_j^2 \leqslant 1-L} (1-L)^2 \langle x^\dagger, v_j\rangle^2 \\
&= (1-L)^2 \sum_{\sigma_j^2 \leqslant 1-L} \langle x^\dagger, v_j\rangle^2.
\end{aligned} \tag{4.11}$$

Using assumption (4.9), we obtain

$$(1-L)^2 \sum_{\sigma_j^2 \leqslant 1-L} \langle x^\dagger, v_j\rangle^2 = \mathcal{O}\left((1-L)^2 \psi\left(1-L\right)^2\right). \tag{4.12}$$

Regarding the second part, it holds

$$\begin{aligned}\|b\|^2 &= \sum_j b_j^2 = \sum_j \max\left\{0, 1-L-\sigma^2\right\}^2 \mu_j^2 \\ &= \sum_{\sigma_j^2 \leqslant 1-L} \left(1-L-\sigma_j^2\right)^2 \mu_j^2 \leqslant (1-L)^2 \sum_{\sigma_j^2 \leqslant 1-L} \mu_j^2. \end{aligned} \tag{4.13}$$

Since the values μ_j form by definition (see lemma 4.2) an ℓ^2 sequence, there exists an index function $\tilde{\psi} \geqslant \psi$ such that

$$\sum_{\sigma_j^2 \leqslant 1-L} \mu_j^2 = \mathcal{O}\left(\tilde{\psi}(1-L)^2\right). \tag{4.14}$$

Overall, we obtain

$$\|\left(Id - W - A\right)x^\dagger\| + \|b\| = \mathcal{O}\left((1-L)^2\,\tilde{\psi}\,(1-L)^2\right), \tag{4.15}$$

thus, the local approximation property (3.1) is fulfilled. □

While (4.9) is quite a technical source condition, standard source conditions of the form

$$\exists w \in X, \mu > 0: \quad x^\dagger = A^\mu w \tag{4.16}$$

imply it. In this case, the index function ψ is of the form $\psi(\beta) = \beta^\mu$. The proof of the following corollary reveals the exact relation between standard source conditions and (4.9).

Corollary 4.2. *We assume that the setting of lemma 4.2 holds. Let $\mathcal{A}$ be compact. Then, for any $x^\dagger \in \mathcal{N}(\mathcal{A})^\perp$ the local approximation property (3.1) is fulfilled (i.e. $S = \mathcal{N}(\mathcal{A})^\perp$).*

Proof. Let $x^\dagger \in \mathcal{N}(\mathcal{A})^\perp$ be arbitrary. By construction, $A = \mathcal{A}^*\mathcal{A}$ is self-adjoint and non-negative. As A is also compact, we can deduce from [29, theorem 1]: For any $\varepsilon > 0$ there exists an index function $\psi : [0,1] \to [0,\infty)$ such that $x^\dagger \in \{x \in X \,|\, x = \psi(A)w, \|w\| \leqslant (1+\varepsilon)\|x^\dagger\|\}$.

This implies that there exists a $w \in X$ such that $x^\dagger = \psi(A)w$. We thus obtain for $\beta > 0$

$$\begin{aligned}\sum_{\sigma_j^2 \leqslant \beta} \langle x^\dagger, v_j\rangle^2 &= \sum_{\sigma_j^2 \leqslant \beta} \langle \psi\left(A\right) w, v_j\rangle^2 = \sum_{\sigma_j^2 \leqslant \beta} \psi\left(\sigma_j^2\right)^2 \langle w, v_j\rangle^2 \\ &\leqslant \sum_{\sigma_j^2 \leqslant \beta} \psi\left(\beta\right)^2 \langle w, v_j\rangle^2 \leqslant \|w\|^2 \psi\left(\beta\right)^2 \leqslant (1+\varepsilon)^2 \|x^\dagger\|^2 \psi\left(\beta\right)^2. \end{aligned} \tag{4.17}$$

By lemma 4.3, we get the desired local approximation property (3.1). □

The verification of regularization properties by theorem 3.1 in terms of the local approximation property is thus in line with the existing theory exploited in lemma 3.3, and it further illustrates the character of the local approximation property combining the approximation capabilities of the residual network and a source condition on the desired solution.

4.3. Linear network with ReLU activation

In the following, we include selected nonlinearities as activation functions in a shallow network, allowing for an analytical investigation of the resulting reconstruction scheme. Here, we start with a ReLU activation, which requires a certain assumption on the training dataset depending on the chosen nonlinearity. For simplicity, we do not include a bias in the architecture, in contrast to the architecture from the last section.

Lemma 4.4. *Let $(v_j, \sigma_j^2)_j$ be the eigenvectors and eigenvalues of A and $\varphi_\theta = Id - f_\theta$ be an iResNet which solves (4.1) with*

(i) *$f_\theta(x) = \phi(Wx)$, where $\theta = W$ and $\Theta = \{W \in L(X) \,|\, \exists : (w_j)_{j\in\mathbb{N}} : W = \sum_{j\in\mathbb{N}} w_j \langle \cdot, v_j\rangle v_j\}$ and ϕ being the ReLU function w.r.t. the eigenvectors, i.e. $\phi(x) = \sum_{j\in\mathbb{N}} \max(0, \langle v_j, x\rangle) v_j$,*

(ii) *for every eigenvector v_j of A, the training dataset $\{x^{(i)}\}_{i\in\{1,\dots,N\}}$ contains at least one $x^{(i)}$ s.t. $\langle x^{(i)}, v_j\rangle > 0$.*

Then, the solution of (4.1) is $W = \sum_{j\in\mathbb{N}} w_j \langle \cdot, v_j\rangle v_j$, $w_j = \min\{1 - \sigma_j^2, L\}$ and (3.15) holds with

$$r_L\left(\sigma^2, s\right) = \begin{cases} \frac{1}{\max\{\sigma^2, 1-L\}} & \text{if } s \geqslant 0, \\ 1 & \text{if } s < 0. \end{cases} \tag{4.18}$$

The proof of the lemma can be found in appendix A.3. The obtained filter function is now characterized by a varying behavior depending on the actual measurement y. This is expressed via the variable s which represents the coefficients $\langle z, v_i\rangle = \sigma_i \langle y, u_i\rangle$ (see (3.15) and (3.16)). Whenever the coefficient is positive, the reconstruction scheme behaves like the squared soft TSVD without bias discussed in section 4.2, i.e. they share the same filter function for those $x^\dagger \in \mathcal{N}(A)$. In all other cases, the reconstruction method does not change the coefficients of the data, i.e. it behaves like the identity. Due to the relationship to the squared soft TSVD, we can immediately specify those $x^\dagger$ fulfilling the local approximation property, i.e. for theAQPLease provide the full form of the abbreviations 'ReLU' and 'LISTA' at their first occurrence.ReLU network, we have

$$S = \left\{ x \in \mathcal{N}(\mathcal{A})^\perp \,|\, \forall j \in \mathbb{N} : \langle v_j, x\rangle \geqslant 0 \right\}. \tag{4.19}$$

Thus, the nonlinearity in the network architecture introduces restricted approximation capabilities as well as convergence guarantees on a limited subset of X only.

4.4. Linear network with soft thresholding activation

At last, we want to analyze a network with soft thresholding activation, e.g. known from LISTA [18]. This function is known to promote sparsity since it shrinks all coefficients and sets those under a certain threshold to zero. That is why only training data with sufficiently large coefficients matters for the result of the training, and the condition $|\langle x, v_j\rangle| > \frac{\alpha_j}{L}$ is crucial in the following lemma.

Lemma 4.5. *Let $(v_j, \sigma_j^2)_j$ be the eigenvectors and eigenvalues of A, $\alpha = (\alpha_j)_j, \alpha_j \geqslant 0$ and $\varphi_\theta = Id - f_\theta$ be an iResNet which solves (4.1) with*

(i) *$f_\theta(x) = \phi_\alpha(Wx)$, where $\theta = W$, $\Theta = \{W \in L(X) \,|\, \exists (w_j)_{j\in\mathbb{N}} : W = \sum_{j\in\mathbb{N}} w_j \langle \cdot, v_j\rangle v_j\}$ and ϕ_α is the soft thresholding function w.r.t. the eigenvectors, i.e. $\phi_\alpha(x) = \sum_{j\in\mathbb{N}} \operatorname{sign}(\langle x, v_j\rangle) \max(0, |\langle x, v_j\rangle| - \alpha_j) v_j$,*

(ii) *the training data set is* $\{x^{(i)}\}_{i\in\{1,\dots,N\}}$*, where for any* $j\in\mathbb{N}:\exists i:\ |\langle x^{(i)},v_j\rangle|>\frac{\alpha_j}{L}$.

Then, a solution of (4.1) is $W=\sum_{j\in\mathbb{N}} w_j\langle\cdot,v_j\rangle v_j$, $w_j=\min\{\frac{\alpha_j}{p_{L,j}}+1-\sigma_j^2,L\}$, $p_{L,j}=\frac{\sum_{i\in I_j(L)}|\langle x^{(i)},v_j\rangle|^2}{\sum_{i\in I_j(L)}|\langle x^{(i)},v_j\rangle|}$, $I_j(L)=\{i\in\{1,\dots,N\}|\ |\langle x^{(i)},v_j\rangle|>\frac{\alpha_j}{L}\}$ *and (3.15) holds with*

$$r_L\left(\sigma_j^2,s\right)=\begin{cases}\frac{1}{\max\left\{\sigma_j^2-\frac{\alpha_j}{p_{L,j}},1-L\right\}}\frac{|s|-\alpha_j}{|s|} & \text{if } |s|>\frac{\alpha_j}{w_j},\\ 1 & \text{if } |s|\leqslant\frac{\alpha_j}{w_j}.\end{cases}\tag{4.20}$$

For singular values $\sigma_j^2=1$, w_j *is not uniquely determined.*

The proof of the lemma can be found in appendix A.4. It follows the same line of reasoning as in the previous sections but is more technical due to the effects of the nonlinearity.

So far, the filter function r_L in lemma 4.5 is only defined on the discrete values σ_j^2 (and not continuous for $\sigma^2\in[0,1]$) since it depends on the coefficients $p_{L,j}$, α_j and w_j. However, if we assume continuous extensions $p_L(\sigma^2)$ with $p_L(\sigma_j^2)=p_{L,j}$, $w_L(\sigma^2)$ with $w_L(\sigma_j^2)=w_j$, and $\alpha(\sigma^2)$ with $\alpha(\sigma_j^2)=\alpha_j$, the function $r_L=r_L(\sigma^2,s)$ also becomes continuous. The continuity at the point $|s|=\frac{\alpha(\sigma^2)}{w_L(\sigma^2)}$ is assured by

$$\begin{aligned}&\frac{1}{\max\left\{\sigma^2-\frac{\alpha(\sigma^2)}{p_L(\sigma^2)},1-L\right\}}\frac{\frac{\alpha(\sigma^2)}{w_L(\sigma^2)}-\alpha\left(\sigma^2\right)}{\frac{\alpha(\sigma^2)}{w_L(\sigma^2)}}\\ &\quad=\frac{1-w_L\left(\sigma^2\right)}{\max\left\{\sigma^2-\frac{\alpha(\sigma^2)}{p_L(\sigma^2)},1-L\right\}}=1=r_L\left(\sigma^2,s\right).\end{aligned}\tag{4.21}$$

To be able to interpret the filter function, suitable values for s representing the coefficients $\langle z,v_i\rangle=\sigma_i\langle y,u_i\rangle$ (see (3.15) and (3.16)) need to be considered. One reasonable option is to choose s according to the data on which φ_θ has been trained. We thus consider an eigenvector v_j scaled with the coefficient $p_{L,j}$. Since φ_θ minimizes (4.1), we expect $\varphi_\theta(p_{L,j}v_j)\approx p_{L,j}Av_j=p_{L,j}\sigma_j^2v_j$ and $\varphi_\theta^{-1}(p_{L,j}\sigma_j^2v_j)\approx p_{L,j}v_j$, respectively. Accordingly, we choose $|s|$ proportional to $p_L(\sigma^2)\sigma^2$, i.e. $|s|=\gamma p_L(\sigma^2)\sigma^2$ for different values $\gamma>0$. Hence, the case $\gamma=1$ corresponds to a test vector z with coefficients $|\langle z,v_i\rangle|=|s|=p_L(\sigma_j^2)\sigma_j^2$, which perfectly fits to the training data. Analogously, the cases $\gamma<1$ (and $\gamma>1$, respectively) correspond to test data whose coefficients are smaller (or bigger, respectively) than the average coefficients of the training data.

For $\gamma=1$, the filter function r_L can be written in a form, which allows for an easier interpretation. It holds

$$r_L\left(\sigma^2,\pm p_L\left(\sigma^2\right)\sigma^2\right)=\begin{cases}1 & \text{if } \sigma^2\leqslant\frac{\alpha(\sigma^2)}{Lp_L(\sigma^2)},\\ \frac{1}{1-L}\left(1-\frac{\alpha(\sigma^2)}{p_L(\sigma^2)\sigma^2}\right) & \text{if } \frac{\alpha(\sigma^2)}{Lp_L(\sigma^2)}<\sigma^2\leqslant\frac{\alpha(\sigma^2)}{p_L(\sigma^2)}+1-L,\\ \frac{1}{\sigma^2} & \text{if } \sigma^2>\frac{\alpha(\sigma^2)}{p_L(\sigma^2)}+1-L.\end{cases}\tag{4.22}$$

The derivation can be found in appendix A.5. Note that the filter function depends especially on the quotient of $\alpha(\sigma^2)$ (soft thresholding parameter) and $p_L(\sigma^2)$ (property of training data). To visualize the influence, we depicted the graph in figure 5 for two different (constant) values

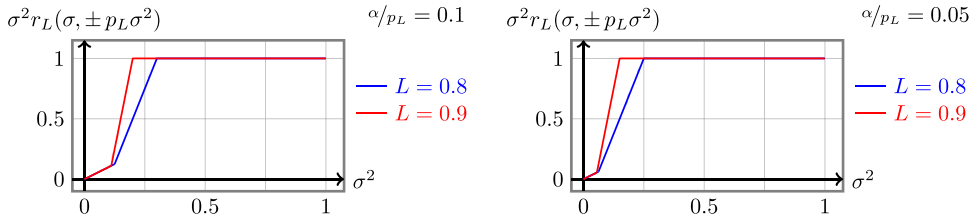


Figure 5. The plots depict graphs of $\sigma^2 r_L(\sigma^2, \pm p_L(\sigma^2)\sigma^2)$ as given in (4.20) and (4.22). The choice $s = p_L(\sigma^2)\sigma^2$ corresponds to the case that the test data is similar to the average training data. In this case, r_L shows a quite similar behavior as the squared soft TSVD function.

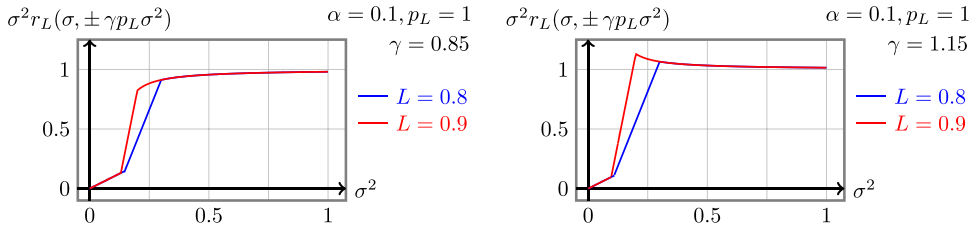


Figure 6. The plots depict graphs of $\sigma^2 r(\sigma^2, \pm\gamma p_L(\sigma^2)\sigma^2)$ as given in (4.20). The case $\gamma \neq 1$ corresponds to test data, which differs from the average training data. Especially for larger singular values σ^2 (to the right of the two kinks), the filter function shows a suboptimal behavior compared to the squared soft TSVD filter function.

of $\frac{\alpha(\sigma^2)}{p_L(\sigma^2)}$. As can be seen, the graph of $\sigma^2 \mapsto \sigma^2 r(\sigma^2, p_L(\sigma^2)\sigma^2)$ has two kinks. The first one depends mainly on the choice of $\frac{\alpha(\sigma^2)}{p_L(\sigma^2)}$, the second one mainly on the choice of L.

For $\gamma \neq 1$, the filter functions cannot be written in a similarly compact and easy form as in (4.22). Instead, we illustrate them in figure 6. In contrast to the case of $\gamma = 1$, the graph of $\sigma^2 \mapsto \sigma^2 r(\sigma^2, p_L(\sigma^2)\sigma^2$ is not equal to one for the larger values of σ^2.

Finally, we want to analyze to which extent φ_θ fulfills the local approximation property (3.1). The illustration of the filter function in figure 5 implies that convergence can only be possible if both kinks tend to zero. So, we need $L \to 1$ and $\alpha \to 0$. Thus, the structure of the nonlinearity (soft thresholding) has a severe influence on the regularization properties of the reconstruction scheme. But in contrast to the ReLU architecture, the soft thresholding operator provides the opportunity to control its similarity to a linear operator, which can be controlled via the coefficients α. In the following lemma, we discuss in more detail how α can be chosen depending on the regularization parameter L to obtain the desired regularization properties.

Lemma 4.6. *Let all assumptions of lemma 4.5 hold. Further, assume that $\mathcal{A}$ is compact and let $p^\dagger = \sum_{j\in\mathbb{N}} p_j^\dagger v_j$ be given by $p_j^\dagger = \frac{\sum_{i=1}^N |\langle x^{(i)}, v_j\rangle|^2}{\sum_{i=1}^N |\langle x^{(i)}, v_j\rangle|}$. In addition, consider $x^\dagger \in X$ as well as strictly monotonic and continuous architecture parameter choices $\alpha_j, \beta : (0,1) \to [0,\infty)$ with*

$$\alpha_j(L) \leqslant p_j^\dagger \beta(L), \text{with } \beta(L) = \mathcal{O}((1-L)\psi(1-L)) \tag{4.23}$$

for an index function $\psi : [0,1] \to [0,\infty)$.

Then, the local approximation property (3.1) holds for any $x^\dagger \in \mathcal{N}(A)^\perp$.

Proof. We first verify that $p_L, p^\dagger \in X$ ($p_L = \sum_{j\in\mathbb{N}} p_{L,j} v_j$). For this, we exploit for each $j \in \mathbb{N}$ that

$$\frac{|\langle x^{(i)}, v_j\rangle|}{\sum_{i=1}^N |\langle x^{(i)}, v_j\rangle|} \leqslant 1 \tag{4.24}$$

for any i. We thus obtain via Young's inequality

$$\|p^\dagger\|^2 = \sum_{j\in\mathbb{N}} \left(p_j^\dagger\right)^2 \leqslant \sum_{j\in\mathbb{N}} \left(\sum_{i=1}^N |\langle x^{(i)}, v_j\rangle|\right)^2 \leqslant 2^{N-1} \sum_{i=1}^N \|x^{(i)}\|^2 < \infty. \tag{4.25}$$

Analogously for p_L. Due to the finite nature of the data set and the properties of α we immediately have $p_L \to p^\dagger$ for $L \to 1$ and $p_L = p^\dagger$ for L being sufficiently close to 1.

We now continue with the approximation property, making use of the notation $x_j := \langle x^\dagger, v_j\rangle$ and a sufficiently large $L < 1$ where $w_j = \min\{\frac{\alpha_j(L)}{p_j^\dagger} + 1 - \sigma_j^2, L\} \geqslant 0$:

$$\begin{aligned}
\|\varphi_{\theta(L)}\left(x^\dagger\right) - Ax^\dagger\|^2 &= \sum_{j\in\mathbb{N}} \left(\max\left(0, w_j|x_j| - \alpha_j(L)\right) - \left(1-\sigma_j^2\right)|x_j|\right)^2 \\
&\leqslant 2\sum_{j\in\mathbb{N}} \left(1-\sigma_j^2 - w_j\right)^2 |x_j|^2 + \left(w_j|x_j| - \max\left(0, w_j|x_j| - \alpha_j(L)\right)\right)^2 \\
&= 2\sum_{j\in\mathbb{N}} \left(1-\sigma_j^2 - w_j\right)^2 |x_j|^2 + \left(\max\left(0, w_j|x_j|\right) - \max\left(0, w_j|x_j| - \alpha_j(L)\right)\right)^2 \\
&\leqslant 2\sum_{j\in\mathbb{N}} \left(1-\sigma_j^2 - w_j\right)^2 |x_j|^2 + \alpha_j(L)^2 \\
&\leqslant 2\sum_{j\in\mathbb{N}} \left(1-\sigma_j^2 - w_j\right)^2 |x_j|^2 + 2\beta(L)^2 \|p^\dagger\|^2
\end{aligned} \tag{4.26}$$

due to Young's inequality, the Lipschitz continuity of the ReLU function, and the assumption on α_j. We further obtain

$$\begin{aligned}
\sum_{j\in\mathbb{N}} \left(1-\sigma_j^2 - w_j\right)^2 |x_j|^2 = &\underbrace{\sum_{j:\sigma_j^2 \leqslant 1-L+\frac{\alpha_j(L)}{p_j^\dagger}} \left(1-\sigma_j^2 - w_j\right)^2 |x_j|^2}_{=:(I)} \\
&+ \underbrace{\sum_{j:\sigma_j^2 > 1-L+\frac{\alpha_j(L)}{p_j^\dagger}} \left(1-\sigma_j^2 - w_j\right)^2 |x_j|^2}_{=:(II)}.
\end{aligned} \tag{4.27}$$

We thus need to derive estimates for (I) and (II).

$$\begin{aligned}(I) &= \sum_{j:\sigma_j^2 \leqslant 1-L+\frac{\alpha_j(L)}{p_j^\dagger}} \left(1-\sigma_j^2-L\right)^2 |x_j|^2 \\ &\leqslant \sum_{j:\sigma_j^2 \leqslant 1-L+\beta(L)} \left(1-\sigma_j^2-L\right)^2 |x_j|^2 \\ &\leqslant \sum_{j:\sigma_j^2 \leqslant 1-L} \left(1-\sigma_j^2-L\right)^2 |x_j|^2 + \sum_{j:\sigma_j^2 > 1-L \wedge \sigma_j^2 \leqslant 1-L+\beta(L)} \left(1-\sigma_j^2-L\right)^2 |x_j|^2 \\ &\leqslant \sum_{j:\sigma_j^2 \leqslant 1-L} (1-L)^2 |x_j|^2 + \sum_{j:\sigma_j^2 > 1-L \wedge \sigma_j^2 \leqslant 1-L+\beta(L)} \beta(L)^2 |x_j|^2 \\ &\leqslant (1-L)^2 \sum_{j:\sigma_j^2 \leqslant 1-L} |x_j|^2 + \beta(L)^2 \|x^\dagger\|^2, \end{aligned} \tag{4.28}$$

where we again exploit Young's inequality and the properties of a_j. For (II) we immediately obtain

$$(II) = \sum_{j:\sigma_j^2 > 1-L+\frac{\alpha_j(L)}{p_j^\dagger}} \left(\frac{\alpha_j(L)}{p_j^\dagger}\right)^2 |x_j|^2 \leqslant \beta(L)^2 \|x^\dagger\|^2. \tag{4.29}$$

Due to the properties of β, (II) already has the desired characteristics for $L \to 1$. In contrast, (I) requires some further calculations. Following the same line of reasoning as in the proof of corollary 4.2 we can find an index function ψ' for any $\varepsilon > 0$ such that

$$\begin{aligned}\sum_{j:\sigma_j^2 \leqslant 1-L} \langle x^\dagger, v_j\rangle^2 &\leqslant (1+\varepsilon)^2 \|x^\dagger\|^2 \psi'(1-L)^2 \\ &= \mathcal{O}\left(\psi'(1-L)^2\right). \end{aligned} \tag{4.30}$$

Combining this with (4.26), (4.28) and (4.29) we obtain the desired result. □

Remark 4.3. Analogous to the line of reasoning in the proof of lemma 4.3, we split the series into two sums, (I) and (II). (I) takes care of small singular values and needs to be related to a property of the element $x^\dagger$ in terms of a certain kind of source condition. The second sum (II) is somehow related to the approximation properties of the underlying architecture. In the proof of lemma 4.3 it is immediately zero for any $x^\dagger$ which is due to the linear network assumption therein. In contrast, in the previous proof we had to carefully control the behavior of this term since it is strongly influenced by the structure of the nonlinearity.

In general, the previous lemma serves as an example of the interplay between the approximation capabilities of the network and the resulting regularization properties.

5. Numerical experiments

In this section, we present numerical results in order to compare our theoretical findings from section 3 and section 4 to its corresponding numerical implementations and extend these by experiments in a more general setting by learning from noisy measurements. To this end, we

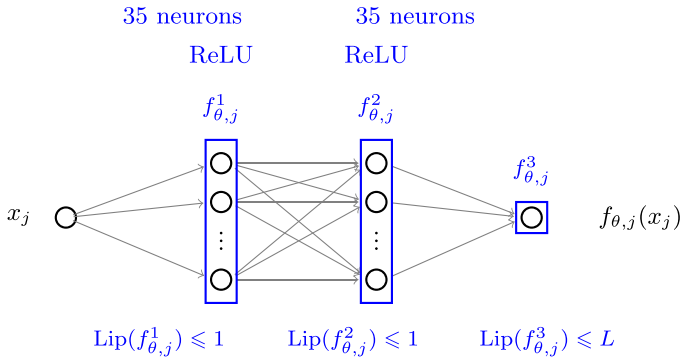


Figure 7. Illustration of the subnetworks $f_{\theta,j} : \mathbb{R} \to \mathbb{R}$, $j = 1, \ldots, n$. Each having 1366 trainable parameters.

use a Radon transform with 30 angles and 41 detector pixels as our forward operator. We discretize the Radon transform for 28×28 px images. This results in a linear forward operator $\mathcal{A} : \mathbb{R}^{28\times 28} \to \mathbb{R}^{30\times 41}$. For the training of the different iResNet architectures, we use the well-known Modified National Institute of Standards and Technology (MNIST) dataset of handwritten digits [25]. This dataset is split into 60 000 images for training and 10 000 images for testing. For our experiments, we treat the 28×28 images as column vectors of length 784 and use fully connected layers in all network architectures. We optimize the networks using Adam [23] and a learning rate scheduling scheme.

There are multiple ways to satisfy the Lipschitz constraint during training. Bungert *et al* [11] use an additional penalty term in the loss function. However, this method does not strictly enforce the constraint. Behrmann *et al* [8] use contractive nonlinearities and only constrain the Lipschitz constant of each individual linear mapping. They compute the Lipschitz constant using a power iteration and normalize the weights after each training step. We observe an improvement of convergence when directly parameterizing the weights to fulfill a specific Lipschitz constant by extending the approach in [30].

In section 5.1, we show experiments on the local approximation property in theorem 3.1. To this end, we use the diagonal architecture proposed in section 3.2. The resulting data-dependent filter functions are visualized in section 5.2, and the convergence property is considered in section 5.3.

We implement each subnetwork $f_{\theta,j}$, $j = 1, \ldots, n$, (see (3.12)) as a small fully-connected neural network with independent weights for each j, where n denotes the total number of singular values in the discrete setting. Each subnetwork consists of three layers, where the first two layers $f^k_{\theta,j}$, $k = 1, 2$ each contain 35 hidden neurons, and its final layer $f^3_{\theta,j}$ contains 1 neuron. Every layer is equipped with a linear matrix multiplication and corresponding additive bias and a ReLU activation function ($k = 1, 2$). Accordingly, each subnetwork has 1366 trainable parameters. An illustration of the architecture is provided in figure 7. Altogether, the parameters of the subnetworks determine the parameters $\theta \in \Theta(L)$ of $\varphi_\theta = \mathrm{Id} - f_\theta$, where $\Theta(L)$ includes the network parameters as well as the Lipschitz constraint being realized by constraints on the network parameter. Here, we enforce the Lipschitz constants $\mathrm{Lip}(f_{\theta,j}) \leqslant L$, $j = 1, \ldots, n$, by constraining $\mathrm{Lip}(f^k_{\theta,j}) \leqslant 1$ for $k = 1, 2$ and $\mathrm{Lip}(f^3_{\theta,j}) \leqslant L$. In the following, we thus write $\theta(L)$ in order to give emphasis to the regularization parameter. Our training objective is the minimization of the *approximation loss* (4.1), i.e. minimize the loss

$$l(\varphi_\theta, A) = \frac{1}{N}\sum_i \|\varphi_\theta(x^{(i)}) - Ax^{(i)}\|^2 \tag{5.1}$$

subject to $\mathrm{Lip}(f_\theta) \leqslant L$.

The source code corresponding to the experiments in this section is openly available [44].

5.1. Local approximation property

According to theorem 3.1, we expect $\varphi_{\theta(L)}^{-1}$ to act as a regularizer as soon as the local approximation property (3.1) is fulfilled. Note that the architecture does not change for varying L in our experiments such that we write $\varphi_{\theta(L)}$ instead of $\varphi_{\theta(L),L}$. To be able to observe this behavior of our trained network, the local approximation property needs to be verified with respect to the trained network parameters. Therefore, we evaluate the approximation error with respect to fixed data samples

$$\mathcal{E}_{x^{(i)}}\left(\varphi_{\theta(L)}, A\right) = \|\varphi_{\theta(L)}(x^{(i)}) - Ax^{(i)}\| \tag{5.2}$$

as well as the mean approximation error over the test data set

$$\mathcal{E}_{\mathrm{mean}}\left(\varphi_{\theta(L)}, A\right) = \frac{1}{N}\sum_i \|\varphi_{\theta(L)}(x^{(i)}) - Ax^{(i)}\|. \tag{5.3}$$

Figure 8 indicates superlinear convergence of $\mathcal{E}_{\mathrm{mean}}$ for $L \to 1$, i.e. the existence of an index function ψ such that the local approximation property (3.1) is satisfied within our test data set on average. This underpins the capability of the chosen network architecture and training to locally approximate the operator A in terms of (3.1). Furthermore, the evaluation of $\mathcal{E}_{x^{(1)}}$ shows that the chosen sample $x^{(1)}$ fulfills the superlinear convergence and behaves very similarly to the mean error over the test data set. However, from the selection of data sample $x^{(2)}$ and corresponding error $\mathcal{E}_{x^{(2)}}$, we notice that the local approximation property does not hold for all samples of our test data set. Figure 8 additionally shows that some coefficients $x_j^{(2)}$ of $x^{(2)}$, corresponding to large singular values, severely deviate from the corresponding mean values with respect to the data distribution. This effect is reduced for $x^{(1)}$. Therefore, the slow, respectively non-, convergence of $\mathcal{E}_{x^{(2)}}$ could possibly be explained by the fact that structures outside of the core of the distribution of chosen singular values have not been learned properly during network training.

5.2. Data-dependent filter functions for diagonal architecture

For the experiments in this and the subsequent section, we train networks $\varphi_{\theta(L)}$ with Lipschitz bounds $L \in \{L_m = 1 - 1/3^m \,|\, m = 1, \ldots 5\}$. We also include additive Gaussian noise in the network training via minimizing the approximation loss (4.1). More precisely, for each training sample $x^{(i)}$ we generate $Ax^{(i)} + \eta^{(i)}$ with $\eta^{(i)} \sim \mathcal{N}(0, \delta_\ell \mathrm{Id})$ and relative noise level $\delta_\ell = \hat{\delta}_\ell \cdot \mathrm{std}_{\mathrm{MNIST}}$, where $\mathrm{std}_{\mathrm{MNIST}}$ denotes the averaged standard deviation of the coefficients $\langle x^{(i)}, v_j\rangle$ of the MNIST data set (standard deviation with respect to i, mean with respect to j) and

$$\hat{\delta}_\ell = \begin{cases} \left(\frac{1}{3}\right)^{7-\ell} & \text{for } 0 < \ell < 7 \\ 0 & \text{for } \ell = 0. \end{cases} \tag{5.4}$$

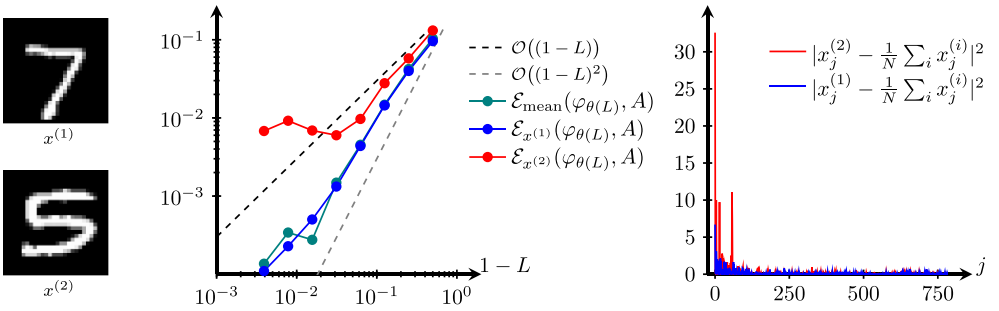


Figure 8. Test samples $x^{(1)}$ and $x^{(2)}$ *(left)*. Evaluations of $\mathcal{E}_{\text{mean}}(\varphi_{\theta(L_m)},A)$, $\mathcal{E}_{x^{(1)}}(\varphi_{\theta(L_m)},A)$ and $\mathcal{E}_{x^{(2)}}(\varphi_{\theta(L_m)},A)$ for $L_m = 1 - 1/2^m$ with $m = 1,\dots,8$ on the MNIST test data set *(middle)*. Squared absolute differences of the coefficients $x_j^{(k)} = \langle x^{(k)}, v_j\rangle$ for $k = 1,2$ to the mean value of corresponding coefficients in the test data set *(right)*.

The loss function l then amounts to

$$\min_{\theta\in\Theta(L)} l(\varphi_\theta,A) = \min_{\theta\in\Theta(L)} \frac{1}{N}\sum_{i=1}^{N} \|\varphi_\theta(x^{(i)}) - Ax^{(i)} - \eta^{(i)}\|^2. \tag{5.5}$$

Note that this includes (4.1) in the noise-free case. Trained networks on noisy samples with noise level δ with a particular Lipschitz bound L are denoted by $\varphi_{\theta(L,\delta)}$. The noise-free training outcome is denoted by $\varphi_{\theta(L)}$.

Utilizing identical network architectures as in the previous subsection, we evaluate the learned filter functions of chosen networks. Analogously to (3.19) the reconstruction can be written as

$$T_L(y) = \varphi_\theta^{-1}(\mathcal{A}^* y) = \hat{b}_L + \sum_{j\in\mathbb{N}} \hat{r}_L(\sigma_j^2, \sigma_j\langle y,u_j\rangle)\sigma_j\langle y,u_j\rangle v_j \tag{5.6}$$

for $\hat{b}_L \in X = \mathbb{R}^n$. As stated in section 3.2, the learned filter function of a trained network with diagonal architecture follows immediately from its subnetworks. Due to the additional bias, we make a small adaption to (3.17), which gives

$$(\mathrm{Id} - f_{\theta,j})^{-1}(s) - \hat{b}_{L,j} = r_L(\sigma_j^2, s)s \qquad \text{for } s\in\mathbb{R}, \tag{5.7}$$

where each entry $\hat{b}_{L,j} = \langle b_L, v_j\rangle$ corresponds to the axis intercept of a subnetwork $\varphi_{\theta,j}$, $j = 1,\dots,n$, i.e. $\hat{b}_{L,j} = (\mathrm{Id} - f_{\theta,j})^{-1}(0)$. Since $f_{\theta,j}$ corresponds to σ_j or σ_j^2, respectively, we also write $\hat{b}_{L,j} = \hat{b}_L(\sigma_j^2)$.

Adapting to the mean values with respect to the distribution of each coefficient, we compute

$$\mu_j := \frac{1}{N}\sum_{i=1}^{N}\langle x^{(i)}, v_j\rangle \quad \text{for } j = 1,\dots,n \tag{5.8}$$

and evaluate the filter function with respect to its second variable at the continuous extension $\sigma^2\mu(\sigma^2)$ with $\mu(\sigma_j^2) = \mu_j, j = 1,\dots,n$.

From figure 9 we notice that the regularization of smaller singular values increases with decreasing Lipschitz bound L for the mean MNIST data sample. This is in line with the regularization theory, as L serves as our regularization parameter. Thus, the filter plots in figure 9

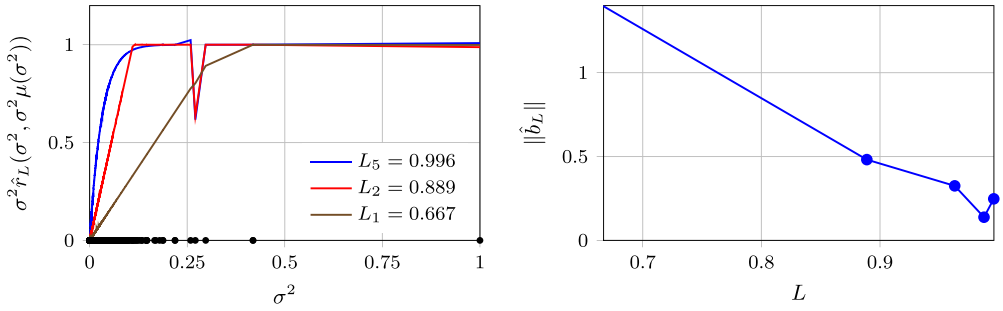


Figure 9. Learned filter functions $r_L(\sigma^2, \sigma^2\mu(\sigma^2))$ of the iResNet with diagonal architecture and zero noise level δ_0 for different Lipschitz constraints L *(left)* where the eigenvalues σ_j^2 are highlighted as black bullets on the x-axis. $\|\hat{b}_L\|$ corresponding to the axis intersection $\hat{b}_L = \sum_j \hat{b}_{L,j} v_j$, evaluated for L_m, $m = 1, \ldots, 5$ *(right)*.

visualize how L acts as a regularization parameter. The trained diagonal networks $\varphi_{\theta(L_m)}$ for L_2 and L_5 show a slightly different behavior at $\sigma_j^2 \approx 0.27$. The distributions of the coefficients $\langle x, v_j \rangle_{j=1,\ldots,n}$ are treated independently and in this particular singular value we observed a wider distribution of coefficients $\langle x, v_j \rangle$ in the dataset. The inherent structure within the dataset might be one possible explanation for this outlier. In general, when neglecting the previously mentioned outlier, for the mean MNIST sample one can observe a similarity to the squared soft TSVD filter function (see section 4.2) to some extent. In addition, the observed decay of $\|\hat{b}_L\|$ with increased L is also in line with the necessary condition for a classical filter function with bias to become a convergent regularization (see lemma 3.3). The observed increase for the largest L is most likely caused by a numerical instability when evaluating $(\mathrm{Id} - f_{\theta,j})^{-1}(0)$ via the fixed point iteration performed for 30 iterations.

Figure 10 includes reconstructions of a fixed sample from the test data set. This example illustrates the effect of a regularization induced by the Lipschitz bound even if the training takes noise into account. It becomes stronger for small values of L, which coincides with our previous observations. Reconstructions resulting from increased noise levels require stronger regularizations in order to improve reconstruction quality. Therefore, the best reconstructions in case of $\hat{\delta}_0$ and $\hat{\delta}_1$ result from $\varphi^{-1}_{\theta(L_3,\delta_0)}$ and $\varphi^{-1}_{\theta(L_3,\delta_1)}$. In comparison, $\varphi^{-1}_{\theta(L_2,\delta_4)}$ and $\varphi^{-1}_{\theta(L_2,\delta_3)}$ provide improved reconstructions for the cases $\hat{\delta}_4$ and $\hat{\delta}_3$. Moreover, at L_1 we notice similar blurred structures in the background of the reconstructed digit for all noise levels. One might argue that its structure compares to a handwritten digit itself, making the learned background pattern being encoded in the bias $\hat{b}_L$ suitable to the data set. These additional observations from figure 10 indicate a dependency of the regularization on learned data structures. The corresponding filter functions being illustrated in the right column of figure 10 show a similar behavior for all training noise levels which underpins that the outcome of the approximation training is independent of the noise for a sufficiently large number of training samples. The outlier in the filter function for the mean sample $(\mu_j)_j$ can also be observed in figure 10. In addition, this test sample has a slightly different behavior with respect to the second singular value. Note that the seemingly linear behavior for $\sigma_j^2 > 0.4$ is only due to the fact that this is the linear continuous extension between the first and second singular value. In summary, the resulting filter behaves similarly to the squared soft TSVD filter independent of the training noise and with exceptions at two singular values.

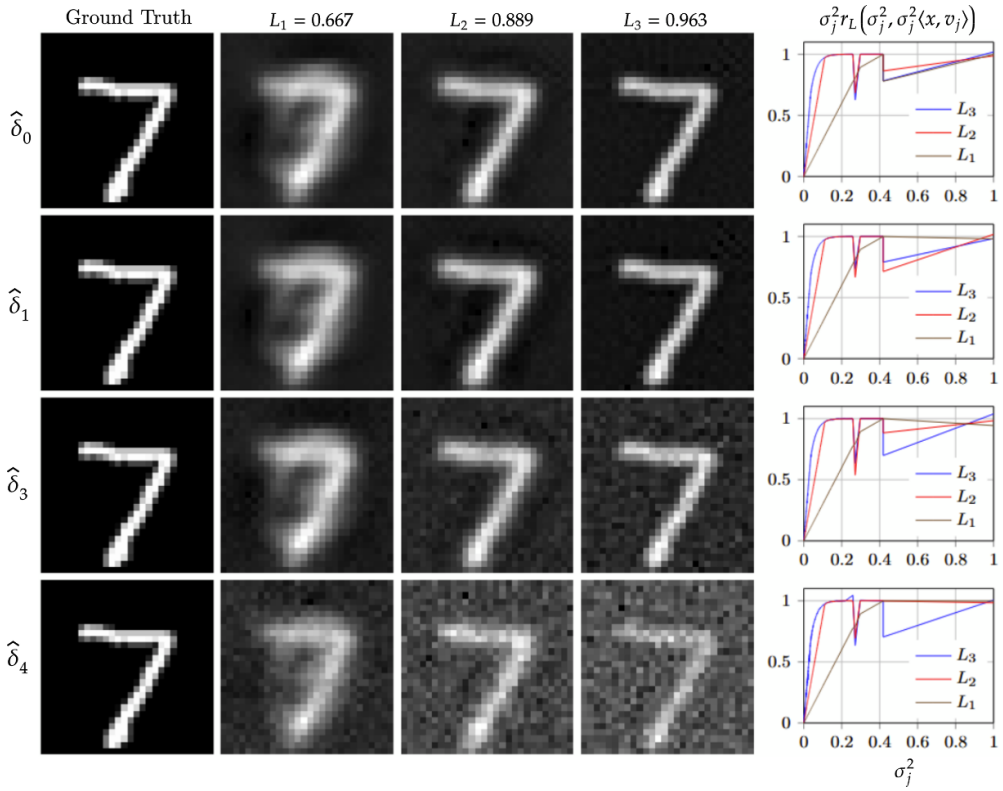


Figure 10. Reconstructions of an MNIST sample $\tilde{x}$ from the test data set by computing $\varphi^{-1}_{\theta(L_m,\delta_\ell)}(A\tilde{x}+\tilde{\eta})$ with $\tilde{\eta} \sim \mathcal{N}(0, \delta_\ell Id)$ for Lipschitz bounds L_m *(columns)* and noise levels $\delta_\ell = \hat{\delta}_\ell \cdot \mathrm{std}_{\mathrm{MNIST}}$ with $\ell = 0, 1, 3, 4$ *(rows)*. The last column depicts the filter functions at L_1, L_2, and L_3 for each noise level with respect to the sample $\tilde{x}$.

5.3. Convergence property

After investigating approximation capabilities of trained networks with respect to the operator A in section 5.1 in the noise-free case and extending the training to the noisy case in section 5.2, we continue verifying the convergence property with respect to different noise levels. We analyze the convergence property by computing the averaged *reconstruction error*

$$\mathrm{MSE}^{\delta_\ell}_{\mathrm{reco}}\left(\varphi_{\theta(L,\delta)}, A\right) = \frac{1}{N}\sum_{i=1}^{N} \|x^{(i)} - \varphi^{-1}_{\theta(L,\delta)}(Ax^{(i)} + \eta^{(i)})\|^2, \tag{5.9}$$

including the noise levels δ_ℓ and noise samples $\eta^{(i)} \sim \mathcal{N}(0, \delta_\ell Id)$ in the reconstruction process where the network has been trained on noise level δ. We thus can evaluate the noise-free training case, which solely aims to impart data dependency, on reconstructions from noisy data, and the noisy training case where training and test noise share the same level. Reconstructions, i.e. the inverse of the iResNet, are obtained by using 30 fixed point iterations.

Figure 11 shows that $\varphi^{-1}_{\theta(L,\delta_0)}$ as well as $\varphi^{-1}_{\theta(L,\delta_\ell)}$ provides more accurate reconstructions with respect to (5.9) at large L and low noise levels, whereas this behavior is reversed for decreasing

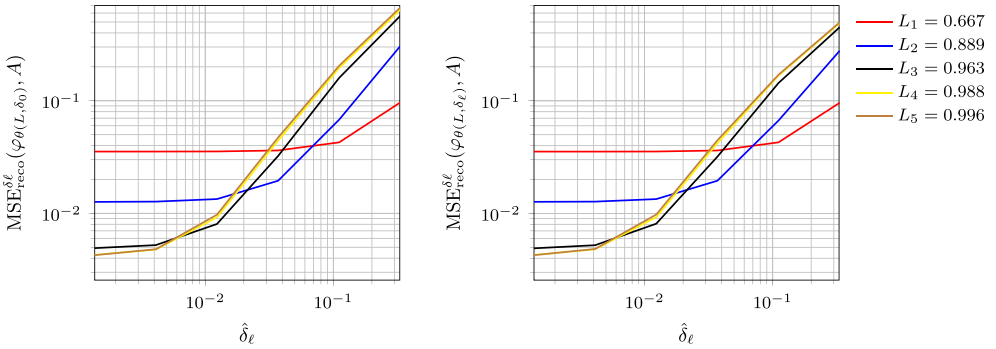


Figure 11. Reconstruction errors $\mathrm{MSE}^{\delta_\ell}_{\mathrm{reco}}(\varphi_{\theta(L,\delta_0)},A)$ with training on noise-free samples *(left)* and $\mathrm{MSE}^{\delta_\ell}_{\mathrm{reco}}(\varphi_{\theta(L,\delta_\ell)},A)$ with training and reconstruction on the same noise level δ_ℓ *(right)* for $\varphi_{\theta(L_m,\delta_\ell)}$ over different noise levels $\delta_\ell = \hat{\delta}_\ell \cdot \mathrm{std}_{\mathrm{MNIST}}$, $\ell = 0,\dots,6$, and Lipschitz bounds L_m, $m = 1,\dots,5$.

Lipschitz bound and increasing noise levels. This is consistent with the regularization theory and the visualized reconstructions in figure 10, as high noise levels require strong regularizations and vice versa. The behavior of $\mathrm{MSE}^{\delta_\ell}_{\mathrm{reco}}(\varphi_{\theta(L,\delta_\ell)},A)$ for small L and small noise levels δ_ℓ is rather intuitive, since its approximation capability is limited as a consequence of strong regularization. In addition, from figure 11 one can also extract a suitable candidate for the parameter choice $L(\delta)$ to obtain convergence. The similarity in the behavior of $\varphi^{-1}_{\theta(L,\delta_0)}$ and $\varphi^{-1}_{\theta(L,\delta_\ell)}$ underpins that the outcome of the approximation training is independent of noise if data and noise are independent.

6. Discussion and outlook

In the present work, we developed and investigated the regularization theory for the proposed iResNet reconstruction approach providing a learned method from data samples. The network's local approximation property is fundamental to delivering a convergent regularization scheme. It comprises approximation properties of the architecture and training, a definition of solution type, and a source condition. The approximation loss used for training is motivated by this property. In the most general version, the framework can be understood as a fully-learned end-to-end reconstruction scheme with minor limitations as it relies on the concatenation with $\mathcal{A}^*$, i.e. some *a priori* knowledge on the forward operator is at least included in a hand-crafted way in the reconstruction scheme. Introducing a diagonal architecture type relying on the SVD of the forward operator $\mathcal{A}$ allowed for an analytical investigation of the resulting learned nonlinear spectral reconstruction method, which becomes a convergent regularization when fulfilling the local approximation property. The analysis of trained shallow architectures revealed the link between the classical linear filter-based regularization theory and the concept of the local approximation property, and it illustrated the interplay between the approximation capability of the nonlinear network and the source condition. In addition, we validated and extended the theoretical findings by a series of numerical experiments on the MNIST data set and provided further insights into the learned nonlinear spectral regularization, such as similarity to the analytically determined linear regularization (squared soft TSVD) and data-dependency of the learned regularization.

Our iResNet method using the diagonal architecture can be seen as a generalization of the learned linear spectral regularization considered in [22]. Using a different loss, which measures the reconstruction error, the authors of [22] obtain learned filter functions corresponding to Tikhonov regularization with data- and noise-dependent regularization parameters for each singular vector. An extension of the iResNet approach to this kind of training loss is thus desirable for future comparison.

The present work also serves as a starting point for various future research directions:

As we obtain the inverse of the residual network via a fixed point iteration, findings in the context of the iResNet reconstruction may be related to learned optimization methods [6], maximally monotone operators [37], or potentially to plug-and-play methods [16]. Investigations at the interface to these methods are far beyond the present work and will remain future work.

The concept of a local criterion to guarantee convergence of the iResNet reconstruction method provides further opportunities to investigate the regularization properties of learned reconstruction schemes. In a different context, Aspri *et al* [5] also consider a local property. They consider a learned projection onto subspaces concatenated with the forward operator, i.e. they consider a projected linear problem and combine it with Tikhonov regularization. There, the authors guarantee convergence of the regularization method by the Tikhonov theory, including a different but also localized assumption. The findings in the present work and the literature thus reveal the necessity to explicitly take the approximation capabilities of the learned schemes and the training procedure into account. Immediate future research includes a comprehensive numerical study on how architectural choices such as particular activation functions, e.g. being strictly monotone but nonlinear, influence the desired regularization properties and reconstruction quality. Further potential future investigations in this context are twofold. On the one hand, one may exploit universal approximation properties of the underlying architectures to guarantee convergence for a sufficiently large subset of X, e.g. by verifying the local approximation property. On the other hand, one may adapt the definition of regularization to the approximate and data-based nature of the learning-based setting.

Besides the more general open research questions at the interface to other methods and data-based concepts, the iResNet reconstruction approach itself provides various potential future works. The observed similarity of the learned nonlinear filter functions in the numerical experiments to the analytical solution of the affine network with bias (section 4.2) immediately raises the question: How much does the data's structure influence the resulting reconstruction scheme? Phrased differently, one needs to test the limits of the proposed approximation training. Here, it would be desirable to change the perspective to a Bayesian one considering data and noise distributions and further investigate the training outcome, including additional variations of the training loss and its influence on the local approximation property. Natural extension and generalizations, such as including learned solution types, e.g. by combining our approach with null space networks [41] or relaxing the diagonal assumption of the architecture, remain future work.

In summary, the present first work on learned iResNet regularization schemes builds the basis for various future works of theoretical and numerical nature.

Data availability statement

The data that support the findings of this study are openly available at the following URL/DOI: https://gitlab.informatik.uni-bremen.de/inn4ip/iresnet-regularization.

Acknowledgments

A Denker, N Heilenkötter, M Iske, and J Nickel acknowledge the support by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation)—Project Number 281474342/GRK2224/2. T Kluth acknowledges support from the DELETO project funded by the Federal Ministry of Education and Research (BMBF, Project Number 05M20LBB). P Maass acknowledges support by the BAB-project PY2DLL (EFRE).

Appendix. Proofs of section 4

A.1. Proof of lemma 4.1

Equivalent filter function to the one-parameter-network.

Proof. At first, we observe that the Lipschitz constraint in (4.1) is fulfilled for $|k| \leqslant L$. Thus, (4.1) is equivalent to

$$\begin{aligned} & \min_{|k|\leqslant L} \sum_i \|\varphi_\theta(x^{(i)}) - Ax^{(i)}\|^2 \\ \Leftrightarrow \quad & \min_{|k|\leqslant L} \sum_i \|x^{(i)} - kx^{(i)} + kAx^{(i)} - Ax^{(i)}\|^2 \\ \Leftrightarrow \quad & \min_{|k|\leqslant L} \sum_i (1-k)^2 \|x^{(i)} - Ax^{(i)}\|^2. \end{aligned} \tag{A.1}$$

Since there is one $x^{(i)}$ s.t. $Ax_i \neq x^{(i)}$ the solution is obviously $k = L$ and we have $\varphi_\theta(x) = x - L(x - Ax)$.

Now we use the SVD of $\mathcal{A}$ to solve $\varphi_\theta(\bar{x}) = z$ for $z \in \mathcal{R}(A)$ component wise. For all $j \in \mathbb{N}$ it holds

$$\begin{aligned} & \langle \bar{x} - L(\bar{x} - A\bar{x}), v_j \rangle = \langle z, v_j \rangle \\ \Leftrightarrow \quad & (1-L)\langle \bar{x}, v_j \rangle + L\langle \bar{x}, Av_j \rangle = \langle z, v_j \rangle \\ \Leftrightarrow \quad & \left(1 - L + L\sigma_j^2\right)\langle \bar{x}, v_j \rangle = \langle z, v_j \rangle \\ \Leftrightarrow \quad & \langle \bar{x}, v_j \rangle = \frac{1}{1 - L + L\sigma_j^2}\langle z, v_j \rangle \end{aligned} \tag{A.2}$$

Thus, the filter function

$$r_L\left(\sigma^2, s\right) = \frac{1}{1 - L + L\sigma^2} = \frac{1}{L} \cdot \frac{1}{\alpha + \sigma^2}, \qquad \alpha = \frac{1-L}{L} \tag{A.3}$$

fulfills (3.15). □

A.2. Proof of lemma 4.2

Equivalent filter function to the linear network.

Proof. At first, we observe that the Lipschitz constraint in (4.1) is fulfilled if the eigenvalues $(w_j)_j$ of W are restricted by $|w_j| \leqslant L$. Defining $\Theta_L = \left\{(W, b) \in \Theta \,\middle|\, |w_j| \leqslant L\right\}$, (4.1) is equivalent to

$$\begin{aligned}
&\min_{\theta\in\Theta_L}\sum_{i=1}^{N}\|\varphi_\theta\left(x^{(i)}\right)-Ax^{(i)}\|^2\\
\Leftrightarrow\quad &\min_{(W,b)\in\Theta_L}\sum_{i=1}^{N}\left\|x^{(i)}-Wx_i-b-Ax^{(i)}\right\|^2\\
\Leftrightarrow\quad &\min_{(W,b)\in\Theta_L}\sum_{i=1}^{N}\|P_{\mathcal{N}(A)}\left(x^{(i)}\right)\|^2+\left\|P_{\mathcal{N}(A)^\perp}\left(x^{(i)}\right)-Wx_i-b-Ax^{(i)}\right\|^2. \qquad \text{(A.4)}
\end{aligned}$$

The part with $P_{\mathcal{N}(A)}(x^{(i)})$ is independent of W and b and therefore not relevant. Furthermore, it holds

$$\begin{aligned}
P_{\mathcal{N}(A)^\perp}\left(x^{(i)}\right)-Wx_i-b-Ax^{(i)} &= \sum_{j\in\mathbb{N}}\langle x,v_j\rangle v_j-\sum_{j\in\mathbb{N}}\left(w_j\langle x,v_j\rangle+b_j\right)v_j-\sum_{j\in\mathbb{N}}\sigma_j^2\langle x,v_j\rangle v_j\\
&= \sum_{j\in\mathbb{N}}\left(\left(1-w_j-\sigma_j^2\right)\langle x,v_j\rangle-b_j\right)v_j. \qquad \text{(A.5)}
\end{aligned}$$

Thus, the minimizing problem can be written as

$$\min_{|w_j|\leqslant L,\,(b_j)\in\ell^2}\sum_{i=1}^{N}\sum_{j\in\mathbb{N}}\left(\left(1-w_j-\sigma_j^2\right)\langle x,v_j\rangle-b_j\right)^2. \qquad \text{(A.6)}$$

We can solve this problem for each j separately. For determining b_j we set the derivative to zero and get

$$\begin{aligned}
&0\overset{!}{=}-2\sum_{i=^1}^{N}\left(\left(1-w_j-\sigma_j^2\right)\langle x^{(i)},v_j\rangle-b_j\right)\\
\Leftrightarrow\quad &b_j\overset{!}{=}\frac{1}{N}\left(1-w_j-\sigma_j^2\right)\sum_{i=1}^{N}\langle x^{(i)},v_j\rangle=\left(1-w_j-\sigma_j^2\right)\mu_j. \qquad \text{(A.7)}
\end{aligned}$$

Since for every i, the sequence $(\langle x^{(i)},v_j\rangle)_{j\in\mathbb{N}}$ is in ℓ^2, $(b_j)\in\ell^2$ is also fulfilled. It remains to solve

$$\min_{|w_j|\leqslant L,}\sum_{i=1}^{N}\left(\left(1-w_j-\sigma_j^2\right)\left(\langle x,v_j\rangle-\mu_j\right)\right)^2. \qquad \text{(A.8)}$$

By assumption, for every v_j there is at least one $x^{(i)}$ s.t. $\langle x^{(i)},v_j\rangle\neq\mu_j$, thus, the problem can be simplified to

$$\min_{|w_j|\leqslant L}\left(1-w_j-\sigma_j^2\right)^2\qquad\forall j\in\mathbb{N}. \qquad \text{(A.9)}$$

The obvious solution is $w_j=\min\{1-\sigma_j^2,L\}$, since $1-\sigma_j^2$ is by assumption always positive.

Now we use the SVD of $\mathcal{A}$ to solve $\varphi_\theta(\bar{x}) = z$ for some $z \in \mathcal{R}(A)$ component-wise. For all $j \in \mathbb{N}$ it holds

$$\begin{aligned}
& \langle \bar{x}, v_j \rangle - w_j \langle \bar{x}, v_j \rangle - b_j = \langle z, v_j \rangle \\
\Leftrightarrow \quad & (1 - w_j) \langle \bar{x}, v_j \rangle = \langle z, v_j \rangle + b_j \\
\Leftrightarrow \quad & \langle \bar{x}, v_j \rangle = \frac{1}{1 - w_j} \langle z, v_j \rangle + \frac{b_j}{1 - w_j} \\
\Leftrightarrow \quad & \langle \bar{x}, v_j \rangle = \frac{1}{\max\left\{\sigma_j^2, 1 - L\right\}} \langle z, v_j \rangle + \frac{\max\left\{0, 1 - L - \sigma_j^2\right\}}{\max\left\{\sigma_j^2, 1 - L\right\}} \mu_j. \qquad \text{(A.10)}
\end{aligned}$$

Thus, using the filter function and bias

$$\hat{r}_L\left(\sigma^2\right) = \frac{1}{\max\left\{\sigma^2, 1 - L\right\}}, \qquad \hat{b}_L = \sum_{\sigma_j^2 < 1 - L} \frac{1 - L - \sigma^2}{1 - L} \mu_j v_j, \qquad \text{(A.11)}$$

the reconstruction schemes $\varphi_\theta^{-1} \circ \mathcal{A}^*$ and T_L from (3.19) are equivalent. □

A.3. Proof of lemma 4.4

Equivalent filter function to the ReLU-network.

Proof. At first, we observe that the Lipschitz constraint in (4.1) is fulfilled if the eigenvalues $(w_j)_j$ of W are restricted by $|w_j| \leqslant L$. Thus, (4.1) is equivalent to

$$\begin{aligned}
& \min_{(w_j),\, |w_j| \leqslant L} \sum_i \left\| \varphi_\theta\left(x^{(i)}\right) - A x^{(i)} \right\|^2 \\
\Leftrightarrow \quad & \min_{(w_j),\, |w_j| \leqslant L} \sum_i \left\| x^{(i)} - \phi\left(W x_i\right) - A x^{(i)} \right\|^2 \\
\Leftrightarrow \quad & \min_{(w_j),\, |w_j| \leqslant L} \sum_i \Bigg\| P_{\mathcal{N}(A)}\left(x^{(i)}\right) + \sum_{j \in \mathbb{N}} \langle x^{(i)}, v_j \rangle v_j - \sum_{j \in \mathbb{N}} \max\left\{0, w_j \langle x^{(i)}, v_j \rangle\right\} v_j \\
& \quad - \sum_{j \in \mathbb{N}} \sigma_j^2 \langle x^{(i)}, v_j \rangle v_j \Bigg\|^2 \\
\Leftrightarrow \quad & \min_{(w_j),\, |w_j| \leqslant L} \sum_i \left\| P_{\mathcal{N}(A)}\left(x^{(i)}\right) \right\|^2 + \left\| \sum_{j \in \mathbb{N}} \left(\left(1 - \sigma_j^2\right) \langle x^{(i)}, v_j \rangle - \max\left\{0, w_j \langle x^{(i)}, v_j \rangle\right\} \right) v_j \right\|^2 \\
\Leftrightarrow \quad & \min_{(w_j),\, |w_j| \leqslant L} \sum_i \sum_{j \in \mathbb{N}} \left(\left(1 - \sigma_j^2\right) \langle x^{(i)}, v_j \rangle - \max\left\{0, w_j \langle x^{(i)}, v_j \rangle\right\} \right)^2. \qquad \text{(A.12)}
\end{aligned}$$

At first we focus on the components $\langle x^{(i)}, v_j \rangle$ which are negative or zero. They do not influence the solution of the problem since

$$\min_{|w_j| \leqslant L} \left(-\left(1 - \sigma_j^2\right) - \max\left\{0, -w_j\right\} \right)^2 \qquad \forall j \in \mathbb{N} \qquad \text{(A.13)}$$

is solved by any $w_j \in [0, L]$, as $1 - \sigma_j^2$ is by assumption always positive. Thus, it suffices to consider the cases $\langle x^{(i)}, v_j \rangle > 0$, which lead to

$$\min_{|w_j| \leqslant L} \left(1 - \sigma_j^2 - \max\{0, w_j\}\right)^2 \qquad \forall j \in \mathbb{N}. \tag{A.14}$$

The obvious solution is $w_j = \min\{1 - \sigma_j^2, L\}$.

Now we use the SVD of $\mathcal{A}$ to solve $\varphi_\theta(\bar{x}) = z$ for $z \in \mathcal{R}(A)$ or

$$\bar{x} - \sum_{j \in \mathbb{N}} \max\{0, w_j \langle \bar{x}, v_j \rangle\} v_j = \sum_{j \in \mathbb{N}} \langle z, v_j \rangle v_j, \tag{A.15}$$

respectively, component-wise. For all $j \in \mathbb{N}$ it holds

$$\begin{aligned} & \langle \bar{x}, v_j \rangle - \max\{0, w_j \langle \bar{x}, v_j \rangle\} = \langle z, v_j \rangle \\ \Leftrightarrow \quad & \min\{\langle \bar{x}, v_j \rangle, (1 - w_j) \langle \bar{x}, v_j \rangle\} = \langle z, v_j \rangle. \end{aligned} \tag{A.16}$$

Since $1 - w_j > 0$, the signs of $\langle \bar{x}, v_j \rangle$ and $\langle z, v_j \rangle$ must coincide. In case of positive signs, the minimum on the left hand side takes the value $(1 - w_j)\langle \bar{x}, v_j \rangle$, in case of negative signs, it is $\langle \bar{x}, v_j \rangle$. Thus, we get

$$\langle \bar{x}, v_j \rangle = \begin{cases} \frac{1}{1 - w_j} \langle z, v_j \rangle & \text{if } \langle z, v_j \rangle \geqslant 0, \\ \langle z, v_j \rangle & \text{if } \langle z, v_j \rangle < 0. \end{cases} \tag{A.17}$$

Using $1 - w_j = \max\{\sigma_j^2, 1 - L\}$, it can be seen that

$$r_L\left(\sigma^2, s\right) = \begin{cases} \frac{1}{\max\{\sigma^2, 1 - L\}} & \text{if } s \geqslant 0, \\ 1 & \text{if } s < 0 \end{cases} \tag{A.18}$$

fulfills (3.15). □

A.4. Proof of lemma 4.5

Equivalent filter function to the soft-thresholding-network.

Proof. At first, we observe that the Lipschitz constraint in (4.1) is fulfilled if the eigenvalues $(w_j)_j$ of W are restricted by $|w_j| \leqslant L$. Besides, $f_\theta(x)$ can be written as

$$\begin{aligned} \phi_\alpha(Wx) &= \sum_{j \in \mathbb{N}} \operatorname{sign}(\langle Wx, v_j \rangle) \max(0, |\langle Wx, v_j \rangle| - \alpha_j) v_j \\ &= \sum_{j \in \mathbb{N}} \operatorname{sign}(w_j \langle x, v_j \rangle) \max(0, |w_j \langle x, v_j \rangle| - \alpha_j) v_j. \end{aligned} \tag{A.19}$$

For any $x^{(i)}$ from the training data set, it follows

$$\begin{aligned} & x^{(i)} - \phi_\alpha\left(Wx^{(i)}\right) - Ax^{(i)} \\ & \quad = P_{\mathcal{N}(A)}\left(x^{(i)}\right) + \sum_{j\in\mathbb{N}} \langle x^{(i)}, v_j\rangle v_j - \sum_{j\in\mathbb{N}} \operatorname{sign}\left(w_j\langle x^{(i)}, v_j\rangle\right) \max\left(0, |w_j\langle x^{(i)}, v_j\rangle| - \alpha_j\right) v_j \\ & \qquad - \sum_{j\in\mathbb{N}} \sigma_j^2 \langle x^{(i)}, v_j\rangle v_j \\ & \quad = P_{\mathcal{N}(A)}\left(x^{(i)}\right) + \sum_{j\in\mathbb{N}} \left[x_j^{(i)} - \operatorname{sign}\left(w_j x_j^{(i)}\right) \max\left(0, |w_j x_j^{(i)}| - \alpha_j\right) - \sigma_j^2 x_j^{(i)}\right] v_j \end{aligned} \tag{A.20}$$

where $x_j^{(i)} := \langle x^{(i)}, v_j\rangle$. Therefore, (4.1) is equivalent to

$$\begin{aligned} & \min_{(w_j),\,|w_j|\leqslant L} \sum_i \|\varphi_\theta\left(x^{(i)}\right) - Ax^{(i)}\|^2 \\ \Leftrightarrow \quad & \min_{(w_j),\,|w_j|\leqslant L} \sum_i \left\|x^{(i)} - \phi_\alpha\left(Wx^{(i)}\right) - Ax^{(i)}\right\|^2 \\ \Leftrightarrow \quad & \min_{(w_j),\,|w_j|\leqslant L} \sum_j \sum_i \left(x_j^{(i)} - \operatorname{sign}\left(w_j x_j^{(i)}\right) \max\left(0, |w_j x_j^{(i)}| - \alpha_j\right) - \sigma_j^2 x_j^{(i)}\right)^2. \end{aligned} \tag{A.21}$$

For each j, we have to find the solution of

$$\begin{aligned} & \min_{|w_j|\leqslant L} \sum_i \left(x_j^{(i)} - \operatorname{sign}\left(w_j x_j^{(i)}\right) \max\left(0, |w_j x_j^{(i)}| - \alpha_j\right) - \sigma_j^2 x_j^{(i)}\right)^2 \\ \Leftrightarrow \quad & \min_{|w_j|\leqslant L} \sum_i \left(\left(1-\sigma_j^2\right)\operatorname{sign}\left(x_j^{(i)}\right)|x_j^{(i)}| - \operatorname{sign}(w_j)\operatorname{sign}\left(x_j^{(i)}\right) \max\left(0, |w_j x_j^{(i)}| - \alpha_j\right)\right)^2 \\ \Leftrightarrow \quad & \min_{|w_j|\leqslant L} \sum_i \left(\left(1-\sigma_j^2\right)|x_j^{(i)}| - \operatorname{sign}(w_j) \max\left(0, |w_j x_j^{(i)}| - \alpha_j\right)\right)^2. \end{aligned} \tag{A.22}$$

Note that for $\sigma_j^2 = 1$, any $w_j \in \left[-\frac{\alpha_j}{\max^{(i)}|x_j^{(i)}|}, \frac{\alpha_j}{\max_i|x_j^{(i)}|}\right]$ solves the problem and hence the solution is not unique. In all other cases, we have $1-\sigma_j^2 > 0$, thus, only a positive sign of w_j makes sense and we restrict ourselves to this case. Due to $w_j \geqslant 0$, we have

$$\begin{aligned} & \left(\left(1-\sigma_j^2\right)|x_j^{(i)}| - \max\left(0, |w_j x_j^{(i)}| - \alpha_j\right)\right)^2 \\ & \quad = \begin{cases} \left(\left(1-\sigma^2\right)x_j^{(i)}\right)^2 & \text{if } |w_j x_j^{(i)}| \leqslant \alpha_j, \\ \left(\left(1-\sigma_j^2-w_j\right)|x_j^{(i)}| + \alpha_j\right)^2 & \text{if } |w_j x_j^{(i)}| > \alpha_j. \end{cases} \end{aligned} \tag{A.23}$$

As the first case is an upper bound of the second case, it is highly desirable to choose w_j large enough such that all training samples are contained in the second case. But we also need to take into account the upper bound $w_j \leqslant L$. Due to the data set assumption (ii), the minimization problem (A.22) becomes

$$\min_{w_j\in[0,L]} \sum_{i\in I_j(L)} \left(\left(1-\sigma_j^2-w_j\right)|x_j^{(i)}| + \alpha_j\right)^2. \tag{A.24}$$

The minimizer is

$$w_j = \begin{cases} \frac{\alpha_j}{p_j} + 1 - \sigma_j^2 & \text{if } \frac{\alpha_j}{p_j} + 1 - \sigma_j^2 \leqslant L, \\ L & \text{else,} \end{cases} \tag{A.25}$$

where $p_{L,j} = p_j = \frac{\sum_{i\in I_j(L)} |x_j^{(i)}|^2}{\sum_{i\in I_j(L)} |x_j^{(i)}|}$. Equivalently, we can write $w_j = \min\{\frac{\alpha_j}{p_j} + 1 - \sigma_j^2, L\}$.

Now we use the SVD of $\mathcal{A}$ to solve $\varphi_\theta(\bar{x}) = z$ for $z \in \mathcal{R}(A)$ or

$$\bar{x} - \sum_{j\in\mathbb{N}} \operatorname{sign}\left(w_j\langle \bar{x}, v_j\rangle\right) \max\left(0, |w_j\langle \bar{x}, v_j\rangle| - \alpha_j\right) v_j = \sum_{j\in\mathbb{N}} \langle z, v_j\rangle v_j, \tag{A.26}$$

respectively, componentwise. For all $j \in \mathbb{N}$ it holds

$$\begin{aligned} & \langle \bar{x}, v_j\rangle - \operatorname{sign}\left(w_j\langle \bar{x}, v_j\rangle\right) \max\left(0, |w_j\langle \bar{x}, v_j\rangle| - \alpha_j\right) = \langle z, v_j\rangle \\ \Leftrightarrow \quad & \operatorname{sign}\left(\langle \bar{x}, v_j\rangle\right) \left(|\langle \bar{x}, v_j\rangle| - \max\left(0, w_j|\langle \bar{x}, v_j\rangle| - \alpha_j\right)\right) = \langle z, v_j\rangle. \end{aligned} \tag{A.27}$$

Thus, the sign of $\langle \bar{x}, v_j\rangle$ is the same as the one of $\langle z, v_j\rangle$. Assuming $w_j|\langle \bar{x}, v_j\rangle| \leqslant \alpha_j$, we obtain

$$|\langle \bar{x}, v_j\rangle| = |\langle z, v_j\rangle| \qquad \text{if } |\langle z, v_j\rangle| \leqslant \frac{\alpha_j}{w_j} \tag{A.28}$$

and assuming $w_j|\langle \bar{x}, v_j\rangle| > \alpha_j$, we obtain

$$|\langle \bar{x}, v_j\rangle| = \frac{|\langle z, v_j\rangle| - \alpha_j}{1 - w_j} \qquad \text{if } \frac{|\langle z, v_j\rangle|}{1 - w_j} - \frac{\alpha_j}{1 - w_j} > \frac{\alpha_j}{w_j}. \tag{A.29}$$

In total, this leads to

$$\langle \bar{x}, v_j\rangle = \begin{cases} \langle z, v_j\rangle & \text{if } |\langle z, v_j\rangle| \leqslant \frac{\alpha_j}{w_j}, \\ \operatorname{sign}\left(\langle z, v_j\rangle\right) \frac{|\langle z, v_j\rangle| - \alpha_j}{1 - w_j} & \text{if } |\langle z, v_j\rangle| > \frac{\alpha_j}{w_j}. \end{cases} \tag{A.30}$$

Using $1 - w_j = \max\{\sigma_j^2 - \frac{\alpha_j}{p_j}, 1 - L\}$, we see that the filter function

$$r_L\left(\sigma_j^2, s\right) = \begin{cases} 1 & \text{if } |s| \leqslant \frac{\alpha_j}{w_j}, \\ \frac{1}{\max\left\{\sigma_j^2 - \frac{\alpha_j}{p_j}, 1 - L\right\}} \frac{|s| - \alpha_j}{|s|} & \text{if } |s| > \frac{\alpha_j}{w_j} \end{cases} \tag{A.31}$$

fulfills (3.15). F_L is then derived by exploiting (3.18). □

A.5. Computation of filter function (4.22)

Assuming continuous extensions $p_L(\sigma^2)$ s.t. $p_L(\sigma_j^2) = p_{L,j}$, $w_L(\sigma^2)$ s.t. $w_L(\sigma_j^2) = w_j$, and $\alpha(\sigma^2)$ s.t. $\alpha(\sigma_j^2) = \alpha_j$, we can write the filter function from lemma 4.5 as

$$r_L\left(\sigma^2, s\right) = \begin{cases} \frac{1}{\max\left\{\sigma^2 - \frac{\alpha(\sigma^2)}{p_L(\sigma^2)}, 1 - L\right\}} \frac{|s| - \alpha\left(\sigma^2\right)}{|s|} & \text{if } |s| > \frac{\alpha\left(\sigma^2\right)}{w_L(\sigma^2)}, \\ 1 & \text{if } |s| \leqslant \frac{\alpha\left(\sigma^2\right)}{w_L(\sigma^2)}. \end{cases} \tag{A.32}$$

Besides, it holds $w_L(\sigma^2) = \min\left\{\frac{\alpha(\sigma^2)}{p_L(\sigma^2)} + 1 - \sigma^2, L\right\} \leqslant L$, also by lemma 4.5. Now, we choose $|s| = p_L(\sigma^2)\sigma^2$ and distinguish between three different cases (small, medium and large values of σ^2).

At first, we consider the case $\sigma^2 \leqslant \frac{\alpha(\sigma^2)}{Lp_L(\sigma^2)}$. Then, it holds

$$|s| = p_L\left(\sigma^2\right)\sigma^2 \leqslant \frac{\alpha\left(\sigma^2\right)}{L} \leqslant \frac{\alpha\left(\sigma^2\right)}{w_L\left(\sigma^2\right)}. \tag{A.33}$$

Thus, it follows

$$r_L\left(\sigma^2, \pm p_L\left(\sigma^2\right)\sigma^2\right) = 1. \tag{A.34}$$

Secondly, we consider the case $\frac{\alpha(\sigma^2)}{Lp_L(\sigma^2)} < \sigma^2 \leqslant \frac{\alpha(\sigma^2)}{p_L(\sigma^2)} + 1 - L$. We shortly make sure, that the lower bound is in fact smaller than the upper one. The assumptions of lemma 4.5 imply that $p_L(\sigma^2) > \frac{\alpha(\sigma^2)}{L}$ holds. From this, we can follow

$$\begin{aligned} & p_L\left(\sigma^2\right)(1-L) > \frac{\alpha\left(\sigma^2\right)}{L} - \frac{L\alpha\left(\sigma^2\right)}{L} \\ \Rightarrow \quad & \alpha\left(\sigma^2\right) + p_L\left(\sigma^2\right)(1-L) > \frac{\alpha\left(\sigma^2\right)}{L} \\ \Rightarrow \quad & \frac{\alpha\left(\sigma^2\right)}{p_L\left(\sigma^2\right)} + 1 - L > \frac{\alpha\left(\sigma^2\right)}{Lp_L\left(\sigma^2\right)}. \end{aligned} \tag{A.35}$$

Besides, note that $\sigma^2 \leqslant \frac{\alpha(\sigma^2)}{p_L(\sigma^2)} + 1 - L$ implies $w_L(\sigma^2) = L$. Thus, it holds

$$|s| = p_L\left(\sigma^2\right)\sigma^2 > \frac{\alpha\left(\sigma^2\right)}{L} = \frac{\alpha\left(\sigma^2\right)}{w_L\left(\sigma^2\right)}. \tag{A.36}$$

Accordingly, we obtain

$$\begin{aligned} r_L\left(\sigma^2, \pm p_L\left(\sigma^2\right)\sigma^2\right) &= \frac{1}{\max\left\{\sigma^2 - \frac{\alpha(\sigma^2)}{p_L(\sigma^2)}, 1-L\right\}} \frac{p_L\left(\sigma^2\right)\sigma^2 - \alpha\left(\sigma^2\right)}{p_L\left(\sigma^2\right)\sigma^2} \\ &= \frac{1}{1-L} \frac{p_L\left(\sigma^2\right)\sigma^2 - \alpha\left(\sigma^2\right)}{p_L\left(\sigma^2\right)\sigma^2} = \frac{1}{1-L}\left(1 - \frac{\alpha\left(\sigma^2\right)}{p_L\left(\sigma^2\right)\sigma^2}\right). \end{aligned} \tag{A.37}$$

At last, we consider the case $\sigma^2 > \frac{\alpha(\sigma^2)}{p_L(\sigma^2)} + 1 - L$. Note that this implies $w_L(\sigma^2) = \frac{\alpha(\sigma^2)}{p_L(\sigma^2)} + 1 - \sigma^2$. Besides it holds $\sigma^2 - \frac{\alpha(\sigma^2)}{p_L(\sigma^2)} > 0$, which implies

$$\begin{aligned} & \left(\sigma^2 - \frac{\alpha\left(\sigma^2\right)}{p_L\left(\sigma^2\right)}\right)\left(1-\sigma^2\right) > 0 \\ \Rightarrow \quad & \sigma^2\left(\frac{\alpha\left(\sigma^2\right)}{p_L\left(\sigma^2\right)} + 1 - \sigma^2\right) > \frac{\alpha\left(\sigma^2\right)}{p_L\left(\sigma^2\right)} \\ \Rightarrow \quad & |s| = p_L\left(\sigma^2\right)\sigma^2 > \frac{\alpha\left(\sigma^2\right)}{\frac{\alpha(\sigma^2)}{p_L(\sigma^2)} + 1 - \sigma^2} = \frac{\alpha\left(\sigma^2\right)}{w_L\left(\sigma^2\right)}. \end{aligned} \tag{A.38}$$

Accordingly, we obtain

$$\begin{aligned} r_L\left(\sigma^2, \pm p_L\left(\sigma^2\right)\sigma^2\right) &= \frac{1}{\max\left\{\sigma^2 - \frac{\alpha(\sigma^2)}{p_L(\sigma^2)}, 1-L\right\}} \frac{p_L\left(\sigma^2\right)\sigma^2 - \alpha\left(\sigma^2\right)}{p_L\left(\sigma^2\right)\sigma^2} \\ &= \frac{1}{\sigma^2 - \frac{\alpha(\sigma^2)}{p_L(\sigma^2)}} \frac{p_L\left(\sigma^2\right)\sigma^2 - \alpha\left(\sigma^2\right)}{p_L\left(\sigma^2\right)\sigma^2} \\ &= \frac{p_L\left(\sigma^2\right)\sigma^2 - \alpha\left(\sigma^2\right)}{\left(p_L\left(\sigma^2\right)\sigma^2 - \alpha\left(\sigma^2\right)\right)\sigma^2} = \frac{1}{\sigma^2}. \end{aligned} \tag{A.39}$$

Thus, in total, it holds

$$r_L\left(\sigma^2, \pm p_L\left(\sigma^2\right)\sigma^2\right) = \begin{cases} 1 & \text{if } \sigma^2 \leqslant \frac{\alpha\left(\sigma^2\right)}{L p_L(\sigma^2)}, \\ \frac{1}{1-L}\left(1 - \frac{\alpha\left(\sigma^2\right)}{p_L(\sigma^2)\sigma^2}\right) & \text{if } \frac{\alpha\left(\sigma^2\right)}{L p_L(\sigma^2)} < \sigma^2 \leqslant \frac{\alpha\left(\sigma^2\right)}{p_L(\sigma^2)} + 1 - L, \\ \frac{1}{\sigma^2} & \text{if } \sigma^2 > \frac{\alpha\left(\sigma^2\right)}{p_L(\sigma^2)} + 1 - L. \end{cases} \tag{A.40}$$

ORCID iDs

Clemens Arndt https://orcid.org/0000-0001-5607-4074
Alexander Denker https://orcid.org/0000-0002-7265-261X
Nick Heilenkötter https://orcid.org/0000-0001-7693-8618
Meira Iske https://orcid.org/0009-0009-1198-6358
Tobias Kluth https://orcid.org/0000-0003-4814-142X
Peter Maass https://orcid.org/0000-0003-1448-8345
Judith Nickel https://orcid.org/0000-0001-6599-7540

References

[1] Adler J and Öktem O 2018 Learned primal-dual reconstruction *IEEE Trans. Med. Imaging* **37** 1322–32
[2] Alberti G S, De Vito E, Lassas M, Ratti L and Santacesaria M 2021 Learning the optimal Tikhonov regularizer for inverse problems *Advances in Neural Information Processing Systems* vol 34 pp 25205–16
[3] Arndt C 2022 Regularization theory of the analytic deep prior approach *Inverse Problems* **38** 115005
[4] Arridge S, Maass P, Öktem O and Schönlieb C-B 2019 Solving inverse problems using data-driven models *Acta Numer.* **28** 1–174
[5] Aspri A, Korolev Y and Scherzer O 2020 Data driven regularization by projection *Inverse Problems* **36** 125009
[6] Banert S, Ringh A, Adler J, Karlsson J and Oktem O 2020 Data-driven nonsmooth optimization *SIAM J. Optim.* **30** 102–31
[7] Bauermeister H, Burger M and Moeller M 2020 Learning spectral regularizations for linear inverse problems *NeurIPS 2020 Workshop on Deep Learning and Inverse Problems*
[8] Behrmann J, Grathwohl W, Chen R T, Duvenaud D and Jacobsen J-H 2019 Invertible residual networks *Int. Conf. on Machine Learning* (PMLR) pp 573–82
[9] Benning M and Burger M 2018 Modern regularization methods for inverse problems *Acta Numer.* **27** 1–111
[10] Bora A, Jalal A, Price E and Dimakis A G 2017 Compressed sensing using generative models *Int. Conf. on Machine Learning* (PMLR) pp 537–46

[11] Bungert L, Raab R, Roith T, Schwinn L and Tenbrinck D 2021 Clip: cheap Lipschitz training of neural networks *Scale Space and Variational Methods in Computer Vision: 8th Int. Conf. (SSVM 2021) (Virtual Event, 16–20 May 2021)* (Springer) pp 307–19
[12] Chung J, Chung M and O'Leary D P 2011 Designing optimal spectral filters for inverse problems *SIAM J. Sci. Comput.* **33** 3132–52
[13] Denker A, Schmidt M, Leuschner J and Maass P 2021 Conditional invertible neural networks for medical imaging *J. Imaging* **7** 243
[14] Dittmer S, Kluth T, Maass P and Otero Baguer D 2020 Regularization by architecture: a deep prior approach for inverse problems *J. Math. Imaging Vis.* **62** 456–70
[15] Ebner A, Frikel J, Lorenz D, Schwab J and Haltmeier M 2023 Regularization of inverse problems by filtered diagonal frame decomposition *Appl. Comput. Harmon. Anal.* **62** 66–83
[16] Ebner A and Haltmeier M 2022 Plug-and-play image reconstruction is a convergent regularization method (arXiv:2212.06881)
[17] Engl H W, Hanke M and Neubauer A 1996 *Regularization of Inverse Problems* vol 375 (Springer)
[18] Gregor K and LeCun Y 2010 Learning fast approximations of sparse coding *Proc. 27th Int. Conf. on Machine Learning (ICML '10) (Madison, WI, USA)* (Omnipress) pp 399–406
[19] Hauptmann A, Lucka F, Betcke M, Huynh N, Adler J, Cox B, Beard P, Ourselin S and Arridge S 2018 Model-based learning for accelerated, limited-view 3D photoacoustic tomography *IEEE Trans. Med. Imaging* **37** 1382–93
[20] He J, Wang Y and Ma J 2020 Radon inversion via deep learning *IEEE Trans. Med. Imaging* **39** 2076–87
[21] Jin K H, McCann M T, Froustey E and Unser M 2017 Deep convolutional neural network for inverse problems in imaging *IEEE Trans. Image Process.* **26** 4509–22
[22] Kabri S, Auras A, Riccio D, Bauermeister H, Benning M, Moeller M and Burger M 2022 Convergent data-driven regularizations for CT reconstruction (arXiv:2212.07786)
[23] Kingma D P and Ba J 2015 Adam: a method for stochastic optimization *3rd Int. Conf. on Learning Representations (ICLR 2015) (San Diego, CA, USA, 7–9 May 2015)* ed Y Bengio and Y LeCun
[24] Laumont R, Bortoli V D, Almansa A, Delon J, Durmus A and Pereyra M 2022 Bayesian imaging using plug and play priors: when Langevin meets tweedie *SIAM J. Imaging Sci.* **15** 701–37
[25] LeCun Y 1998 The MNIST database of handwritten digits (available at: http://yann.lecun.com/exdb/mnist/)
[26] Li H, Schwab J, Antholzer S and Haltmeier M 2020 NETT: solving inverse problems with deep neural networks *Inverse Problems* **36** 065005
[27] Louis A K 1989 *Inverse und Schlecht Gestellte Probleme* (Teubner)
[28] Lunz S, Öktem O and Schönlieb C-B 2018 Adversarial regularizers in inverse problems *Advances in Neural Information Processing Systems* vol 31
[29] Mathé P and Hofmann B 2008 How general are general source conditions? *Inverse Problems* **24** 015009
[30] Miyato T, Kataoka T, Koyama M and Yoshida Y 2018 Spectral normalization for generative adversarial networks (arXiv:1802.05957)
[31] Mukherjee S, Dittmer S, Shumaylov Z, Lunz S, Öktem O and Schönlieb C-B 2020 Learned convex regularizers for inverse problems (arXiv:2008.02839)
[32] Mukherjee S, Hauptmann A, Öktem O, Pereyra M and Schönlieb C-B 2023 Learned reconstruction methods with convergence guarantees: a survey of concepts and applications *IEEE Signal Process. Mag.* **40** 164–82
[33] Mukherjee S, Schönlieb C-B and Burger M 2021 Learning convex regularizers satisfying the variational source condition for inverse problems *NeurIPS 2021 Workshop on Deep Learning and Inverse Problems*
[34] Obmann D and Haltmeier M 2023 Convergence analysis of equilibrium methods for inverse problems (arXiv:2306.01421)
[35] Obmann D, Nguyen L, Schwab J and Haltmeier M 2021 Augmented NETT regularization of inverse problems *J. Phys. Commun.* **5** 105002
[36] Oh C, Kim D, Chung J-Y, Han Y and Park H 2018 ETER-net: end to end MR image reconstruction using recurrent neural network *Machine Learning for Medical Image Reconstruction: 1st Int. Workshop, MLMIR 2018, Held in Conjunction with MICCAI 2018 (Granada, Spain, 16 September 2018)* (Springer) pp 12–20
[37] Pesquet J-C, Repetti A, Terris M and Wiaux Y 2021 Learning maximally monotone operators for image recovery *SIAM J. Imaging Sci.* **14** 1206–37

[38] Rieder A 2013 *Keine Probleme mit Inversen Problemen: Eine Einführung in Ihre Stabile Lösung* (Springer)
[39] Ryu E, Liu J, Wang S, Chen X, Wang Z and Yin W 2019 Plug-and-play methods provably converge with properly trained denoisers *Int. Conf. on Machine Learning* (PMLR) pp 5546–57
[40] Schuster T, Kaltenbacher B, Hofmann B and Kazimierski K S 2012 *Regularization Methods in Banach Spaces* (De Gruyter)
[41] Schwab J, Antholzer S and Haltmeier M 2019 Deep null space learning for inverse problems: convergence analysis and rates *Inverse Problems* **35** 025008
[42] Ulyanov D, Vedaldi A and Lempitsky V 2020 Deep image prior *Int. J. Comput. Vis.* **128** 1867–88
[43] Venkatakrishnan S V, Bouman C A and Wohlberg B 2013 Plug-and-play priors for model based reconstruction *2013 IEEE Global Conf. on Signal and Information Processing* (IEEE) pp 945–8
[44] Arndt C, Denker A, Dittmer S, Heilenkötter N, Iske M, Kluth T, Maass P and Nickel J 2023 iResNet Regularization (available at: https://gitlab.informatik.uni-bremen.de/inn4ip/iresnet-regularization)

OPEN ACCESS
IOP Publishing
Inverse Problems
Inverse Problems **40** (2024) 045021 (49pp) https://doi.org/10.1088/1361-6420/ad2aaa

Bayesian view on the training of invertible residual networks for solving linear inverse problems*

Clemens Arndt[1,**], **Sören Dittmer**[1,2], **Nick Heilenkötter**[1], **Meira Iske**[1], **Tobias Kluth**[1] and **Judith Nickel**[1,**]

[1] Center for Industrial Mathematics, University of Bremen, 28359 Bremen, Germany
[2] Cambridge Image Analysis Group, University of Cambridge, Cambridge CB3 0WA, United Kingdom

E-mail: carndt@uni-bremen.de and judith.nickel@uni-bremen.de

Received 1 August 2023; revised 26 January 2024
Accepted for publication 19 February 2024
Published 6 March 2024

Abstract

Learning-based methods for inverse problems, adapting to the data's inherent structure, have become ubiquitous in the last decade. Besides empirical investigations of their often remarkable performance, an increasing number of works address the issue of theoretical guarantees. Recently, Arndt *et al* (2023 *Inverse Problems* **39** 125018) exploited invertible residual networks (iResNets) to learn provably convergent regularizations given reasonable assumptions. They enforced these guarantees by approximating the linear forward operator with an iResNet. Supervised training on relevant samples introduces data dependency into the approach. An open question in this context is to which extent the data's inherent structure influences the training outcome, i.e. the learned reconstruction scheme. Here, we address this delicate interplay of training design and data dependency from a Bayesian perspective and shed light on opportunities and limitations. We resolve these limitations by analyzing reconstruction-based training of the inverses of iResNets, where we show that this optimization strategy introduces a level of data-dependency that cannot be achieved by

* The authors are listed in alphabetical order.
** Authors to whom any correspondence should be addressed.

approximation training. We further provide and discuss a series of numerical experiments underpinning and extending the theoretical findings.

Keywords: iResNet, learned regularization, linear inverse problems, Bayesian inverse problems

1. Introduction

The mathematical field of inverse problems has many applications, e.g. imaging, image processing, and several more. Inverse problems come with characteristic difficulties summarized under the term 'ill-posedness.' Typically, one wants to recover causes x, which discontinuously depend on some observed measurements z. However, a good reconstruction algorithm needs to be stable; otherwise, it cannot handle noisy measurement data. Still, one naturally wants the reconstructor to be as accurate as possible. This results in a compromise called regularization (see [6] for a recent survey on regularization theory). The more stable the reconstructor becomes, the more the set of causes for which it provides accurate results is restricted.

Hence, this set of accurate performance is critical, and one typically chooses it using *prior knowledge* about the application-specific data. In imaging problems, this knowledge often amounts to solutions x looking somehow 'natural.' However, the mathematical characterization of natural images is challenging. Thus, learned methods often outperform in this area, learning stable and accurate reconstructions from given training data (see, e.g. the early survey [4]).

While many experimental studies confirm the impressive performance of learned methods, the theoretical understanding remains limited. In particular, learned methods often lack stability guarantees. However, the topic is gaining in importance [21]. In the present work, we address this issue by studying invertible residual networks (iResNets) [5]. As proposed in [3], their invertibility makes them readily applicable to linear inverse problems. Arndt *et al* [3] approximates the forward operation ($x \mapsto z$) using the iResNet, the iResNet's inverse, then solves the inverse problem ($z \mapsto x$). Here, one can control the regularization strength by choosing a hyperparameter of the iResNet that directly controls its stability.

Arndt *et al* [3] also develop a regularization theory for these iResNets. For this purpose, they considered particular architectures and uncovered equivalences to filter functions from classical regularization theory. In the present article, we now analyze what iResNets actually learn in practice from the given training data. For this purpose, the Bayesian view is suitable, as it encodes prior knowledge on x and the measurement noise in z as probability distributions. We consider two different ways of training, via the forward and via the inverse mapping, and investigate to which extent the iResNet uses the given information about the data to regularize inverse problems.

The manuscript is structured as follows: Section 2 introduces the problem setting, basic assumptions, and the reconstruction approach using iResNets. The subsequent two sections contain the theoretical analysis of the iResNet's training in a Bayesian setting. First, section 3 considers the so-called approximation training, where the network is trained supervisedly to approximate the forward operator. In particular, we investigate what information the network learns from the training data distribution (i.e. the effect of prior distribution and noise on the network). Second, section 4 considers the so-called reconstruction training, where the iResNet's inverse is trained to map from noisy measurements to a reconstruction. Section 5 complements the theoretical analyses with extensive numerical experiments. We implement the two training types and underpin the theoretical findings of the previous sections.

1.1. Related literature

The Bayesian theory for inverse problems differs from the functional analytic regularization theory. While the functional analytic theory focuses mainly on the stability and convergence of regularization operators, the Bayesian perspective considers the full posterior distribution and uncertainty estimation for reconstructions. Detailed introductions are given in [9, 12, 25]. An overview of the basic theory and Bayesian methods for solving inverse problems is also contained in [4]. Learning-based methods are particularly powerful for solving Bayesian inverse problems, e.g. Adler and Öktem [1] describes two different general concepts for an efficient application of neural networks in this framework.

Laumont *et al* [16] proposes a method that demonstrates the Bayesian theory's advantages for inverse problems using a trained denoiser in a plug-and-play Langevin algorithm. The denoiser is assumed to fulfill a Lipschitz condition (similar to the iResNet, see section 2), implying guaranteed convergence of the algorithm to the posterior distribution. Sherry *et al* [24] leverages convex analysis to create nonexpansive residual networks and uses them to solve inverse problems. This is particularly desirable for denoising and plug-and-play schemes. Furthermore, invertible neural networks are also of interest to generative modeling. In [8], iResNets act as normalizing flows, i.e. learn to map from a base distribution to a target distribution and perform competitive or even superior to alternative architectures. Similar to iResNets, [22] also makes use of invertibility and Lipschitz constraints to get a suitable architecture for the use in convergent Gauss-Newton methods. Arndt *et al* [3] provides a more detailed discussion of the literature concerning learned convergent regularization schemes. Similar to our work, [19] also addresses Bayesian analysis of learning forward and inverse problems. But there, the focus is on a certain 2×2-example and a trivial linear network architecture to illustrate some general properties.

2. Problem setting and basic properties

We consider linear inverse problems based on the operator equation

$$Ax = z \tag{2.1}$$

where $A \in L(X,X)$ is a self-adjoint and positive semidefinite operator and X is a finite-dimensional inner product space, here $X = \mathbb{R}^n$. For simplification, we assume $\|A\| = 1$, which a scaling of the operator can easily obtain. In practice, neural network domains tend to be finite-dimensional; this justifies the restriction to the finite-dimensional case. Also, the Bayesian perspective becomes less standard if the underlying probability theory uses infinite-dimensional probability spaces, and the presented theory would require the extension to Bochner integrals. We, however, expect our observations to generalize to the infinite-dimensional case and aim to treat this in future research.

Due to the properties of A, there exist eigenvalues $\sigma_j^2 \in (0,1]$ and corresponding eigenvectors v_j, such that $\mathcal{N}(A)^\perp = \operatorname{span}\{v_j \,|\, j = 1,\dots,\tilde{n}\}$, $\tilde{n} \leqslant n$. We use this eigendecomposition in some of our theoretical analyses.

The aim is to recover the unknown ground truth vector $x^\dagger$ as well as possible by only having access to a noisy observation $z^\delta = Ax^\dagger + \eta$. The noise η is assumed to be distributed according to a probability density function (pdf) $p_H \colon X \to \mathbb{R}_{\geqslant 0}$. Since there may exist arbitrarily many solutions x which could explain the data z^δ, it is important to incorporate prior knowledge about the unknown solutions. The pdf $p_X \colon X \to \mathbb{R}_{\geqslant 0}$ encodes this knowledge. In practice, p_X may describe the distribution of natural-looking images or the typical structure of a cross-section of the human body (e.g. in CT problems).

To solve the inverse problem (2.1), we use the approach of [3], i.e. we approximate the forward operator A with a (single-layer) invertible residual network (iResNet)

$$\varphi_\theta = \mathrm{Id} - f_\theta, \tag{2.2}$$

where $f_\theta : X \to X$ is some residual function modeled as a (small) neural network. This is done by a supervised training of φ_θ for which a paired dataset $\{x^{(i)}, z^{\delta,(i)}\}_{i=1,\dots,N}$ of i.i.d. (independent and identically distributed) samples $x^{(i)} \sim p_X$, $z^{\delta,(i)} - Ax^{(i)} \sim p_H$ is needed. One can then use the trained network to compute a regularized solution of (2.1) by $\varphi_\theta^{-1}(z^\delta)$. Invertibility of φ_θ is guaranteed using the constraint

$$\mathrm{Lip}\,(f_\theta) \leqslant L \tag{2.3}$$

for some $L < 1$, where the inverse is stable and fulfills $\mathrm{Lip}(\varphi_\theta^{-1}) \leqslant {}^1\!/\!_{(1-L)}$ (see [3, lemma 2.1], [5]).

Remark 2.1. The assumption of a positive semidefinite forward operator A is due to the fact that the invertibility condition (2.3) can also be interpreted as some kind of monotonicity condition for φ_θ. Thus, φ_θ cannot approximate arbitrary linear operators but in particular positive (semi-)definite ones.

A more general linear inverse problem

$$\tilde{A}x = y \tag{2.4}$$

with an arbitrary linear operator, $\tilde{A} \in L(X, Y)$ (X and Y being different spaces), can be translated into the above setting by considering $A = \tilde{A}^*\tilde{A}$ and $z = \tilde{A}^*y$.

In this case the noise η on z may arise from noise $\tilde{\eta}$ on y via $\eta = \tilde{A}^*\tilde{\eta}$. To illustrate this, let us consider the example of Gaussian noise $\tilde{\eta} \sim \mathcal{N}(0, \Sigma)$. Then, it holds $\eta = \tilde{A}^*\tilde{\eta} \sim \mathcal{N}(0, \tilde{A}^*\Sigma\tilde{A})$, which means that $\tilde{A}^*$ transforms the covariance matrix Σ. If $\tilde{A}$ has a nontrivial nullspace, the distribution of $\tilde{A}^*\tilde{\eta}$ is singular, and there exists no pdf. Nevertheless, it is possible to approximate the distribution, e.g. by adding $\varepsilon\,\mathrm{Id}$ to the covariance matrix or restricting the problem to $\mathcal{N}(\tilde{A})^\perp$.

While implicit knowledge about p_X and p_H via the given dataset is sufficient for training φ_θ, we derive some theoretical results using these pdfs explicitly. For this purpose, we need to make the following assumptions.

Assumption 2.1. Let

- $p_X \colon X \to \mathbb{R}_{\geqslant 0}$ be a pdf $\left(\text{i.e. } \int_X p_X(x)\,\mathrm{d}x = 1\right)$ with existing first and second moments (i.e. $p_X(x)\|x\|$ and $p_X(x)\|x\|^2$ are Lebesgue-integrable) and expected value

$$\mu_X = \int_X p_X(x)\,x\,\mathrm{d}x, \tag{2.5}$$

- $p_H \colon X \to \mathbb{R}_{\geqslant 0}$ be a pdf $\left(\text{i.e. } \int_X p_H(\eta)\,\mathrm{d}\eta = 1\right)$ with existing first and second moments (i.e. $p_H(\eta)\|\eta\|$ and $p_H(\eta)\|\eta\|^2$ Lebesgue-integrable) and zero expectation $\left(\text{i.e. } \int_X p_H(\eta)\eta\,\mathrm{d}\eta = 0\right)$, and
- the random variables $x \sim p_X$, $\eta \sim p_H$ be stochastically independent.

The crucial condition to guarantee the invertibility of φ_θ is $\mathrm{Lip}(f_\theta) \leqslant L < 1$. Consequently, the inverse $\psi_\theta = \varphi_\theta^{-1}$ fulfills a property describable as a combination of coercivity and Lipschitz-continuity, which, in turn, trivially implies strong monotonicity. We formulate this equivalence in the following lemma.

Lemma 2.1 (inverse of iResNet). *For $\varphi : X \to X$ and $0 \leqslant L < 1$, the following two conditions are equivalent:*

(1) $\exists f\colon X \to X$ *with* $\mathrm{Lip}(f) \leqslant L$ *such that* $\varphi = \mathrm{Id} - f$
(2) $\exists \psi : X \to X$ *with*

$$\forall z_1, z_2 \in X\colon \qquad \left(1 - L^2\right) \|\psi(z_1) - \psi(z_2)\|^2 + \|z_1 - z_2\|^2 \leqslant 2\langle z_1 - z_2, \psi(z_1) - \psi(z_2)\rangle \tag{2.6}$$

such that $\varphi = \psi^{-1}$.

In particular, (2.6) guarantees the invertibility of ψ.

Proof. We begin with (1) $\Rightarrow$ (2). For arbitrary $x_1, x_2 \in X$, the condition $\mathrm{Lip}(\mathrm{Id} - \varphi) \leqslant L$ implies

$$\begin{aligned} &\|(x_1 - \varphi(x_1)) - (x_2 - \varphi(x_2))\|^2 \leqslant L^2 \|x_1 - x_2\|^2 \\ \Leftrightarrow\ &\|x_1 - x_2\|^2 - 2\langle x_1 - x_2, \varphi(x_1) - \varphi(x_2)\rangle + \|\varphi(x_1) - \varphi(x_2)\|^2 \leqslant L^2 \|x_1 - x_2\|^2 \\ \Leftrightarrow\ &\left(1 - L^2\right)\|x_1 - x_2\|^2 + \|\varphi(x_1) - \varphi(x_2)\|^2 \leqslant 2\langle x_1 - x_2, \varphi(x_1) - \varphi(x_2)\rangle. \end{aligned} \tag{2.7}$$

Since $\mathrm{Lip}(f) \leqslant L$ implies invertibility of φ (see [3, lemma 2.1], [5]), we can define $\psi = \varphi^{-1}$ and $z_i = \varphi(x_i)$. This yields

$$\left(1 - L^2\right)\|\psi(z_1) - \psi(z_2)\|^2 + \|z_1 - z_2\|^2 \leqslant 2\langle z_1 - z_2, \psi(z_1) - \psi(z_2)\rangle \tag{2.8}$$

for arbitrary $z_1, z_2 \in X$.

For the converse implication, we now prove that (2.6) guarantees the invertibility of ψ. Injectivity and Lipschitz continuity follow directly by applying the Cauchy–Schwarz inequality to (2.6), which yields

$$\|z_1 - z_2\|^2 \leqslant 2\|z_1 - z_2\|\|\psi(z_1) - \psi(z_2)\|, \tag{2.9}$$

$$\left(1 - L^2\right)\|\psi(z_1) - \psi(z_2)\|^2 \leqslant 2\|z_1 - z_2\|\|\psi(z_1) - \psi(z_2)\|. \tag{2.10}$$

To prove surjectivity, we construct a convergent sequence (z_k) such that $\psi(z_k)$ converges to an arbitrary $x \in X$. We recursively define

$$z_{k+1} = z_k + \left(1 - L^2\right)(x - \psi(z_k)), \qquad z_0 \in X. \tag{2.11}$$

It can be observed that

$$\begin{aligned} 2\langle x - \psi(z_k), \psi(z_{k+1}) - \psi(z_k)\rangle &= 2\langle z_{k+1} - z_k, \psi(z_{k+1}) - \psi(z_k)\rangle \frac{1}{1 - L^2} \\ &\geqslant \|\psi(z_{k+1}) - \psi(z_k)\|^2 + \frac{1}{1 - L^2}\|z_{k+1} - z_k\|^2 \\ &= \|\psi(z_{k+1}) - \psi(z_k)\|^2 + \frac{1}{1 - L^2}\|\left(1 - L^2\right)(x - \psi(z_k))\|^2 \\ &= \|\psi(z_{k+1}) - \psi(z_k)\|^2 + \left(1 - L^2\right)\|x - \psi(z_k)\|^2 \end{aligned} \tag{2.12}$$

holds. Using this, it follows

$$\begin{aligned}\|x-\psi(z_{k+1})\|^2 &= \|(x-\psi(z_k))-(\psi(z_{k+1})-\psi(z_k))\|^2\\ &= \|x-\psi(z_k)\|^2 - 2\langle x-\psi(z_k),\psi(z_{k+1})-\psi(z_k)\rangle + \|\psi(z_{k+1})-\psi(z_k)\|^2\\ &\leqslant \|x-\psi(z_k)\|^2 - \|\psi(z_{k+1})-\psi(z_k)\|^2 - \left(1-L^2\right)\|x-\psi(z_k)\|^2 + \|\psi(z_{k+1})-\psi(z_k)\|^2\\ &= L^2\|x-\psi(z_k)\|^2. \end{aligned} \tag{2.13}$$

Thus, we have $\|x-\psi(z_{k+1})\| \leqslant L\|x-\psi(z_k)\|$, which implies $\|x-\psi(z_k)\| \leqslant L^k\|x-\psi(z_0)\|$. Hence, it holds $\psi(z_k)\to x$ and (2.6) guarantees convergence of (z_k). Since x was arbitrary, ψ is surjective and therefore invertible. With the argumentation from the beginning in reversed order, we obtain the implication $(2)\Rightarrow(1)$. □

The following remark simplifies condition (2.6) for $X=\mathbb{R}$.

Remark 2.2. In case of $X=\mathbb{R}$ (one-dimensional space), condition (2.6) becomes

$$\forall z_1,z_2\in\mathbb{R}: \qquad \frac{1}{1+L} \leqslant \frac{\psi(z_1)-\psi(z_2)}{z_1-z_2} \leqslant \frac{1}{1-L}, \tag{2.14}$$

which is a constraint on the slope of ψ from above and from below.

This motivates us to think of the condition on ψ as a Lipschitz constraint similar to the one that applies to an iResNet. The following remark shows a direct connection between the iResNet and its inverse.

Remark 2.3 (inverses of iResNets are iResNets). From lemma 2.1, we can deduce that one can write the inverse of an iResNet as a scaled iResNet. The constraint (2.6) is equivalent to

$$\begin{aligned}\left(1-L^2\right)^2\|\psi(z_1)-\psi(z_2)\|^2 - 2\left(1-L^2\right)\langle z_1-z_2,\psi(z_1)-\psi(z_2)\rangle + \|z_1-z_2\|^2 \leqslant L^2\|z_1-z_2\|^2\\ \Leftrightarrow \left\|\left(\mathrm{Id}-\left(1-L^2\right)\psi\right)(z_1)-\left(\mathrm{Id}-\left(1-L^2\right)\psi\right)(z_2)\right\| \leqslant L\|z_1-z_2\|\\ \Leftrightarrow \mathrm{Lip}\left(\mathrm{Id}-\left(1-L^2\right)\psi\right)\leqslant L. \end{aligned} \tag{2.15}$$

By defining $g := \mathrm{Id}-(1-L^2)\psi$ we obtain

$$\psi = \frac{1}{1-L^2}(\mathrm{Id}-g) \quad \text{where } \mathrm{Lip}(g)\leqslant L, \tag{2.16}$$

which is a scaled iResNet $\mathrm{Id}-g$ where g satisfies the same Lipschitz constraint as f in the forward mapping.

3. Approximation training

In [3], the *approximation training* is introduced, in which the iResNet φ_θ is trained to approximate A, i.e. to solve

$$\min_{\theta\in\Theta_L}\frac{1}{N}\sum_{i=1}^{N}\left\|\varphi_\theta\left(x^{(i)}\right)-z^{\delta,(i)}\right\|^2 \tag{3.1}$$

for a given dataset of N pairs $(x^{(i)}, z^{\delta,(i)}) \in X \times X$, $z^{\delta,(i)} = Ax^{(i)} + \eta^{(i)}$. The parameter space Θ_L encodes the architecture choice, and the Lipschitz constraint $\mathrm{Lip}(f_\theta) \leqslant L$. This setting was partly motivated by the so-called local approximation property ([3, theorem 3.1]) characterizing convergence guarantees for the regularized solution $\varphi_\theta^{-1}(z^\delta)$ as $\delta \to 0$. In [3], specific network architectures were trained according to the approximation training and analyzed under which conditions they satisfy the properties of a convergent regularization scheme. This revealed a connection to the classical linear filter-based regularization theory.

In contrast, we now aim to derive more general results without making restrictions on the architecture of the iResNet apart from the constraint on the Lipschitz constant of f. This enables us to analyze the influence of the noise and prior distribution on the trained network and, especially, the regularized solution. To this end, we consider the case of an infinite amount of training data, allowing us to interpret equation (3.1) from a Bayesian point of view. To be more precise, taking the limit $N \to \infty$ in equation (3.1) and exploiting the independence of x and η (assumption 2.1) results in

$$\min_{\theta \in \Theta_L} \mathbb{E}_{x \sim p_X} \mathbb{E}_{\eta \sim p_H} \left(\|\varphi_\theta(x) - Ax - \eta\|^2 \right). \tag{3.2}$$

The Euclidian norm can be decomposed into $\|\varphi_\theta(x) - Ax - \eta\|^2 = \|\varphi_\theta(x) - Ax\|^2 - 2\langle \varphi_\theta(x) - Ax, \eta \rangle + \|\eta\|^2$. Again, because of the independence of x and η and due to $\mathbb{E}_{p_H}(\eta) = 0$, the mixed term vanishes in expectation. Therefore, we obtain

$$\begin{aligned} & \min_{\theta \in \Theta_L} \mathbb{E}_{x \sim p_X} \left(\|\varphi_\theta(x) - Ax\|^2 + \mathbb{E}_{\eta \sim p_H} \left(\|\eta\|^2 \right) \right) \\ \Leftrightarrow \quad & \min_{\theta \in \Theta_L} \mathbb{E}_{x \sim p_X} \left(\|\varphi_\theta(x) - Ax\|^2 \right). \end{aligned} \tag{3.3}$$

Consequently, the noise does not influence the training. We could interpret this positively since the noise cannot lead to approximation errors of φ_θ. However, a big drawback is that φ_θ^{-1}, which shall regularize the inverse problem, neither depends on the noise level. Accordingly, the amount of regularization has to be set manually by choice of L for the noise level δ (see [3]) and is not data-dependent.

What remains is the influence of the prior distribution p_X on the training of φ_θ. We are especially interested in how φ_θ acts on the different eigenspaces of A to analyze the dependence on the size of the eigenvalues. Therefore, we make the rather strong assumption of stochastic independence of the components $x_j = \langle x, v_j \rangle$:

Assumption 3.1. Let $x_j \sim p_{X,j}$ with $p_X(x) = \prod_j p_{X,j}(x_j)$.

Observe that this assumption is implicitly made, for example, when using Tikhonov regularization with $\|\cdot\|^2$-penalty term. Furthermore, assumption 2.1 implies that $p_{X,j}$ has existing first and second moments with

$$\mu_{X,j} = \int_{\mathbb{R}} p_{X,j}(x_j)\, x_j \, \mathrm{d}x, \tag{3.4}$$

which follows from Fubini's theorem and the independence of the components. In this setting, a diagonal structure of the network

$$f_\theta(x) = \sum_j f_{j,\theta}\left(\langle x, v_j \rangle \right) v_j \quad \text{with} \quad f_{j,\theta} \colon \mathbb{R} \to \mathbb{R}, \tag{3.5}$$

with respect to eigenvectors v_j of A, which was also used in [3], is sufficient to account for the structure of the distribution according to assumption 3.1. Hence, the above minimization

problem can be analyzed for each component separately due to properties of the eigendecomposition, and we get

$$\min_{\theta\in\Theta_L} \mathbb{E}_{x_j\sim p_{X,j}} \left(\left| \left(1-\sigma_j^2\right) x_j - f_{j,\theta}\left(x_j\right) \right|^2 \right). \tag{3.6}$$

This is equivalent to a 1d-setting with $A\colon \mathbb{R}\to\mathbb{R}$, $x_j \mapsto \sigma_j^2 x_j$.

In the following, instead of minimizing over a parameter space Θ_L, we directly consider a function space $\mathcal{F}$ encoding the Lipschitz constraint ($\mathrm{Lip}(f)\leqslant L$) and the architecture choice. For simplicity, in what follows, we omit the index j and consider

$$\min_{f\in\mathcal{F}} \int_{\mathbb{R}} p_X(x) \left| \left(1-\sigma^2\right) x - f(x) \right|^2 \mathrm{d}x. \tag{3.7}$$

If $\mathcal{F}$ allows for (affine) linear functions and in case of $1-\sigma^2\leqslant L$, we can indicate the trivial solution $f=(1-\sigma^2)\,\mathrm{Id}$. Obviously, this solution is unique on $\mathrm{supp}(p_X)$. Thus, for eigenvalues σ^2, which are not too small, the training leads to a perfect approximation of the forward operator and no regularization of the inverse problem. For $1-\sigma^2>L$, the minimization problem gets more interesting due to the Lipschitz constraint. First, we derive the following result, which builds the basis for a subsequent generalization.

Lemma 3.1. *Let* $\mathcal{F}=\{f\in C(\mathbb{R}) \,|\, \exists m\in[-L,L], b\in\mathbb{R}\colon\ f(x)=mx+b\}$ *and* $L<1-\sigma^2$. *Then,*

$$f^*(x) = Lx + \left(1-\sigma^2-L\right)\mu_X \tag{3.8}$$

is the unique solution of the minimization problem (3.7).

Proof. The minimizer can be calculated by using the necessary KKT conditions. A detailed proof can be found in appendix A.1. □

The previous lemma provides the prerequisite for the following theorem, where $\mathcal{F}$ contains arbitrary Lipschitz continuous functions with constrained Lipschitz constant.

Theorem 3.1. *Let* $\mathcal{F}=\{f\in C^{0,1}(\mathbb{R}) \,|\, \mathrm{Lip}(f)\leqslant L\}$, *where* $C^{0,1}$ *denotes the Hölder space of Lipschitz continuous functions. Then,*

$$f^*(x) = \begin{cases} \left(1-\sigma^2\right)x & \text{if } 1-\sigma^2\leqslant L, \\ Lx+\left(1-\sigma^2-L\right)\mu_X & \text{if } 1-\sigma^2>L \end{cases} \tag{3.9}$$

is the solution of the minimization problem (3.7). This solution is unique on $\mathrm{supp}(p_X)$ *and for* $1-\sigma^2>L$ *even on the convex hull of* $\mathrm{supp}(p_X)$.

Proof. We define $F\colon \mathcal{F} \to \mathbb{R}$,

$$F(f) = \int_{\mathbb{R}} p_X(x) \,|\, \left(1-\sigma^2\right) x - f(x) \,|^2 \,\mathrm{d}x \tag{3.10}$$

and start with the case $1-\sigma^2 \leqslant L$. Obviously, it holds $F(f^*) = 0$, so f^* is a minimizer. Using the fundamental lemma of the calculus of variations, one can deduce the uniqueness on $\operatorname{supp}(p_X)$.

Now, consider $1-\sigma^2 > L$ and let $g \in C^{0,1}(\mathbb{R})$, $\operatorname{Lip}(g) \leqslant L$ be an arbitrary function. We will show that $F(g) > F(f^*)$ holds, if $g \neq f^*$ on the convex hull of $\operatorname{supp}(p_X)$.

First, we verify that F is well-defined, i.e, for $f \in \mathcal{F}$

$$\begin{aligned}
F(f) &\leqslant 2 \int_{\mathbb{R}} p_X(x) \left(\left(1-\sigma^2\right)^2 x^2 + |f(x)|^2 \right) \mathrm{d}x \\
&= 2 \int_{\mathbb{R}} p_X(x) \left(\left(1-\sigma^2\right)^2 x^2 + |f(x) - f(0) + f(0)|^2 \right) \mathrm{d}x \\
&\leqslant 2 \int_{\mathbb{R}} p_X(x) \left(1-\sigma^2\right)^2 x^2 \,\mathrm{d}x + 4 \int_{\mathbb{R}} p_X(x) \,|f(x) - f(0)|^2 \,\mathrm{d}x + 4 \int_{\mathbb{R}} p_X(x) \,|f(0)|^2 \,\mathrm{d}x \\
&\leqslant 2 \left(1-\sigma^2\right)^2 \int_{\mathbb{R}} p_X(x) x^2 \,\mathrm{d}x + 4L^2 \int_{\mathbb{R}} p_X(x) x^2 \,\mathrm{d}x + 4 f(0)^2 < \infty
\end{aligned} \tag{3.11}$$

holds, as the second moment of p_X exists.

Due to $L < 1-\sigma^2$, the function g has always a smaller slope than $(1-\sigma^2)\,\mathrm{Id}$, which implies that there exists an intersection point x_0 such that $g(x_0) = (1-\sigma^2)\, x_0$. The affine linear function $\tilde{f}(x) = L(x-x_0) + (1-\sigma^2)\, x_0$ possesses the same intersection point.

In case of $g = \tilde{f}$ on the convex hull of $\operatorname{supp}(p_X)$, we simply apply lemma 3.1. This shows that g can be the minimizer only if $\tilde{f} = f^*$.

In the case of $g \neq \tilde{f}$, let us examine the integrand of $F(g)$. For any $x \in \mathbb{R}$, it holds

$$\begin{aligned}
|\left(1-\sigma^2\right) x - g(x)|^2 &= |\left(1-\sigma^2\right) x - \left(g(x) - \tilde{f}(x)\right) - \tilde{f}(x)|^2 \\
&= |\left(1-\sigma^2\right) x - \tilde{f}(x)|^2 - 2\left(\left(1-\sigma^2\right) x - \tilde{f}(x)\right)\left(g(x) - \tilde{f}(x)\right) \\
&\quad + |g(x) - \tilde{f}(x)|^2.
\end{aligned} \tag{3.12}$$

For $x \leqslant x_0$, we have $(1-\sigma^2)x - \tilde{f}(x) \leqslant 0$ and $\operatorname{Lip}(g) \leqslant L$ implies $g(x) - \tilde{f}(x) \geqslant 0$. Thus, we obtain

$$-2\left(\left(1-\sigma^2\right) x - \tilde{f}(x)\right)\left(g(x) - \tilde{f}(x)\right) \geqslant 0, \tag{3.13}$$

which implies $|(1-\sigma^2)x - g(x)|^2 \geqslant |(1-\sigma^2)x - \tilde{f}(x)|^2$. Analogously, for $x \geqslant x_0$, we observe that $(1-\sigma^2)x - \tilde{f}(x) \geqslant 0$ and $g(x) - \tilde{f}(x) \leqslant 0$, which also implies $|(1-\sigma^2)x - g(x)|^2 \geqslant |(1-\sigma^2)x - \tilde{f}(x)|^2$. Therefore, it holds $F(g) \geqslant F(\tilde{f})$.

Finally, we show that $F(g) = F(\tilde{f})$ implies $\tilde{f} = g$ on the convex hull of $\operatorname{supp}(p_X)$. If $F(g) = F(\tilde{f})$, it holds

$$\int_{\Omega} p_X(x) \,|\, \left(1-\sigma^2\right) x - g(x) \,|^2 - p_X(x) \,|\, \left(1-\sigma^2\right) x - \tilde{f}(x) \,|^2 \,\mathrm{d}x = 0. \tag{3.14}$$

for any measurable $\Omega \subset \mathbb{R}$, since the term under the integral is always greater than or equal to zero. The fundamental lemma of the calculus of variations then implies

$$p_X(x) \,|\, \left(1-\sigma^2\right) x - g(x) \,|^2 = p_X(x) \,|\, \left(1-\sigma^2\right) x - \tilde{f}(x) \,|^2 \tag{3.15}$$

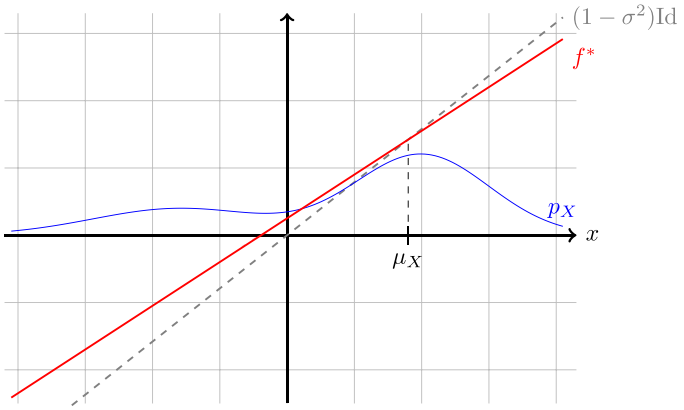


Figure 1. The residual function f^* which results from the approximation training (theorem 3.1) is affine linear and only depends on σ^2, L and μ_X. In case of $\sigma^2 < 1-L$, f^* exhibits the maximum possible slope of L and intersects $(1-\sigma)^2\,\mathrm{Id}$ at the mean μ_X of the prior distribution.

for all $x \in \mathbb{R}$ (since g and $\tilde{f}$ are continuous). Thus, for any $x \in \mathrm{supp}(p_X)$, it holds $g(x) = \tilde{f}(x)$ and for $x_1, x_2 \in \mathrm{supp}(p_X)$, we obtain $g(x_1) - g(x_2) = L(x_1 - x_2)$. Consequently, for any x in between of x_1 and x_2, $g(x) = \tilde{f}(x)$ must also hold, otherwise $\mathrm{Lip}(g) \leqslant L$ would be violated. Hence, g and $\tilde{f}$ coincide on the convex hull of $\mathrm{supp}(p_X)$. □

Figure 1 exemplifies the solution f^* for a Gaussian mixture prior p_X. The inverse φ_θ^{-1} corresponding to the minimizer of (3.7) derived in the previous theorem provides a convergent regularization scheme, which we discuss in the following remark.

Remark 3.1. Due to the affine linear structure of f^*, one can express φ_θ^{-1} as an affine filter-based regularization scheme. The affine linear diagonal architecture was already analyzed in [3, lemma 4.2], i.e. for

$$f_j(x_j) = \min\left\{1-\sigma_j^2, L\right\} x_j + \max\left\{0, 1-\sigma_j^2-L\right\} \mu_{X,j} \tag{3.16}$$

(which coincides with the solution f^* in (3.9) in theorem 3.1), it holds

$$\varphi_\theta^{-1}(z) = \hat{b}_L + \sum_j \hat{r}_L\left(\sigma_j^2\right) \langle z, v_j \rangle v_j, \tag{3.17}$$

$$\hat{r}_L\left(\sigma_j^2\right) = \frac{1}{\max\left\{\sigma_j^2, 1-L\right\}}, \quad \hat{b}_L = \sum_{\sigma_j^2 < 1-L} \frac{1-\sigma_j^2-L}{1-L} \mu_{X,j} v_j. \tag{3.18}$$

By [3, lemma 3.3], this filter scheme with bias defines a convergent regularization method for $L \to 1$ in case of vanishing noise and a suitable parameter choice $L(\delta)$.

The previous results show that approximation training of a diagonal architecture always leads to an affine linear φ_θ, independent of prior and noise distribution (p_X, p_H). Hence, an affine linear residual layer is the best architecture choice for this task. This implies that φ_θ^{-1} is a reconstruction scheme with minimal data dependency since only the mean μ_X of the prior distribution has an influence. Furthermore, φ_θ^{-1} is equivalent to a classical regularization scheme, where one predefines the amount of regularization by choosing the parameter L depending on the noise level.

For the general approximation training problem

$$\min_{f\in C(\mathbb{R}^n,\mathbb{R}^n),\ \mathrm{Lip}(f)\leqslant L} \mathbb{E}_{x\sim p_X}\left(\|\left(\mathrm{Id}-f\right)(x)-Ax\|^2\right) \tag{3.19}$$

the previous investigations suggest that the solution depends on the second moments of the prior distribution p_X at most. A detailed consideration of the general setting for the approximation training is beyond the scope of the present work.

4. Reconstruction training

The results in the last section show that the approximation training of iResNets is capable to provide a convergent regularization but it turns out that it is insufficient for learning a noise- and more data-dependent regularization. To address this, we instead consider the training

$$\min_{\theta\in\Theta}\frac{1}{N}\sum_{i=1}^{N}\left\|\varphi_\theta^{-1}\left(Ax^{(i)}+\eta^{(i)}\right)-x^{(i)}\right\|^2 \quad \text{s.t. } \mathrm{Lip}\left(f_\theta\right)\leqslant L \tag{4.1}$$

for given training data $\{x^{(i)}\}_i\subset X$, noise realizations $\{\eta^{(i)}\}_i\in X$ and $\varphi_\theta=\mathrm{Id}-f_\theta$. This is also motivated by sufficient conditions for the convergence analysis in [3, remark 4.1]. We refer to this training scheme as the *reconstruction training*. One can also interpret this reconstruction training as a supervised training on data pairs $(x^{(i)},z^{\delta,(i)})$ for $\varphi_\theta^{-1}(z^{\delta,(i)})\approx x^{(i)}$ with $z^{\delta,(i)}=Ax^{(i)}+\eta^{(i)}$.

Using lemma 2.1, we know that

$$\begin{aligned}&\min_{\theta\in\Theta}\frac{1}{N}\sum_{i=1}^{N}\left\|\psi_\theta\left(Ax^{(i)}+\eta^{(i)}\right)-x^{(i)}\right\|^2\\ &\text{s.t. } \left(1-L^2\right)\|\psi_\theta\left(z_1\right)-\psi_\theta\left(z_2\right)\|^2+\|z_1-z_2\|^2\leqslant 2\langle\psi_\theta\left(z_1\right)-\psi_\theta\left(z_2\right),z_1-z_2\rangle \quad \forall z_1,z_2\in X\end{aligned} \tag{4.2}$$

is an equivalent problem, assuming that the architectures of φ_θ and ψ_θ can approximate any continuous function.

Similar to the approximation training, we analyze the case of an unlimited amount of training data with $x\sim p_X$ and $\eta\sim p_H$ fulfilling assumption 2.1. Thus, we obtain the minimization problem

$$\min_{\psi\in\Psi}\int_X\int_X p_X(x)\,p_H(\eta)\,\|\psi\left(Ax+\eta\right)-x\|^2\,\mathrm{d}\eta\,\mathrm{d}x, \tag{4.3}$$

where the set of functions Ψ represents the choice of the architecture and the constraints on the parameters. In this context, we also make use of the density function of $z^\delta=Ax+\eta$, which is given by

$$p_Z\left(z^\delta\right)=\int_X p_X(x)\,p_H\left(z^\delta-Ax\right)\,\mathrm{d}x. \tag{4.4}$$

With this, we can define the space of p_Z-weighted L^2-functions as

$$L^2_{p_Z}(X,X) = \{\psi : X \to X \,|\, \|\psi\|_{p_Z,2} < \infty\}, \tag{4.5}$$

$$\|\psi\|^2_{p_Z,2} = \int_X p_Z(z)\, \|\psi(z)\|^2_X \,\mathrm{d}z, \tag{4.6}$$

which is a Hilbert space. Note that functions from $L^2_{p_Z}(X,X)$ are (only) well-defined on $\mathrm{supp}(p_Z) \subset X$, which is sufficient for our purposes.

At first, we consider the unconstrained case of $\Psi = L^2_{p_Z}(X,X)$. In this setting, the conditional mean[3] $\hat{\psi}(z^\delta) = \mathbb{E}(x|z^\delta)$ is the solution of (4.3) which is in line with the established theory in statistical inverse problems (see, e.g. conditional mean estimator in the discussion of Bayes cost estimators in [12] or [1, proposition 2]).

Lemma 4.1. *Let assumption 2.1 hold and* $\Psi = L^2_{p_Z}(X,X)$. *Then,*

$$\hat{\psi} = \left(z^\delta \mapsto \mathbb{E}\left(x|z^\delta\right)\right) = \left(z^\delta \mapsto \int_X p\left(x|z^\delta\right) x \,\mathrm{d}x\right) \tag{4.7}$$

is the solution of (4.3), which is unique w.r.t. the $L^2_{p_Z}$*-norm.*

Proof. The minimization problem (4.3) can be solved via the first-order optimality condition. A detailed proof can be found in appendix A.2. □

Next, we consider the constrained reconstruction training, where we encode an arbitrary constraint, e.g. (4.2), by choosing Ψ to be a suitable subset of $L^2_{p_Z}(X,X)$.

Lemma 4.2. *Let assumption 2.1 hold,* Ψ *be an arbitrary subset of* $L^2_{p_Z}(X,X)$ *and let* $\hat{\psi} : X \to X$, $z^\delta \mapsto \mathbb{E}(x|z^\delta)$ *be the conditional mean estimator. Then, the minimization problem (4.3) is equivalent to*

$$\min_{\psi\in\Psi} \int_X p_Z\left(z^\delta\right) \|\psi\left(z^\delta\right) - \hat{\psi}\left(z^\delta\right)\|^2 \,\mathrm{d}z^\delta. \tag{4.8}$$

Note that the existence of an actual minimizer is only guaranteed for closed Ψ.

Proof. The minimization problem (4.3) is equivalent to

$$\min_{\psi\in\Psi} \int_X \int_X p_X(x)\, p_H\left(z^\delta - Ax\right) \left(\|\psi\left(z^\delta\right) - x\|^2 - \|\hat{\psi}\left(z^\delta\right) - x\|^2\right) \mathrm{d}z^\delta \,\mathrm{d}x. \tag{4.9}$$

In the proof of lemma 4.1, we have already established that the integrals are finite. To split the integral term into two parts, we use

$$\|\psi\left(z^\delta\right) - x\|^2 - \|\hat{\psi}\left(z^\delta\right) - x\|^2 = \|\psi\left(z^\delta\right)\|^2 - \|\hat{\psi}\left(z^\delta\right)\|^2 - \langle 2x, \psi\left(z^\delta\right) - \hat{\psi}\left(z^\delta\right)\rangle. \tag{4.10}$$

Fubini's theorem and the definition of p_Z (4.4) implies

$$\begin{aligned}&\int_X \int_X p_X(x)\, p_H\left(z^\delta - Ax\right) \left(\|\psi\left(z^\delta\right)\|^2 - \|\hat{\psi}\left(z^\delta\right)\|^2\right) \mathrm{d}z^\delta \,\mathrm{d}x \\ &\quad = \int_X p_Z\left(z^\delta\right) \left(\|\psi\left(z^\delta\right)\|^2 - \|\hat{\psi}\left(z^\delta\right)\|^2\right) \mathrm{d}z^\delta.\end{aligned} \tag{4.11}$$

[3] expected value corresponding to $p(x|z^\delta) = \frac{p_H(z^\delta - Ax)p_X(x)}{p_Z(z^\delta)}$.

Again using Fubini's theorem and the definition of $\hat{\psi}(z^\delta) = \mathbb{E}(x|z^\delta)$ (see proof of lemma 4.1), we obtain

$$\begin{aligned}
&\int_X \int_X p_X(x)\, p_H\left(z^\delta - Ax\right) \langle 2x, \psi\left(z^\delta\right) - \hat{\psi}\left(z^\delta\right)\rangle \,\mathrm{d}z^\delta \,\mathrm{d}x \\
&= \int_X \left\langle \int_X p_X(x)\, p_H\left(z^\delta - Ax\right) 2x \,\mathrm{d}x, \psi\left(z^\delta\right) - \hat{\psi}\left(z^\delta\right) \right\rangle \mathrm{d}z^\delta \\
&= \int_X p_Z\left(z^\delta\right) \left\langle 2\int_X \frac{p_X(x)\, p_H\left(z^\delta - Ax\right)}{p_Z\left(z^\delta\right)} x \,\mathrm{d}x, \psi\left(z^\delta\right) - \hat{\psi}\left(z^\delta\right) \right\rangle \mathrm{d}z^\delta \\
&= \int_X p_Z\left(z^\delta\right) \left\langle 2\hat{\psi}\left(z^\delta\right), \psi\left(z^\delta\right) - \hat{\psi}\left(z^\delta\right) \right\rangle \mathrm{d}z^\delta. \qquad (4.12)
\end{aligned}$$

Thus, (4.9) is equivalent to

$$\min_{\psi \in \Psi} \int_X p_Z\left(z^\delta\right) \left(\|\psi\left(z^\delta\right)\|^2 - \|\hat{\psi}\left(z^\delta\right)\|^2 - \left\langle 2\hat{\psi}\left(z^\delta\right), \psi\left(z^\delta\right) - \hat{\psi}\left(z^\delta\right) \right\rangle \right) \mathrm{d}z^\delta. \qquad (4.13)$$

Now, the assertion follows from $\|\psi(z^\delta)\|^2 - \|\hat{\psi}(z^\delta)\|^2 - \langle 2\hat{\psi}(z^\delta), \psi(z^\delta) - \hat{\psi}(z^\delta)\rangle$ $= \|\psi(z^\delta) - \hat{\psi}(z^\delta)\|^2$. □

Thus, in the constraint case, the function ψ^*, obtained by reconstruction training, aims to approximate the conditional mean estimator for the p_Z-weighted L^2-norm. In other words, reconstruction training with a constraint corresponds to a projection of the conditional mean estimator onto the constraint set with respect to the p_Z-weighted L^2-norm.

Remark 4.1. Since we know from remark 2.3 that the inverse network ψ can be interpreted as a scaled iResNet, we can further compare the minimization problem to the case of approximation training. In the notation of an iResNet, the problem formulated in (4.8) is equivalent to

$$\min_{g \in C(\mathbb{R}^n, \mathbb{R}^n),\ \mathrm{Lip}(g) \leqslant L} \int_X p_Z\left(z^\delta\right) \| \left(\mathrm{Id} - g\right)\left(z^\delta\right) - \left(1 - L^2\right) \mathbb{E}\left(x|z^\delta\right) \|^2 \,\mathrm{d}z^\delta. \qquad (4.14)$$

Thus, the reconstruction training is equivalent to training an iResNet with residual function g ($\mathrm{Lip}(g) \leqslant L$) to fit a scaled version of the posterior expectation estimator. In contrast, approximation training aims to fit the same architecture type to the linear operator A. Overall, this indicates that theoretical and numerical properties (such as data-dependence) for the two strategies are the sole consequences of the training approach, and there is no additional bias due to the architecture choice of an iResNet as the forward mapping when assuming sufficient approximation capability.

So far, the distribution of the noise p_H was fixed. Now, we want to consider a variable noise level $\delta > 0$ by introducing the pdf $p_{H,\delta} \colon X \to \mathbb{R}_{\geqslant 0}$. We do not specify the exact relation of $p_{H,\delta}$ on δ but make the rather informal assumption that $\eta^\delta \sim p_{H,\delta}$ implies $\|\eta^\delta\| \sim \delta$ with high probability. So, $\delta \to 0$ corresponds to the case of vanishing noise. Analogously, let $p_{Z,\delta}$ be defined according to (4.4) s.t. $z^\delta \sim p_{Z,\delta}$ holds for $z^\delta = Ax + \eta^\delta$. The posterior mean now also depends on δ and may therefore be defined as

$$\hat{\psi}_\delta(z) = \int_X \frac{p_{H,\delta}(z - Ax)\, p_X(x)}{p_{Z,\delta}(z)} x \,\mathrm{d}x. \qquad (4.15)$$

Further, we want to specify the set $\Psi \subset L^2_{p_Z}(X, X)$, which represents the inverses of possible iResNet architectures depending on δ and L. To encode the side constraint of (4.2), we define

$$\Psi_L^\delta = \left\{ \psi \in L^2_{p_{Z,\delta}}(X,X) \cap C(\operatorname{supp}(p_{Z,\delta}), X) \mid (4.17) \text{ holds } \forall z_1, z_2 \in \operatorname{supp}(p_{Z,\delta}) \right\}, \tag{4.16}$$

$$\left(1 - L^2\right) \|\psi(z_1) - \psi(z_2)\|^2 + \|z_1 - z_2\|^2 \leqslant 2\langle z_1 - z_2, \psi(z_1) - \psi(z_2)\rangle. \tag{4.17}$$

The set Ψ_L^δ is closed and convex in $L^2_{p_{Z,\delta}}(X,X)$ for any $L < 1$ and $\delta > 0$. Consequently, the minimization problem in lemma 4.2 is well-defined and admits a unique solution. This solution, w.r.t. Ψ_L^δ, $p_{Z,\delta}$ and $\hat{\psi}_\delta$, is denoted by $\psi^*_{L,\delta}$.

The parameter L controls the stability of the elements from Ψ_L^δ and can therefore be interpreted as a regularization parameter. The question remains whether we can also obtain a convergence result, i.e. $\psi^*_{L,\delta}(Ax^\dagger + \eta^\delta) \to x^\dagger$ for $\|\eta^\delta\| \leqslant \delta$, $\delta \to 0$ (vanishing noise) and $L \to 1$, analogous to the convergence theorems of classical methods like Tikhonov's. In the following remark, we discuss difficulties and supporting facts regarding a convergence result.

Remark 4.2. There are different results in the literature for the posterior distribution to converge to one single point $x^\dagger$ (or its respective delta distribution) [7, theorems 1 and 2], [26]. Thus, it might be realistic to expect that the conditional mean estimator $\hat{\psi}_\delta(Ax^\dagger)$ also converges to $x^\dagger$. In classical convergence theorems [10], the ground truth $x^\dagger$ is mostly assumed to be a minimum-norm-solution of (2.1). However, in a Bayesian setting with a learned reconstruction scheme, a criterion based on p_X for characterizing $x^\dagger$ is more appropriate, e.g.

$$x^\dagger = \arg\max_{Ax=z} p_X(x) \qquad \text{for } z \in \mathcal{R}(A) \tag{4.18}$$

as in [7] or the one provided in the subsequent lemma 4.3.

Since $\psi^*_{L,\delta}$ is the projection of $\hat{\psi}_\delta$ onto the set Ψ_L^δ (lemma 4.2), it is not unlikely that $\psi^*_{L,\delta}(Ax^\dagger)$ also converges to $x^\dagger$. However, the projection is w.r.t. the $L^2_{p_{z,\delta}}$-norm, and we are actually interested in pointwise convergence. Besides, all functions ψ from the sets Ψ_L^δ are Lipschitz continuous and they fulfill the monotonicity condition

$$\|z_1 - z_2\|^2 \leqslant 2\langle z_1 - z_2, \psi(z_1) - \psi(z_2)\rangle. \tag{4.19}$$

Thus, one cannot approximate arbitrary $L^2_{p_{z,\delta}}$-functions.

There are two facts that partly address this difficulty. First, the posterior mean $\hat{\psi}_\delta$ is also a Lipschitz continuous function under certain assumptions [9, theorem 4.5, remark 4.6]. Second, a generalized inverse on $\mathcal{R}(A)$ of the form $A^\dagger : z \mapsto x^\dagger$ would indeed fulfill (4.19). This follows from A being self-adjoint and positive semidefinite (we can write $A = \tilde{A}^*\tilde{A}$) and $\|A\| = 1$ by

$$\begin{aligned} \|z_1 - z_2\|^2 = \|Ax_1^\dagger - Ax_2^\dagger\|^2 \leqslant \|\tilde{A}^*\|^2 \|\tilde{A}(x_1 - x_2)\|^2 &= \left\langle \tilde{A}\left(x_1^\dagger - x_2^\dagger\right), \tilde{A}\left(x_1^\dagger - x_2^\dagger\right)\right\rangle \\ &\leqslant \left\langle A\left(x_1^\dagger - x_2^\dagger\right), x_1^\dagger - x_2^\dagger\right\rangle = \langle z_1 - z_2, A^\dagger z_1 - A^\dagger z_2\rangle. \end{aligned} \tag{4.20}$$

Assuming that $\psi^*_{L,\delta}$ converges pointwise to $A^\dagger$ on $\mathcal{R}(A)$ would be sufficient to obtain a convergence result for noisy data as well. If $\|z^\delta - z\| \leqslant \delta$ holds and one chooses L in a way that $L \to 1$ and $\frac{\delta}{1-L} \to 0$ for $\delta \to 0$, the desired convergence

$$\begin{aligned} \|\psi^*_{L,\delta}(z^\delta) - A^\dagger(z)\| &\leqslant \|\psi^*_{L,\delta}(z^\delta) - \psi^*_{L,\delta}(z)\| + \|\psi^*_{L,\delta}(z) - A^\dagger(z)\| \\ &\leqslant \frac{1}{1-L}\|z^\delta - z\| + \|\psi^*_{L,\delta}(z) - A^\dagger(z)\| \to 0 \end{aligned} \tag{4.21}$$

would follow, since $\operatorname{Lip}(\psi^*_{L,\delta}) \leqslant \frac{1}{1-L}$ [3, lemma 2.1].

In the following, we provide a result for a potential candidate for $x^\dagger$ for the convergence analysis.

Lemma 4.3. *Let assumption 2.1 hold with $p_{H,\delta} = p_H$ indicating the dependence on δ. In addition, let $\hat{\psi}_\delta : X \to X$, $z^\delta \mapsto \mathbb{E}(x|z^\delta)$ be the conditional mean estimator with*

$$p\left(x|z^\delta\right) = \frac{p_{H,\delta}\left(z^\delta - Ax\right) p_X(x)}{p_{Z,\delta}\left(z^\delta\right)} \tag{4.22}$$

and $p_{Z,\delta}(z^\delta) = \int_X p_X(x)\, p_{H,\delta}(z^\delta - Ax)\,\mathrm{d}x$. We further make the following assumptions:

(i) *Noise on $\mathcal{R}(A)^\perp = \mathcal{N}(A)$ and $\mathcal{R}(A) = \mathcal{N}(A)^\perp$ (as $A = A^*$ and X is finite-dimensional) is stochastically independent, i.e. there exist pdfs $p^\dagger_{H,\delta} : \mathcal{N}(A)^\perp \to \mathbb{R}_{\geqslant 0}$ and $p^0_{H,\delta} : \mathcal{N}(A) \to \mathbb{R}_{\geqslant 0}$ such that $p_{H,\delta}(\eta) = p^0_{H,\delta}(P_{\mathcal{N}(A)}\eta) \cdot p^\dagger_{H,\delta}(P_{\mathcal{N}(A)^\perp}\eta)$,*
(ii) *$p^0_{H,\delta}$ and $p^\dagger_{H,\delta}$ define Dirac sequences with respect to $\delta \to 0$,*
(iii) *p_X is compactly supported and continuous. We define $\mathcal{R}_{p_X}(A) := \{Ax \,|\, x \in \mathrm{supp}(p_X)\} \subset \mathcal{R}(A)$,*
(iv) *For any $z \in \mathcal{R}_{p_X}$ there exists a $\bar{\delta}$ such that for all $\delta \in (0, \bar{\delta}]$ it holds $z \in \mathrm{supp}(p_{Z,\delta})$.*

We then have pointwise convergence of $\hat{\psi}_\delta$ for $z \in \mathcal{R}_{p_X}(A)$ such that it holds

$$\begin{aligned} &\hat{\psi}_\delta(z) \underset{\delta\to 0}{\longrightarrow} A^\dagger z + \int_{\mathcal{N}(A)} p\left(x_0|A^\dagger z\right) x_0 \,\mathrm{d}x_0 \\ &\text{with } p\left(x_0|A^\dagger z\right) = \frac{p_X\left(x_0 + A^\dagger z\right)}{\int_{\mathcal{N}(A)} p_X\left(x_0' + A^\dagger z\right)\,\mathrm{d}x_0'} \end{aligned} \tag{4.23}$$

i.e. it converges to the minimum-norm solution $A^\dagger z$ plus the conditional expectation $\mathbb{E}(x_0|A^\dagger z) \in \mathcal{N}(A)$ in the nullspace.

Proof. The proof can be found in appendix A.3. □

4.1. Reconstruction training for diagonal architecture

In order to derive more specific results for the minimizer of (4.3), we make the assumption of stochastic independence of the components $x_j = \langle x, v_j\rangle \sim p_{X,j}$ and $\eta_j = \langle \eta, v_j\rangle \sim p_{H,j}$ similar to the setting in the last section on the approximation training:

Assumption 4.1. Let $p_X(x) = \prod_j p_{X,j}(x_j)$ and $p_H(\eta) = \prod_j p_{H,j}(\eta_j)$.

Observe that the first and second moments of $p_{X,j}$ and $p_{H,j}$ exist due to assumption 2.1 with

$$\mu_{X,j} = \int_{\mathbb{R}} p_{X,j}(x_j)\, x_j \,\mathrm{d}x_j \qquad \text{and} \qquad \mu_{H,j} = \int_{\mathbb{R}} p_{H,j}(\eta_j)\, \eta_j \,\mathrm{d}\eta_j = 0 \tag{4.24}$$

for all $j \in \mathbb{N}$. In addition, the density of $z_j = \sigma_j^2 x_j + \eta_j$ is given by

$$p_{Z,j}(z_j) = \int_{\mathbb{R}} p_{X,j}(x_j)\, p_{H,j}\left(z_j - \sigma_j^2 x_j\right)\,\mathrm{d}x_j \tag{4.25}$$

with

$$\mu_{Z,j} = \int_{\mathbb{R}} p_{Z,j}(z_j)\, z_j \,\mathrm{d}z_j = \int_{\mathbb{R}}\int_{\mathbb{R}} z_j\, p_{X,j}(x_j)\, p_{H,j}\left(z_j - \sigma_j^2 x_j\right)\,\mathrm{d}x_j\,\mathrm{d}z_j = \sigma_j^2\,\mu_{X,j}. \tag{4.26}$$

Analogously to the approximation training, in this setting, it is sufficient to consider a diagonal structure of φ, which then implies the same structure for ψ, i.e.

$$\psi(z) = \sum_j \psi_j(\langle z, v_j\rangle) v_j \tag{4.27}$$

and the resulting optimization problem to train each ψ_j now reads

$$\min_{\psi_j \in \Psi_j} \int_{\mathbb{R}} \int_{\mathbb{R}} p_{X,j}(x_j)\, p_{H,j}(\eta_j)\, \|\psi_j(\sigma_j^2 x_j + \eta_j) - x_j\|^2 \,\mathrm{d}\eta_j \,\mathrm{d}x_j \tag{4.28}$$

with a suitable set of functions Ψ_j. Recall that $\psi_j : \mathbb{R} \to \mathbb{R}$ represents the inverse of an iResNet if ψ_j satisfies

$$\frac{1}{1+L} \leqslant \frac{\psi_j(z_1) - \psi_j(z_2)}{z_1 - z_2} \leqslant \frac{1}{1-L} \qquad \forall z_1, z_2 \in \mathbb{R} \tag{4.29}$$

for some $0 \leqslant L < 1$, cf remark 2.2. Therefore, we consider the set

$$\Psi_j = \left\{ \psi_j : \mathbb{R} \to \mathbb{R} \,\middle|\, \psi_j(z_j) = m z_j + b \text{ for } z_j \in \mathbb{R} \text{ with } m \in \mathbb{R}, \frac{1}{1+L} \leqslant m \leqslant \frac{1}{1-L}, b \in \mathbb{R} \right\} \tag{4.30}$$

for some $0 \leqslant L < 1$. The following lemma provides a formula for the minimizer of problem (4.28). For better readability, we leave out the index j in the subsequent derivations.

Lemma 4.4. *The unique solution to the minimization problem (4.28) with Ψ_j as in (4.30) is given by*

$$\psi^*(z) = mz + \left(1 - \sigma^2 m\right) \mu_X \quad \text{for } z \in \mathbb{R} \tag{4.31}$$

with

$$m = \begin{cases} \frac{1}{1+L} & \text{if } \frac{\sigma^2 Var_{p_X}(x)}{\sigma^4 Var_{p_X}(x) + Var_{p_H}(\eta)} < \frac{1}{1+L}, \\ \frac{\sigma^2 Var_{p_X}(x)}{\sigma^4 Var_{p_X}(x) + Var_{p_H}(\eta)} & \text{if } \frac{1}{1+L} \leqslant \frac{\sigma^2 Var_{p_X}(x)}{\sigma^4 Var_{p_X}(x) + Var_{p_H}(\eta)} \leqslant \frac{1}{1-L}, \\ \frac{1}{1-L} & \text{if } \frac{\sigma^2 Var_{p_X}(x)}{\sigma^4 Var_{p_X}(x) + Var_{p_H}(\eta)} > \frac{1}{1-L}. \end{cases} \tag{4.32}$$

Proof. The minimization problem (4.28) can be solved using the necessary KKT conditions. A detailed proof can be found in appendix A.4. □

Note that the inverse of the function $\psi^* : \mathbb{R} \to \mathbb{R}$ is given by $\varphi^*(x) = x - f^*(x)$ with

$$f^*(x) = \begin{cases} -Lx + \left(1 + L - \sigma^2\right) \mu_X & \text{if } \frac{\sigma^2 \mathrm{Var}_{p_X}(x)}{\sigma^4 \mathrm{Var}_{p_X}(x) + \mathrm{Var}_{p_H}(\eta)} < \frac{1}{1+L}, \\ \left(1 - \frac{\sigma^4 \mathrm{Var}_{p_X}(x) + \mathrm{Var}_{p_H}(\eta)}{\sigma^2 \mathrm{Var}_{p_X}(x)}\right) x + \frac{\mathrm{Var}_{p_H}(\eta)}{\sigma^2 \mathrm{Var}_{p_X}(x)} \mu_X & \text{if } \frac{1}{1+L} \leqslant \frac{\sigma^2 \mathrm{Var}_{p_X}(x)}{\sigma^4 \mathrm{Var}_{p_X}(x) + \mathrm{Var}_{p_H}(\eta)} \leqslant \frac{1}{1-L}, \\ Lx + \left(1 - L - \sigma^2\right) \mu_X & \text{if } \frac{\sigma^2 \mathrm{Var}_{p_X}(x)}{\sigma^4 \mathrm{Var}_{p_X}(x) + \mathrm{Var}_{p_H}(\eta)} > \frac{1}{1-L} \end{cases} \tag{4.33}$$

for $x \in \mathbb{R}$. In the case of noiseless data, i.e. $\mathrm{Var}_{p_H}(\eta) = 0$, f^* corresponds to the function f^* derived in lemma 3.1 and theorem 3.1.

The function ψ^* plays an important role in the case of Gaussian prior and noise distributions, which we will deal with in the following corollary.

Corollary 4.1. *Assume that $p_X : \mathbb{R} \to \mathbb{R}$ and $p_H : \mathbb{R} \to \mathbb{R}$ are Gaussian probability density functions. Then, the function ψ^* of lemma 4.4 is a solution to the minimization problem (4.28) with*

$$\Psi = \left\{ \psi \in C^1(\mathbb{R}) \cap L^2_{p_Z}(\mathbb{R}) \;\middle|\; \frac{1}{1+L} \leqslant \psi'(z) \leqslant \frac{1}{1-L} \text{ for all } z \in \mathbb{R} \right\}. \quad (4.34)$$

Proof. In lemma 4.1 we have seen that the unconstrained solution of problem (4.28) is given by $\hat{\psi}(z) = \mathbb{E}(x|z)$ for all $z \in \mathbb{R}$. In the case of Gaussian noise and prior distributions, $\mathbb{E}(x|z)$ can be expressed as

$$\mathbb{E}(x|z) = \frac{\sigma^2 \mathrm{Var}_{p_X}(x)}{\sigma^4 \mathrm{Var}_{p_X}(x) + \mathrm{Var}_{p_H}(\eta)} z + \left(1 - \frac{\sigma^4 \mathrm{Var}_{p_X}(x)}{\sigma^4 \mathrm{Var}_{p_X}(x) + \mathrm{Var}_{p_H}(\eta)}\right) \mu_X \quad (4.35)$$

for all $z \in \mathbb{R}$ [25, theorem 6.20 and equation (2.16a)], which is an element of $C^1(\mathbb{R}) \cap L^2_{p_Z}(\mathbb{R})$. In combination with lemma (4.2), minimization problem (4.3) can be rewritten as

$$\min_{\psi \in \Psi} \int_{\mathbb{R}} p_Z(z) \left(\psi(z) - \frac{\sigma^2 \mathrm{Var}_{p_X}(x)}{\sigma^4 \mathrm{Var}_{p_X}(x) + \mathrm{Var}_{p_H}(\eta)} z - \left(1 - \frac{\sigma^4 \mathrm{Var}_{p_X}(x)}{\sigma^4 \mathrm{Var}_{p_X}(x) + \mathrm{Var}_{p_H}(\eta)}\right) \mu_X \right)^2 \mathrm{d}z. \quad (4.36)$$

The same reasoning as in the proof of theorem 3.1 now shows that ψ^* of lemma 4.4 is a solution to the minimization problem. □

Figure 2 illustrates the behavior of the unconstrained solution $\hat{\psi}$ and the constrained solution ψ^* in case of Gaussian probability density functions for varying noise and small singular values ($1 - \sigma^2 > L$). Note that both solutions can be rewritten to depend on μ_Z instead of on μ_X using $\mu_Z = \sigma^2 \mu_X$. It can be observed that the noise level affects the slope of the unconstrained solution, with decreasing values at higher noise levels. Thus, $\hat{\psi}$ violates the invertibility condition (4.29) for very small and very large values of $\mathrm{Var}_{p_H}(\eta)$ leading to $\psi^* \neq \hat{\psi}$ in these cases (left and right image of figure 2).

4.1.1. General behavior of ψ^*. The previous results deal with special cases where either the architecture or the probability density functions are known. In order to derive more general results, we make use of the theory of optimal control. For this, we need to restrict ourselves to piecewise continuously differentiable functions ψ with bounded domain, i.e. we consider the set

$$\Psi = \left\{ \psi \in C^0([z_0, z_1]) \;\middle|\; \psi \text{ piecewise continuously differentiable with } \frac{1}{1+L} \leqslant \psi'(z) \leqslant \frac{1}{1-L} \right\} \quad (4.37)$$

with fixed $z_0, z_1 \in \mathbb{R}$ and $\Pr(z \leqslant z_0)$[4] $\leqslant \varepsilon$, $\Pr(z \geqslant z_1) \leqslant \varepsilon$ for some small $\varepsilon > 0$ to stay close to the previous setting. Furthermore, to apply the optimal control theory, we need to split the optimization problem into two successive minimization problems. First, we minimize over all

[4] Pr denotes the probability w.r.t. $z \sim p_Z$.

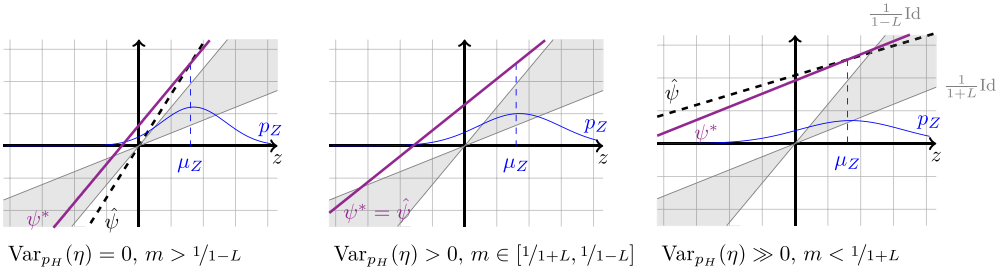


Figure 2. Illustration of the constrained solution ψ^* and the unconstrained solution $\hat{\psi}$ in the case of Gaussian probability density functions p_X and p_H, cf corollary 4.1. The slope of the unconstrained solution $\hat{\psi}$ is denoted by m, i.e. $m = \frac{\sigma^2 \mathrm{Var}_{p_X}(x)}{\sigma^4 \mathrm{Var}_{p_X}(x) + \mathrm{Var}_{p_H}(\eta)}$. The plots exemplify the behavior of ψ^* and $\hat{\psi}$ for small singular values ($1 - \sigma^2 > L$) assuming a fixed prior distribution p_X but increasing variance of p_H. In the case that $\mathrm{Var}_{p_H}(\eta) = 0$ (left), the slope of the unconstrained solution exceeds $\frac{1}{1-L}$. If the noise increases, the slope of the unconstrained solution decreases, resulting in $m \in [\frac{1}{1+L}, \frac{1}{1-L}]$ (middle). For very noisy data, the slope of the unconstrained solution is smaller than $\frac{1}{1+L}$ (right), again resulting in $\psi^* \neq \hat{\psi}$. Observe that ψ^* and $\hat{\psi}$ are equal to $\frac{1}{\sigma^2}\mu_Z = \mu_X$ for $z = \mu_Z = \sigma^2 \mu_X$ in all cases.

functions $\psi \in \Psi$ with fixed starting point $\psi(z_0) = \psi^0 \in \mathbb{R}$. Then, the starting point minimizing the objective function is determined. In combination with lemma 4.2, the minimization problem thus reads

$$\min_{\psi^0 \in \mathbb{R}} \left(\min_{\psi \in \Psi \cap \{\psi \,|\, \psi(z_0) = \psi^0\}} \frac{1}{2} \int_{z_0}^{z_1} p_Z(z) \, |\psi(z) - \hat{\psi}(z)|^2 \, \mathrm{d}z \right). \tag{4.38}$$

We would like to stress that the minimization problem defined in lemma 4.2 is not equivalent to the initial one of equation (4.3) due to the bounded domain of ψ. However, this error is negligible for small ε and the two minimization problems coincide if $\mathrm{supp}(p_Z) \subset [z_0, z_1]$.

Remark 4.3. The restriction to a bounded domain of ψ might seem artificial at first. Nevertheless, in applications, the dataset rarely contains samples belonging to low-density regions of p_Z, and thus, these cases are covered in our setting.

The inner minimization problem can be solved with the help of Pontryagin's maximum principle, resulting in the following necessary and sufficient conditions for the derivative of ψ.

Lemma 4.5. *Let $\psi_0 \in \Psi \cap \{\psi \,|\, \psi(z_0) = \psi^0\}$ be a solution of the minimization problem*

$$\min_{\psi \in \Psi \cap \{\psi \,|\, \psi(z_0) = \psi^0\}} \frac{1}{2} \int_{z_0}^{z_1} p_Z(z) \, |\psi(z) - \hat{\psi}(z)|^2 \, \mathrm{d}z. \tag{4.39}$$

Then, in all points of differentiability, the derivative ψ_0' must satisfy the necessary and sufficient conditions

$$\psi_0'(z) = \begin{cases} \frac{1}{1+L} & \text{if } \lambda(z) > 0 \\ f_0(z) & \text{if } \lambda(z) = 0 \\ \frac{1}{1-L} & \text{if } \lambda(z) < 0 \end{cases} \quad \text{with } z \in [z_0, z_1] \tag{4.40}$$

for some $f_0 : [z_0, z_1] \to \mathbb{R}$ *satisfying*

$$\frac{1}{1+L} \leqslant f_0(z) \leqslant \frac{1}{1-L} \quad \forall z \in [z_0, z_1] \tag{4.41}$$

and $\lambda : [z_0, z_1] \to \mathbb{R}$ *with*

$$\lambda'(z) = -p_Z(z)\left(\psi_0(z) - \hat{\psi}(z)\right) \text{ and } \lambda(z_1) = 0. \tag{4.42}$$

Proof. Let us denote the set of all points $z \in [z_0, z_1]$ where ψ_0 is differentiable by D. For problem (4.39), Pontryagin's maximum principle, see [18, *9.6 theorem 1] and [23, theorem 1], provides the necessary conditions

$$\psi_0'(z) = u_0(z) \quad \text{for all } z \in D \text{ and some piecewise continuous function } u_0 : [z_0, z_1] \to \mathbb{R} \tag{4.43}$$

$$\frac{1}{1+L} \leqslant u_0(z) \leqslant \frac{1}{1-L} \quad \forall z \in [z_0, z_1] \tag{4.44}$$

$$\lambda'(z) = -p_Z(z)\left(\psi_0(z) - \hat{\psi}(z)\right) \text{ with } \lambda(z_1) = 0 \tag{4.45}$$

$$H(\psi_0, u_0, \lambda, z) \leqslant H(\psi_0, u, \lambda, z) \quad \forall u \text{ satisfying (4.44)} \tag{4.46}$$

with the Hamiltonian function

$$H(\psi, u, \lambda, z) = \lambda(z)\, u(z) + \frac{1}{2} p_Z(z)\, |\psi(z) - \hat{\psi}(z)|^2. \tag{4.47}$$

Condition (4.46) is equivalent to setting

$$u_0(z) = \begin{cases} \frac{1}{1+L} & \text{if } \lambda(z) > 0 \\ f_0(z) & \text{if } \lambda(z) = 0 \\ \frac{1}{1-L} & \text{if } \lambda(z) < 0 \end{cases} \qquad \text{for some function } f_0 \text{ satisfying (4.41)}. \tag{4.48}$$

Furthermore, For a function ψ_0 satisfying the conditions of Pontryagin's maximum principle to be a solution of problem (4.39), the Hamiltonian needs to be jointly convex in ψ and u and the constrained set defined by equation (4.44) needs to be convex, see [23, theorem 2]. Both of these conditions are satisfied in our setting, and thus, the proof is complete. □

We would like to remark that λ can be expressed as

$$\lambda(z) = \int_z^{z_1} p_Z(\tilde{z}) \left(\psi_0(\tilde{z}) - \hat{\psi}(\tilde{z})\right) \mathrm{d}\tilde{z}. \tag{4.49}$$

whenever p_Z and $\hat{\psi}$ are continuous. To illustrate the previous lemma, let us look at a very simple example. Assume that $\hat{\psi}'(z) > {}^1\!/_{1-L}$ for all $z \in [z_0, z_1]$. Then, lemma 4.5 in combination with a minimization over the starting points ψ^0 shows that a solution to problem (4.38) is given by

$$\psi^*(z) = \frac{1}{1-L} z + \frac{\int_{z_0}^{z_1} p_Z(\tilde{z})\, \hat{\psi}(\tilde{z})\, \mathrm{d}\tilde{z} - \frac{1}{1-L} \int_{z_0}^{z_1} p_Z(\tilde{z})\, \tilde{z}\, \mathrm{d}\tilde{z}}{\int_{z_0}^{z_1} p_Z(\tilde{z})\, \mathrm{d}\tilde{z}} \quad \text{for } z \in [z_0, z_1]. \tag{4.50}$$

In addition, the general behavior of the solution to problem (4.38) is illustrated in figure 3. This exemplifies that the best possible architecture choice, when considering the reconstruction training loss, is not necessarily an affine linear one, unlike in the approximation training studied in the last section. This is partly because of the influence of the noise distribution p_H, which

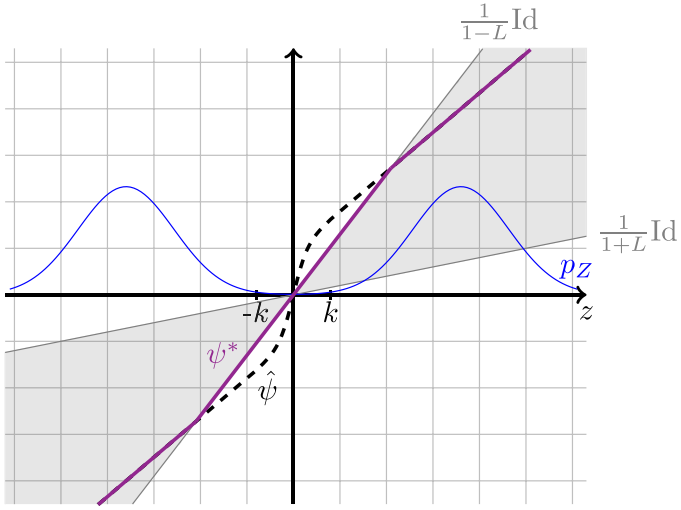


Figure 3. Behavior of the solution ψ^* of (4.38) in the case that p_z can be represented as a Gaussian mixture, cf remark 5.1. Observe that the slope of the unconstrained solution $\hat{\psi}$ exceeds $1/1-L$ in the interval $[-k,k]$ resulting in $\lambda(z)<0$ for $z\in[-k-\varepsilon,k+\varepsilon]$ with $\varepsilon>0$. As a result, ψ^* is equal to $z\mapsto 1/1-Lz$ for $z\in[-k-\varepsilon,k+\varepsilon]$ and to $\hat{\psi}$ outside this interval.

cancels out when using the approximation training loss. Moreover, the variance of the prior and noise distribution influences the best architecture and parameter choice, which is not the case in the approximation training setting, where only the expectation of the prior distribution influences the solution.

5. Numerical experiments

To study the implications of the previously developed theory for the practical application of iResNets for solving inverse problems, we perform experiments on two forward operators, where we compare approximation training (3.1) to reconstruction training (4.1). In all experiments, we train single-layer iResNets with diagonal (3.5) structure where the residual functions f_θ comprise multiple layers.

In the setting presented in the following sections, we consider a discrete convolution with a smoothing kernel $a\in\mathbb{R}^{9\times 9}$ that is depicted in figure 4 and zero padding to preserve dimensionality. Since the resulting Toeplitz matrix M_a that performs the convolution with a is symmetric and positive definite, this serves immediately as a self-adjoint operator $A=M_a$. The second inverse problem we aim to solve is given by a discrete Radon operator $\tilde{A}:\mathbb{R}^{28\times 28}\to\mathbb{R}^{30\times 41}$, such that $A=\tilde{A}^*\tilde{A}$, which is in line with the setting used in the prior work [3]. We restrict the discussion of the numerical results to the convolution operator, and for the sake of completeness, the results for the Radon operator are provided in appendix D in figures 12–16 and table 2.

In both cases, we train our models on the MNIST handwritten digits dataset [17], where we treat images as flattened vectors in $\mathbb{R}^{28\cdot 28}$. In addition, we study an artificially generated bimodal Gaussian dataset for which we sample the prior distribution in every singular vector independently from a (bimodal) Gaussian mixture distribution. The pdf in every j therefore reads

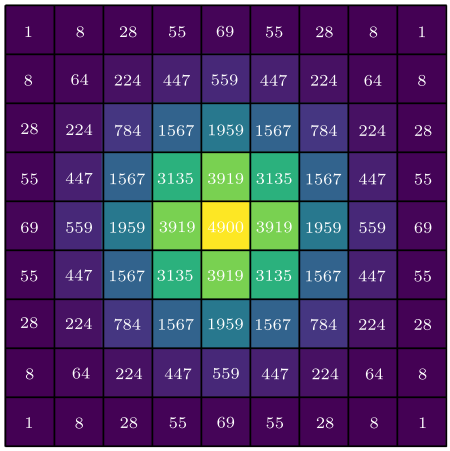


Figure 4. The filter kernel a used in the convolution operator M_a.

$$p_{X,j}(x_j) = \rho_1\, g_{v,t_1}(x_j) + \rho_2\, g_{v,t_2}(x_j) \tag{5.1}$$

where $\rho_1 = 0.35$, $\rho_2 = 0.65$, $v = 0.15$, $t_1 = -0.6$, $t_2 = 0.6$ and $g_{s,t}$ is the pdf of a Gaussian with standard deviation s and expected value t. This bimodal structure enables us to further explore the data dependency of the optimized models.

The architecture of the subnetworks included in the diagonal architecture and their training is realized identical to [3]: Each subnetwork consists of a three-layer fully connected network, equipped with 35 hidden neurons in each of the first two layers and one output neuron. In addition, we apply the ReLU activation function to the first and second layers of each subnetwork. Figure 7 of [3] depicts the architecture. The network weights are optimized using Adam [13] with a learning rate of 0.001, reduced by the factor 0.96 after every epoch. Extending the approach in [20], we parameterize the network weights in the k-th linear layer of all subnetworks to fulfill the Lipschitz constraint $L^{(k)}$. We do so by choosing the weights $W_{j,k}$ as $W_{j,k} = \min(1, L^{(k)}/\mathrm{Lip}(\tilde{W}_{j,k}))\tilde{W}_{j,k}$, where $\tilde{W}_{j,k}$ are unconstrained matrices. We choose the individual layer Lipschitz constants as $L^{(1)} = L^{(2)} = 1$, $L^{(3)} = L$.

Reconstruction training is accomplished by computing the inverse of the iResNets through the usual fixed point iteration and backpropagating through the unrolled iteration to optimize the network weights. To ensure sufficient accuracy of the inverse, especially for the models with a high Lipschitz bound, we need to make a suitable choice for the number of fixed point iterations. For this purpose, we utilize the models that were trained via approximation training. This is motivated by our observation that each of these models reaches its Lipschitz bound and is therefore as unstable as permitted within the constraint. For each L, we select a number of fixed point iterations that results in approximately 2% error of $\varphi^{-1}_{\theta(L)}$ evaluated on the test dataset. As a result, one has to run the iterative inversion in every training step, resulting in a much greater computational effort for the reconstruction training than approximation training. The Lipschitz bound, realized by a proper parameterization of the f_θ, has to be computed only once per iteration.

Of course, a numerically more efficient training approach would be to extend on remark 2.3 and construct ψ, the inverse of an iResNet, as a scaled iResNet that is trained on reversed data points but similar to the approximation approach. However, we currently do not have guarantees on the approximation capability of the involved (fully-connected) iResNets architectures

and their inverses. To provide a fair comparison, we aim to enforce the same inductive bias in both training methods by choosing the same forward mapping architecture. The source code corresponding to the experiments in this section is openly available [27].

In the described settings, we perform experiments for varying noise levels δ_ℓ and Lipschitz constants L_m, where we choose

$$\hat{\delta}_\ell = \begin{cases} \left(\frac{1}{3}\right)^{7-\ell} & \text{for } 0 < \ell < 7 \\ 0 & \text{for } \ell = 0. \end{cases} \qquad \ell = 0, \dots, 6 \tag{5.2}$$

$$L_m = 1 - \left(\frac{1}{3}\right)^m, \qquad m = 1, \dots, 5 \tag{5.3}$$

and the resulting noise η is Gaussian noise with standard deviation $\delta_\ell = \hat{\delta}_\ell \cdot \mathrm{std}_{\mathrm{dataset}}$, where $\mathrm{std}_{\mathrm{dataset}}$ denotes the averaged standard deviation of the coefficients $\mathrm{std}_j := \mathrm{std}(\langle x^{(i)}, v_j \rangle_{i=1,\dots,N})$ of the current dataset (i.e. standard deviation with respect to i, mean with respect to j). The corresponding numbers of fixed point iterations for the chosen error bound are $i_{1,2} = 30, i_3 = 100$, $i_4 = 300$ and $i_5 = 800$. In the following, we discuss the results in terms of the learned solutions, the resulting data-dependent filter functions, and the regularization and approximation properties of the models.

5.1. Learned inverse mappings

To compare the characteristics of the approximation and reconstruction training to our theoretical findings, we plot the learned one-dimensional inverse mappings ψ_j in the different components j (corresponding to the singular values σ_j) for the bimodal dataset. We visualize the results in figure 5 for a large and a small eigenvalue of A.

In the case of approximation training, we observe the predicted affine linear behavior in the support of the data distribution, limited by the Lipschitz constraints and aligned to the expected value of the training data. This result is independent of the noise and optimal only for small noise levels. At the boundaries of the data distribution seen during training, the proper behavior of the optimal solution from theorem 3.1 is not learned properly.

For the case of reconstruction training, we can again corroborate our theoretical findings numerically. For this purpose, we compute the posterior expectation $\mathbb{E}(x_j | z_j^\delta)$ for our setting.

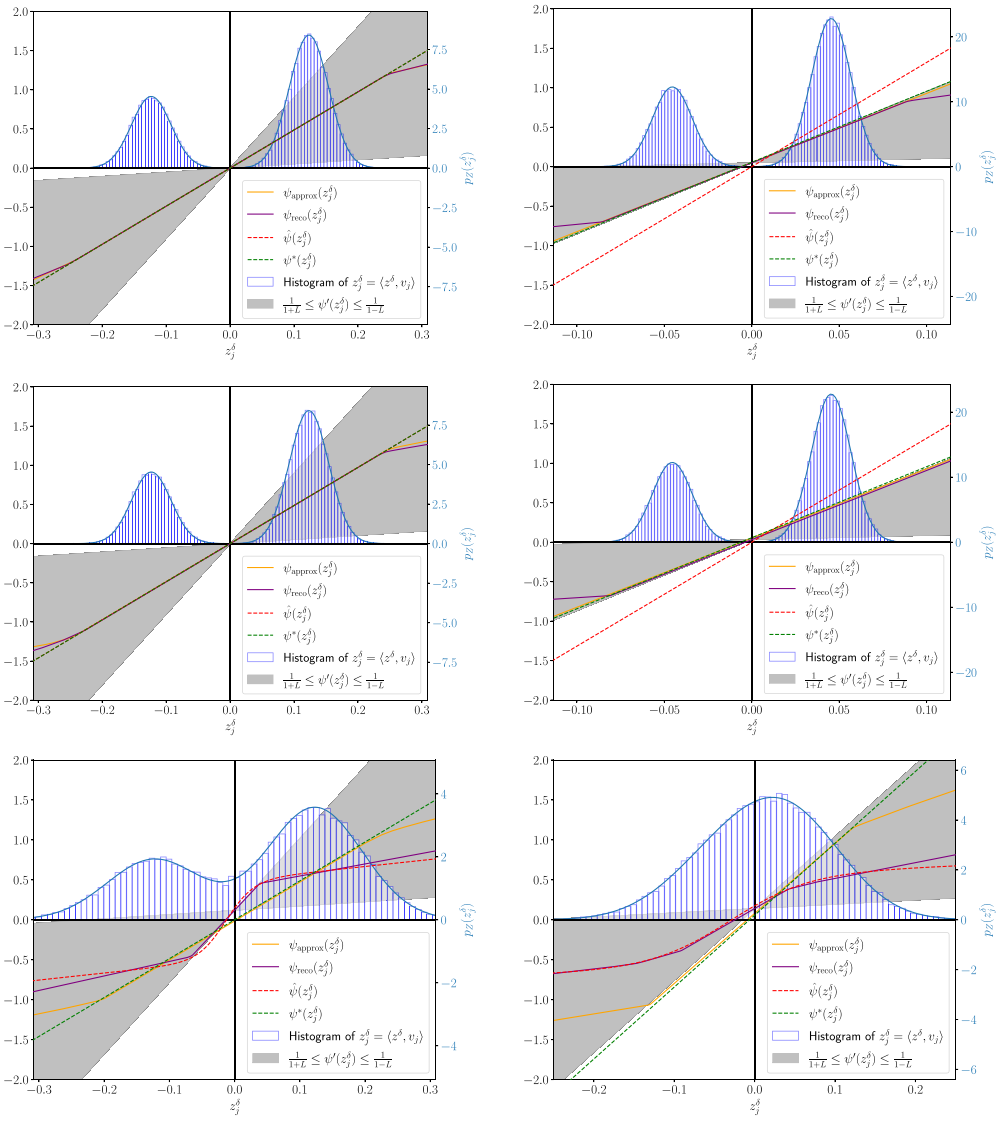
Remark 5.1. For the multimodal Gaussian distribution with pdf p_X and Gaussian noise p_H,

$$p_H(z) = g_{w,0}(z) \tag{5.4}$$

$$p_X(x) = \sum_{k=1}^{K} \rho_k \, g_{v_k, t_k}(x), \tag{5.5}$$

we have

$$p_z(z) = \sum_{k=1}^{K} \rho_k \, g_{u_k, t_k \sigma^2}(z) \tag{5.6}$$



(a) $\sigma_j^2 = 0.205,\ j = 95$ (b) $\sigma_j^2 = 0.075,\ j = 150$

Figure 5. Reconstructions $\psi^*_{\text{approx}}(z_j^\delta)$ trained via approximation training and $\psi^*_{\text{reco}}(z_j^\delta)$ trained via reconstruction training at Lipschitz bound L_2 for different singular values and for noise levels 'zero' (δ_0, *top row*), 'small' (δ_1, *middle row*) and 'large' (δ_5, *bottom row*) for $A = M_a$.

and the posterior expectation $\hat{\psi}$ reads

$$\begin{aligned}\mathbb{E}(x|z) &= \frac{\sum_{k=1}^{K} \frac{\rho_k}{u_k^2} \left(\sigma^2 v_k^2 z + t_k w^2\right) g_{u_k, t_k\sigma^2}(z)}{\sum_{k=1}^{K} \rho_k g_{u_k, t_k\sigma^2}(z)} \\ &= \sigma^2 z \frac{\sum_{k=1}^{K} \frac{v_k^2 \rho_k}{u_k^2} g_{u_k, t_k\sigma^2}(z)}{\sum_{k=1}^{K} \rho_k g_{u_k, t_k\sigma^2}(z)} + w^2 \frac{\sum_{k=1}^{K} \frac{t_k \rho_k}{u_k^2} g_{u_k, t_k\sigma^2}(z)}{\sum_{k=1}^{K} \rho_k g_{u_k, t_k\sigma^2}(z)},\end{aligned} \tag{5.7}$$

where $u_k = \sqrt{w^2 + (\sigma^2 v_k)^2}$. We note that this recovers the linear behavior $\mathbb{E}(x|z) = z/\sigma^2$ in the noise-free case while it adds a correction term in the noisy case that pulls and pushes data points towards more likely results.

For regions within the support of the data distribution, where the constraint (2.6) permits the model to approximate $\mathbb{E}(x_j|z_j^\delta)$, we observe in figure 5 that the learned solutions match well with the posterior expectation. If the model reaches the limiting constraint, it exhausts the possible slope to be as close to the posterior expectation as possible. This results in a much more data-dependent inversion scheme, where reconstructions that were more likely to appear during training are favored. Consequently, the model can compensate for larger noise levels based on additional learned knowledge about the data. The behavior of the learned solution thus coincides with the theoretically founded one in figure 3.

In the case of large noise, the reconstruction-based model regularizes and does not necessarily exhaust the Lipschitz constraint, while the approximation model always tries to fit the operator as well as possible. If noise is absent, the learned mappings coincide for both training strategies.

5.2. Learned filter functions

As a link to classical regularization theory, we visualize the data-dependent filter functions that correspond to the learned models. For this purpose, we evaluate the filter r_L, where

$$(\mathrm{Id} - f_{\theta,j})^{-1}(s(q)) - \hat{b}_{L,j} = r_L\left(\sigma_j^2, s(q)\right) s(q) \text{ for } s \in \mathbb{R}, \tag{5.8}$$

for each singular value σ_j at data points $s(q) := \sigma_j^2(\mu_{X,j} + q \cdot \mathrm{std}_j)$, where we subtract the axis intercept $\hat{b}_{L,j} = (\mathrm{Id} - f_{\theta,j})^{-1}(0)$. The variable $q \in \mathbb{R}$ determines the number of standard deviations std_j away from the mean value $\mu_{X,j} = \frac{1}{N}\sum_{i=1}^{N} \langle x^{(i)}, v_j \rangle$ in the image of the dataset with respect to the fixed singular value of the operator. For simplicity, we define

$$R_L(\sigma_j, q) := r_L\left(\sigma_j^2, s(q)\right) s(q). \tag{5.9}$$

The results for approximation and reconstruction training are visualized as surface plots for both datasets in figures 6 and 11. In all cases, the r_L show a sensible behavior, damping small singular values and roughly satisfying $r_L \to 1$ as $\sigma^2 \to 1$.

On the bimodal dataset in figure 6, data dependency occurs in all filter functions. In the case of approximation training, this is visible only in the sense that proper regularizing behavior is learned exclusively within the support of the data distribution. Reconstruction training shows a more complex dependency since the posterior mean aims to push points that lie close to 0 towards the two peaks of the bimodal data distribution if noise is present in the data, as can also be seen in figure 5. As a result, the medium singular values, in which an amplified slope in the solution is, on the one hand, permitted by the Lipschitz constraint and, on the other hand, also necessary due to the present influence of noise, are elevated near the center of

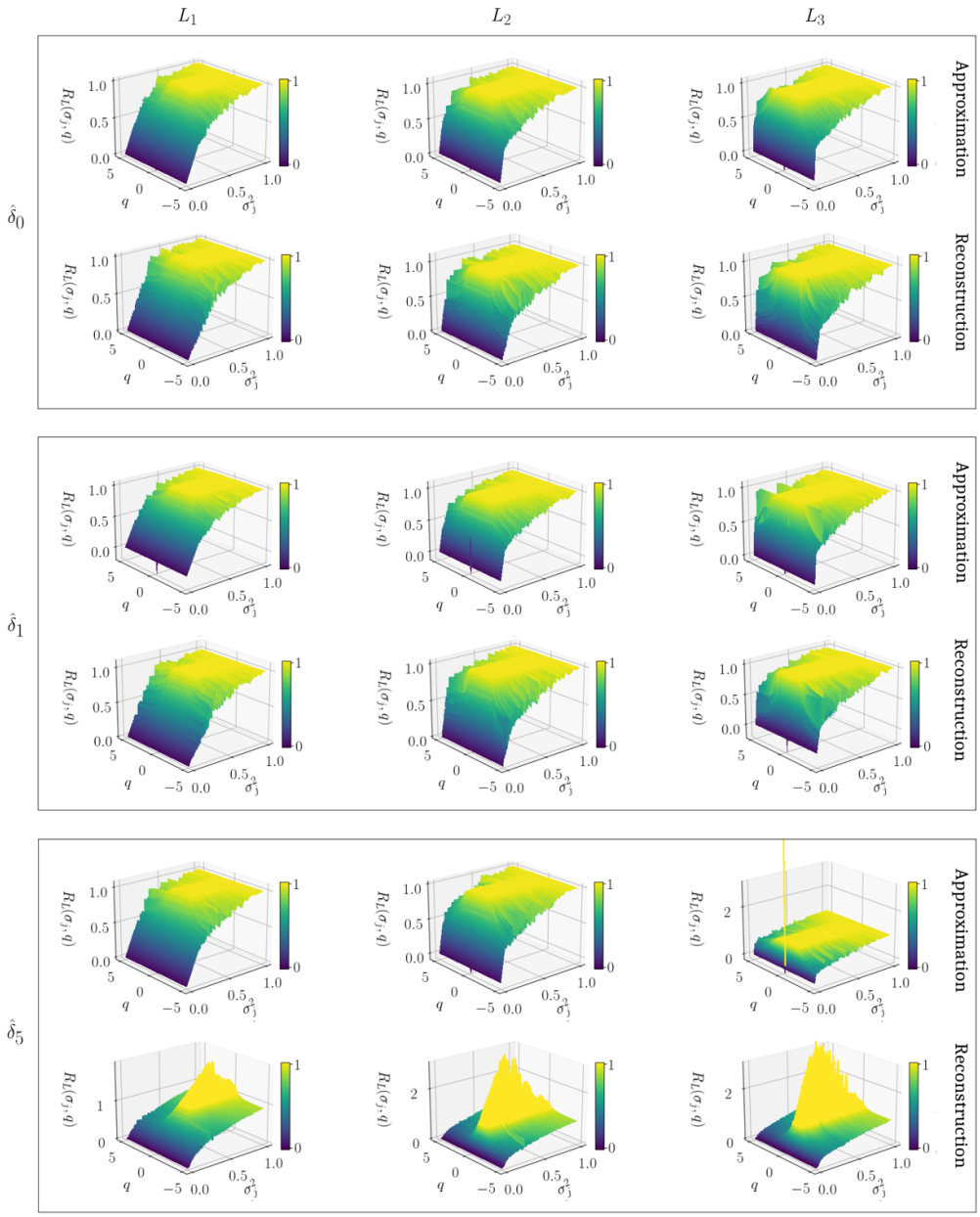


Figure 6. Filter functions $R_L(\sigma_j, q)$ as defined in (5.9) corresponding to trained networks $\varphi_{\theta(L_m),\delta_\ell}$ for $m = 1, 2, 3$ *(columns)* and $\ell = 0, 1, 5$ *(rows)*, trained via approximation training *(top)* and via reconstruction training *(bottom)* on the bimodal dataset for $A = M_a$.

the distribution. This results in filter functions that may not lead to convergent regularization schemes but include data-driven corrections for the observed noise.

On MNIST in figure 11, the stronger data dependence of the reconstruction training is not directly visible; all filters appear to be approximately constant in the range of 5 standard deviations around the mean. This is likely to be due to the fact that the distributions in the singular values are all approximately unimodal. Therefore, the correction could be similar to the simple regularizing behavior of the neural networks in the approximation training approach. In return, the action of L as a regularization parameter becomes visible. Especially for small L, the filter functions show similarities to the expected filter function; see remark 3.1. The learned filter functions in approximation training are very similar for every noise level, which is in line with the developed theory. In contrast, the filters learned in reconstruction training show stronger regularization (i.e. more dampening of small singular values) for larger noise levels, adapting to the data seen during training. This again fits well within the theoretical results developed in section 4 and is especially visible for large $L = L_3$, where the model is, in principle, able to fit the operator well. However, the filter functions for the largest noise $\hat{\delta}_5$ and L_2, L_3 look very similar, indicating that the model learns strong regularization from data at the cost of a worse operator approximation.

5.3. Reconstruction quality and convergence

To compare the performance of the different models in terms of reconstruction quality, we show images and filter functions for a single MNIST digit in figure 7.

Especially for large noise, the visual quality clearly benefits from the reconstruction training compared to the approximation approach. This supports the argument that additional data dependence may be desirable for large-noise applications. In the same case, the approximation training shows its regularization properties and indicates proper parameter choice rules: The image quality improves for smaller Lipschitz constraints, which imply a strong regularization. This behavior is not visible in the reconstruction approach, where a large L generally seems to improve the quality. The filter plots we provide for the given sample in figure 7(right) also underlined this. In this case, they show a similar graph for L_2 and L_3. The filter plots also reveal that the models optimized via approximation training are independent of the noise seen during training and, therefore, learn identical filter functions for all noise levels. Table 1 also reveals an advantage of this method for small noise levels.

Another way to study the convergence in L and δ of the trained models is to evaluate the approximation properties in terms of the overall errors on the dataset with respect to different error measures. To evaluate the localized approximation property that has been introduced in [3, theorem 3.1], we define

$$\mathcal{E}_{\text{mean}}\left(\varphi_{\theta(L)}, A\right) = \frac{1}{N}\sum_{i=1}^{N}\left\|\varphi_{\theta(L)}\left(x^{(i)}\right) - Ax^{(i)}\right\|, \tag{5.10}$$

$$\mathcal{E}_{x^{(m)}}\left(\varphi_{\theta(L)}, A\right) = \left\|\varphi_{\theta(L)}\left(x^{(m)}\right) - Ax^{(m)}\right\|, \tag{5.11}$$

estimating the approximation error of the trained model for the whole dataset or a single sample. In figure 8, we plot this error for the models trained without noise and varying L. In the case of approximation training, we observe slightly superlinear convergence in the dataset on average, indicating that the property is satisfied for many samples. As proven in prior work [3], this implies that the training constructs a convergent regularization scheme for these samples. For reconstruction-based training, this is not fulfilled on average. To preserve some

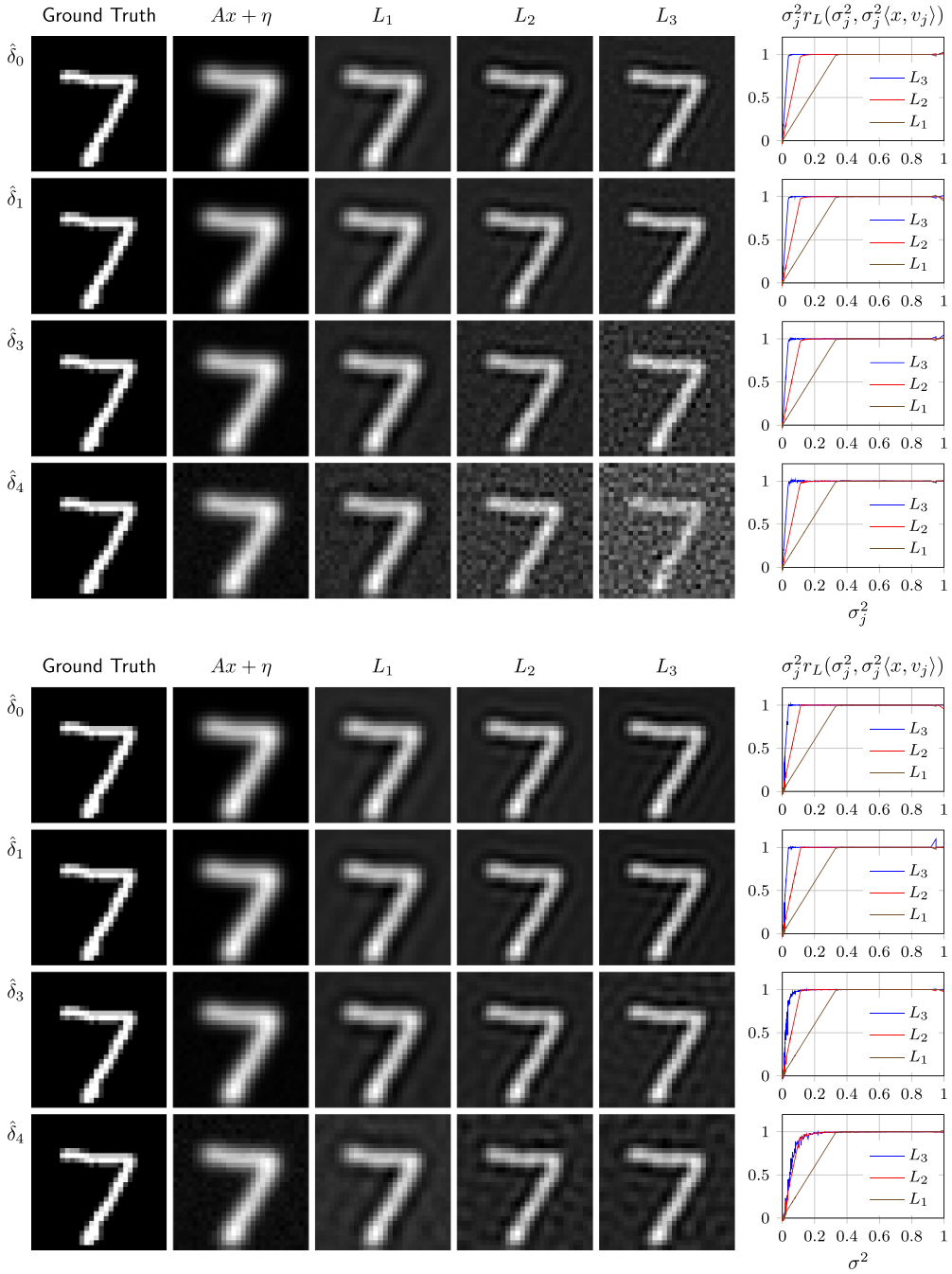


Figure 7. Reconstructions of an MNIST sample $x = x^{(1)}$ from the test dataset by computing $\varphi^{-1}_{\theta(L_m,\delta_\ell)}(Ax+\tilde{\eta})$ with $\tilde{\eta} \sim \mathcal{N}(0, \delta_\ell \mathrm{Id})$ for Lipschitz bounds L_m with $m = 1,2,3$ *(columns)* and noise levels $\delta_\ell = \hat{\delta}_\ell \cdot \mathrm{std}_{\mathrm{MNIST}}$ with $\ell = 0,1,3,4$ *(rows)* together with corresponding filter functions for $A = M_a$. The top subfigure depicts the reconstructions from networks trained via approximation training, and the bottom subfigure corresponds to the networks optimized via reconstruction training.

Table 1. SSIM and MSE measures corresponding to reconstructions of $x^{(1)}$ in figure 7. Bold values indicate the best reconstruction quality with respect to the corresponding error measure for a given noise level.

	SSIM			MSE		
Approximation training	L_1	L_2	L_3	L_1	L_2	L_3
δ_0	0.7401	0.7984	**0.8106**	0.0101	0.0058	**0.0037**
δ_1	0.7377	0.7930	**0.8219**	0.0103	0.0056	**0.0036**
δ_3	0.7261	**0.7455**	0.6575	0.0105	**0.0063**	0.0094
δ_4	**0.6791**	0.6236	0.5085	**0.0112**	0.0132	0.0469
	SSIM			MSE		
Reconstruction training	L_1	L_2	L_3	L_1	L_2	L_3
δ_0	0.7417	0.8046	**0.8255**	0.0102	0.0057	**0.0040**
δ_1	0.7403	0.8051	**0.8240**	0.0103	0.0056	**0.0039**
δ_3	0.7336	0.7774	**0.7961**	0.0105	0.0061	**0.0046**
δ_4	0.7208	**0.7242**	0.7096	0.0106	0.0070	**0.0066**

insights on the convergence of the method, we extend on the weaker but sufficient condition in [3, remark 3.2] and define

$$\widetilde{\mathcal{E}}_{\text{mean}}\left(\varphi_{\theta(L)},A\right)=\frac{1}{N}\sum_{i=1}^{N}\left\|x^{(i)}-\varphi_{\theta(L)}^{-1}\left(Ax^{(i)}\right)\right\| \tag{5.12}$$

$$\widetilde{\mathcal{E}}_{x^{(m)}}\left(\varphi_{\theta(L)},A\right)=\left\|x^{(m)}-\varphi_{\theta(L)}^{-1}\left(Ax^{(m)}\right)\right\|, \tag{5.13}$$

which is closer to the target in reconstruction training. In this case, $\widetilde{\mathcal{E}}_{x^{(m)}}(\varphi_{\theta(L)},A)\xrightarrow{L\to 1}0$ would be sufficient for local convergence. Figure 8 indicates that this property can still be satisfied, however, with slow convergence rates.

In addition, we evaluate the reconstruction error of the training approaches for varying noise levels. Figure 9 depicts the results for the mean squared error

$$\text{MSE}_{\text{reco}}^{\delta_\ell}\left(\varphi_{\theta(L,\delta_\ell)},A\right)=\frac{1}{N}\sum_{i=1}^{N}\left\|x^{(i)}-\varphi_{\theta(L,\delta_\ell)}^{-1}\left(Ax^{(i)}+\eta^{(i)}\right)\right\|^2 \tag{5.14}$$

and averaged structural similarity index measure (SSIM) as defined in [28], computed for the dataset by

$$\text{SSIM}^{\delta_\ell}\left(\varphi_{\theta(L,\delta_\ell)},A\right)=\frac{1}{N}\sum_{i=1}^{N}\text{SSIM}\left(x^{(i)},\varphi_{\theta(L,\delta_\ell)}^{-1}\left(Ax^{(i)}+\eta^{(i)}\right)\right). \tag{5.15}$$

Here, one can again see that the approximation training comes with a typical parameter choice rule known from regularization theory, where one has to choose $L\to 1$ while $\delta\to 0$. In contrast, the reconstruction training performs best with the largest L among all noise levels since the regularization emanates from the data. Overall, the reconstruction training method outperforms the approximation training for all large δ but stays slightly behind for very small noise.

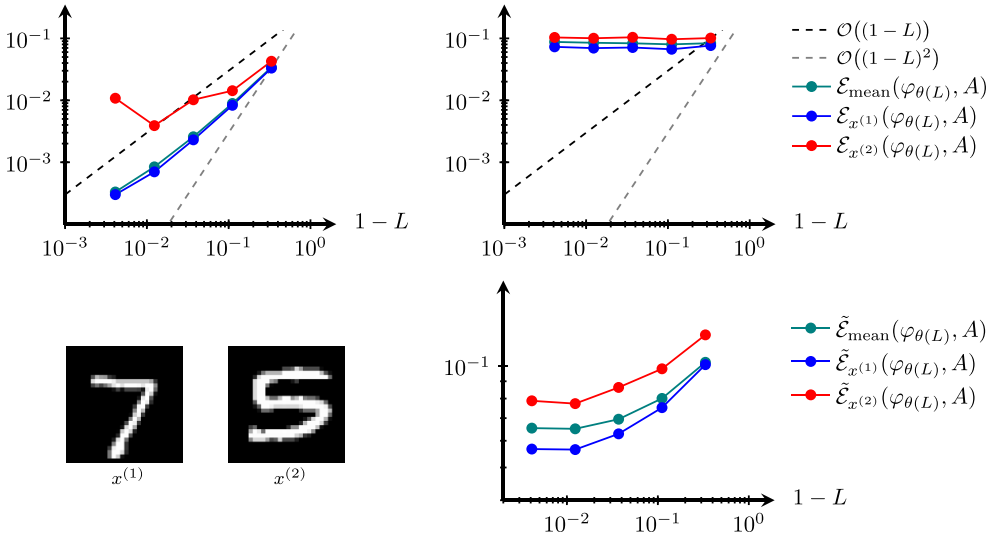


Figure 8. Test samples $x^{(1)}$ and $x^{(2)}$ *(bottom left)*. Evaluations of the local approximation property via $\mathcal{E}_{\mathrm{mean}}(\varphi_{\theta(L_m)},A)$, $\mathcal{E}_{x^{(1)}}(\varphi_{\theta(L_m)},A)$ and $\mathcal{E}_{x^{(2)}}(\varphi_{\theta(L_m)},A)$ for the approximation training *(top left)* and the reconstruction training *(top right)*, and evaluations of the generalized approximation property via $\tilde{\mathcal{E}}_{\mathrm{mean}}(\varphi_{\theta(L_m)},A)$, $\tilde{\mathcal{E}}_{x^{(1)}}(\varphi_{\theta(L_m)},A)$ and $\tilde{\mathcal{E}}_{x^{(2)}}(\varphi_{\theta(L_m)},A)$ for the reconstruction training *(bottom right)* for $L_m = 1 - 1/3^m$ with $m = 1,\dots,5$, $A = M_a$ on the MNIST test dataset.

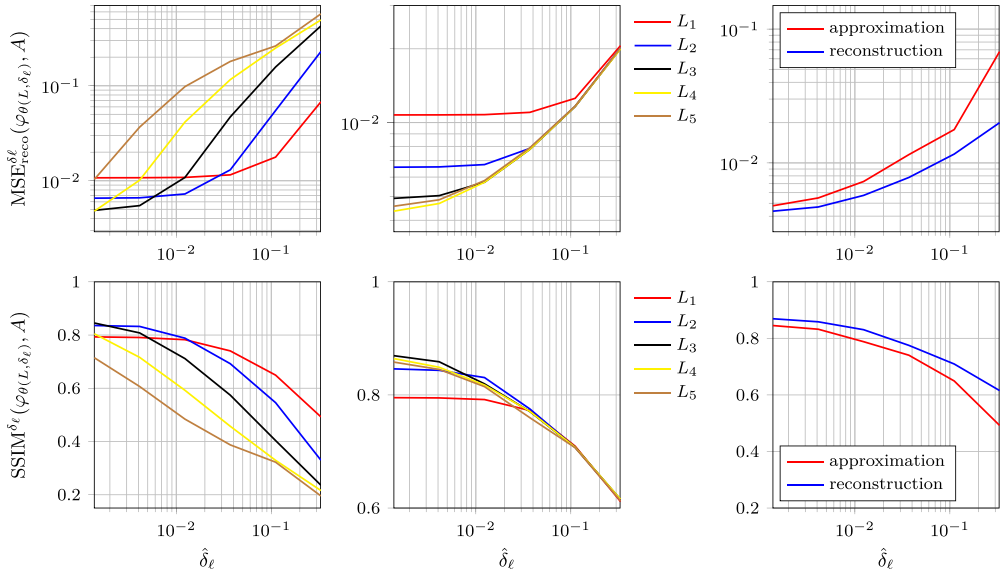


Figure 9. Reconstruction errors $\mathrm{MSE}^{\delta_\ell}_{\mathrm{reco}}(\varphi_{\theta(L,\delta_\ell)},A)$ *(top row)* and $\mathrm{SSIM}^{\delta_\ell}(\varphi_{\theta(L,\delta_\ell)},A)$ *(bottom row)* for networks trained on noisy samples with noise levels δ_ℓ for $\ell = 0,\dots,6$ and reconstructions from noisy samples of the same noise level for the approximation training *(left)* and for the reconstruction training *(middle)* with Lipschitz bounds L_m on the MNIST dataset for $A = M_a$. Outcomes of optimal parameter choices for both training strategies over different noise levels can be seen on the right-hand side.

6. Discussion and outlook

The present work can be seen as a continuation of [3]. There, the authors investigated regularization properties of the proposed iResNet reconstruction approach for specific network architectures trained according to the approximation training on samples to impart data dependency. Here, we have extended the theory by focusing on the question of to what extent the training data distribution influences the optimal parameters of the iResNet and, in turn, the resulting reconstruction scheme. To this end, we considered the training of the iResNets from a Bayesian perspective and focused on two different loss functions, namely the approximation training and the reconstruction training.

In the approximation training, our results for the diagonal architecture show that for all possible prior and noise distributions, the best-suited residual function f is an affine linear one whose optimal parameters depend on the mean of the prior distribution, the eigenvalue σ_j^2 and the Lipschitz constant $L < 1$. Thus, in this setting, the data dependency on the training outcome is minimal, with no influence from the noise distribution and, especially, the regularization properties of the resulting reconstruction scheme. Instead, the amount of regularization of φ_θ^{-1} is solely controlled by the Lipschitz constant L, which also becomes apparent by the observed equivalence of φ_θ^{-1} to a convergent filter-based regularization scheme with bias.

In contrast, the prior and noise distributions significantly impact the optimal architecture and parameters when considering the reconstruction training. Here, we realize the network by an iResNet's inverse and train it to approximate A^{-1}, resulting in a stable reconstruction scheme. We showed that we can interpret the optimal network as an approximation of the conditional mean estimator $z^\delta \mapsto \mathbb{E}(x|z^\delta)$ w.r.t. the p_Z-weighted L^2-norm, where p_Z is the density function of $Ax + \eta$. Hence, the optimal architecture choice and the corresponding optimal parameters depend on the prior and the noise distribution. Consequently, this indicates that the amount of regularization of the trained network is controlled by the Lipschitz constant L and possibly by the amount of noise in the training data.

The theoretical findings are validated and further corroborated by a series of numerical investigations on the MNIST dataset and an artificially generated dataset following a bimodal Gaussian distribution for two different forward operators. In particular, the results highlight that in reconstruction training, the noise distribution influences the regularization properties of the network. In the approximation training, the noise does not influence the regularization properties; they are solely controlled by the Lipschitz constant L. As a result, the reconstruction training leads to superior regularizations in high noise regimes, whereas the approximation training is more suitable in low noise regimes due to better convergence properties.

These investigations of the approximation and reconstruction training illustrate how the loss function determines the influence of the prior and noise distribution on the reconstruction scheme and shed light on which architectures are suitable. Investigating a link to MAP estimation and how it could be represented in terms of an iResNet might allow for revealing further links to regularization theory in future works (see also a more detailed discussion in appendix B).

The presented results allow further investigations and can serve as a foundation for future research directions. In the approximation training case, it might be desirable to relax the naive Bayes assumption and to consider the general non-diagonal network case (3.19). In line with the found limited data dependency in the diagonal case, we conjecture a dependency on second moments of the prior distribution p_X at most. In the general network case, we showed that reconstruction training leads to a data-dependent and stable reconstruction scheme that approximates the mean of the posterior distribution, where the degree of stability can be controlled by the Lipschitz constant L. What is left to prove is a convergence property as discussed in

remark 4.2, which could, in principle, further manifest the superiority of the reconstruction training approach and provide additional guarantees. Here, potential generalizations also could incorporate alternative loss functions in the integrand of the reconstruction training loss, which might result in approximations of alternative estimators induced by Bayes costs [12]. In this context, as a starting point, one may limit oneself to linear estimators to further investigate the relation to learned MAP estimators, respectively Tikhonov regularization, as in [2, 11]. In order to obtain convergence guarantees as well as data dependence, when desirable, one can explore a noise-controlled convex combination of both training losses. Theoretical as well as numerical investigations in this direction remain future research.

In addition, remark 2.3 serves as a basis to improve the numerical implementation of the reconstruction training by constructing the network as a scaled iResNet, resulting in a more efficient training approach as mentioned in section 5. This is a reasonable approach when one is not interested in directly comparing the approximation and reconstruction training. Numerical investigations, including the general non-diagonal network architecture, remain immediate future research.

Besides these improvements, extending our results to deeper network architectures, e.g. a concatenation of iResNets, could be beneficial. This would allow for more expressive network architectures and further improve the reconstruction quality of the networks. Finally, it might be worthwhile to generalize the results to nonlinear inverse problems, allowing for an application to a larger number of operators.

Data availability statement

The data that support the findings of this study are openly available at the following URL/DOI: https://gitlab.informatik.uni-bremen.de/inn4ip/iresnet-regularization.

Acknowledgments

N Heilenkötter, M Iske, and J Nickel acknowledge the support of the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - Project number 281474342/GRK2224/2. T Kluth acknowledges support from the DELETO project funded by the Federal Ministry of Education and Research (BMBF, Project Number 05M20LBB).

Appendix A. Proofs

A.1. Proof of lemma 3.1

Proof. For a function f of the form $f(x) = mx + b$ with the constraint $m^2 \leqslant L^2$, we can solve (3.7) by using the Lagrangian

$$K(m,b,\lambda) = \int_{\mathbb{R}} p_X(x) \left| \left(1 - \sigma^2 - m\right) x - b \right|^2 \mathrm{d}x + \lambda \left(m^2 - L^2\right), \tag{A.1}$$

where the integral is well-defined due to the existence of the first and second moments of p_X. The convexity, coercivity, and continuity of the integral term w.r.t. (m, b) imply that a minimizer exists. Therefore, we can calculate the minimizer using the necessary conditions (KKT conditions)

$$\frac{\partial K}{\partial m}(m,b,\lambda) = -\int_{\mathbb{R}} 2p_X(x)\left(\left(1-\sigma^2-m\right)x-b\right)x\,\mathrm{d}x + 2\lambda m \overset{!}{=} 0, \tag{A.2}$$

$$\frac{\partial K}{\partial b}(m,b,\lambda) = -\int_{\mathbb{R}} 2p_X(x)\left(\left(1-\sigma^2-m\right)x-b\right)\mathrm{d}x \overset{!}{=} 0, \tag{A.3}$$

$$\lambda\left(m^2-L^2\right) \overset{!}{=} 0, \tag{A.4}$$

$$\lambda \geqslant 0. \tag{A.5}$$

Rearranging (A.2) for m and (A.3) for b leads to

$$m = \frac{\int_{\mathbb{R}} 2p_X(x)\left(\left(1-\sigma^2\right)x-b\right)x\,\mathrm{d}x}{\int_{\mathbb{R}} 2p_X(x)x^2\,\mathrm{d}x + 2\lambda} = \frac{\left(1-\sigma^2\right)\mathbb{E}_{p_X}\left(x^2\right)-b\mu_X}{\mathbb{E}_{p_X}\left(x^2\right)+\lambda}, \tag{A.6}$$

$$b = \frac{\int_{\mathbb{R}} 2p_X(x)\left(1-\sigma^2-m\right)x\,\mathrm{d}x}{\int_{\mathbb{R}} 2p_X(x)\,\mathrm{d}x} = \left(1-\sigma^2-m\right)\mu_X, \tag{A.7}$$

where we use the abbreviated notation $\mathbb{E}_{p_X}$ for $\mathbb{E}_{x\sim p_X}$. Now, plugging $b=(1-\sigma^2-m)\mu_X$ into the equation for m implies

$$\begin{aligned}
& m = \frac{\left(1-\sigma^2\right)\mathbb{E}_{p_X}\left(x^2\right)-\left(1-\sigma^2-m\right)\mu_X^2}{\mathbb{E}_{p_X}\left(x^2\right)+\lambda} \\
\Leftrightarrow \quad & \left(1-\frac{\mu_X^2}{\mathbb{E}_{p_X}\left(x^2\right)+\lambda}\right)m = \frac{\left(1-\sigma^2\right)\left(\mathbb{E}_{p_X}\left(x^2\right)-\mu_X^2\right)}{\mathbb{E}_{p_X}\left(x^2\right)+\lambda} \\
\Leftrightarrow \quad & \frac{\mathrm{Var}_{p_X}(x)+\lambda}{\mathbb{E}_{p_X}\left(x^2\right)+\lambda}m = \frac{\left(1-\sigma^2\right)\mathrm{Var}_{p_X}(x)}{\mathbb{E}_{p_X}\left(x^2\right)+\lambda} \\
\Leftrightarrow \quad & m = \left(1-\sigma^2\right)\frac{\mathrm{Var}_{p_X}(x)}{\mathrm{Var}_{p_X}(x)+\lambda}.
\end{aligned} \tag{A.8}$$

Since $1-\sigma^2 > L$ holds by assumption, we need $\lambda > 0$ to ensure $m \leqslant L$. Then, (A.4) directly implies $m = L$. As m is uniquely determined we also know that b is unique with $b = (1-\sigma^2-L)\,\mu_X$. □

A.2. Proof of lemma 4.1

Proof. We denote the objective function by $F\colon L^2_{p_Z}(X,X) \to [0,\infty)$,

$$F(\psi) = \int_X \int_X p_X(x)\,p_H\left(z^\delta - Ax\right)\,\|\psi\left(z^\delta\right)-x\|^2\,\mathrm{d}z^\delta\,\mathrm{d}x, \tag{A.9}$$

which coincides with (4.3) with the substitution $z^\delta = Ax+\eta$. Note that for all $\psi \in L^2_{p_Z}(X,X)$, it holds $F(\psi) < \infty$ since

$$\|\psi\left(z^\delta\right)-x\|^2 = \|\psi\left(z^\delta\right)\|^2 - 2\langle\psi\left(z^\delta\right),x\rangle + \|x\|^2 \leqslant 2\|\psi\left(z^\delta\right)\|^2 + 2\|x\|^2 \tag{A.10}$$

and the integrals

$$\int_X \int_X p_X(x)\, p_H\left(z^\delta - Ax\right) \|\psi(z)\|^2 \,\mathrm{d}z^\delta \,\mathrm{d}x = \int_X p_Z\left(z^\delta\right) \|\psi\left(z^\delta\right)\|^2 = \|\psi\|^2_{p_Z,2},$$

$$\begin{aligned}\int_X \int_X p_X(x)\, p_H\left(z^\delta - Ax\right) \|x\|^2 \,\mathrm{d}z^\delta &= \int_X p_X(x)\, \|x\|^2 \int_X p_H\left(z^\delta - Ax\right) \mathrm{d}z^\delta \,\mathrm{d}x \\ &= \int_X p_X(x)\, \|x\|^2 \,\mathrm{d}x \end{aligned} \tag{A.11}$$

are both finite.

Besides, note that F is convex w.r.t. ψ since $\psi \mapsto \|\psi(z^\delta) - x\|^2$ is convex for any $x, z^\delta \in X$. Thus, we can find the minimizer of F by setting its derivative to zero.

To compute the derivative of F, we consider

$$\begin{aligned} F(\psi + h) &= \int_X \int_X p_X(x)\, p_H\left(z^\delta - Ax\right) \|\psi\left(z^\delta\right) + h\left(z^\delta\right) - x\|^2 \,\mathrm{d}z^\delta \,\mathrm{d}x \\ &= \int_X \int_X p_X(x)\, p_H\left(z^\delta - Ax\right) \|\psi\left(z^\delta\right) - x\|^2 \,\mathrm{d}z^\delta \,\mathrm{d}x \\ &\quad + 2\int_X \int_X p_X(x)\, p_H\left(z^\delta - Ax\right) \langle \psi\left(z^\delta\right) - x, h\left(z^\delta\right)\rangle \,\mathrm{d}z^\delta \,\mathrm{d}x \\ &\quad + \int_X \int_X p_X(x)\, p_H\left(z^\delta - Ax\right) \|h\left(z^\delta\right)\|^2 \,\mathrm{d}z^\delta \,\mathrm{d}x. \end{aligned} \tag{A.12}$$

The first of the three summands is $F(\psi)$, the last one equals $\|h\|^2_{p_Z,2}$ and the second summand equals $\partial F(\psi)h$. Note that the second summand is finite since $F(\psi)$ and $\|h\|^2_{p_Z,2}$ are both finite. Using Fubini's theorem, we obtain

$$\begin{aligned} \partial F(\psi)\, h &= 2\int_X \int_X p_X(x)\, p_H\left(z^\delta - Ax\right) \langle \psi\left(z^\delta\right) - x, h\left(z^\delta\right)\rangle \,\mathrm{d}z^\delta \,\mathrm{d}x \\ &= 2\int_X \left\langle \int_X p_X(x)\, p_H\left(z^\delta - Ax\right) \left(\psi\left(z^\delta\right) - x\right) \mathrm{d}x, h\left(z^\delta\right) \right\rangle \mathrm{d}z^\delta. \end{aligned} \tag{A.13}$$

We are looking for $\hat{\psi}$ such that $\partial F(\hat{\psi})h = 0$ for any $h \in L^2_{p_Z}(X,X)$. Hence, the fundamental lemma of the calculus of variations implies

$$\begin{aligned} & \int_X p_X(x)\, p_H\left(z^\delta - Ax\right) \left(\hat{\psi}\left(z^\delta\right) - x\right) \mathrm{d}x = 0 \\ \Leftrightarrow \quad & p_Z\left(z^\delta\right) \cdot \hat{\psi}\left(z^\delta\right) = \int_X p_X(x)\, p_H\left(z^\delta - Ax\right) x \,\mathrm{d}x \\ \Leftrightarrow \quad & \hat{\psi}\left(z^\delta\right) = \frac{\int_X p_X(x)\, p_H\left(z^\delta - Ax\right) x \,\mathrm{d}x}{p_Z\left(z^\delta\right)} \end{aligned} \tag{A.14}$$

for almost all $z^\delta \in \mathrm{supp}(p_Z)$. According to Bayes' formula

$$p\left(x|z^\delta\right) = \frac{p\left(z^\delta|x\right) p_X(x)}{p_Z\left(z^\delta\right)} = \frac{p_H\left(z^\delta - Ax\right) p_X(x)}{p_Z\left(z^\delta\right)}, \tag{A.15}$$

$\hat{\psi}(z^\delta)$ is the expected value of the posterior density function $p(x|z^\delta)$.

Finally, we need to make sure that $\hat{\psi}(z^\delta) = \mathbb{E}(x|z^\delta)$ is an $L^2_{p_Z}$-function. For this, we use Jensen's inequality for conditional expectations [14, theorem 8.20, corollary 8.21] and obtain

$$\|\mathbb{E}\left(x|z^\delta\right)\|^2 = \sum_{j=1}^{n} |\mathbb{E}\left(x_j|z^\delta\right)|^2 \leqslant \sum_{j=1}^{n} \mathbb{E}\left(|x_j|^2 \,|\, z^\delta\right) = \mathbb{E}\left(\|x\|^2 \,|\, z^\delta\right). \tag{A.16}$$

Then, by the definition of conditional expectations, we get

$$\int_X p_Z\left(z^\delta\right) \mathbb{E}\left(\|x\|^2 \,|\, z^\delta\right) \mathrm{d}z^\delta = \mathbb{E}_{p_Z}\left(\mathbb{E}\left(\|x\|^2 \,|\, z^\delta\right)\right) = \mathbb{E}\left(\|x\|^2\right), \tag{A.17}$$

which is finite, and the proof is complete. □

A.3. Proof of lemma 4.3

Proof. Due to (iii), it immediately follows that p_X is uniformly continuous. From (iii), we also deduce that the marginal of p_X on $\mathcal{N}(A)^\perp$, $p_{X,\mathcal{N}(A)^\perp} : \mathcal{N}(A)^\perp \to \mathbb{R}_{\geqslant 0}$ with

$$p_{X,\mathcal{N}(A)^\perp}(x) = \int_{\mathcal{N}(A)} p_X(x_0 + x)\ \mathrm{d}x_0 \tag{A.18}$$

is compactly supported and bounded and thus also uniformly continuous. Analogously, we can deduce uniform continuity of the mappings $g_0 : \mathcal{N}(A)^\perp \to \mathcal{N}(A)$ and $g_\dagger : \mathcal{N}(A)^\perp \to \mathcal{N}(A)^\perp$ with

$$g_0(x) = \int_{\mathcal{N}(A)} p_X(x_0 + x)\, x_0\ \mathrm{d}x_0 \tag{A.19}$$

and

$$g_\dagger(x) = \int_{\mathcal{N}(A)} p_X(x_0 + x)\ \mathrm{d}x_0\, x. \tag{A.20}$$

Next, by using $X = \mathcal{N}(A) \oplus \mathcal{N}(A)^\perp$ we now observe that for arbitrary $z \in X$ due to (i) it holds

$$\begin{aligned} p_{Z,\delta}(z) &= \int_X p_{H,\delta}(z - Ax)\, p_X(x)\ \mathrm{d}x \\ &= \int_{\mathcal{N}(A)^\perp} p_{H,\delta}(z - Ax_1) \int_{\mathcal{N}(A)} p_X(x_0 + x_1)\ \mathrm{d}x_0\, \mathrm{d}x_1 \\ &= p^0_{H,\delta}\left(P_{\mathcal{N}(A)} z\right) \int_{\mathcal{R}(A)} p^\dagger_{H,\delta}\left(P_{\mathcal{N}(A)^\perp} z - y_1\right) \int_{\mathcal{N}(A)} p_X\left(x_0 + A^\dagger y_1\right)\ \mathrm{d}x_0\, \mathrm{d}y_1 |\det A^\dagger|, \end{aligned} \tag{A.21}$$

where the transformation $x_1 = A^\dagger y_1$, with generalized inverse $A^\dagger : \mathcal{R}(A) \to \mathcal{N}(A)^\perp$, is used in the last equality. Analogously, we further obtain for arbitrary $z \in \mathrm{supp}(p_{Z,\delta}) \subset X$

$$\begin{aligned}\hat{\psi}_\delta(z) &= \frac{1}{p_{Z,\delta}(z)} \int_X p_{H,\delta}(z - Ax)\, p_X(x)\, x \,\mathrm{d}x \\ &= \frac{1}{p_{Z,\delta}(z)} \int_{\mathcal{N}(A)^\perp} p_{H,\delta}(z - Ax_1) \int_{\mathcal{N}(A)} p_X(x_0 + x_1)\,(x_0 + x_1)\ \mathrm{d}x_0\, \mathrm{d}x_1 \\ &= \frac{p^0_{H,\delta}\left(P_{\mathcal{N}(A)} z\right)}{p_{Z,\delta}(z)} \int_{\mathcal{R}(A)} p^\dagger_{H,\delta}\left(P_{\mathcal{N}(A)^\perp} z - y_1\right) \\ &\qquad \int_{\mathcal{N}(A)} p_X\left(x_0 + A^\dagger y_1\right)\left(x_0 + A^\dagger y_1\right)\ \mathrm{d}x_0\, \mathrm{d}y_1 |\det A^\dagger|. \end{aligned} \tag{A.22}$$

Exploiting the previously derived representation of $p_{Z,\delta}$ and (i) yields

$$\begin{aligned}\hat{\psi}_\delta(z) &= \frac{\int_{\mathcal{R}(A)} p^\dagger_{H,\delta}\left(P_{\mathcal{N}(A)^\perp} z - y_1\right)\left(\int_{\mathcal{N}(A)} p_X\left(x_0 + A^\dagger y_1\right)\left(x_0 + A^\dagger y_1\right)\ \mathrm{d}x_0\right)\ \mathrm{d}y_1}{\int_{\mathcal{R}(A)} p^\dagger_{H,\delta}\left(P_{\mathcal{N}(A)^\perp} z - y_1\right) \int_{\mathcal{N}(A)} p_X\left(x_0 + A^\dagger y_1\right)\ \mathrm{d}x_0\, \mathrm{d}y_1} \\ &= \frac{\int_{\mathcal{R}(A)} p^\dagger_{H,\delta}\left(P_{\mathcal{N}(A)^\perp} z - y_1\right)\left(g_0\left(A^\dagger y_1\right) + g_\dagger\left(A^\dagger y_1\right)\right)\ \mathrm{d}y_1}{\int_{\mathcal{R}(A)} p^\dagger_{H,\delta}\left(P_{\mathcal{N}(A)^\perp} z - y_1\right) p_{X,\mathcal{N}(A)^\perp}\left(A^\dagger y_1\right)\ \mathrm{d}y_1}. \end{aligned} \tag{A.23}$$

We now consider the denominator and nominator separately. In the denominator we first observe that the mapping $p_{X,\mathcal{N}(A)^\perp}(A^\dagger y_1)$ is also uniformly continuous due to the continuity of $A^\dagger$. Exploiting $\mathcal{R}(A) = \mathcal{R}(A^*) = \mathcal{N}(A)^\perp$, the approximation property [15, sec II] of the Dirac sequence $p^\dagger_{H,\delta}$ delivers uniform convergence of the denominator, i.e. this implies pointwise convergence such that for any $z \in X$ it holds

$$\int_{\mathcal{R}(A)} p^\dagger_{H,\delta}\left(P_{\mathcal{N}(A)^\perp} z - y_1\right) p_{X,\mathcal{N}(A)^\perp}\left(A^\dagger y_1\right)\ \mathrm{d}y_1 \underset{\delta\to 0}{\longrightarrow} p_{X,\mathcal{N}(A)^\perp}\left(A^\dagger P_{\mathcal{N}(A)^\perp} z\right). \tag{A.24}$$

Analogous arguments apply to the nominator of (A.23) by exploiting the approximation property of the Dirac sequence implying uniform convergence and thus pointwise convergence such that for any $z \in X$, it holds

$$\begin{aligned}&\int_{\mathcal{R}(A)} p^\dagger_{H,\delta}\left(P_{\mathcal{N}(A)^\perp} z - y_1\right)\left(g_0\left(A^\dagger y_1\right) + g_\dagger\left(A^\dagger y_1\right)\right)\ \mathrm{d}y_1 \\ &\quad \underset{\delta\to 0}{\longrightarrow} g_0\left(A^\dagger P_{\mathcal{N}(A)^\perp} z\right) + g_\dagger\left(A^\dagger P_{\mathcal{N}(A)^\perp} z\right). \end{aligned} \tag{A.25}$$

Due to (iv) for any fixed $z \in \mathcal{R}_{p_X}(A)$ the representation of $\hat{\psi}_\delta$ in (A.23) is well-defined for sufficiently small δ. Consequently, we have convergent sequences in the nominator and in the denominator such that the quotient converges by standard sequence arguments. As $\mathcal{R}_{p_X}(A) \subset \mathcal{R}(A) = \mathcal{N}(A)^\perp$ we thus obtain the desired pointwise convergence

$$\hat{\psi}_\delta(z) \underset{\delta\to 0}{\longrightarrow} \frac{g_\dagger\left(A^\dagger z\right) + g_0\left(A^\dagger z\right)}{p_{X,\mathcal{N}(A)^\perp}\left(A^\dagger z\right)} = A^\dagger z + \int_{\mathcal{N}(A)} \frac{p_X\left(x_0 + A^\dagger z\right)}{\int_{\mathcal{N}(A)} p_X\left(x_0' + A^\dagger z\right)\ \mathrm{d}x_0'} x_0 \,\mathrm{d}x_0. \tag{A.26}$$

□

A.4. Proof of lemma 4.4

Proof. We solve the minimization problem (4.28) by applying the KKT conditions. To this end, consider the Lagrange functional

$$K(m,b,\lambda_1,\lambda_2) = \int_{\mathbb{R}}\int_{\mathbb{R}} p_X(x)\, p_H(\eta)\, \left(m\left(\sigma^2 x+\eta\right)+b-x\right)^2 \mathrm{d}\eta\,\mathrm{d}x \\ + \lambda_1\left(m-\frac{1}{1-L}\right)+\lambda_2\left(\frac{1}{1+L}-m\right) \tag{A.27}$$

for $m,b,\lambda_1,\lambda_2\in\mathbb{R}$. Observe that the integral is well-defined as the first and second moments of p_X and p_H exist by assumption. Moreover, K is convex and coercive w.r.t. (m,b) and the integral is continuous w.r.t. (m,b). Consequently, there exists a minimizer of problem (4.28) which satisfies the necessary KKT conditions

$$\frac{\partial K}{\partial m}(m,b,\lambda_1,\lambda_2) = \int_{\mathbb{R}}\int_{\mathbb{R}} 2p_X(x)\, p_H(\eta)\, \left(\sigma^2 x+\eta\right)\left(m\left(\sigma^2 x+\eta\right)+b-x\right)\mathrm{d}\eta\,\mathrm{d}x \\ + \lambda_1 m-\lambda_2 m \overset{!}{=} 0, \tag{A.28}$$

$$\frac{\partial K}{\partial b}(m,b,\lambda_1,\lambda_2) = \int_{\mathbb{R}}\int_{\mathbb{R}} 2p_X(x)\, p_H(\eta)\, \left(m\left(\sigma^2 x+\eta\right)+b-x\right)\mathrm{d}\eta\,\mathrm{d}x \overset{!}{=} 0, \tag{A.29}$$

$$\lambda_1\left(m-\frac{1}{1-L}\right)\overset{!}{=}0, \tag{A.30}$$

$$\lambda_2\left(\frac{1}{1+L}-m\right)\overset{!}{=}0, \tag{A.31}$$

$$\lambda_1,\lambda_2\geqslant 0. \tag{A.32}$$

Case $\lambda_1=\lambda_2=0$: equation (A.28) implies

$$\begin{aligned}
&\int_{\mathbb{R}}\int_{\mathbb{R}} p_X(x)\, p_H(\eta)\, \left(\sigma^2 x+\eta\right)\left(m\left(\sigma^2 x+\eta\right)+b-x\right)\mathrm{d}\eta\,\mathrm{d}x \\
&\quad= \int_{\mathbb{R}}\int_{\mathbb{R}} p_X(x)\, p_H(\eta)\, \left(m\sigma^4x^2+2m\sigma^2x\eta+m\eta^2+b\sigma^2x+b\eta-\sigma^2x^2-\eta x\right)\mathrm{d}\eta\,\mathrm{d}x \\
&\quad= m\sigma^4\mathbb{E}_{p_X}\left(x^2\right)+2m\sigma^2\mu_X\mu_H+m\mathbb{E}_{p_H}\left(\eta^2\right)+b\sigma^2\mu_X+b\mu_H-\sigma^2\mathbb{E}_{p_X}\left(x^2\right)-\mu_H\mu_X \\
&\quad= m\sigma^4\mathbb{E}_{p_X}\left(x^2\right)+m\mathbb{E}_{p_H}\left(\eta^2\right)+b\sigma^2\mu_X-\sigma^2\mathbb{E}_{p_X}\left(x^2\right) \\
&\quad\overset{!}{=} 0
\end{aligned} \tag{A.33}$$

and equation (A.29) gives

$$\int_{\mathbb{R}}\int_{\mathbb{R}} 2p_X(x)p_H(\eta)\left(m\left(\sigma^2x+\eta\right)+b-x\right)\mathrm{d}\eta\,\mathrm{d}x = m\sigma^2\mu_X+m\mu_H+b-\mu_X\overset{!}{=}0 \tag{A.34}$$

resulting in

$$b = \mu_X - m\sigma^2\mu_X. \tag{A.35}$$

Inserting the last equation into equation (A.33) yields

$$\begin{aligned} & m\sigma^4\mathbb{E}_{p_X}\left(x^2\right) + m\mathbb{E}_{p_H}\left(\eta^2\right) + \sigma^2\mu_X^2 - m\sigma^4\mu_X^2 - \sigma^2\mathbb{E}_{p_X}\left(x^2\right) \overset{!}{=} 0 \\ \Leftrightarrow \quad & m\sigma^4\mathrm{Var}_{p_X}(x) + m\mathrm{Var}_{p_H}(\eta) - \sigma^2\mathrm{Var}_{p_X}(x) = 0 \\ \Leftrightarrow \quad & m = \frac{\sigma^2\mathrm{Var}_{p_X}(x)}{\sigma^4\mathrm{Var}_{p_X}(x) + \mathrm{Var}_{p_H}(\eta)} \end{aligned} \tag{A.36}$$

and

$$b = \left(1 - \frac{\sigma^4\mathrm{Var}_{p_X}(x)}{\sigma^4\mathrm{Var}_{p_X}(x) + \mathrm{Var}_{p_H}(\eta)}\right)\mu_X. \tag{A.37}$$

The formulas for m and b correspond to the unconstrained solution of problem (4.28). In order to satisfy the constraint on m, we need to guarantee that

$$\frac{1}{1+L} \leqslant \frac{\sigma^2\mathrm{Var}_{p_X}(x)}{\sigma^4\mathrm{Var}_{p_X}(x) + \mathrm{Var}_{p_H}(\eta)} \leqslant \frac{1}{1-L}. \tag{A.38}$$

If this is not satisfied, we must require $\lambda_1 > 0$ or $\lambda_2 > 0$, which we will deal with in the following paragraphs.
Case $\lambda_1 > 0$, $\lambda_2 = 0$: If $\lambda_1 > 0$, equation (A.30) directly yields

$$m = \frac{1}{1-L}. \tag{A.39}$$

For b, we again obtain

$$b = \mu_X - m\sigma^2\mu_X = \left(1 - \frac{\sigma^2}{1-L}\right)\mu_X \tag{A.40}$$

as equation (A.29) is independent of λ_1 and λ_2. Furthermore, equation (A.28) implies

$$\begin{aligned} & \int_{\mathbb{R}}\int_{\mathbb{R}} p_X(x)\, p_H(\eta)\left(\sigma^2 x + \eta\right)\left(m\left(\sigma^2 x + \eta\right) + b - x\right) \mathrm{d}\eta\, \mathrm{d}x + \lambda_1 m \overset{!}{=} 0 \\ \Leftrightarrow \quad & \lambda_1 = 2\,\frac{\sigma^2\mathrm{Var}_{p_X}(x) - m\sigma^4\mathrm{Var}_{p_X}(x) - m\mathrm{Var}_{p_H}(\eta)}{m}. \end{aligned} \tag{A.41}$$

In combination with the condition $\lambda_1 > 0$ we now obtain

$$\frac{\sigma^2\mathrm{Var}_{p_X}(x)}{\sigma^4\mathrm{Var}_{p_X}(x) + \mathrm{Var}_{p_H}(\eta)} > \frac{1}{1-L}. \tag{A.42}$$

Case $\lambda_1 = 0$, $\lambda_2 > 0$: The same line of reasoning as in the preceding case yields

$$m = \frac{1}{1+L} \tag{A.43}$$

$$b = \mu_X - m\sigma^2\mu_X = \left(1 - \frac{\sigma^2}{1+L}\right)\mu_X \tag{A.44}$$

$$\frac{\sigma^2 \mathrm{Var}_{p_X}(x)}{\sigma^4 \mathrm{Var}_{p_X}(x) + \mathrm{Var}_{p_H}(\eta)} < \frac{1}{1+L}. \tag{A.45}$$

Observe that the case $\lambda_1 > 0$ and $\lambda_2 > 0$ is not possible as there is no m satisfying equations (A.30) and (A.31) simultaneously for $0 < L < 1$. The uniqueness of the solution directly follows from the uniqueness of m and b, which concludes the proof. □

Appendix B. MAP estimation and its interpretation as an iResNet

Consider a linear inverse problem $\tilde{A}x = y$ with $\tilde{A} \in L(X, Y)$ as in remark 2.1, where noisy data $y^\delta \in Y$ is given. One established approach in Bayesian inverse problems is the so-called MAP (maximum *a posteriori*) estimator

$$x_{\mathrm{MAP}} = \arg\max_{x \in X} p\left(x|y^\delta\right). \tag{B.1}$$

The posterior density $p(x|y^\delta)$ can be derived via Bayes rule from the pdf's of the prior ($x \sim p_X$), the noise ($y^\delta - Ax \sim \tilde{p}_H$) and the data ($y^\delta \sim p_Y$). Using this and the monotonicity of the logarithm, one obtains

$$\begin{aligned} x_{\mathrm{MAP}} &= \arg\max_{x \in X} \frac{\tilde{p}_H\left(y^\delta - \tilde{A}x\right) p_X(x)}{p_Y\left(y^\delta\right)} \\ \Leftrightarrow x_{\mathrm{MAP}} &= \arg\max_{x \in X} \tilde{p}_H\left(y^\delta - \tilde{A}x\right) p_X(x) \\ \Leftrightarrow x_{\mathrm{MAP}} &= \arg\min_{x \in X} -\log\left(\tilde{p}_H\left(y^\delta - \tilde{A}x\right)\right) - \log\left(p_X(x)\right). \end{aligned} \tag{B.2}$$

Here, one can observe a well-known similarity to variational regularization schemes. In the case of Gaussian noise with noise level $\delta > 0$, i.e.

$$\tilde{p}_H(\tilde{\eta}) \propto \exp\left(-\frac{1}{2\delta^2}\|\tilde{\eta}\|^2\right) \tag{B.3}$$

it holds

$$x_{\mathrm{MAP}} = \arg\min_{x \in X} -\frac{1}{2}\|\tilde{A}x - y^\delta\|^2 - \delta^2 \log\left(p_X(x)\right). \tag{B.4}$$

The negative log-likelihood (NLL) $-\log p_X$ can be interpreted as a penalty term, weighted with the squared noise level δ^2. If $-\log p_X$ is differentiable, we can use the first-order optimality condition and derive

$$\begin{aligned} 0 &= \tilde{A}^*\left(\tilde{A}x_{\mathrm{MAP}} - y^\delta\right) - \delta^2 \partial\left(\log p_X\right)\left(x_{\mathrm{MAP}}\right) \\ \Rightarrow \tilde{A}^* y^\delta &= \tilde{A}^*\tilde{A}x_{\mathrm{MAP}} - \delta^2 \partial\left(\log p_X\right)\left(x_{\mathrm{MAP}}\right) \\ \Rightarrow x_{\mathrm{MAP}} &= \left(\mathrm{Id} - \left(\mathrm{Id} - \tilde{A}^*\tilde{A}x_{\mathrm{MAP}} + \delta^2 \partial\left(\log p_X\right)\right)\right)^{-1}\left(\tilde{A}^* y^\delta\right), \end{aligned} \tag{B.5}$$

where the last implication only holds if $\tilde{A}^*\tilde{A} - \delta^2\partial(\log p_X)$ is invertible (which is guaranteed, e.g. in case of a convex NLL).

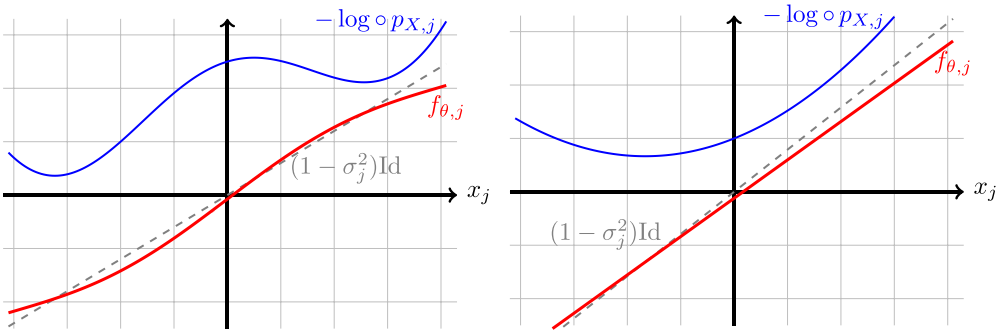


Figure 10. MAP estimation with iResNet. For large singular values $\sigma_j^2 > 1-L$, the NLL is not very restricted (can be nonconvex) to guarantee $\mathrm{Lip}(f_{\theta,j}) \leqslant L < 1$. For small singular values $\sigma_j^2 \leqslant 1-L$, the NLL has to be (strongly) convex to guarantee that the slope of $f_{\theta,j}$ is smaller than $1-\sigma_j^2$ (and smaller than L).

Now, we can interpret (B.5) as an iResNet approach for solving the inverse problem

$$Ax = z \tag{B.6}$$

where $A = \tilde{A}^*\tilde{A}$ and $z = \tilde{A}^* y$. It holds $x_{\mathrm{MAP}} = \varphi_\theta^{-1}(z^\delta) = (\mathrm{Id} - f_\theta)^{-1}(\tilde{A}^* y^\delta)$ if the residual layer is given by

$$f_\theta = \mathrm{Id} - A + \delta^2 \partial(\log p_X). \tag{B.7}$$

Thus, MAP estimation with an iResNet is possible as long as the above-defined f_θ has a Lipschitz constant of at most $L < 1$.

We can derive conditions for the prior p_X and the noise level δ from this Lipschitz constraint by making use of assumption 3.1, i.e. stochastic independence of the components $x_j \sim p_{X,j}$ and the eigendecomposition of A. In this setting, the components can be handled separately and, thus,

$$f_{\theta,j} = \left(1-\sigma_j^2\right)\mathrm{Id} + \delta^2 \partial(\log p_{X,j}). \tag{B.8}$$

To obtain further insights, we distinguish between large eigenvalues (i.e. $\sigma_j^2 > 1-L$) and small ones (i.e. $\sigma_j^2 \leqslant 1-L$).

Remark B.1. The prior $p_{X,j}$ corresponding to a large eigenvalue can have a rather arbitrary character. The only important property is that the derivative of the NLL (i.e. $\partial(\log p_{X,j})$) must be Lipschitz continuous. Because in this case, for small enough δ, it holds

$$\mathrm{Lip}\left(f_{\theta,j}\right) \leqslant \left(1-\sigma^2\right) + \delta^2 \mathrm{Lip}\left(\partial(\log p_{X,j})\right) \leqslant L. \tag{B.9}$$

This is visualized in the left plot of figure 10.

However, for smaller eigenvalues, it holds $1-\sigma^2 \geqslant L$. Hence, the prior must decrease the slope of $f_{\theta,j}$. This holds true if the NLL of the prior is convex (or even strongly convex). Because then, $\partial(\log p_{X,j})$ is monotonously decreasing and δ can again be chosen s.t.

$$\mathrm{Lip}\left(f_{\theta,j}\right) = \mathrm{Lip}\left(\left(1-\sigma^2\right)\mathrm{Id} + \partial(\log p_{X,j})\right) \leqslant L \tag{B.10}$$

holds. An example for such a $p_{X,j}$ is a Gaussian prior, where $\partial(\log p_{X,j})$ is a linear function with a negative slope (see the right plot of figure 10 and the subsequent derivations).

B.1. Example prior distributions

To exemplify the previous observations, let us look at two commonly used prior distributions, namely the Gaussian distribution and the Laplace distribution. In the case of the Gaussian distribution, we have

$$p_{X,j}(x) = \frac{1}{\sqrt{2\pi\, b_j^2}} \exp\left(-\frac{(x-\mu_j)^2}{2b_j^2}\right) \tag{B.11}$$

for all $j \in \mathbb{N}$ with mean $\mu_j \in \mathbb{R}$ and variance $b_j^2 > 0$. Consequently, for the residual layer, we obtain

$$f_j(x) = \left(1 - \sigma_j^2 - \frac{\delta^2}{b_j^2}\right) x + \frac{\delta^2 \mu_j}{b_j^2} \quad \text{for } x \in \mathbb{R} \tag{B.12}$$

with Lipschitz constant $\mathrm{Lip}(f_j) = |1 - \sigma_j^2 - {\delta^2}/{b_j^2}|$. Hence, similar to the observations in the previous remark, for singular values with $1 - \sigma_j^2 < L$, the Lipschitz constraint $\mathrm{Lip}(f_j) \leqslant L$ is fulfilled for all δ and b_j. In the case $1 - \sigma_j^2 \geqslant L$, δ and b_j need to satisfy ${\delta^2}/{b_j^2} \geqslant 1 - \sigma_j^2 - L$ to guarantee $\mathrm{Lip}(f_j) \leqslant L$.

In the case of the prior distribution being a Laplacian, i.e.

$$p_{X,j}(x) = \frac{1}{\sqrt{2b_j}} \exp\left(-\frac{|x-\mu_j|}{b_j}\right) \quad \text{for } x \in \mathbb{R} \tag{B.13}$$

with mean $\mu_j \in \mathbb{R}$ and variance $2b_j^2 > 0$, the subgradient of $\log p_{X,j}$ is given by

$$\partial(\log p_{X,j})(x) = \begin{cases} \frac{1}{b_j} & x < \mu_j \\ \left[-\frac{1}{b_j}, \frac{1}{b_j}\right] & x = \mu_j \\ -\frac{1}{b_j} & x > \mu_j \end{cases} \quad \text{for } x \in \mathbb{R}. \tag{B.14}$$

Consequently, the residual layer for the Laplacian prior distribution is given by

$$f_j(x) = \begin{cases} \left(1 - \sigma_j^2\right) x + \frac{\delta^2}{b_j} & x < \mu_j \\ \left[\left(1 - \sigma_j^2\right) x - \frac{\delta^2}{b_j}, \left(1 - \sigma_j^2\right) x + \frac{\delta^2}{b_j}\right] & x = \mu_j \\ \left(1 - \sigma_j^2\right) x - \frac{\delta^2}{b_j} & x > \mu_j \end{cases} \quad \text{for } x \in \mathbb{R}, \tag{B.15}$$

which is not Lipschitz-continuous. As a result, the Laplace distribution does not satisfy the conditions of remark B.1.

In summary, the previous considerations illustrate that MAP estimation with a Gaussian noise model can also be represented by the proposed iResNet approach for certain prior distributions, guaranteeing invertibility.

Appendix C. Approximation training in diagonal architecture with dependent data and noise distribution

In section 3, we assumed that the random variables $x \sim p_X$ and $\eta \sim p_H$ are independent. However, one can obtain a more general version of lemma 3.1 with less restrictive assumptions on the joint data and noise distribution. To this end, we denote by $p\colon \mathbb{R}^2 \to [0,\infty)$ the joint probability density function with marginal distributions $p_X(x) = \int_{\mathbb{R}} p(x,\eta)\,\mathrm{d}\eta$, $p_H(\eta) =$

$\int_{\mathbb{R}} p(x,\eta)\,\mathrm{d}x$ and assume that the respective first and second moments exist. In this setting, the minimization problem of the approximation training reads

$$\min_{f\in\mathcal{F}} \int_{\mathbb{R}^2} p\,(x,\eta)\;|\left(1-\sigma^2\right)x-\eta-f(x)\,|^2\,\mathrm{d}\,(x,\eta) \tag{C.1}$$

and the subsequent lemma provides a closed-form solution of the minimizer of problem (C.1) in the case $L<1-\sigma^2$.

Lemma C.1. *Let* $\mathcal{F}=\{f\in C(\mathbb{R})\,|\,\exists m\in[-L,L], b\in\mathbb{R}\colon\ f(x)=mx+b\}$ *and* $L<1-\sigma^2$. *Then,*

$$f^*\,(\hat{x}) = \begin{cases} L\hat{x}+\left(1-\sigma^2-L\right)\mu_X-\mu_H & \text{if } \frac{\mathrm{Cov}_p(x,\eta)}{\mathrm{Var}_{p_X}(x)} < 1-\sigma^2-L \\ \left(1-\sigma^2-\frac{\mathrm{Cov}_p(x,\eta)}{\mathrm{Var}_{p_X}(x)}\right)\hat{x}+\frac{\mathrm{Cov}_p(x,\eta)}{\mathrm{Var}_{p_X}(x)}\mu_X-\mu_H & \text{if } \frac{\mathrm{Cov}_p(x,\eta)}{\mathrm{Var}_{p_X}(x)} \in \left[1-\sigma^2-L,1-\sigma^2+L\right] \\ -L\hat{x}+\left(1-\sigma^2+L\right)\mu_X-\mu_H & \text{if } \frac{\mathrm{Cov}_p(x,\eta)}{\mathrm{Var}_{p_X}(x)} > 1-\sigma^2+L \end{cases} \tag{C.2}$$

is the unique solution of the minimization problem (C.1), where μ_X, μ_H *denote the expected values of the marginal distributions,* Cov_p *the covariance w.r.t.* $(x,\eta)\sim p$ *and* Var_{p_X} *the variance w.r.t.* $x\sim p_X$.

Proof. For a function f of the form $f(x)=mx+b$ with the constraint $m^2\leqslant L^2$, we can solve (C.1) by using the Lagrangian

$$K(m,b,\lambda)=\int_{\mathbb{R}^2} p\,(x,\eta)\;|\left(1-\sigma^2-m\right)x-\eta-b|^2\,\mathrm{d}\,(x,\eta)+\lambda\left(m^2-L^2\right). \tag{C.3}$$

Observe that the integral is well-defined due to the existence of the first and second moments of p. In addition, the convexity, coercivity, and continuity of the integral term w.r.t. (m,b) implies that a minimizer exists. The minimizer must satisfy the necessary conditions (KKT conditions)

$$\frac{\partial K}{\partial m}\,(m,b,\lambda) = -2\int_{\mathbb{R}^2} p\,(x,\eta)\left(\left(1-\sigma^2-m\right)x-\eta-b\right)x\,\mathrm{d}\,(x,\eta)+2\lambda m \overset{!}{=} 0, \tag{C.4}$$

$$\frac{\partial K}{\partial b}\,(m,b,\lambda) = -2\int_{\mathbb{R}^2} p\,(x,\eta)\left(\left(1-\sigma^2-m\right)x-\eta-b\right)\mathrm{d}\,(x,\eta) \overset{!}{=} 0, \tag{C.5}$$

$$\lambda\left(m^2-L^2\right) \overset{!}{=} 0, \tag{C.6}$$

$$\lambda \geqslant 0. \tag{C.7}$$

Exploiting the marginal distribution p_X, p_H and rearranging (C.4) for m and (C.5) for b leads to

$$m=\frac{\left(1-\sigma^2\right)\mathbb{E}_{p_X}\left(x^2\right)-b\,\mu_X-\mathbb{E}_p\left(x\cdot\eta\right)}{\mathbb{E}_{p_X}\left(x^2\right)+\lambda}, \tag{C.8}$$

$$b=\left(1-\sigma^2-m\right)\mu_X-\mu_H, \tag{C.9}$$

where we use the abbreviated notation $\mathbb{E}_p$ for $\mathbb{E}_{(x,\eta)\sim p}$, $\mathbb{E}_{p_X}$ for $\mathbb{E}_{x\sim p_X}$ and $\mathbb{E}_{p_H}$ for $\mathbb{E}_{\eta\sim p_H}$. Combining both equations yields

$$\begin{aligned}
&\left(1-\frac{\mu_X^2}{\mathbb{E}_{p_X}(x^2)+\lambda}\right)m = \frac{\left(1-\sigma^2\right)\left(\mathbb{E}_{p_X}\left(x^2\right)-\mu_X^2\right)-\left(\mathbb{E}_p\left(x\cdot\eta\right)-\mu_X\mu_H\right)}{\mathbb{E}_p\left(x^2\right)+\lambda}\\
\Leftrightarrow\quad &\frac{\mathrm{Var}_{p_X}(x)+\lambda}{\mathbb{E}_{p_X}(x^2)+\lambda}m = \frac{\left(1-\sigma^2\right)\mathrm{Var}_{p_X}(x)-\mathrm{Cov}_p(x,\eta)}{\mathbb{E}_{p_X}(x^2)+\lambda}\\
\Leftrightarrow\quad &m = \left(1-\sigma^2\right)\frac{\mathrm{Var}_{p_X}(x)}{\mathrm{Var}_{p_X}(x)+\lambda}-\frac{\mathrm{Cov}_p(x,\eta)}{\mathrm{Var}_{p_X}(x)+\lambda}. \qquad \text{(C.10)}
\end{aligned}$$

In order to determine the value of λ, we need to distinguish two cases.

(I): $\mathrm{Cov}_p(x,\eta)\leqslant 0$:
Since $1-\sigma^2>L$ holds by assumption, the case $\lambda=0$ is not possible, and we need $\lambda>0$ to ensure $m\leqslant L$. Then, (C.6) directly implies $m=L$ as (C.10) cancels out the possibility $m=-L$. Thus we also know $b=(1-\sigma^2-L)\mu_X-\mu_H$.

(II): $\mathrm{Cov}_p(x,\eta)>0$:
In this case, we need to distinguish the cases $\lambda=0$ and $\lambda>0$.
(IIa): $\lambda=0$:
In this case, we have

$$m = \left(1-\sigma^2\right)-\frac{\mathrm{Cov}_p(x,\eta)}{\mathrm{Var}_{p_X}(x)}. \qquad \text{(C.11)}$$

The constraint that $m\leqslant L$ and $m\geqslant -L$ only holds true if

$$\mathrm{Cov}_p(x,\eta)\geqslant\left(1-\sigma^2-L\right)\mathrm{Var}_{p_X}(x) \qquad \text{(C.12)}$$

$$\wedge\ \mathrm{Cov}_p(x,\eta)\leqslant\left(1-\sigma^2+L\right)\mathrm{Var}_{p_X}(x) \qquad \text{(C.13)}$$

taking into account that $1-\sigma^2>L$.
(IIa): $\lambda>0$:
In this case, we can have either $m=L$ or $m=-L$. Rearranging (C.10) yields

$$\lambda = \frac{1}{m}\left(\left(1-\sigma^2-m\right)\mathrm{Var}_{p_X}(x)-\mathrm{Cov}_p(x,\eta)\right). \qquad \text{(C.14)}$$

From this we deduce that $\lambda>0$ holds if either

* $m=L$ and $\mathrm{Cov}_p(x,\eta)<(1-\sigma^2-L)\mathrm{Var}_{p_X}(x)$, or
* $m=-L$ and $\mathrm{Cov}_p(x,\eta)>(1-\sigma^2+L)\mathrm{Var}_{p_X}(x)$.

Exploiting (C.9) yields b in either case, which provides the desired f^*. In combination with the observation that m and b are uniquely determined, the proof is complete. □

Appendix D. Additional numerical experiments

The numerical results in section 5 are illustrated for the convolution operator $A=M_a$. In the following, additional illustrations for the convolution operator and all corresponding results for the Radon operator, described in section 5, are provided.

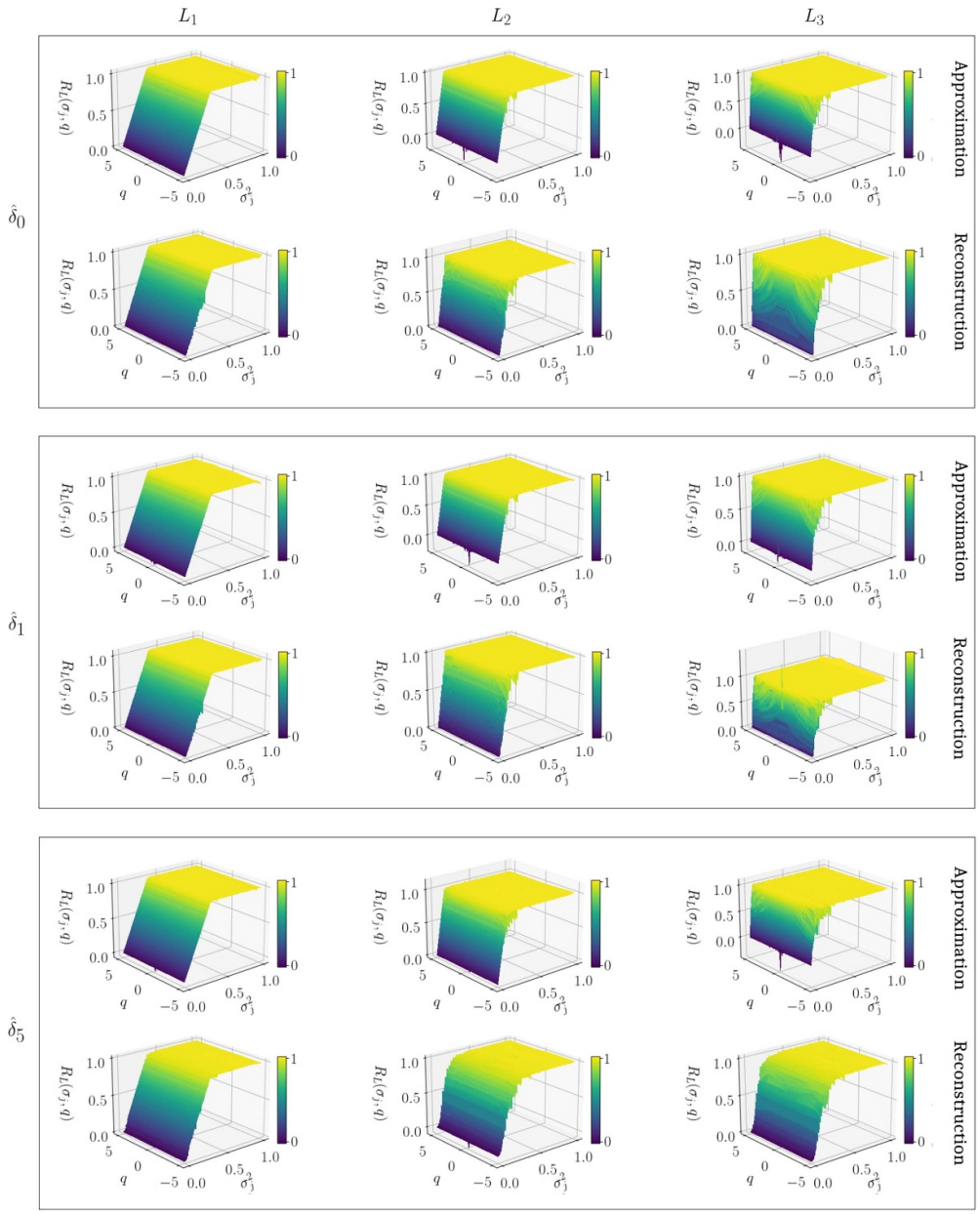
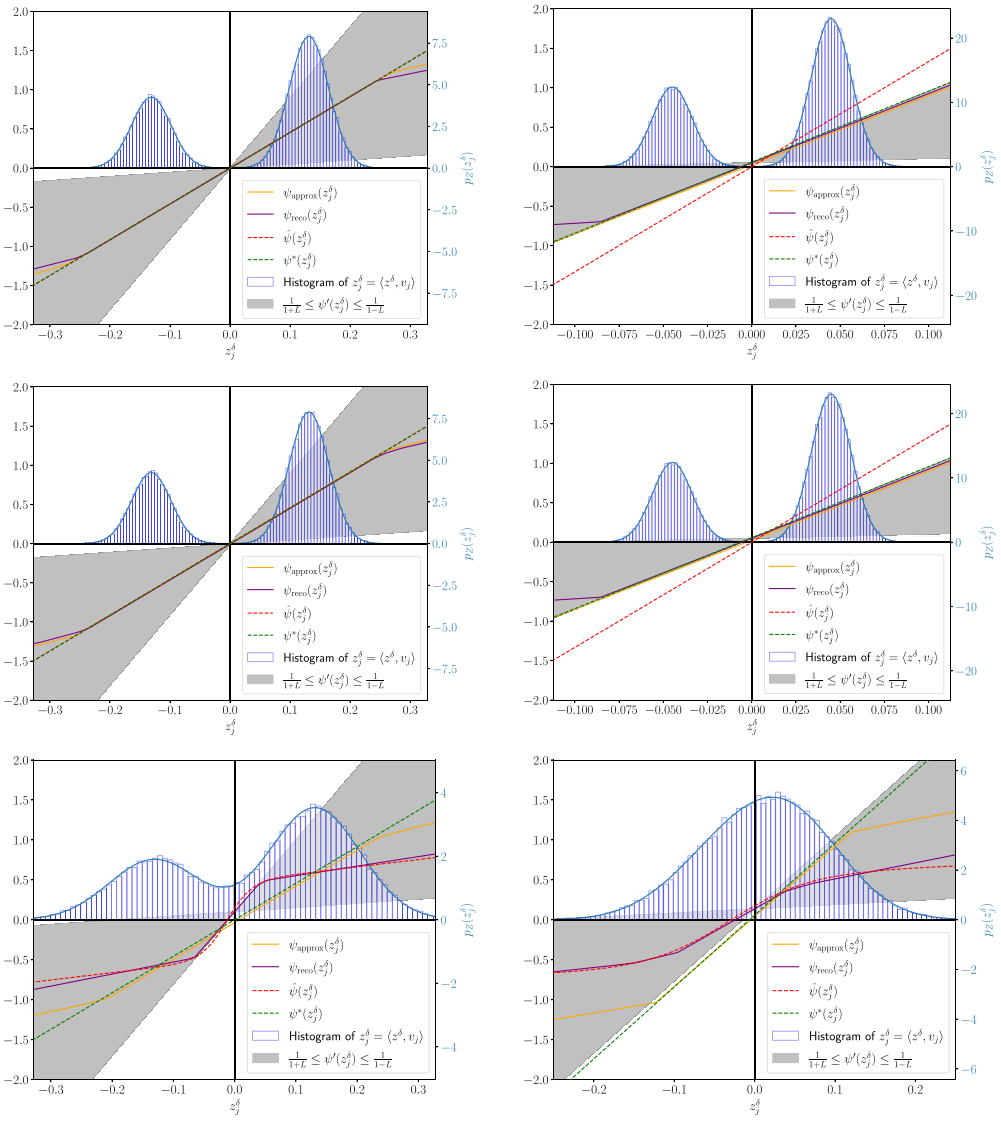


Figure 11. Filter functions $R_L(\sigma_j, q)$ as defined in (5.9) corresponding to trained networks $\varphi_{\theta(L_m,\delta_\ell)}$ for $m = 1,2,3$ *(columns)* and $\ell = 0,1,5$ *(rows)*, trained via approximation training *(top)* and via reconstruction training *(bottom)* on the MNIST dataset for $A = M_a$.



(a) $\sigma_j^2 = 0.219,\ j = 8$ (b) $\sigma_j^2 = 0.075,\ j = 51$

Figure 12. Reconstructions $\psi^*_{\text{approx}}(z_j^\delta)$ trained via approximation training and $\psi^*_{\text{reco}}(z_j^\delta)$ trained via reconstruction training at Lipschitz bound L_2 for different singular values and for noise levels 'zero' (δ_0, *top row*), 'small' (δ_1, *middle row*) and 'large' (δ_5, *bottom row*) for $\tilde{A}$ the Radon operator.

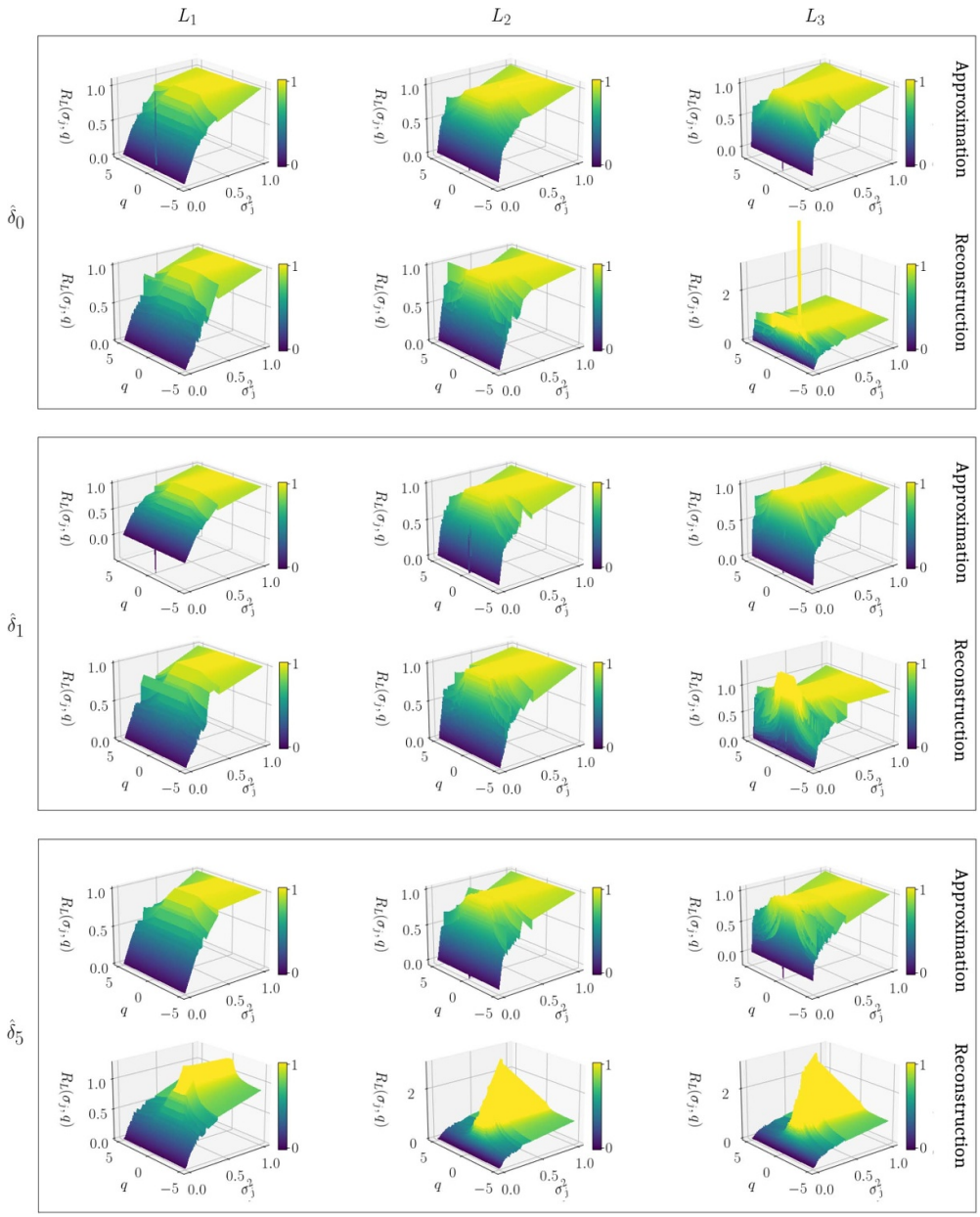


Figure 13. Filter functions $R_L(\sigma_j, q)$ as defined in (5.9) corresponding to trained networks $\varphi_{\theta(L_m,\delta_\ell)}$ for $m = 1, 2, 3$ *(columns)* and $\ell = 0, 1, 5$ *(rows)*, trained via approximation training *(top)* and via reconstruction training *(bottom)* on the bimodal dataset for $\tilde{A}$ the Radon operator.

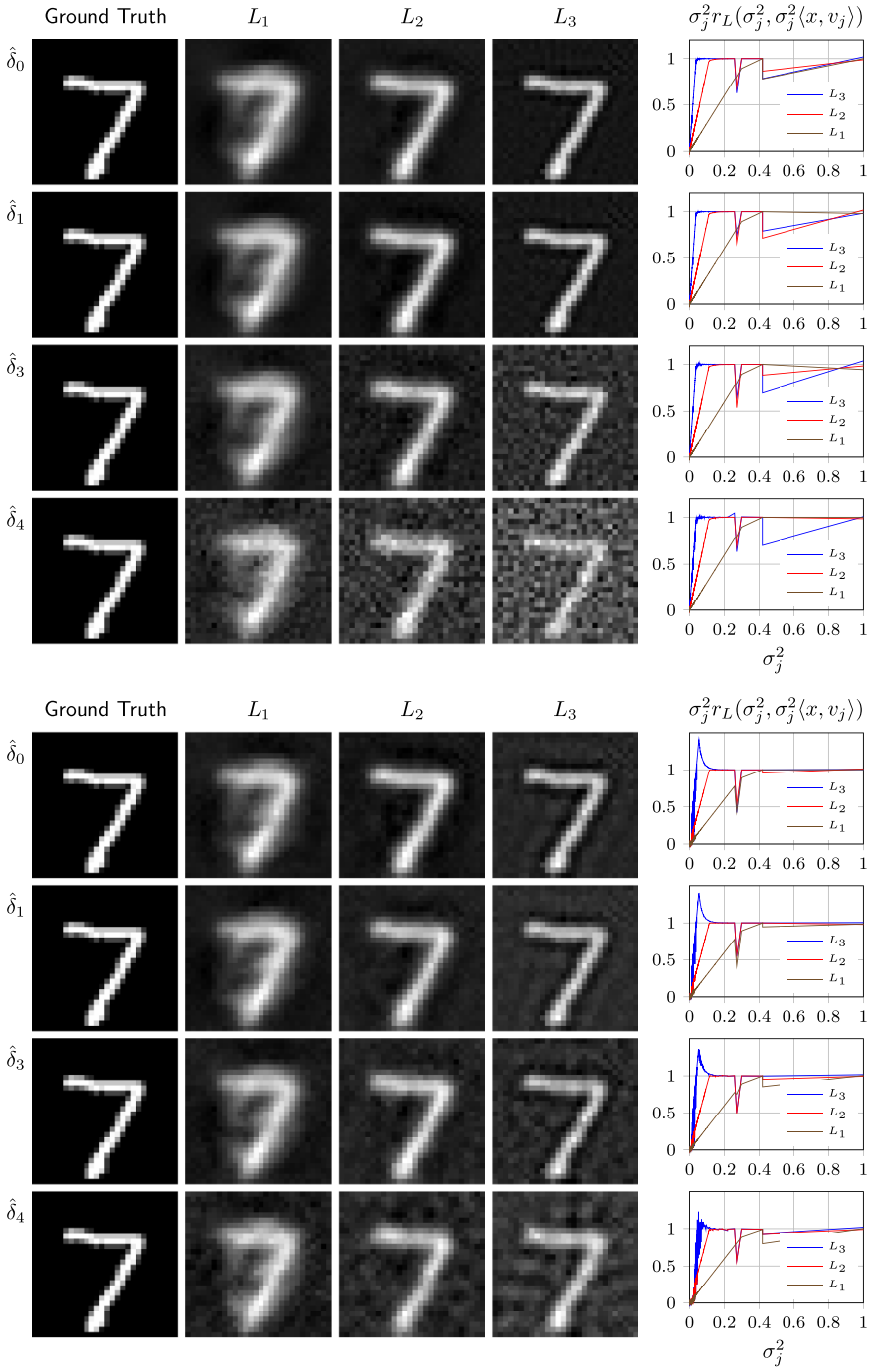


Figure 14. Reconstructions of an MNIST sample $x = x^{(1)}$ from the test dataset by computing $\varphi^{-1}_{\theta(L_m,\delta_\ell)}(Ax + \tilde{\eta})$ with $\tilde{\eta} \sim \mathcal{N}(0, \delta_\ell \mathrm{Id})$ for Lipschitz bounds L_m with $m = 1, 2, 3$ *(columns)* and noise levels $\delta_\ell = \hat{\delta}_\ell \cdot \mathrm{std}_{\mathrm{MNIST}}$ with $\ell = 0, 1, 3, 4$ *(rows)* together with corresponding filter functions for $\tilde{A}$ the Radon operator. The top subfigure depicts the reconstructions from networks trained via approximation training, and the bottom subfigure corresponds to the networks optimized via reconstruction training.

Table 2. SSIM and MSE measures corresponding to reconstructions of $x^{(1)}$ in figure 14. Bold values indicate the best reconstruction quality with respect to the corresponding error measure for a given noise level.

	SSIM			MSE		
Approximation training	L_1	L_2	L_3	L_1	L_2	L_3
δ_0	0.3950	0.7202	**0.8671**	0.0307	0.0117	**0.0023**
δ_1	0.3913	0.7083	**0.8521**	0.0306	0.0120	**0.0025**
δ_3	0.3871	0.6569	**0.6620**	0.0306	0.0129	**0.0087**
δ_4	0.3774	**0.5480**	0.5033	0.0311	**0.0190**	0.0459
	SSIM			MSE		
Reconstruction training	L_1	L_2	L_3	L_1	L_2	L_3
δ_0	0.4161	0.6899	**0.8545**	0.0290	0.0124	**0.0029**
δ_1	0.3989	0.6865	**0.8464**	0.0304	0.0126	**0.0031**
δ_3	0.3913	0.6512	**0.7267**	0.0300	0.0134	**0.0055**
δ_4	0.3934	0.5990	**0.6254**	0.0298	0.0149	**0.0126**

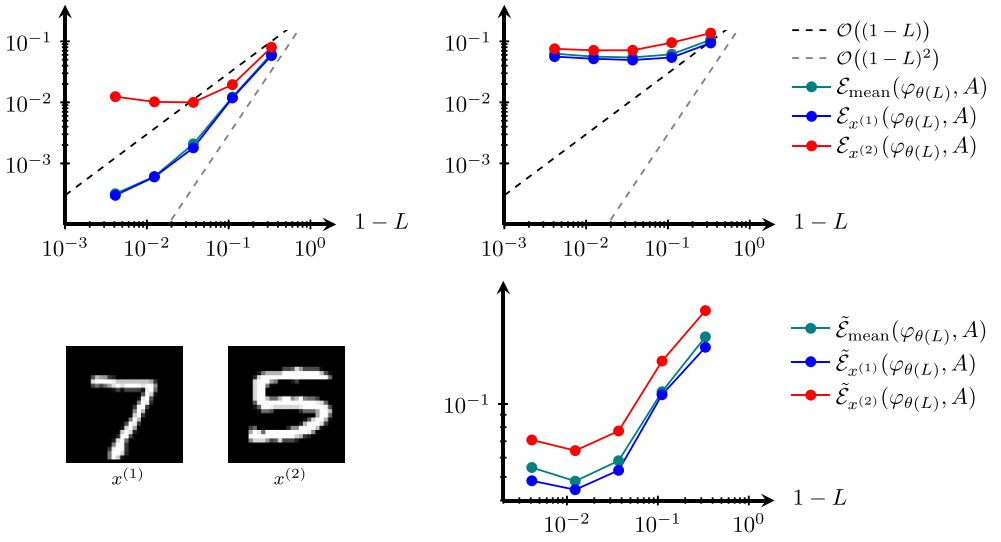


Figure 15. Test samples $x^{(1)}$ and $x^{(2)}$ *(bottom left)*. Evaluations of the local approximation property via $\mathcal{E}_{\text{mean}}(\varphi_{\theta(L_m)},A)$, $\mathcal{E}_{x^{(1)}}(\varphi_{\theta(L_m)},A)$ and $\mathcal{E}_{x^{(2)}}(\varphi_{\theta(L_m)},A)$ for the approximation training *(top left)* and the reconstruction training *(top right)*, and evaluations of the generalized approximation property via $\tilde{\mathcal{E}}_{\text{mean}}(\varphi_{\theta(L_m)},A)$, $\tilde{\mathcal{E}}_{x^{(1)}}(\varphi_{\theta(L_m)},A)$ and $\tilde{\mathcal{E}}_{x^{(2)}}(\varphi_{\theta(L_m)},A)$ for the reconstruction training *(bottom right)* for $L_m = 1 - 1/3^m$ with $m = 1,\dots,5$ and $\tilde{A}$ the Radon operator on the MNIST test dataset.

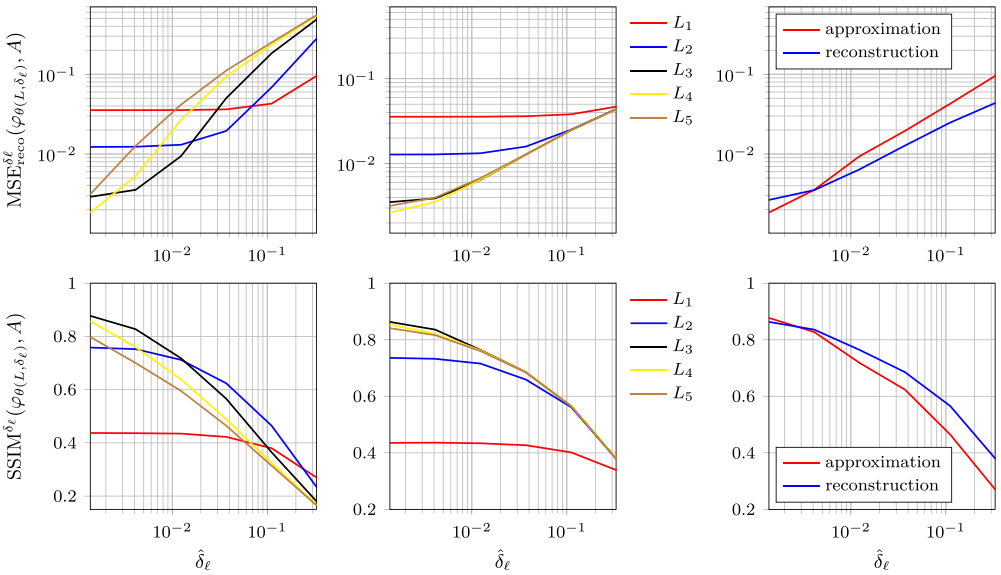


Figure 16. Reconstruction errors $\mathrm{MSE}^{\delta_\ell}_{\mathrm{reco}}(\varphi_{\theta(L,\delta_\ell)},A)$ *(top row)* and $\mathrm{SSIM}^{\delta_\ell}(\varphi_{\theta(L,\delta_\ell)},A)$ *(bottom row)* for networks trained on noisy samples with noise levels δ_ℓ for $\ell = 0,\dots,6$ and reconstructions from noisy samples of the same noise level for the approximation training *(left)* and for the reconstruction training *(middle)* with Lipschitz bounds L_m on the MNIST dataset for the Radon operator $\tilde{A}$. Outcomes of optimal parameter choices for both training strategies over different noise levels can be seen on the right-hand side.

ORCID iDs

Clemens Arndt https://orcid.org/0000-0001-5607-4074
Nick Heilenkötter https://orcid.org/0000-0001-7693-8618
Meira Iske https://orcid.org/0009-0009-1198-6358
Tobias Kluth https://orcid.org/0000-0003-4814-142X
Judith Nickel https://orcid.org/0000-0001-6599-7540

References

[1] Adler J and Öktem O 2018 Deep Bayesian inversion (arXiv:1811.05910)
[2] Alberti G S, De Vito E, Lassas M, Ratti L and Santacesaria M 2021 Learning the optimal tikhonov regularizer for inverse problems *Advances in Neural Information Processing Systems* vol 34 pp 25205–16
[3] Arndt C, Denker A, Dittmer S, Heilenkötter N, Iske M, Kluth T, Maass P and Nickel J 2023 Invertible residual networks in the context of regularization theory for linear inverse problems *Inverse Problems* **39** 125018
[4] Arridge S, Maass P, Öktem O and Schönlieb C-B 2019 Solving inverse problems using data-driven models *Acta Numer.* **28** 1–174
[5] Behrmann J, Grathwohl W, Chen R T, Duvenaud D and Jacobsen J-H 2019 Invertible residual networks *Int. Conf. on Machine Learning* (PMLR) pp 573–82
[6] Benning M and Burger M 2018 Modern regularization methods for inverse problems *Acta Numer.* **27** 1–111

[7] Bochkina N 2013 Consistency of the posterior distribution in generalized linear inverse problems *Inverse Problems* **29** 095010
[8] Chen R T Q, Behrmann J, Duvenaud D K and Jacobsen J-H 2019 Residual flows for invertible generative modeling *Advances in Neural Information Processing Systems* vol 32, ed H Wallach, H Larochelle, A Beygelzimer, F d'Alché-Buc, E Fox and R Garnett (Curran Associates, Inc.)
[9] Dashti M and Stuart A 2017 *The Bayesian Approach to Inverse Problems* (Springer) pp 311–428
[10] Engl H W, Hanke M and Neubauer A 1996 *Regularization of Inverse Problems* vol 375 (Springer)
[11] Kabri S, Auras A, Riccio D, Bauermeister H, Benning M, Moeller M and Burger M 2022 Convergent data-driven regularizations for CT reconstruction (arXiv:2212.07786)
[12] Kaipio J and Somersalo E 2006 *Statistical and Computational Inverse Problems* vol 160 (Springer)
[13] Kingma D P and Ba J 2015 Adam: a method for stochastic optimization *3rd Int. Conf. on Learning Representations, ICLR 2015 (Conf. Track Proc.) (San Diego, CA, USA, 7–9 May 2015)* ed Y Bengio and Y LeCun
[14] Klenke A 2020 *Probability Theory: A Comprehensive Course* (*Universitext*) (Springer)
[15] Königsberger K 2013 *Analysis 2* (Springer)
[16] Laumont R, Bortoli V D, Almansa A, Delon J, Durmus A and Pereyra M 2022 Bayesian imaging using plug & play priors: When langevin meets tweedie *SIAM J. Imaging Sci.* **15** 701–37
[17] LeCun Y 1998 The MNIST database of handwritten digits (available at: http://yann.lecun.com/exdb/mnist/)
[18] Luenberger D G 1969 *Optimization by Vector Space Methods* (*Wiley Professional Paperback Series*) (Wiley)
[19] Maass P 2019 Deep learning for trivial inverse problem *Compressed Sensing and Its Applications* (Springer)
[20] Miyato T, Kataoka T, Koyama M and Yoshida Y 2018 Spectral normalization for generative adversarial networks (arXiv:1802.05957)
[21] Mukherjee S, Hauptmann A, Öktem O, Pereyra M and Schönlieb C-B 2023 Learned reconstruction methods with convergence guarantees: a survey of concepts and applications *IEEE Signal Process. Mag.* **40** 164–82
[22] Scherzer O, Hofmann B and Nashed Z 2023 Gauss–Newton method for solving linear inverse problems with neural network coders *Sampling Theory Signal Process. Data Anal.* **21** 25
[23] Seierstad A and Sydsaeter K 1977 Sufficient conditions in optimal control theory *Int. Econ. Rev.* **18** 367–91
[24] Sherry F, Celledoni E, Ehrhardt M J, Murari D, Owren B and Schönlieb C-B 2018 Designing stable neural networks using convex analysis and odes (arXiv:1802.05957)
[25] Stuart A M 2010 Inverse problems: a Bayesian perspective *Acta Numer.* **19** 451–559
[26] Vollmer S J 2013 Posterior consistency for Bayesian inverse problems through stability and regression results *Inverse Problems* **29** 125011
[27] Arndt C, Dittmer S, Heilenkötter N, Iske M, Kluth T and Nickel J 2024 iResNet Regularization (available at: https://gitlab.informatik.uni-bremen.de/inn4ip/iresnet-regularization)
[28] Wang Z, Bovik A, Sheikh H and Simoncelli E 2004 Image quality assessment: from error visibility to structural similarity *IEEE Trans. Image Process.* **13** 600–12

Invertible ResNets for Inverse Imaging Problems: Competitive Performance with Provable Regularization Properties

Clemens Arndt* Judith Nickel*

December 17, 2024

Abstract

Learning-based methods have demonstrated remarkable performance in solving inverse problems, particularly in image reconstruction tasks. Despite their success, these approaches often lack theoretical guarantees, which are crucial in sensitive applications such as medical imaging. Recent works by Arndt et al (2023 Inverse Problems 39 125018, 2024 Inverse Problems 40 045021) addressed this gap by analyzing a data-driven reconstruction method based on invertible residual networks (iResNets). They revealed that, under reasonable assumptions, this approach constitutes a convergent regularization scheme. However, the performance of the reconstruction method was only validated on academic toy problems and small-scale iResNet architectures. In this work, we address this gap by evaluating the performance of iResNets on two real-world imaging tasks: a linear blurring operator and a nonlinear diffusion operator. To do so, we extend some of the theoretical results from Arndt et al to encompass nonlinear inverse problems and offer insights for the design of large-scale performant iResNet architectures. Through numerical experiments, we compare the performance of our iResNet models against state-of-the-art neural networks, confirming their efficacy. Additionally, we numerically investigate the theoretical guarantees of this approach and demonstrate how the invertibility of the network enables a deeper analysis of the learned forward operator and its learned regularization.

*Center for Industrial Mathematics, University of Bremen, 28359 Bremen, Germany. Equal contribution. Alphabetical author order. Emails: carndt@uni-bremen.de, junickel@uni-bremen.de

1 Introduction

Inverse problems arise in a wide range of applications, such as image processing, medical imaging, and non-destructive testing. These problems share a common objective: recovering an unknown cause from possibly noisy measurement data by inverting the measurement process (i.e., the forward operator). A major challenge is that the causes often depend discontinuously on the observed data, a characteristic that makes these problems ill-posed. To address this, the problems need to be regularized in order to allow for stable reconstructions. However, this adjustment of the problem must be small to ensure that the solutions remain accurate.

The theory behind classical regularization schemes is well established, and provides strong guarantees regarding their regularization properties [7]. Recently, deep learning based approaches have demonstrated a remarkable performance in solving inverse problems. Many of these methods are build on classical algorithms, e.g., unrolled iterative schemes [14] or plug-and-play methods [27]. However, the theoretical understanding in terms of provable guarantees remains limited. A key concern is the lack of stability in most approaches, which can pose significant risks in safety-critical applications. For this reason, there has been a growing body of research aimed at investigating regularization properties for deep learning methods (cf. "related literature" below).

We tackle the challenges of ill-posedness with a supervised learning approach based on invertible residual networks (iResNets). The basic idea is to approximate the forward operator with the forward pass of the network and use the inverse to solve the inverse problem. A general convergence theory, providing theoretical guarantees for iResNets, has been developed in [2]. In addition, [3] analyzed different training strategies, and investigated to what extend iResNets learn a regularization from the training data. However, this approach has only been tested on simplified toy examples so far. Thus, the question arises whether iResNets are competitive with state-of-the-art methods in real-world tasks.

To address this, we evaluate the performance of iResNets through extensive numerical experiments on both a linear blurring and a nonlinear diffusion problem, comparing the results with state-of-the-art methods. Additionally, we provide valuable insights into designing effective invertible architectures. Beyond the theoretical regularization properties of the iResNet reconstruction approach we demonstrate how the invertibility allows for an interpretation of the learned forward operator and its regularization, thereby reducing the "black-box" nature commonly associated with learned regularization techniques.

The manuscript is organized as follows. We begin in Section 2 by introducing iResNets and extending their regularization theory to encompass nonlinear forward operators. Fol-

lowing this, in Section 3, we discuss the types of problems where iResNets are particularly well-suited, with an emphasis on the problem settings for our numerical experiments. In Section 4, we report our numerical results. First, we describe the details of the invertible architecture in Section 4.1. Then, we compare the reconstruction quality of iResNets with several other common deep learning approaches in Section 4.2. Finally, we investigate the learned regularization by comparing the forward pass of the trained iResNet with the true forward operator in Section 4.3. We conclude in Section 5 with a summary of our findings and suggestions for future research directions.

Related literature

In recent years, as neural networks have achieved remarkable performance, the focus on interpretability and providing theoretical guarantees for deep learning algorithms has gained increasing importance. This is reflected in the growing body of research dedicated to addressing these concerns.

A comprehensive overview of deep learning methods with regularization properties can be found in [16]. Furthermore, [18] presents stability and convergence results for deep equilibrium models. To achieve these results, the learned component of the equilibrium equation must meet specific conditions, which can be satisfied using iResNets, for example. Based on the classical theory of variational regularization, [15, 17] introduced the adversarial convex regularizer. This approach relies on architecture restrictions which enforce strong convexity in order to obtain a stable and convergent regularization scheme.

One notable contribution aiming at enhancing the interpretability of the learned solution is the DiffNet architecture [4], which is specifically designed to model potentially nonlinear diffusion processes. Inspired by the structure of numerical solvers, the network consists of multiple blocks representing discrete steps in time. Within each block, a differential operator, computed by a small subnetwork, is applied to the input. This design results in a highly parameter-efficient architecture that is fast to train and facilitates the analysis and interpretation of the differential operations.

In addition to research focused on providing theoretical guarantees or improving the interpretability of learned solutions, several approaches exhibit structural similarities to iResNets. One such method is the proximal residual flow [11], a ResNet designed to be invertible, where the residual functions are constrained to be averaged operators, differing from the Lipschitz constraint we use in this work. This architecture has been used as a normalizing flow for Bayesian inverse problems. In [24], invertibility is achieved by considering numerical solvers of ordinary differential equations with a monotone right-hand-side, resulting in a ResNet-like

architecture with constraints similar to iResNets. This approach is primarily used to obtain learned nonexpansive denoisers, which are desirable for plug-and-play schemes. Similarly, [22] interprets convolutional ResNets as discretized partial differential equations, deriving stability results for the architecture using monotonicity or Lipschitz conditions.

We note that approaches which achieve both a high numerical performance and strong theoretical guarantees such as stability and convergence are still quite rare in general. Therefore, our iResNet approach provides an advantageous combination of highly desirable properties.

2 iResNets for inverse problems

We study iResNets $\varphi_\theta : X \to X$ with network parameters θ acting on a Hilbert space X, where

$$\varphi_\theta = \varphi_{\theta_1,1} \circ ... \circ \varphi_{\theta_N,N} \tag{2.1}$$

with $N \in \mathbb{N}$ and $\varphi_{\theta_i,i} : X \to X$ for $i \in \{1, \dots, N\}$. Each subnetwork $\varphi_{\theta_i,i}$ is defined as

$$\varphi_{\theta_i,i} = \mathrm{Id} - f_{\theta_i,i} \quad \text{for } i \in \{1, \dots, N\}$$

with Lipschitz continuous residual functions $f_{\theta_i,i} : X \to X$ satisfying $\mathrm{Lip}(f_{\theta_i,i}) \leq L_i < 1$. The bound on the Lipschitz constant of $f_{\theta_i,i}$ allows for an inversion of the subnetworks $\varphi_{\theta_i,i}$ via Banach's fixed-point theorem with fixed-point iteration

$$x^{k+1} = z + f_{\theta_i,i}(x^k)$$

converging to $x = \varphi_{\theta_i,i}^{-1}(z)$ for $z \in X$. Moreover, each subnetwork $\varphi_{\theta_i,i}$ and its inverse $\varphi_{\theta_i,i}^{-1}$ satisfy the Lipschitz bounds

$$\mathrm{Lip}(\varphi_{\theta_i,i}) \leq 1 + L_i \quad \text{and} \quad \mathrm{Lip}(\varphi_{\theta_i,i}^{-1}) \leq \frac{1}{1 - L_i}.$$

We refer the reader to [2, 6] for a detailed derivation of these results. The properties of the subnetworks directly translate to the concatenated network φ_θ making it invertible as each subnetwork is invertible with

$$\mathrm{Lip}(\varphi_\theta) \leq \prod_{i=1}^{N} (1 + L_i) \quad \text{and} \quad \mathrm{Lip}(\varphi_\theta^{-1}) \leq \prod_{i=1}^{N} \frac{1}{1 - L_i}. \tag{2.2}$$

The Lipschitz constant of the inverse φ_θ^{-1} is crucial for ensuring the stability of the reconstruction scheme, as we will elaborate on later. To simplify our discussion, we define a Lipschitz parameter L for the inverse φ_θ^{-1} such that

$$\mathrm{Lip}(\varphi_\theta^{-1}) \leq \frac{1}{1-L} = \frac{1}{\prod_{i=1}^{N}(1-L_i)}. \tag{2.3}$$

The aim is to apply the above defined iResNets to solve ill-posed inverse problems. Given that the input and output spaces of iResNets are inherently identical, we restrict our consideration to problems of the form

$$F(x) = z$$

where $F\colon X \to X$ is a (possibly nonlinear) operator, and both the ground truth data $x \in X$ and the observed data $z \in X$ belong to the same Hilbert space X. For an example of how to transfer a linear operator $\mathcal{A} : X \to Y$ between different Hilbert spaces X and Y to the aforementioned setting, we refer the reader to our earlier publications [2] and [3]. The objective is then to recover the unknown ground truth $x^\dagger \in X$ via iResNets, given noisy measurements $z^\delta \in X$ with noise level $\delta > 0$ satisfying $\|z^\delta - F(x^\dagger)\| \leq \delta$. To do so, we follow the subsequent *reconstruction approach*:

(I) The iResNet $\varphi_\theta : X \to X$ is trained to approximate the forward operator F.

(II) The inverse φ_θ^{-1} is used to reconstruct the ground truth $x^\dagger$ from z^δ.

The task of reconstructing $x^\dagger$ from the data z^δ poses two main challenges making it particularly difficult to solve. First, the solutions $x^\dagger$ may not depend continuously on the data z, rendering the problem ill-posed. Second, as already mentioned, the observed data is usually corrupted by noise. Because of the interplay between these two issues, incorporating prior knowledge about potential solutions $x^\dagger$ is essential to regularize the inverse problem. A regularization scheme aims to address the ill-posedness of the problem by using a reconstruction algorithm that ensures the existence and uniqueness of solutions, stability w.r.t. z^δ and convergence to the ground truth $x^\dagger$ as the noise level δ converges to zero. In particular for nonlinear inverse problems, some of these properties are often hard to achieve, even with classical variational regularization schemes. One important reason for this is that the data discrepancy term $\|F(x) - z^\delta\|^2$ might be non-convex w.r.t. x. Thus, the minimizers of regularization functionals are not guaranteed to be unique and the most common stability results only hold for subsequences and (dependent on the penalty functional) sometimes only with weak convergence [23, Proposition 4.2], [12, Theorem 3.2], [9, Theorem 10.2].

The above defined reconstruction approach based on iResNets effectively tackles the challenges posed by inverse problems, which is mainly attributed to its Lipschitz continuous inverse. To be more precise, the existence and uniqueness of $\varphi_\theta^{-1}(z^\delta)$ is synonymous with the invertibility of φ_θ and thus automatically fulfilled. Moreover, the stability, i.e., continuity of φ_θ^{-1}, directly follows from Equation 2.2, c.f. [2, Lemma 3.1]. The only aspect not directly guaranteed is the convergence of $\varphi_\theta^{-1}(z^\delta)$ to the ground truth as $\delta \to 0$, which requires certain additional prerequisites. Simply put, the convergence depends on the success of the training and the expressivity of the trained network. In [2], a convergence result has been derived in the setting of a linear forward operator and an iResNet consisting of a single subnetwork, i.e. $N = 1$. For this, we made use of an index function generalizing the concept of convergence rates. Recall that an index function is a continuous and strictly increasing mapping $\psi\colon \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ with $\psi(0) = 0$. The convergence result of [2] can be directly generalized to nonlinear forward operators and iResNets as defined in Equation (2.1). To do so, the network parameters θ must depend on the Lipschitz parameter L, as introduced in Equation 2.3. We emphasize this dependency by adapting the notation to $\varphi_{\theta(L)}$.

Theorem 2.1 (Convergence - local approximation property, c.f. [2])**.** *For $x^\dagger \in X$, let $z^\delta \in X$ satisfy $\|z^\delta - F(x^\dagger)\| \leq \delta$. Moreover, assume that the network $\varphi_{\theta(L)}$ with network parameters $\theta(L) \in \Theta(L)$ for $L \in [0, 1)$ satisfies the local approximation property*

$$\|F(x^\dagger) - \varphi_{\theta(L)}(x^\dagger)\| = \mathcal{O}((1-L)\psi(1-L)) \quad (\text{as } L \to 1) \tag{2.4}$$

for some index function ψ.

If the Lipschitz parameter $L : (0, \infty) \to [0, 1)$ is chosen such that

$$L(\delta) \to 1 \quad \wedge \quad \frac{\delta}{1 - L(\delta)} \to 0 \qquad \text{for } \delta \to 0, \tag{2.5}$$

then it holds

$$\|\varphi_{\theta(L(\delta))}^{-1}(z^\delta) - x^\dagger\| \to 0 \qquad \text{for } \delta \to 0.$$

Proof. For simplicity, we abbreviate $\varphi_{\theta(L(\delta))}$ by φ_δ in this proof. The Lipschitz continuity of the inverse, c.f. Equation 2.3, directly implies

$$\begin{aligned}
\|\varphi_\delta^{-1}(z^\delta) - x^\dagger\| &\leq \|\varphi_\delta^{-1}(z^\delta) - \varphi_\delta^{-1}(F(x^\dagger))\| + \|\varphi_\delta^{-1}(F(x^\dagger)) - x^\dagger\| \\
&\leq \frac{1}{1 - L(\delta)}\|z^\delta - F(x^\dagger)\| + \|\varphi_\delta^{-1}(F(x^\dagger)) - \varphi_\delta^{-1}(\varphi_\delta(x^\dagger))\| \\
&\leq \frac{\delta}{1 - L(\delta)} + \frac{1}{1 - L(\delta)}\|F(x^\dagger) - \varphi_\delta(x^\dagger)\|.
\end{aligned}$$

The last inequality yields the desired assertion due to (2.4) and (2.5) in combination with $\lim_{L\to 1} \psi(1-L) = 0$. □

This theorem reveals a relation between the convergence of the iResNet and its approximation capabilities and provides a verifiable condition in form of the local approximation property. A weaker (and sufficient) condition for convergence is given by

$$\|\varphi_{\theta(L)}^{-1}(F(x^\dagger)) - x^\dagger\| \to 0 \quad \text{as } L \to 1, \tag{2.6}$$

c.f. [2, Remark 3.2], which is more in line with the reconstruction training setting introduced in [3] and explained in Section 4.

3 Approximation capabilities of iResNets and problem setting

In this section, we discuss the types of problems for which iResNets are a suitable choice and, in particular, the problem setting of our numerical experiments. Due to their architecture constraints, iResNets cannot approximate arbitrary forward operators $F\colon X \to X$. Each subnetwork can only fit Lipschitz continuous (due to $\mathrm{Lip}(\varphi_{\theta,k}) \leq 1 + L_k$) and monotone (due to $\langle \varphi_{\theta,k}(x), x\rangle = \|x\|^2 - \langle f_{\theta,k}(x), x\rangle \geq (1-L_k)\|x\|^2$) functions. This drawback can be partly compensated by using an architecture with multiple subnetworks, see Section 4.1 for more details. Essentially, iResNets are inherently suited for any problem where the input and output spaces are identical, and learning a deviation from identity is beneficial. This includes applications such as image processing (e.g., denoising, deblurring), problems in the context of partial differential equations (PDEs) or pre-processing and post-processing tasks.

For our numerical experiments, we consider a nonlinear diffusion problem and a linear blurring operator. They are particularly interesting, because several works observed an analogy between ResNets and numerical solvers for ODEs [6, 22, 24]. In the following, we briefly review this relationship by focusing on the invertibility of the iResNet-subnetworks.

Let $X = L^2(\Omega)$ be the space of ground truth images and data with some domain $\Omega \subset \mathbb{R}^n$ and $F\colon X \to X$ be an operator which maps a clean image to a diffused version of it. More precisely, we consider the forward problem $F(u_0) = u(T, \cdot)$ for some $T > 0$, where $u\colon \mathbb{R}_{\geq 0} \times \Omega \to \mathbb{R}$ is the solution of the PDE

$$\begin{aligned} \partial_t u &= \mathrm{div}_x\left(g(|\nabla_x u|)\nabla_x u\right) && \text{on } (0,T]\times\Omega, \\ u(0,\cdot) &= u_0 && \text{on } \Omega, \end{aligned} \tag{3.1}$$

known as Perona-Malik diffusion [19], with $g \in C^1(\mathbb{R}_{\geq 0})$ and zero Dirichlet or zero Neumann boundary condition on $\partial\Omega$.

By considering the explicit Euler method

$$u_{t+1} = u_t + h \cdot \operatorname{div}\left(g(|\nabla_x u_t|)\nabla_x u_t\right)$$

for solving the PDE numerically, a similarity to the structure of a ResNet (identity plus differential operator) can be observed. However, since the differential operator is not continuous w.r.t. $u_t \in L^2(\Omega)$, fitting the contractive residual function $f_{\theta,i}$ of an iResNet-subnetwork to it might be challenging.

This changes, when we consider the implicit Euler method

$$u_{t+1} = u_t + h \cdot \operatorname{div}\left(g(|\nabla_x u_{t+1}|)\nabla_x u_{t+1}\right),$$

which is an elliptic PDE for u_{t+1}. For simplicity, we assume $g \equiv 1$ in the following and analyze the solution operator $S_h \colon L^2(\Omega) \to L^2(\Omega)$, $S_h(u_t) = u_{t+1}$. The following Lemma shows that S_h is suitable for being approximated with iResNet-subnetworks.

Lemma 3.1. *For $g \equiv 1$, the solution operator S_h is linear and firmly nonexpansive, i.e.,*

$$\|S_h u\|^2 + \|(Id - S_h)u\|^2 \leq \|u\|^2.$$

For the proof, see Appendix A.

Fitting S_h with a subnetwork $\varphi_{\theta,i}$ requires that $f_{\theta,i} \approx Id - S_h$. According to the previous lemma, this mapping has a Lipschitz constant of at most one. For most inputs u and v, $\|(Id - S_h)u - (Id - S_h)v\|$ will even be strictly less than $\|u - v\|$. Therefore, S_h is well-suited for approximation using iResNets.

4 Numerical Results

To validate the performance of iResNets in solving inverse problems and provide guidelines on investigating their regularization capabilities, we conduct numerical experiments on two forward operators. As discussed in Section 3, diffusion and blurring operators are particularly suitable for this purpose. Therefore, we consider a linear blurring operator defined as a convolution with a Gaussian kernel with a kernel size of 11×11 and standard deviation of $5/3$. Additionally, we examine an anisotropic nonlinear diffusion problem governed by the PDE in Equation (3.1), utilizing the Perona-Malik filter function $g(|\nabla_x u|) = 1/1+\lambda^{-2}|\nabla_x u|^2$

with contrast parameter $\lambda = 0.1$. The diffused image is obtained after 5 steps of Heun's method with step size 0.15 and zero Neumann boundary conditions.

All networks are trained on pairs of distorted and clean grayscale images from the STL-10 dataset [1], where we use 16 284 images for training, 64 for validation and 128 for testing and evaluation. The distorted images are generated by applying the forward operator to the clean images, combined with additive Gaussian white noise with standard deviation $\delta > 0$. We consider three different levels of noise in our numerical experiments, namely $\delta = 0.01$, $\delta = 0.025$ and $\delta = 0.05$.

In [3], two fundamentally different approaches for training an iResNet to solve an inverse problem are discussed. Both methods require supervised training with paired data (x_i, z_i^δ). The first approach, referred to as *approximation training*, aims at training φ_θ to approximate F via the training objective

$$\min_\theta \sum_i \|\varphi_\theta(x_i) - z_i^\delta\|^2.$$

The second approach, referred to as *reconstruction training*, focuses on training the inverse φ_θ^{-1} to reconstruct the ground truth via the training objective

$$\min_\theta \sum_i \|\varphi_\theta^{-1}(z_i^\delta) - x_i\|^2. \tag{4.1}$$

Reconstruction training is preferable for obtaining a data-driven regularization method, since it results in an approximation of the posterior mean estimator of the training data distribution [3, Lemma 4.2]. In contrast, approximation training overlooks many features of the training data distribution, with its regularizing effect primarily arising from the architectural constraints [3, Theorem 3.1]. For this reason, we perform reconstruction training in our numerical experiments. It is important to note that this approach requires computing the network's inverse during training, which is done via a fixed-point iteration within each subnetwork, c.f. Section 2.

The source code for the experiments in this section is available on GitLab[1].

4.1 Architecture

In this section, we discuss key practical considerations in the design of iResNets to achieve high-performing networks, along with details of the architectures used in our numerical experiments. We tested a small and a big architecture (denoted as "small iResNet" and "iResNet", respectively), both designed according to (2.1). Each residual function $f_{\theta_i,i}$ is

[1] `https://gitlab.informatik.uni-bremen.de/junickel/iresnets4inverseproblems`

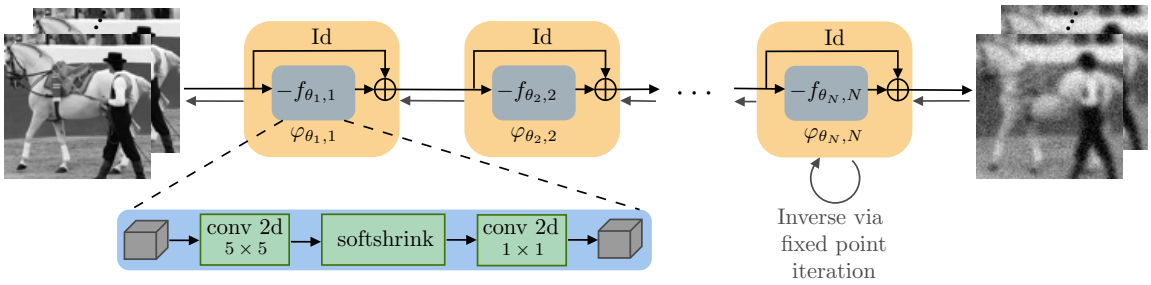


Figure 1: Architecture of the iResNets used in our numerical experiments.

defined as a 5×5-convolution followed by a soft shrinkage activation function with learnable threshold and a 1×1-convolution. Soft shrinkage activation is very suitable for the estimation of Lipschitz constants because, unlike ReLU, its lokal Lipschitz constant equals its global Lipschitz constant everywhere except at a small interval. More details about both architectures are listed in Table 1 and a visualization of the network design is depicted in Figure 1.

We made three different choices for the Lipschitz parameter of the architectures, i.e., $L = 0.95$, $L = 0.99$ and $L = 0.999$, to be able to test different levels of stability. Unless otherwise stated, the reported results correspond to the network with the highest L because it is the least restricted. To control the Lipschitz constants of the residual functions $f_{\theta_i,i}$, we need to regulate the operator norms of each convolution, which can be efficiently computed with power iterations. Thus, before performing a convolution, we apply a projection to the (trainable) network parameters to obtain convolutional weights which fulfill the desired operator norm. It is important to note that during training, automatic differentiation accounts for both the projection step and the computation of operator norms. This strategy was already used in [3] and it prevents the gradient steps computed by the optimizer to be in conflict with the Lipschitz constraint.

We opted for a relatively large number of subnetworks in our architectures ($N = 12$ for the smaller and $N = 20$ for the larger) while keeping the residual functions $f_{\theta,i}$ fairly shallow. There are several reasons behind this design choice, which we will elaborate on below.

First, the Lipschitz constant of a neural network can be estimated from above by the product of the Lipschitz constants of all layers, but this estimation can be far from tight, as noted in [8]. This issue gets worse as the number of layers increases, which is why we chose shallow sub-architectures to mitigate this problem.

Second, if L_i is chosen to be very close to one, the invertibility condition $\mathrm{Lip}(f_{\theta,i}) < 1$ might in fact be violated due to small numerical errors. This can lead to severe problems when attenpting to invert $\varphi_{\theta,i}$. By using a large number of subnetworks N, we can assign

Table 1: Network and training parameters of our iResNets trained on a system with an AMD EPYC 7742 64-Core processor and a GeForce RTX 3090 GPU.

	iResNet	Small iResNet	DiffNet	U-Net	ConvResNet
Subnetworks/Scales	20	12	5	5	20
Input channels	64	32	1	1	64
Hidden channels	128	64	32	16 - 256	128
Kernel size	5	5	3	5	5
Trainable parameters	4 263 700	640 140	101 310	5 468 705	4 263 873
Learning rate	1×10^{-4}	1×10^{-3}	2×10^{-3}	5×10^{-4}	1×10^{-4}
Epochs	500	300	100	500	300
Batch size	32	32	16	64	16
Optimizer	Adam	Adam	Adam	Adam	Adam
Training time	6-7 days	1-2 days	1 hour	2 hours	15 hours

smaller, more stable values to L_i while still achieving a large overall L value (see (2.3)). For our small architecture with $N = 12$, we can, e.g., choose $L_i \approx 0.438$ to obtain $L = 0.999$.

Third, the residual functions $f_{\theta,i}$ must be evaluated in each fixed point iteration for the inversion of $\varphi_{\theta,i}$. Smaller subnetworks make these computations faster. Additionally, smaller L_i values reduce the number of iterations required to reach a specific accuracy level. Although a higher N increases the number of subnetworks to invert, the overall inversion process is faster with shallow residual functions.

Fourth, a single iResNet-subnetwork has a Lipschitz constant of at most $1 + L_i < 2$ in the forward pass. This implies that $\|\varphi_{\theta,i}^{-1}(z_1) - \varphi_{\theta,i}^{-1}(z_2)\| \geq \frac{1}{2}\|z_1 - z_2\|$, which limits the regularization capabilities of $\varphi_{\theta,i}^{-1}$ since it cannot map different data z_1, z_2 arbitrarily close to the same solution $x^\dagger$. Increasing the number of subnetworks alleviates this issue.

Fifth, while individual subnetworks are always monotone, concatenating several of them allows for fitting increasingly non-monotone functions, thereby expanding the set of suitable forward operators.

Finally, as we consider the solution operator of a PDE (see Section 3) as the forward operator for our numerical experiments, a concatenated architecture is a natural choice.

A drawback of using multiple concatenated shallow subnetworks is that the number of channels (i.e., the network's width) can only be expanded within the residual functions. The input and output dimension of all subnetworks has to be the same. To overcome this limitation, we lift the inverse problem $F\colon X \to X$ to a multichannel problem $\tilde{F}\colon X^M \to X^M$, $\tilde{F}(x_j)_k = F(x_j)$ for $j, k = 1, ..., M$. Input and target images $z^\delta, x^\dagger \in X$ for training and evaluation of φ_θ are accordingly stacked to multichannel representations $(z^\delta, ..., z^\delta), (x^\dagger, ..., x^\dagger) \in X^M$. The mapping of the network's output back to the original space X is simply implemented as a mean over all channels. Note that in this setting φ_θ is

not an invertible mapping on X but on X^M.

To perform reconstruction training (4.1), the gradients of φ_θ^{-1} are required. We use the idea of deep equilibrium models [5] to avoid the memory-intensive process of backpropagation through the potentially large number of fixed point iterations. This means that the derivative of the fixed point x of

$$x - f_{\theta,i}(x) = z$$

w.r.t. θ and z is directly calculated via the implicit function theorem. In [10], this technique has already been used for solving inverse problems to simulate an infinite number of layers in unrolled architectures.

4.2 Performance and comparison to other models

In this section, we assess the performance of the iResNet reconstruction scheme and compare it with both traditional and deep-learning-based reconstruction methods.

The performance of the iResNet and the small iResNet w.r.t. PSNR and SSIM across all three noise levels δ depending on the Lipschitz parameter L is illustrated in Figure 2 for the nonlinear diffusion operator and in Figure 3 for the linear blurring operator. One can observe that the reconstruction performance increases as the Lipschitz parameter L approaches 1 for both forward operators and across all noise levels. Additionally, the larger iResNet outperforms the smaller iResNet at Lipschitz parameters close to 1, while their performance is similar on average at lower Lipschitz parameters. This behavior is expected, as the Lipschitz constraint reduces the network's expressiveness, particularly at smaller Lipschitz parameters. Based on these observations, the following investigations will focus on the highest Lipschitz parameter, $L = 0.999$.

To further assess the reconstruction quality of our iResNets, we compare their performance against three other deep learning methods and the classical TV regularization. Specifically, we implement a convolutional ResNet (ConvResNet) without Lipschitz constraints on the residual functions, exhibiting the same network architecture as our iResNet, but with an additional learnable 1×1 convolution at the beginning and end of the network. This modification overcomes the need to transfer the problem to a multichannel problem as discussed in Section 4.1. Additionally, we employ the widely recognized U-Net architecture [21] as well as the DiffNet from [4], a specialized architecture based on non-stationary filters designed specifically for diffusion problems. We optimize the hyperparameters of our iResNets, TV, ConvResNet and U-Net on the validation set using random search. The hyperparameters of DiffNet are adapted from [4], as they focus on a similar forward operator. The architecture, training parameters and corresponding training times for all networks are detailed in Table 1.

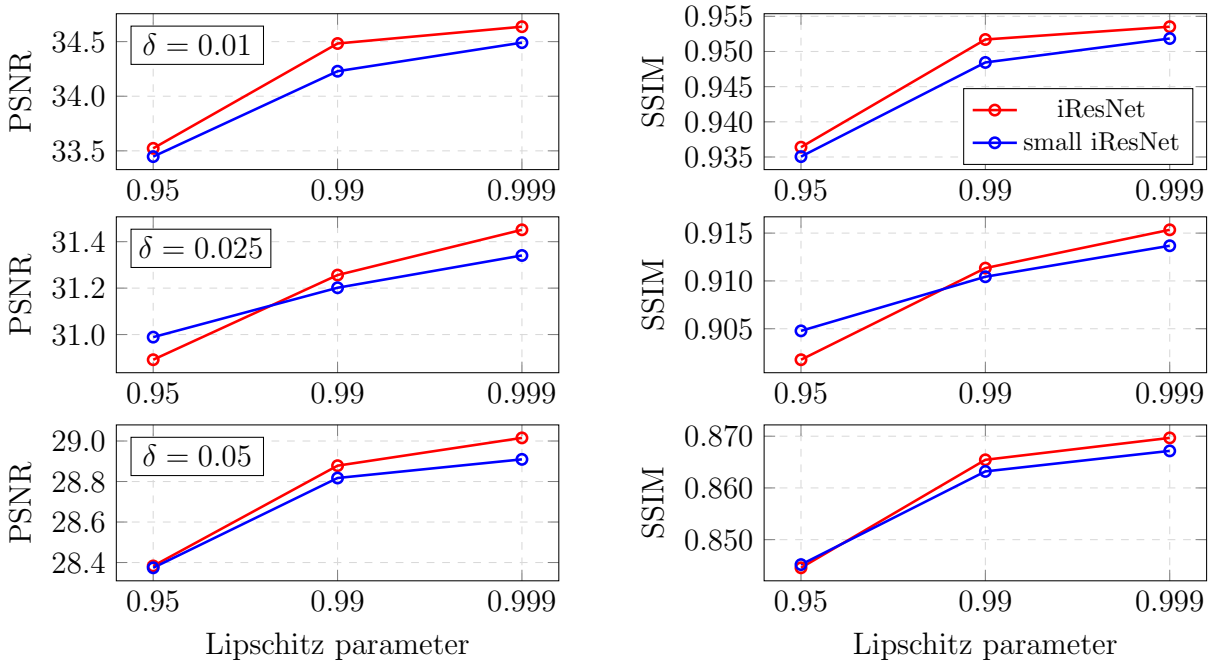


Figure 2: Reconstruction performance of iResNets dependent on the Lipschitz parameter L for the **nonlinear diffusion operator** and different noise levels ($\delta = 0.01, 0.025, 0.05$).

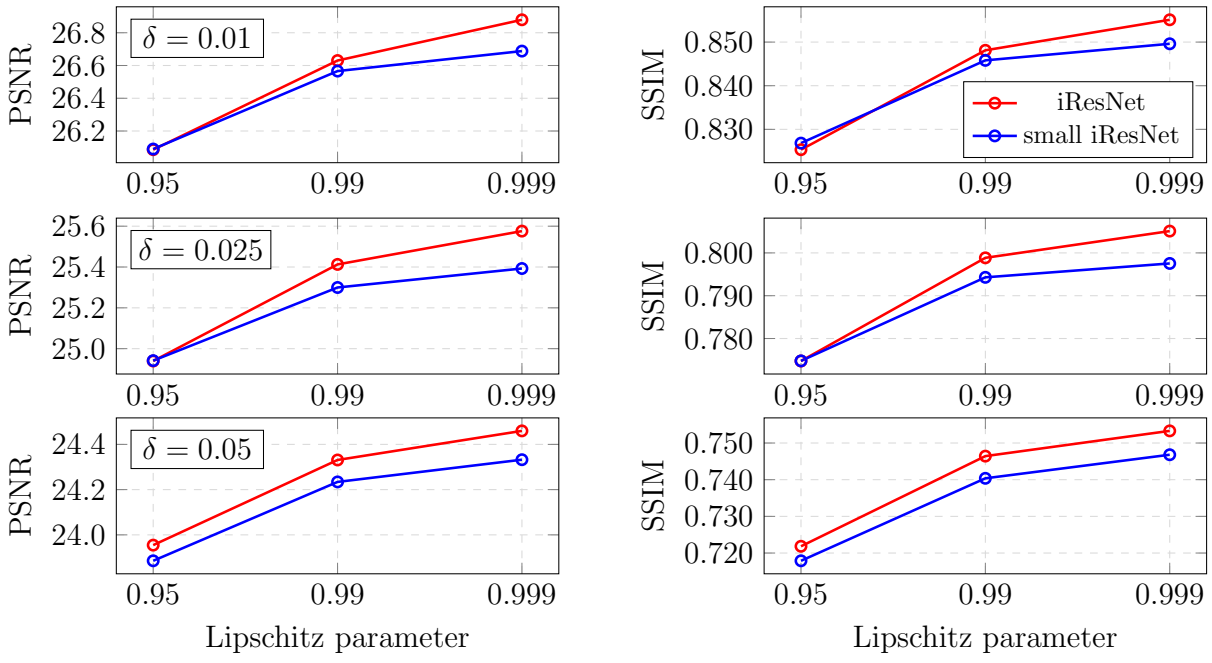


Figure 3: Reconstruction performance of iResNets dependent on the Lipschitz parameter L for the **linear blurring** and different noise levels ($\delta = 0.01, 0.025, 0.05$).

One can observe that the DiffNet has the fewest trainable parameters, followed by the small iResNet, while the (larger) iResNet, U-Net and ConvResNet have a similar and significantly higher number of parameters. Moreover, there is a considerable difference in training times: DiffNet and U-Net converge within $1-2$ hours, whereas our iResNets require several days up to a week to converge. This is caused by the combination of the reconstruction training objective and the network's invertibility, c.f. Section 4.1.

The reconstruction quality in terms of PSNR and SSIM for all methods at different noise levels is depicted in Figure 4 for the nonlinear diffusion and in Figure 5 for the linear blurring. The networks substantially outperform the classical TV reconstruction, with the ConvResNet achieving the highest SSIM and PSNR values across both forward operators. Our iResNet exhibits a similar performance than DiffNet and U-Net. Notably, for the nonlinear diffusion operator, the iResNet performs particularly well at high noise levels, while for the linear blurring it surpasses both DiffNet and U-Net at all noise levels. The small iResNet achieves slightly lower reconstruction quality compared to the larger iResNet, but it still significantly outperforms TV reconstruction.

The quality of reconstructions produced by our iResNets is further validated through comparative examples, as can be seen in Figure 6 for the nonlinear diffusion and in Figure 7 for the linear blurring at both the lowest and highest noise level. For both types of forward operators, the visual performance of our iResNet closely matches that of the ConvResNet, U-Net and DiffNet. Moreover, the TV reconstructions exhibit severe artifacts, especially in the case of high noise and with the linear blurring operator. As anticipated, all methods struggle to recover fine details at high noise levels.

These results demonstrate that our iResNets perform on par with state-of-the-art methods in the literature, albeit with a significantly longer training time due to their invertibility. Among the compared methods, DiffNet appears to strike the best balance between the number of trainable parameters, training time, and performance. However, DiffNet is specifically designed for diffusion problems, whereas our iResNets are more versatile and can be applied to a broader range of forward operators, c.f. Section 3. Additionally, iResNets provide theoretical guarantees, and their invertibility allows for the interpretation of the learned operator, as we will discuss in the following section. Due to the superior performance of our larger iResNet compared to its smaller version, we will focus our subsequent investigations on the former.

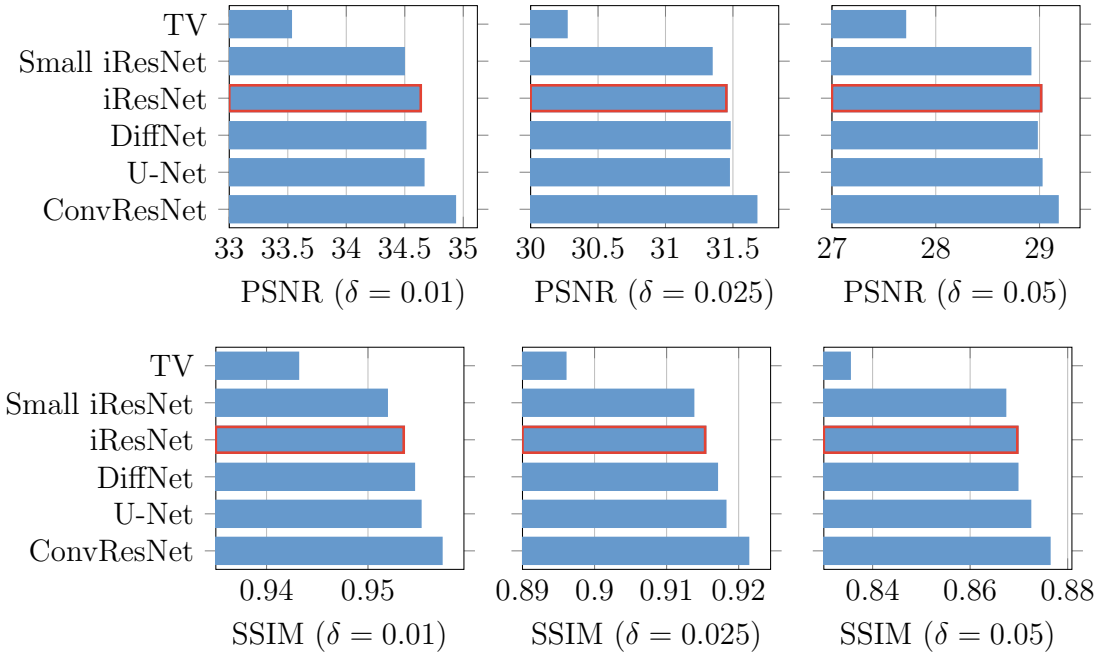


Figure 4: Comparison of the reconstruction quality (top: PSNR, bottom: SSIM) between the iResNet architectures (with $L = 0.999$) and other models for the **nonlinear diffusion operator** across various noise levels.

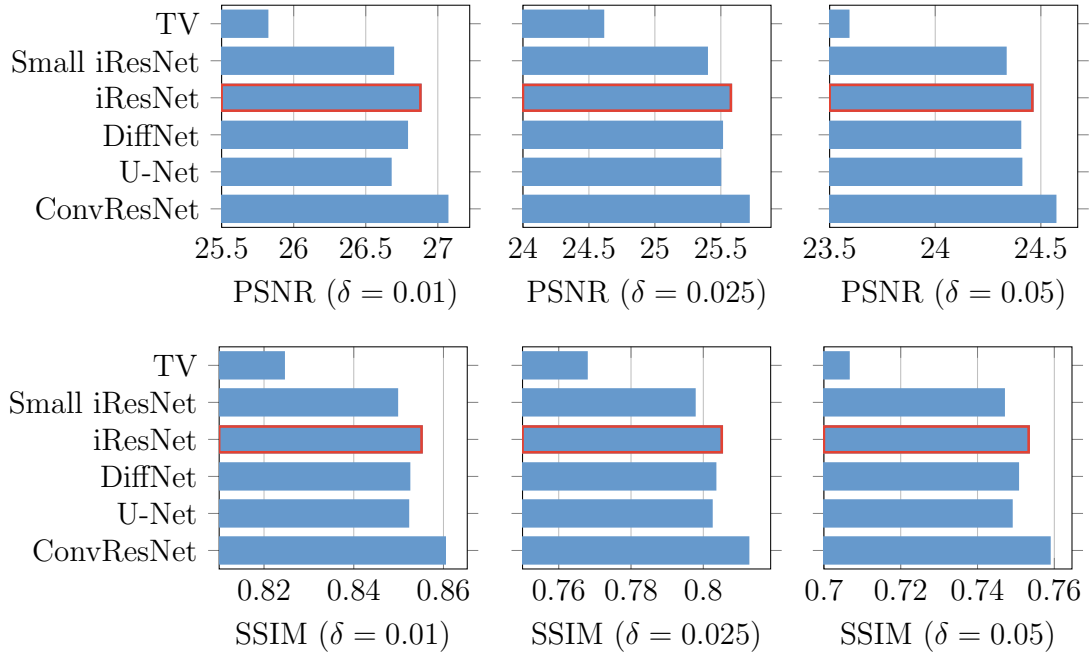


Figure 5: Comparison of the reconstruction quality (top: PSNR, bottom: SSIM) between the iResNet architectures (with $L = 0.999$) and other models for the **linear blurring operator** across various noise levels.

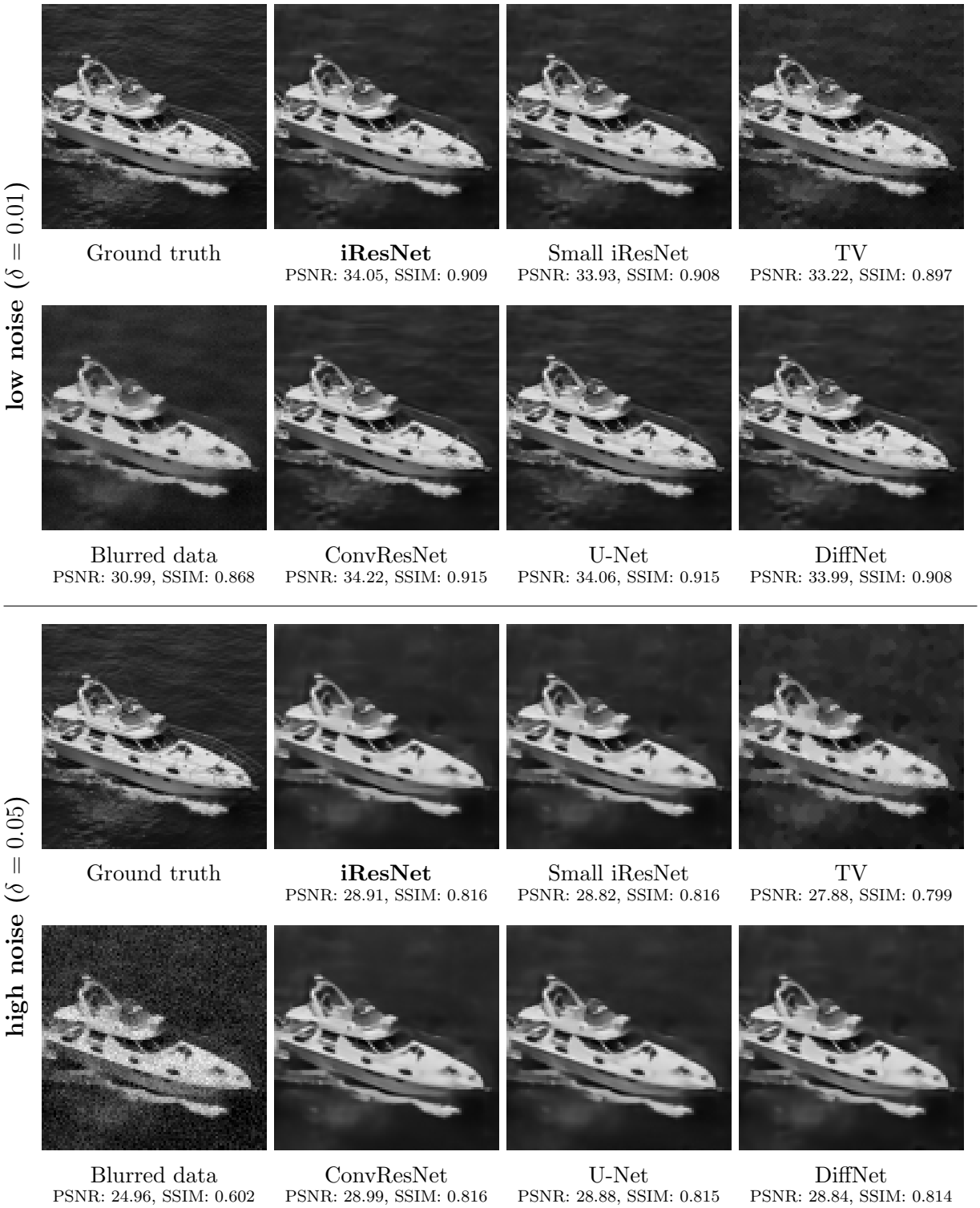
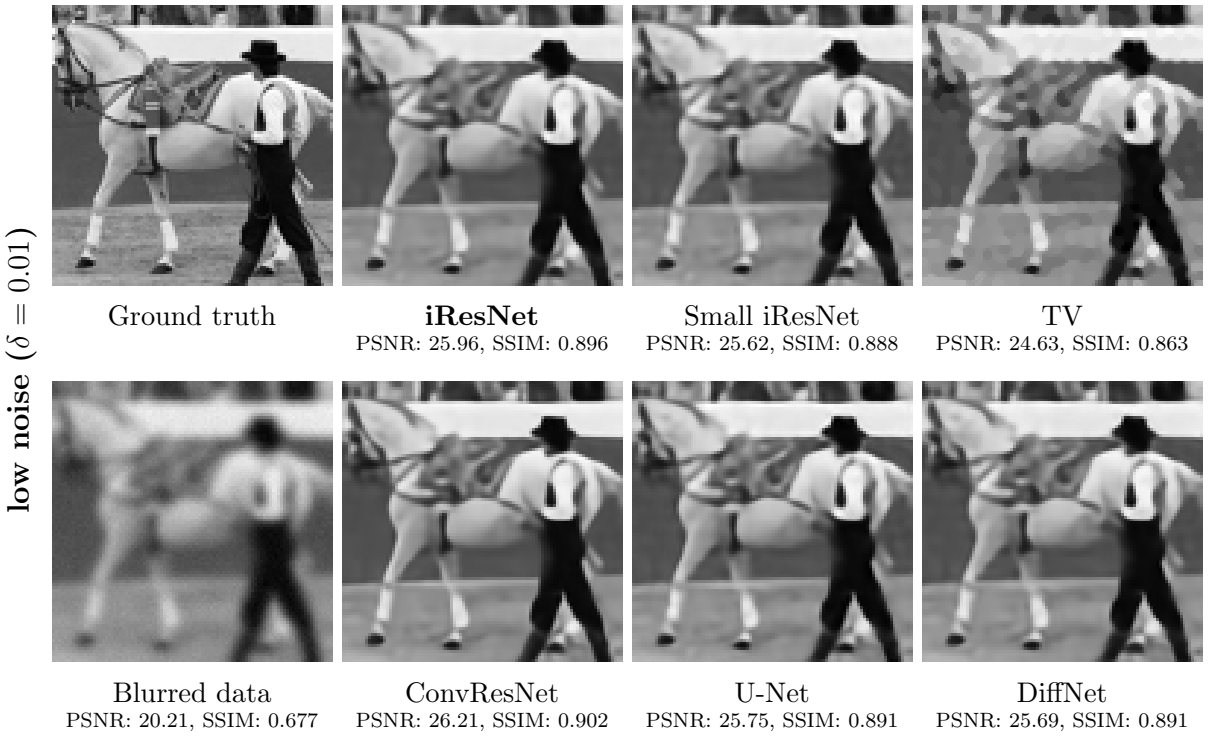


Figure 6: Comparison of reconstructions from different methods for the **nonlinear diffusion operator**.



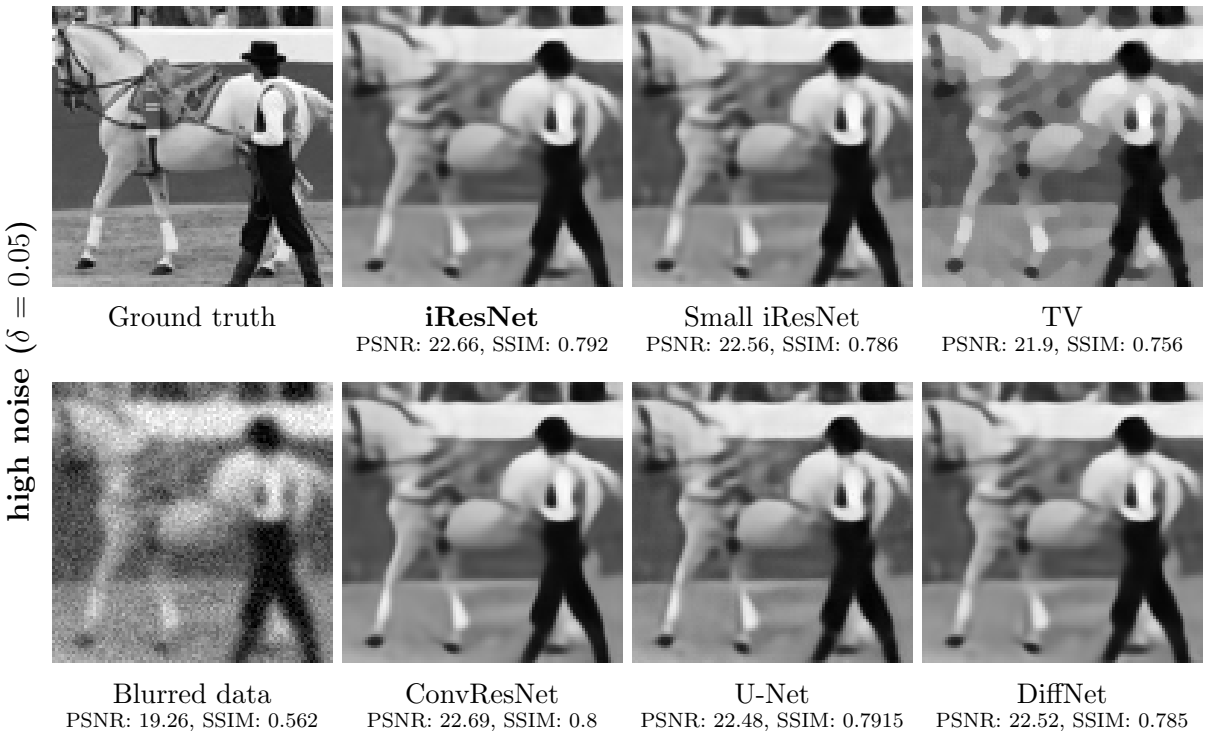


Figure 7: Comparison of reconstructions from different methods for the **linear blurring operator**.

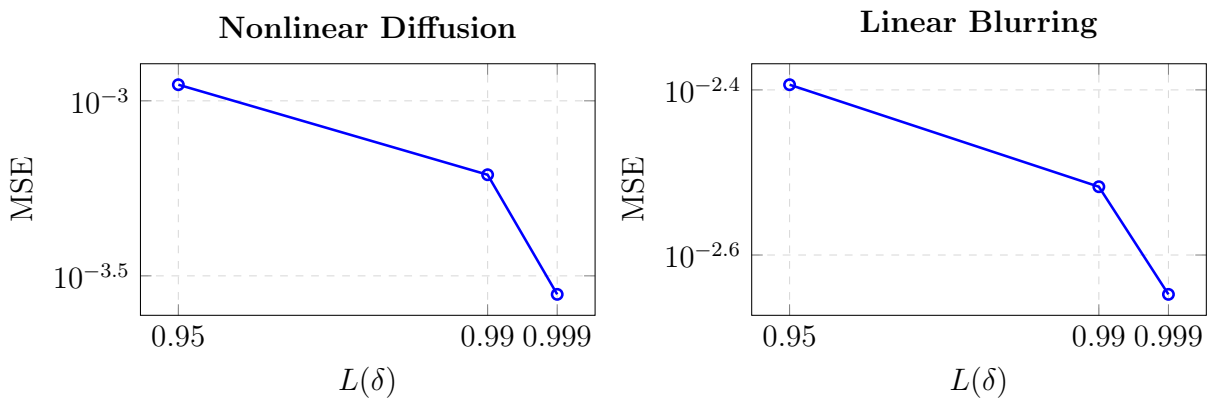


Figure 8: Mean inversion error $\|\varphi^{-1}_{\theta(L),L}(F(x)) - x\|$ on the test set for the nonlinear diffusion (left) and the linear blurring operator (right). The parameter choice $L(\delta)$ required for our convergence result in Theorem 2.1 and Equation (2.6) is given by $L(0.01) = 0.999, L(0.025) = 0.99, L(0.05) = 0.95$. To be more precise, for $L = 0.999$ the network was trained with $\delta = 0.01$, for $L = 0.99$ with $\delta = 0.025$ and for $L = 0.95$ with $\delta = 0.05$.

4.3 Investigation of regularization properties

In the previous section, we demonstrated that the proposed iResNets achieve competitive reconstruction performance. Unlike most other methods, iResNets also come with theoretical guarantees, as explained in Section 2. Specifically, they ensure the existence and uniqueness of solutions $\varphi_\theta^{-1}(z^\delta)$, as well as the stability of the resulting reconstruction scheme w.r.t. z^δ. The last criterion to verify for the iResNet reconstruction approach to qualify as a regularization scheme is the convergence of $\varphi_\theta^{-1}(z^\delta)$ to the ground truth $x^\dagger$ as the noise level δ approaches zero. In Theorem 2.1, we derived a convergence result for iResNets based on the local approximation property (2.4), which we relaxed in Equation (2.6) by considering the inversion error $\|\varphi^{-1}_{\theta(L),L}(Fx^\dagger) - x^\dagger\|$ as $L(\delta) \to 1$ for $\delta \to 0$ with a suitable parameter choice rule $L(\delta)$. We depict the average inversion error on the test set depending on $L(\delta)$ for the nonlinear diffusion (left) and the linear blurring (right) in Figure 8. It can be observed that the error decreases as $L(\delta) \to 1$ for both forward operators, indicating that the iResNet reconstruction approach meets the conditions for a convergent regularization scheme on average.

Besides these theoretical guarantees, the proposed reconstruction approach allows for accessing the learned forward operator. This opens the door to an in-depth investigation of the nature of the learned regularization. To start with, observe that the accuracy of the approximation of the true forward operator, w.r.t. MSE and SSIM, increases for all Lipschitz parameters L as the noise level δ descreases, which can be seen in Figure 9 for the nonlinear diffusion and the linear blurring operator. Moreover, one can observe that

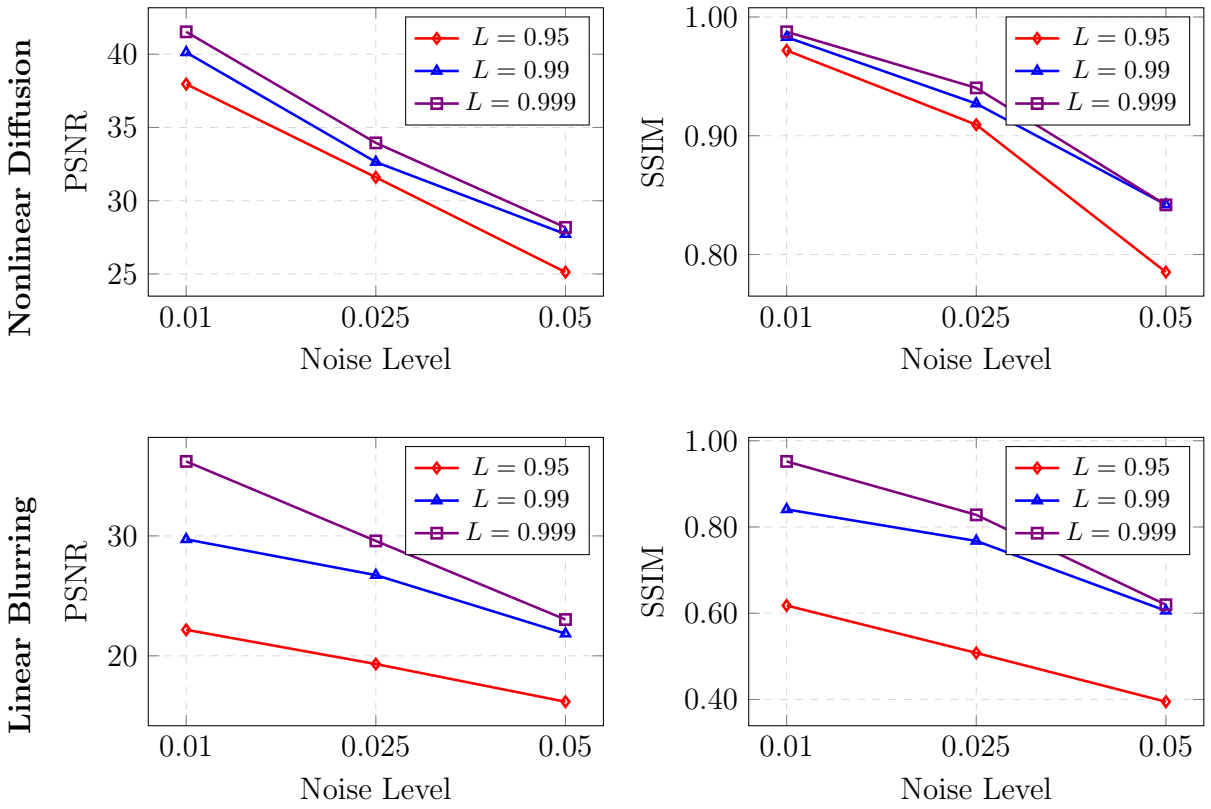


Figure 9: Approximation accuracy of the true forward operator by iResNets trained using reconstruction training on data with varying noise levels (top row: **nonlinear diffusion**, bottom row: **linear blurring**).

the best approximation performance, on average, is achieved with the highest Lipschitz parameter. Consequently, selecting a Lipschitz parameter close to one is beneficial for both target reconstruction and forward operator approximation. For this reason, we will focus all subsequent investigations on $L = 0.999$.

Examples of the true forward operator and the learned forward operator for $\delta = 0.01$ as well as $\delta = 0.05$ can be seen in Figure 10. One can observe that the learned forward operator resembles the true forward operator for small noise ($\delta = 0.01$), displaying slightly less blurring or diffusion in both linear and nonlinear cases. In contrast, at a higher noise level ($\delta = 0.05$), the learned forward operators tend to over-amplify details, especially edges in the images. The reason for this is that for high noise levels the learned operators need to exhibit a stronger regularization to be able to stably reconstruct the target images.

To provide a deeper understanding of the learned regularization, we present two approaches aiming at shedding some light on its nature. To do so, we first compare the local ill-posedness of the true forward operators F with the learned operators φ_θ in Section 4.3.1

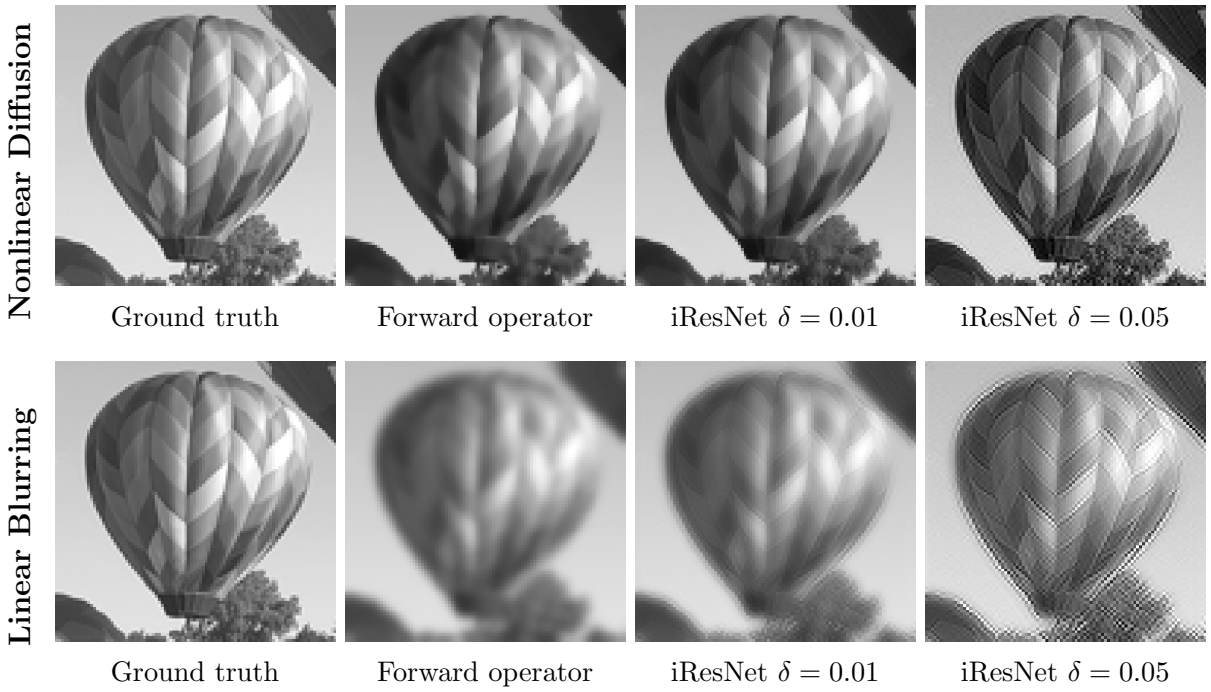
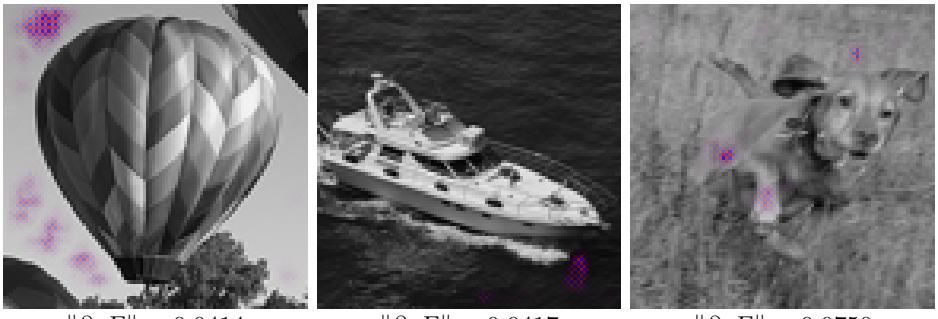


Figure 10: Comparison of the forward operator (top row: **nonlinear diffusion**, bottom row: **linear blurring**) with the trained iResNets for different noise levels. From left to right: Ground truth image, blurred image using the forward operator with no added noise as well as the output (forward pass) of the network trained with Lipschitz parameter $L = 0.999$ for noise level 0.01 and 0.05.

and we then analyze the behavior of φ_θ by clustering blurring kernel emerging from a linearization of the network in Section 4.3.2. This approach allows us to understand how the networks respond to various image structures. We would like to note that we present the results of our investigations solely for the nonlinear diffusion operator, as the results were comparable for the linear blurring operator. Moreover, we emphasize that while this paper focuses on deblurring tasks, the proposed approaches are applicable to various problems and should be viewed as exemplary methods for numerically investigating the regularizing behavior of iResNets.

4.3.1 Directional derivatives and local ill-posedness

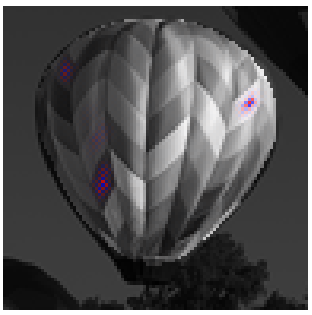
A nonlinear inverse problem is called locally ill-posed in a point $x_0 \in X$, if any open neighborhood of x_0 contains a sequence (x_k) such that $F(x_k) \to F(x_0)$ but $x_k \not\to x_0$ [23, Definition 3.15]. For differentiable F, this often entails that some directional derivatives $\partial_h F(x_0) = F'(x_0)h$ are small. Accordingly, for our nonlinear diffusion operator F we expect $\|\partial_h F(x_0)\|$ to be small for certain directions $h \in X$, $\|h\| = 1$, dependent on the input image



$\|\partial_h F\| = 0.0414$
$\|\partial_h \varphi_\theta\| = 1.27$

$\|\partial_h F\| = 0.0417$
$\|\partial_h \varphi_\theta\| = 1.24$

$\|\partial_h F\| = 0.0750$
$\|\partial_h \varphi_\theta\| = 1.00$



$\|\partial_h F\| = 0.0532$
$\|\partial_h \varphi_\theta\| = 1.18$

$\|\partial_h F\| = 0.126$
$\|\partial_h \varphi_\theta\| = 0.675$

$\|\partial_h F\| = 0.165$
$\|\partial_h \varphi_\theta\| = 0.921$

Figure 11: Computation of directions h for which the difference $\|\partial_h \varphi_\theta(x_0)\| - \|\partial_h F(x_0)\|$ is particularly large (φ_θ trained for the **nonlinear diffusion operator** with $\delta = 0.01$). The direction vector h is displayed in color on the image x_0 (grayscale). It can be interpreted as a direction in which the network has learned a significant regularization. In the bottom row, h is restricted to the subject of the image x_0.

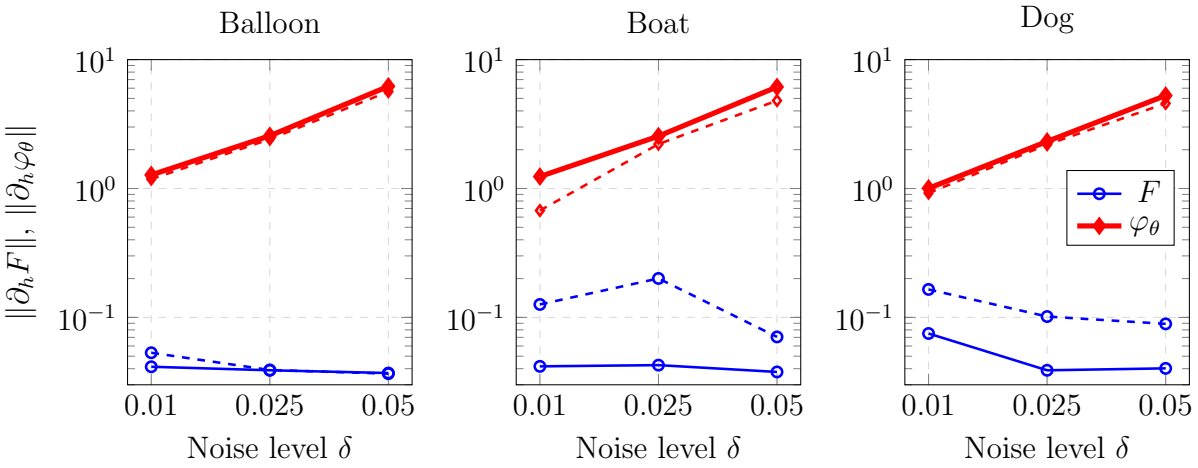


Figure 12: Computation of directions h for which the difference $\|\partial_h \varphi_\theta(x_0)\| - \|\partial_h F(x_0)\|$ is particularly large (φ_θ trained for the **nonlinear diffusion operator** with different noise levels δ). For the three test images "Balloon", "Boat" and "Dog", we display $\|\partial_h \varphi_\theta(x_0)\|$ in red and $\|\partial_h F(x_0)\|$ in blue. The dashed lines show the results, where h is restricted to the subject of the image (cf. Figure 11).

$x_0 \in X$.

In contrast, we expect the trained iResNet φ_θ to have learned a regularization for the inverse problem. Thus, we anticipate $\|\partial_h \varphi_\theta(x_0)\| \gg \|\partial_h F(x_0)\|$ for the same directions h. The Lipschitz constraint even ensures that $\|\partial_h \varphi_\theta(x_0)\| \geq 1 - L$ is guaranteed. Moreover, we can use gradient ascent to find directions $h \in X$, $\|h\| = 1$ such that

$$\|\partial_h \varphi_\theta(x_0)\|^2 - \|\partial_h F(x_0)\|^2$$

is as large as possible. In other words, we look for a direction h in which the network has learned the strongest regularization w.r.t. a given input image x_0. We can also restrict h to certain areas of the image x_0 in order to analyze the regularization inside this area.

Figure 11 shows the resulting directions h of highest regularization for three different test images x_0. Note that gradient ascent does not lead to unique solutions and the directions should therefore rather be seen as examples.

The first striking observation is that h consistently exhibits a checkerboard pattern. This occurs because adding a small perturbation εh to x_0 introduces numerous small edges, which are subsequently blurred by the forward operator F. Since these small edges may represent important image details, regularization by the iResNet is necessary. Additionally, h is primarily concentrated in the smooth parts of the images, such as the uniform areas of the background. In these regions, the introduction of small edges has the most pronounced effect. Moreover, reconstructing a smooth background without any details is a relatively simple task for a neural network, requiring only few information from the noisy and blurred image z^δ and relying more heavily on the learned prior. Therefore, it is optimal to apply strong regularization in these areas. Even when h is restricted to the subject of the images, it remains concentrated in the smoother regions.

An iResNet trained on higher noise levels is expected to apply a stronger regularization. To verify this, we compare the values of $\|\partial_h \varphi_\theta(x_0)\|$ for φ_θ trained on different noise levels δ. The result is illustrated in Figure 12. For all three test images, we indeed observe that the amount of regularization increases with δ.

4.3.2 Investigation of the learned forward operator by linearization

In the previous section, we observed that the iResNet has effectively learned to regularize, particularly in smoother regions of the image. In this section, we aim to gain a deeper understanding of this learned regularization by comparing the local behavior of our iResNet φ_θ with that of the operator F. However, due to the inherent nonlinearity of both the network and the operator, this analysis presents a significant challenge.

To address this challenge, we approximate φ_θ and F around a given image $x_0 \in \mathbb{R}^{n\times m}$ with the help of a first order Taylor expansion. The first order Taylor expansion $T\Psi(\cdot, x_0) : \mathbb{R}^{n\times m} \to \mathbb{R}^{n\times m}$ of a differentiable operator $\Psi : X \to X$ around x_0 restricted to some pixel $(n_0, m_0) \in \{1, \dots, n\} \times \{1, \dots, m\}$ is given by

$$T\Psi(x, x_0)\big|_{(n_0,m_0)} = \Psi(x_0)\big|_{(n_0,m_0)} + \Big\langle J\Psi(x_0)\big|_{(n_0,m_0)}, x - x_0 \Big\rangle + R\Psi(x, x_0)\big|_{(n_0,m_0)}$$

with Jacobian

$$J\Psi(x_0)\big|_{(n_0,m_0)} = \left(\frac{\partial \Psi_{n_0,m_0}}{\partial x_{k,l}}(x_0)\right)_{k=1,\dots,n,\, l=1,\dots,m},$$

$x \in \mathbb{R}^{n\times m}$ and remainder $R\Psi(\cdot, x_0) : \mathbb{R}^{n\times m} \to \mathbb{R}^{n\times m}$. In the setting of Ψ being our iResNet or the nonlinear diffusion operator, the Jacobian can be interpreted as a linear blurring kernel that approximates the nonlinear diffusion operations of the network and the operator, respectively. This perspective allows us to assign a kernel to each pixel in the image x_0, providing a visual interpretation of φ_θ and F. This approach was first introduced in [25] in the context of image classification, leading to the creation of so-called saliency maps, which enable the interpretation of decisions made by image classification networks. While saliency maps in classification tasks are limited to the number of classes, the large number of image pixels in our setting makes manual analysis impractical. To address this, we opted to cluster the kernels of φ_θ and F and then analyze the resulting clusters, respectively.

Before discussing the clustering results, we will first provide an overview of our calculation pipeline. The computation of the Jacobians and the clustering is performed separately for each image in the test set. This approach is necessary because we observed that clustering becomes ineffective or fails to converge when the number of pixels exceeds 15 000. We found that limiting the number of pixels per image to 1500 leads to a good trade-off between cluster accuracy and computational efficiency.

We randomly select a set $\mathcal{P}$ of pixel indices, which remains the same across all images, networks and the operator. For each pixel (n_0, m_0) in the set $\mathcal{P}$ and fixed image x_0, we calculate the Jacobians $J\varphi_\theta(x_0)\big|_{(n_0,m_0)}$ and $JF(x_0)\big|_{(n_0,m_0)}$. For the network φ_θ, this can be achieved efficiently via backpropagation giving access to the gradients of the input image. Each Jacobian is then cropped to a 9×9 region centered around the pixel (n_0, m_0). This cropping is performed because pixel values outside this region are typically near zero and do not contain relevant information. In what follows, the cropped image is referred to as the saliency map or linear blurring kernel. Before clustering, we normalize the values of these saliency maps while preserving the sign of the values.

Finally, the transformed saliency maps are clustered using spectral clustering. In addition to spectral clustering, we also evaluated other clustering techniques, ranging from primitive

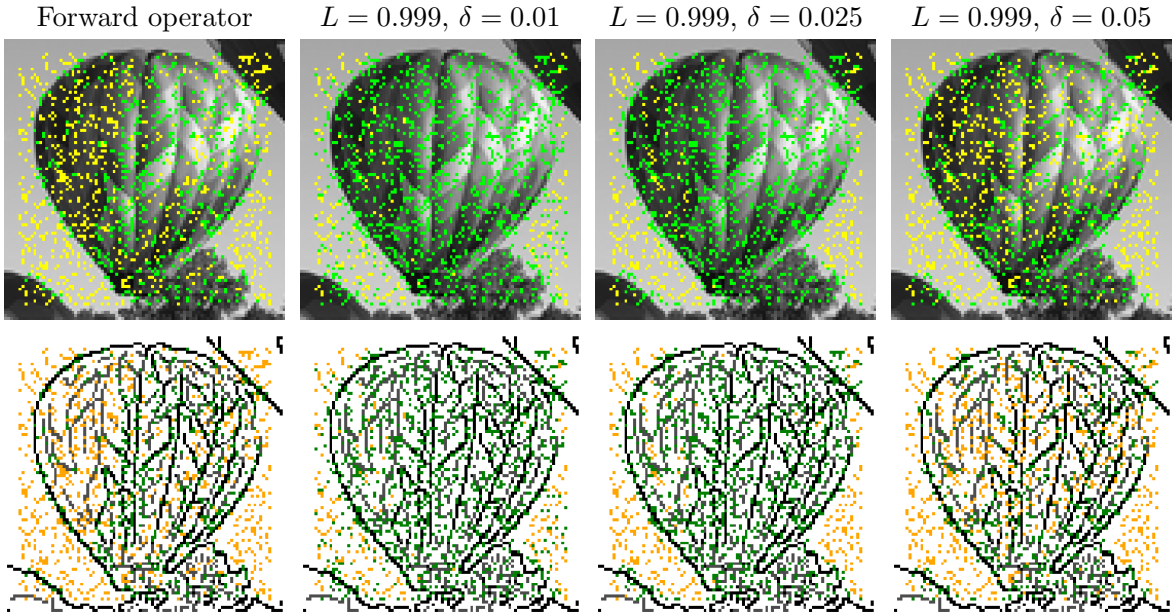


Figure 13: Clustering of the saliency maps of the operator and the network trained with Lipschitz parameter $L = 0.999$ for different noise levels. The first row visualizes the clustering with the ground truth image. The second row visualizes the clustering with the edges of the ground truth image. Weak edges are coloured in grey and strong edges are coloured in black.

methods to more advanced approaches involving principal component analysis (PCA). However, the clustering results were all very similar, which is why we opted for the standard and straightforward spectral clustering algorithm. This method is particularly effective when the clusters are expected to be highly non-convex. We refer the reader to [28] for a more detailed explanation of spectral clustering.

When applying a clustering method, selecting the number of clusters is a critical hyperparameter that must be determined in advance. Several techniques exist to help identify an appropriate number of clusters. To obtain an initial estimate, we compare the predictions from the elbow method [26], the gap statistic [20], and the gap* statistic [13]. While the elbow method is heuristic in nature, both the gap statistic and the gap* statistic provide a more formal approach grounded in statistical analysis. In the examples presented in this section, we found that using two clusters offers a reasonable trade-off between achieving meaningful data separation and maintaining comparability between the network and the operator.

The clustering results for the operator as well as the network trained with different noise levels and fixed Lipschitz parameter $L = 0.999$ are visualized in Figure 13, exemplified on the balloon image from Figure 10. Each pixel for which saliency maps are calculated and clustered is colored based on its cluster assignment. We visualize the results alongside

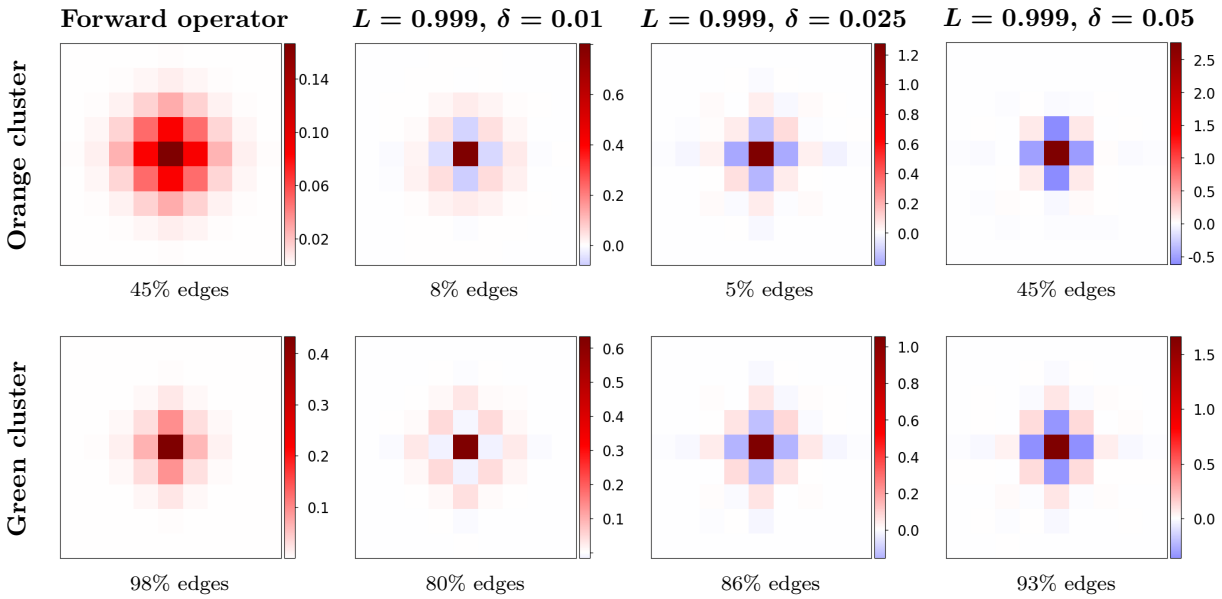


Figure 14: Mean of the saliency maps of the operator and the network trained with Lipschitz parameter $L = 0.999$ and noise level 0.01, 0.025, 0.05. The percentage of pixels on edges for each cluster is indicated below each saliency map.

the ground truth image as well as its edges. In all subsequent plots, the cluster with the highest number of pixels that align with the image edges is highlighted in green. The edges are calculated using the Canny edge detection algorithm. We differentiate between weak and strong edges, where strong edges are defined as those whose value exceeds 20% of the maximum of the intermediate edge image generated during the Canny algorithm after applying Sobel filtering and gradient magnitude thresholding.

Figure 13 shows that the operator's saliency maps are primarily clustered into pixels corresponding to strong edges and those associated with weak edges and smooth areas. In contrast, the clustering for the network trained with $\delta = 0.01$ indicates that the saliency maps are very similar for most pixels, as the majority of pixels fall into the cluster visualized in green. This behaviour is less pronounced for the network trained with $\delta = 0.025$, where the saliency maps are more distinctly clustered into pixels corresponding to edges and those belonging to smoother regions of the image. The clustering of the network trained with $\delta = 0.05$ most closely resembles that of the operator, with weak edges and smooth areas grouped into the same cluster.

To further investigate the clustering, we visualize the mean saliency maps for the individual clusters of the operator and the network trained with $L = 0.999$ for different noise levels in Figure 14. The operator's mean saliency map of the orange cluster (comprising pixels on weak edges and in smooth areas) is much more spread out compared to that of

the green cluster (strong edges). This was to be expected as the strength of the nonlinear diffusion depends on the gradient of the image, with stronger diffusion in areas with smaller gradients.

When comparing the network's saliency maps to those of the operator, it is particularly striking that the spread of the orange cluster is significantly lower and decreases further as the noise level increases. Since the orange cluster primarily includes pixels in smooth regions and on weak edges, this leads to significantly less blurring in these areas and has a regularizing effect, which is in line with our investigations of the local ill-posedness in the previous section. Additionally, for small noise levels ($\delta = 0.01, 0.025$), the dispersion of the network's saliency maps is quite similar across both clusters, which may explain the poor clustering observed in Figure 13.

Furthermore, the network's mean saliency maps exhibit a strong emphasis on the central pixel, displaying significantly higher values compared to those of the operator. This is particularly the case for the cluster containing pixels in smooth areas as well as high noise levels. To counterbalance this and maintain the range of values in the blurred image, the values of the pixels adjacent to the central pixel are negative.

Overall, the behaviour of the saliency maps leads to an overemphasis of edges and significantly reduced blurring in smooth regions, especially for high noise levels. This aligns with the observations presented in Figure 10.

We also examined the clustering results of the network with varying Lipschitz parameters. Our analysis confirms that smaller Lipschitz parameters significantly reduce the network's expressiveness, as previously discussed. As a result, the network's ability to learn an effective regularization from the data is limited for smaller Lipschitz parameters. For the sake of completeness, the results can be found in Appendix B.

So far, the investigations have been based entirely on a single image from the test set, which may lead to biased conclusions. This limitation arises because comparing clustering results across different images, let alone the entire test set, is challenging. Pixel cluster affiliations can vary significantly between different networks and images, making automated comparison difficult. To obtain more reliable results, we decided to manually cluster the pixels of each image in the test set to create comparable clusters. Based on the operator's behavior and the clustering results of the balloon image, we chose to divide the pixels of each image into two groups: those belonging to smooth areas and those belonging to the edges.

We implement this manual clustering using Canny edge detection to extract the edges of each image in the test set, thereby creating two distinct sets of pixels corresponding to smooth regions and edges. To ensure a clear spatial separation between the two clusters, we dilate the edge image using a 3×3 kernel, assigning pixels to the cluster representing smooth

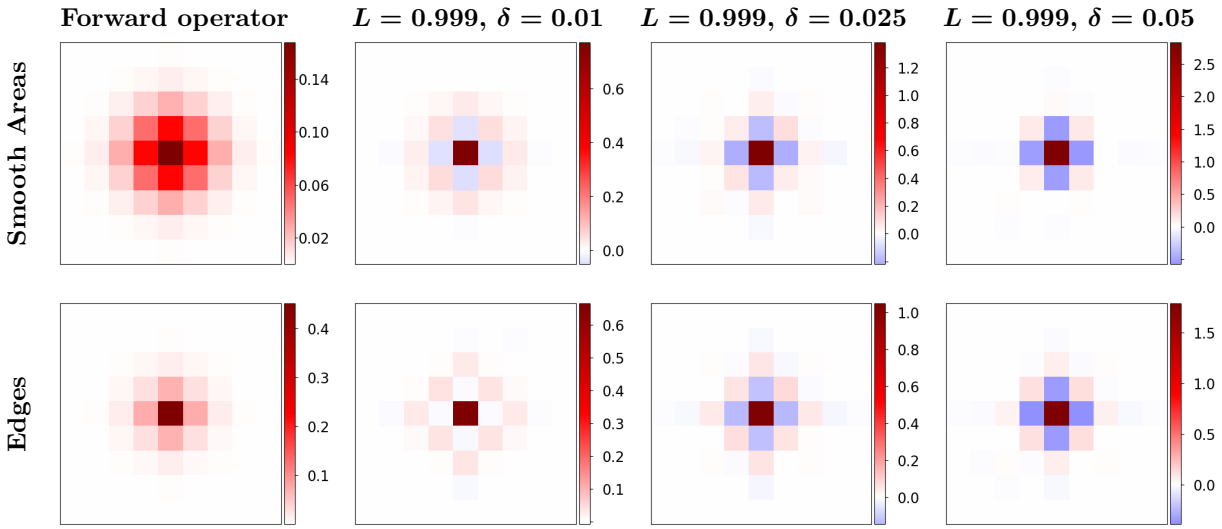


Figure 15: Mean of the saliency maps of the operator and the network trained with Lipschitz parameter $L = 0.999$ and noise level 0.01, 0.025, 0.05. The clustering is done manually on the test set according to edge pixels and smooth areas.

areas only if they are at least two pixels away from an edge. Moreover, the boundary region of the image is excluded from clustering to ensure that the extracted and cropped saliency maps consistently have the same size. For each image in the test set, we then sample 250 pixels equidistantly from each cluster, calculate the corresponding saliency map, and average them within each cluster.

The resulting averaged saliency maps are depicted in Figure 15 for the operator as well as the network trained with noise level $\delta = 0.01, 0.025, 0.05$ and Lipschitz parameter $L = 0.999$. The mean saliency maps exhibit a behavior remarkably similar to that observed in the clustering of the balloon image. This similarity confirms the reliability and significance of the balloon image's clustering results, indicating that this behavior generalizes well to other images.

5 Discussion and Outlook

This present work builds on the theoretical foundation of the iResNet reconstruction approach for solving inverse problems, previously introduced in [2, 3], by conducting extensive numerical experiments on two real-world tasks. Our experiments demonstrate the competitiveness of iResNets compared to state-of-the-art reconstruction methods. While the earlier publications [2, 3] established criteria for rendering the iResNet reconstruction approach a convergent regularization scheme, their efficacy in real-world, high-dimensional tasks had

not yet been confirmed. To address this, we developed a large-scale iResNet architecture and evaluated its performance on both a linear blurring problem and a nonlinear diffusion problem.

Our results indicate that iResNets perform on par with leading image reconstruction networks, such as U-Nets. However, unlike most state-of-the-art methods, iResNets provide verifiable regularization properties. Specifically, they ensure stability of the resulting reconstruction scheme by controlling the hyperparameter L and offer empirically verifiable convergence to the true solution as the noise level decreases. Additionally, the inherent invertibility of iResNets enables exploration of the learned forward operator. This opens the door to in-depth investigations of the nature of the learned regularization. However, these advantages come at the price of significantly higher training times compared to state-of-the-art methods.

In our numerical experiments, the best trade-off between interpretability of the learned solution and performance was achieved by DiffNets, as introduced in [4], though these are specifically tailored for diffusion problems. In contrast, our iResNet reconstruction approach is applicable to a broader class of inverse problems and offers theoretical guarantees in addition to interpretability. Consequently, to validate the versatility of iResNets, future research should explore their performance across a wider range of forward operators. Furthermore, it is essential to investigate possibilities for reducing the training times of iResNets while preserving their performance in order to increase their efficiency in practical applications.

In summary, this work provides numerical evidence supporting the iResNet regularization scheme as an effective and interpretable learned reconstruction method with theoretical guarantees. Consequently, iResNets offer a promising approach to solving complex real-world inverse problems by bridging the gap between high-performance, data-driven methods that often lack theoretical justification and classical methods with guarantees but inferior performance.

Acknowledgments

J. Nickel acknowledges support by the Deutsche Forschungsgemeinschaft (DFG) - project number 281474342/GRK2224/2 - as well as the Bundesministerium für Bildung und Forschung (BMBF) - funding code 03TR07W11A. C. Arndt acknowledges funding by the Bundesministerium für Bildung und Forschung (BMBF) within the collaborative research project KIWi - Project number 05M22LBA. The responsibility for the content of this publication lies with the authors.

A Proof of Lemma 3.1

Proof. Since $g \equiv 1$, $S_h u$ fulfills the linear PDE $u = S_h u - h \cdot \operatorname{div}(\nabla(S_h u))$. It holds

$$\begin{aligned}\|(Id - S_h)u\|^2 &= \|u\|^2 - 2\langle S_h u, u\rangle + \|S_h u\|^2 \\ &= \|u\|^2 - 2\langle S_h u, S_h u - h \cdot \operatorname{div}(\nabla(S_h u))\rangle + \|S_h u\|^2 \\ &= \|u\|^2 + 2h\langle S_h u, \operatorname{div}(\nabla(S_h u))\rangle - \|S_h u\|^2 \ .\end{aligned}$$

Due to the zero Dirichlet or zero Neumann boundary condition on $S_h u$, we can go on with

$$\begin{aligned}\|(Id - S_h)u\|^2 &= \|u\|^2 - 2h\langle \nabla(S_h u), \nabla(S_h u)\rangle - \|S_h u\|^2 \\ &\leq \|u\|^2 - \|S_h u\|^2,\end{aligned}$$

which completes the proof. □

B Cluster comparison for different Lipschitz parameters

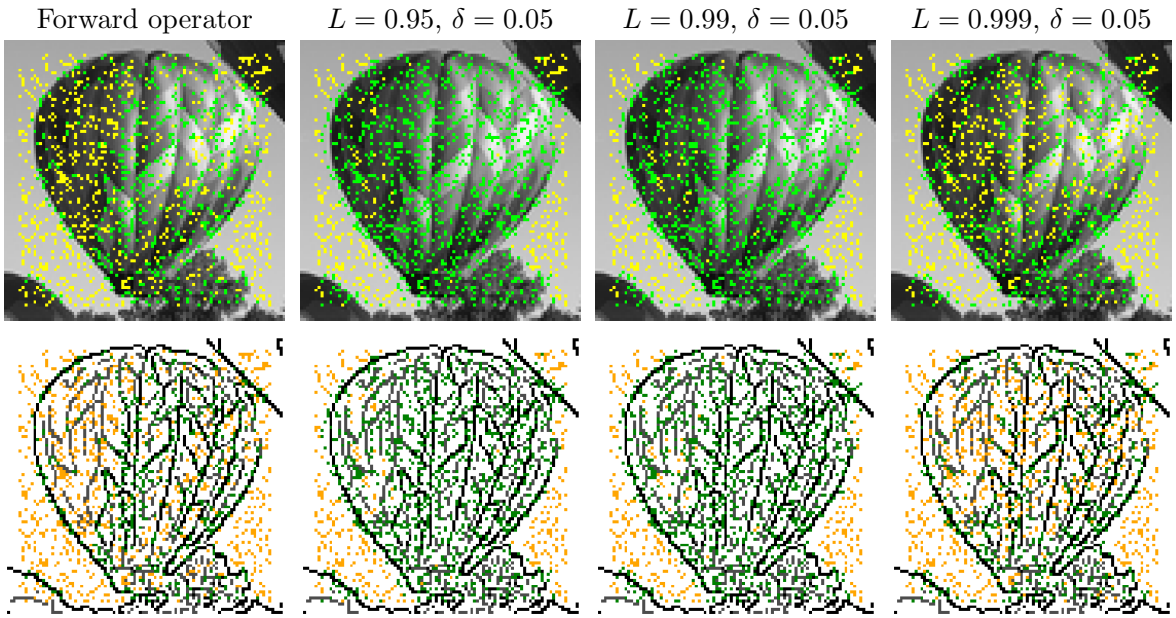


Figure 16: Clustering of the saliency maps of the operator and the network trained with noise level $\delta = 0.05$ for different Lipschitz parameters. The first row visualizes the clustering with the ground truth image and the second row with the corresponding edges (weak edges are grey and strong edges are black).

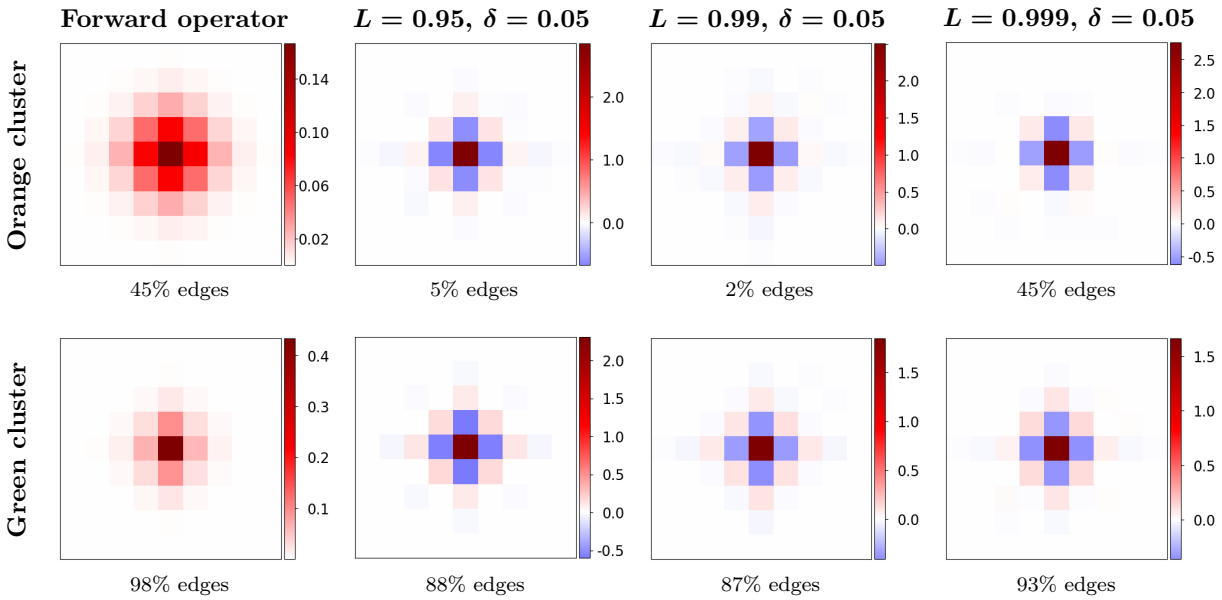


Figure 17: Mean of the saliency maps of the operator and the network trained with Lipschitz parameters $L = 0.95, 0.99, 0.999$ and noise level 0.05. The percentage of pixels on edges for each cluster is indicated below each saliency map.

References

[1] A. Y. N. Adam Coates, Honglak Lee. An analysis of single layer networks in unsupervised feature learning. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.

[2] C. Arndt, A. Denker, S. Dittmer, N. Heilenkötter, M. Iske, T. Kluth, P. Maass, and J. Nickel. Invertible residual networks in the context of regularization theory for linear inverse problems. *Inverse Problems*, 39(12):125018, nov 2023.

[3] C. Arndt, S. Dittmer, N. Heilenkötter, M. Iske, T. Kluth, and J. Nickel. Bayesian view on the training of invertible residual networks for solving linear inverse problems. *Inverse Problems*, 40(4):045021, mar 2024.

[4] S. Arridge and A. Hauptmann. Networks for nonlinear diffusion problems in imaging. *Journal of Mathematical Imaging and Vision*, 62:471–487, 2020.

[5] S. Bai, J. Z. Kolter, and V. Koltun. Deep equilibrium models. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[6] J. Behrmann, W. Grathwohl, R. T. Chen, D. Duvenaud, and J.-H. Jacobsen. Invertible residual networks. In *International Conference on Machine Learning*, pages 573–582. PMLR, 2019.

[7] M. Benning and M. Burger. Modern regularization methods for inverse problems. *Acta Numerica*, 27:1–111, 2018.

[8] L. Bungert, R. Raab, T. Roith, L. Schwinn, and D. Tenbrinck. Clip: Cheap lipschitz training of neural networks. In A. Elmoataz, J. Fadili, Y. Quéau, J. Rabin, and L. Simon, editors, *Scale Space and Variational Methods in Computer Vision*, pages 307–319, Cham, 2021. Springer International Publishing.

[9] H. W. Engl, M. Hanke, and A. Neubauer. *Regularization of Inverse Problems*. Springer Dordrecht, 1st edition, 2000.

[10] D. Gilton, G. Ongie, and R. M. Willett. Deep equilibrium architectures for inverse problems in imaging. *IEEE Transactions on Computational Imaging*, 7:1123–1133, 2021.

[11] J. Hertrich. Proximal residual flows for bayesian inverse problems. In L. Calatroni, M. Donatelli, S. Morigi, M. Prato, and M. Santacesaria, editors, *Scale Space and Variational Methods in Computer Vision*, pages 210–222, Cham, 2023. Springer International Publishing.

[12] B. Hofmann, B. Kaltenbacher, C. Pöschl, and O. Scherzer. A convergence rates result for Tikhonov regularization in Banach spaces with non-smooth operators. *Inverse Problems*, 23(3):987–1010, 2007.

[13] M. Mohajer, K.-H. Englmeier, and V. J. Schmid. A comparison of gap statistic definitions with and without logarithm function. *Department of Statistics: Technical Reports*, 96, 2010.

[14] V. Monga, Y. Li, and Y. C. Eldar. Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing. *IEEE Signal Processing Magazine*, 38(2):18–44, 2021.

[15] S. Mukherjee, S. Dittmer, Z. Shumaylov, S. Lunz, O. Öktem, and C.-B. Schönlieb. Learned convex regularizers for inverse problems. Preprint, arXiv:2008.02839, 2021.

[16] S. Mukherjee, A. Hauptmann, O. Öktem, M. Pereyra, and C.-B. Schönlieb. Learned reconstruction methods with convergence guarantees: A survey of concepts and applications. *IEEE Signal Processing Magazine*, 40(1):164–182, 2023.

[17] S. Mukherjee, C.-B. Schönlieb, and M. Burger. Learning convex regularizers satisfying the variational source condition for inverse problems. In *NeurIPS 2021 Workshop on Deep Learning and Inverse Problems*, 2021.

[18] D. Obmann and M. Haltmeier. Convergence analysis of equilibrium methods for inverse problems. *arXiv preprint arXiv:2306.01421*, 2023.

[19] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):629–639, 1990.

[20] T. H. Robert Tibshirani, Guenther Walther. Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 63:411–423, 2001.

[21] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional networks for biomedical image segmentation. In N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention - MICCAI 2015*, volume 9351, pages 234–241, 2015.

[22] L. Ruthotto and E. Haber. Deep neural networks motivated by partial differential equations. *Journal of Mathematical Imaging and Vision*, 62:352–364, 2020.

[23] T. Schuster, B. Kaltenbacher, B. Hofmann, and K. S. Kazimierski. *Regularization Methods in Banach Spaces*. De Gruyter, Berlin, Boston, 2012.

[24] F. Sherry, E. Celledoni, M. J. Ehrhardt, D. Murari, B. Owren, and C.-B. Schönlieb. Designing stable neural networks using convex analysis and odes. *Physica D: Nonlinear Phenomena*, 463:134159, 2024.

[25] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps, 2014.

[26] R. Thorndike. Who belongs in the family? *Psychometrika*, 18(4):267–276, 1953.

[27] S. V. Venkatakrishnan, C. A. Bouman, and B. Wohlberg. Plug-and-play priors for model based reconstruction. In *2013 IEEE Global Conference on Signal and Information Processing*, pages 945–948, 2013.

[28] U. von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17:395–416, 2007.

Manuscript submitted to AIMS Journal

doi:10.3934/xx.xxxxxxx

IN FOCUS - HYBRID DEEP LEARNING APPROACHES TO THE HDC2021 CHALLENGE

Clemens Arndt*, Alexander Denker, Judith Nickel, Johannes Leuschner, Maximilian Schmidt and

Bibliothekstr. 5, 28359 Bremen, Germany
Center for Industrial Mathematics
University of Bremen

Gael Rigaud

Pfaffenwaldring 57, 70569 Stuttgart, Germany
Department of Mathematics
University of Stuttgart

Abstract. In this work, we present our contribution to the Helsinki Deblur Challenge 2021. The goal of the challenge was to recover images of sequences of letters from progressively out-of-focus photographs. While the blur model was unknown, a dataset of sharp and blurry images was provided. We propose to tackle this problem in a two-step process: (i) the blur models are first extracted and estimated from the provided dataset, and (ii) then incorporated into the reconstruction process. Here, we present three different ways of integrating the estimated model into learning-based methods: (i) an educated deep image prior employing the estimated model in the loss function, (ii) a learned iterative approach that directly employs the estimated model in the architecture and (iii) a fully learned approach where we used the estimated model to simulate additional training data. These three models are improved versions of our original contributions to the challenge. We compare and benchmark them on the released test set of the HDC2021.

1. **Introduction.** Image deblurring consists of the restoration of a degraded image and can generally be formulated as

$$g^\delta = \mathcal{A}f + \eta, \tag{1}$$

where f is the original image, g^δ is the blurred image, and $\mathcal{A}$ stands for a forward operator, which models the blurring process. Additionally, η describes some noise with noise level δ. In the Helsinki Deblur Challenge 2021 (HDC2021) [20], a dataset was made available for participants to evaluate their deblurring methods. The dataset consists of pairs of original and blurred images with 20 levels of progressively increasing out-of-focus blur. However, the specific blurring kernel was not provided. The problem is then known as blind image deblurring [16]. A common way of tackling blind image deblurring is to first estimate the forward model and, in a second step, incorporate this estimated model into a non-blind image deblurring

2020 *Mathematics Subject Classification.* Primary: 68T07, 45Q05.

Key words and phrases. Deblurring, Iterative Reconstruction, Inverse Problems, Deep Learning, Learned Reconstruction.

*Corresponding author: Clemens Arndt.

method [26]. Usually, the forward model is assumed to be a linear convolution operator given by a blurring kernel. The problem of non-blind image deblurring is widely studied, and there exist many algorithms for recovering the original image [8, 24]. Recently, deep learning-based methods have extensively been used for image reconstruction (see [2] for an overview) and, in particular, for non-blind image deblurring [19, 27, 30].

In this paper, we will present and compare three learning-based techniques with different philosophies on how to address the blind deblurring problem. The three methods incorporate a blur model initially estimated from the provided data pairs, see Section 3. In Section 4.2, we cover our first method, an educated deep image prior [4, 19] where the estimated blurring operator is used in the objective function. Section 4.3 introduces the learned gradient descent approach [1], which directly integrates the estimated blurring operator in the network architecture. The *StepNet*, presented in Section 4.4, is a fully learned approach and uses the estimated blurring operator only to simulate additional training data. We evaluate our approaches on the test set of the HDC2021 in Section 5 and draw conclusions in Section 6.

2. **Challenge overview.** The HDC2021 dataset consists, in total, of 4000 pairs of sharp and blurry images split into 20 levels of blur. The experimental setup and data collection are described in [20]. Each pair is a photograph of the same image taken with two identical cameras with different settings. The experimental setup is illustrated in Figure 2(a). The target image consists of three lines of black text on a white background. Two different font types are used: Times and Verdana, cf. Figure 1. Overall, for each font type and each blurring level, there are 100 pairs of images. The various blur levels become more and more severe as one camera is shifted increasingly out-of-focus. In addition to this blurring, the out-of-focus camera uses a higher ISO level (exposure index) [33] and an additional neutral density filter. The filter reduces the amount of light reaching the photo sensor. Normally, this means that a lower ISO value can be used when photographing. In contrast, the higher ISO level used results in considerably more noise on the blurred images. The images are further converted into grayscale images, aligned, and cropped. A few example image pairs are shown in Figure 2(b). For the lower blurring level 4, a human can still read the blurry text. For the last level, 19, this is no longer possible.

The goal of the challenge is to build a reconstruction method to recover the original image from the noisy and out-of-focus version. The quality of the reconstruction is judged by the performance of the Tesseract OCR[1] software [32]. In addition, the methods should pass a sanity check consisting of blurred images without text, which need to be deblurred consistently. To be more precise, the organizers of the challenge produced photographs of natural images under the same conditions. When the reconstruction methods are applied to these natural images, a slight deblurring effect should be visible, and the reconstruction method should not overfit to text images.

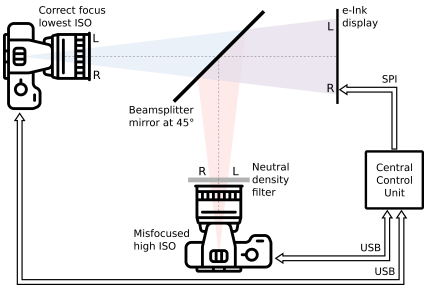
3. **Learning the forward operator.** In many inverse problems, it has been beneficial to introduce model-based knowledge into a learning-based reconstruction method. In recent data challenges, like fastMRI [21, 22], LoDoPab-CT [23], or the AAPM sparse view challenge [18], the best-performing methods always combined

[1]We use the Python wrapper https://pypi.org/project/pytesseract/0.3.7/

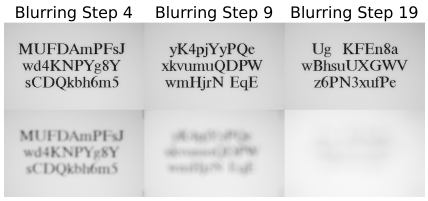
Times ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
0123456789

Verdana ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
0123456789

FIGURE 1. Letters and digits of the fonts Times and Verdana. Both are shown in the same font size.



(a) Experimental setup of the HDC 2021.



(b) Data pairs for steps 4, 9, and 19.

FIGURE 2. The figure (a) is taken from [20], published under CC BY 4.0 license (https://creativecommons.org/licenses/by/4.0/deed.en).

model-based components with deep learning architectures. As the forward operator is unknown, we use the training data to learn an approximation. In our initial submission to the HDC2021, we considered a disk-like point spread function (PSF) and estimated the radius. In addition to the disk-like PSF, we use the implementation of the team *robust-and-stable*[2] from the TU Berlin, which is based on a fully-learned kernel composed with a learned lens distortion model. We start by introducing the disk-like blur model and then present the fully learned blur model with additional distortion.

3.1. **Disk blur.** Out-of-focus blur can be modeled as the convolution of the image with a disk-like PSF

$$k_a(r) = \begin{cases} 1/(\pi a^2) & \text{if } r \leq a, \\ 0 & \text{else} \end{cases} \tag{2}$$

where $r = \sqrt{x^2 + y^2}$ is the distance from the center of the image [17]. This results in a linear forward operator $\bar{A}f = k_a(r) * f$ with $*$ denoting the convolution operator. As this model is not differentiable w.r.t. the radius a, we used a line search over a to fit this model to the training data for each blurring step.

[2] https://github.com/theophil-trippe/HDC_TUBerlin_version_1

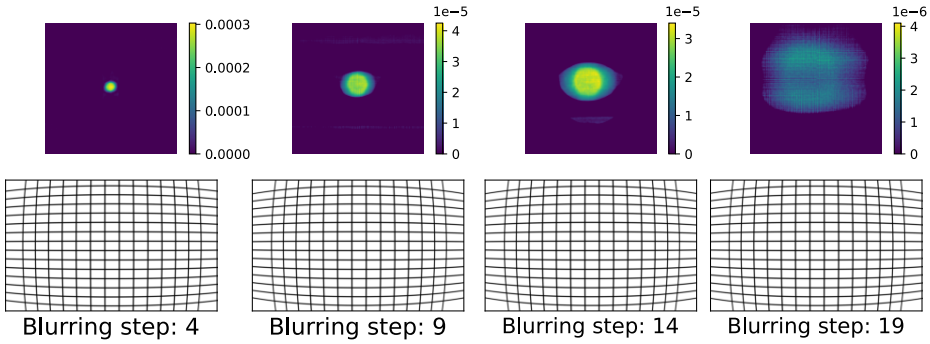


FIGURE 3. First row: Learned blurring kernels k for blurring steps $4, 9, 14$, and 19. Second row: Visualization of the learned lens distortion model D for blurring steps $4, 9, 14$, and 19.

This model is used in the gradient step of the learned gradient descent model in Section 4.3.

3.2. **Combined blur and distortion model.** A more flexible approximation can be built by combining a linear blur operator with a lens distortion model. Instead of restricting the kernel to be a disk, we parameterized the kernel $k \in \mathbb{R}_{\geq 0}^{n_k \times n_k}$ as a $n_k \times n_k$ positive matrix. A general distortion model is given by [6]. It has been reported that lens distortion can be modeled using a radial distortion model of the form [7, 29]:

$$D(f(X_u)) = f(X_d) := f((1 + K_1 r^2 + K_2 r^4) X_u)$$

with $X_u = (x_u, y_u)$ being the undistorted image coordinates, $X_d = (x_d, y_d)$ the distorted coordinates, r the radius, and K_1, K_2 the radial distortion coefficients. The pixel values for the new image coordinates are calculated using bilinear interpolation. This combined blurring and distortion model can be fitted using gradient-based methods for solving the minimization problem

$$\min_{k, K_1, K_2} \frac{1}{N} \sum_{i=1}^{N} \| D(f_i * k) - g_i^\delta \|_2^2,$$

where D describes the distortion process. In the rest of the manuscript, we will denote this model by $\hat{\mathcal{A}} f = D(f * k)$. The learned kernel and a visualization of the learned distortion are shown in Figure 3. As one can see, this fully learned model also gives rise to disk-like blurring kernels. Furthermore, the distortion gets more severe as the blurring level increases.

The combined blur and distortion model is used to simulate additional training data and is incorporated in the deep image prior in Section 4.2.

4. **Reconstruction methods.** In this section, we present the different reconstruction methods used in the challenge. We improved our methods with new ideas from other submissions to the HDC2021. The changes to our initial submission will be explained in the respective sections. In particular, we cover three methods:

- Educated deep image prior using a U-Net architecture[3],

[3] https://github.com/mschmidt25/hdc2021_DIP

- Learned gradient descent (LGD)[4],
- A fully learned approach, which we call *StepNet*[5].

In addition, we trained a simple U-Net [14], which serves as a baseline for our methods.

To train the models, we used 90 images of each font (180 training images in total) per blurring step. The remaining 10 images per font and blurring step were used for validation and fine-tuning of the models. To overcome the small amount of training data, we used 500 additional synthetic text images generated on-the-fly with the fully learned blur and distortion model of Section 3.2. To train first on the synthetic images and afterward on the real images turned out to be the best strategy. For both training stages, we chose the best model parameters according to the validation loss.

4.1. **Baseline U-Net.** As a baseline, we trained a neural network to map directly from the blurry images to the clean ground truth images. For each blurring step, a separate model with the same architecture is trained. This approach is purely data-driven and does not use any knowledge about the forward operator (besides the synthetic training images).

The model architecture consists of a U-Net [14] of depth 6 with skip connections on all scales. The size of the convolution kernels ranges from 7×7 on the upper scales to 3×3 on the lower scales. Batch normalization is performed after each convolution. We adopt nearest neighbor interpolation as an upsampling strategy because it results in non-smooth images, which is favorable for text images. As activation functions, we use LeakyReLU [28] with a negative slope of 0.2 at all intermediate layers and sigmoid in the last layer to enforce the output to be in the interval $[0, 1]$. We train the network with a learning rate of 10^{-4} for 150 epochs on the synthetic images and another 150 epochs on the real images. As optimizer we use Adam [12] and a batch size of one due to limited memory. We use the mean squared error (MSE) as loss function, on the reconstructions.

The described network architecture is similar to the one we used in our submission of the educated deep image prior. One of the main differences is that we trained the U-Net in our submission on downsampled images. In addition, the U-Net had a depth of 5, a fixed kernel size of 3×3, and we used bilinear upsampling instead of nearest neighbor upsampling. We found that our changes further improved the reconstruction quality of the U-Net.

4.2. **Educated Deep Image Prior.** For the deep image prior (DIP) approach [19], a neural network φ_θ is used for parametrizing the solution of an inverse problem. In contrast to other deep learning approaches, the network is only trained on a single data point g^δ without using any ground truth data. Therefore, we use the loss function

$$\frac{1}{2\Sigma}\|\hat{\mathcal{A}}\varphi_\theta(g^\delta) - g^\delta\|_1 + \frac{1}{2\Sigma}\|\hat{\mathcal{A}}\varphi_\theta(g^\delta) - g^\delta\|_2^2 + \kappa \cdot \mathrm{TV}(\varphi_\theta(g^\delta)) \tag{3}$$

where Σ is the size of the images (height $\times$ width) and $\|\cdot\|_p$ is the ℓ^p-Norm. For regularization we use anisotropic total variation [15, 5] with $\kappa = 10^{-6}$. After optimizing the weights of the network, $\hat{f} = \varphi_\theta(g^\delta)$ is the deblurred image.

[4] https://github.com/alexdenker/hdc2021_LGD
[5] https://github.com/mschmidt25/hdc2021_StepNet

There are several reasons which make DIP suitable for the deblurring challenge. While other deep learning approaches may have problems with the small size of the training dataset, DIP only needs a blurry measurement and a forward model of the blurring process. This forward model should be as accurate as possible. That is why we use the fully-learned blur with distortion $\hat{\mathcal{A}}$ from Section 3.2. An additional advantage of DIP is that it results in a solution $\hat{f}$ with small data discrepancy

$$\|\hat{\mathcal{A}}\hat{f} - g^\delta\|_2^2. \tag{4}$$

Because of this so-called data consistency, DIP is a useful supplement for the baseline U-Net (Section 4.1), which has no such guarantees.

Our strategy is to choose φ_θ as a U-Net with the same architecture as the baseline. Following the idea of [4], we initialize φ_θ with the weights θ_0 of the trained network. The DIP approach in combination with an initialization with trained weights is called educated deep image prior, EDIP for short. Before starting the EDIP training, we check whether the data discrepancy is already smaller than some empirically chosen tolerance value TOL > 0. The specific choice of the tolerance value is explained in the following paragraph. If the data discrepancy is smaller than the tolerance value, we assume that the baseline reconstruction $\varphi_{\theta_0}(g^\delta)$ is already good enough and we do not apply EDIP. If the data discrepancy is too high, we start the EDIP training with a learning rate of 10^{-4} and Adam optimizer [12]. The number of training epochs is fixed depending on the blurring step in order to achieve the best reconstruction quality and avoid overfitting (step 4: 2000, step 9: 4000, step 14 and 19: 5000). A visualization of this strategy can be seen in Figure 4. We would like to remark, that the training strategy is the same as in our submission of the educated deep image prior with adaptations in the number of training epochs and tolerance values TOL of the data discrepancy.

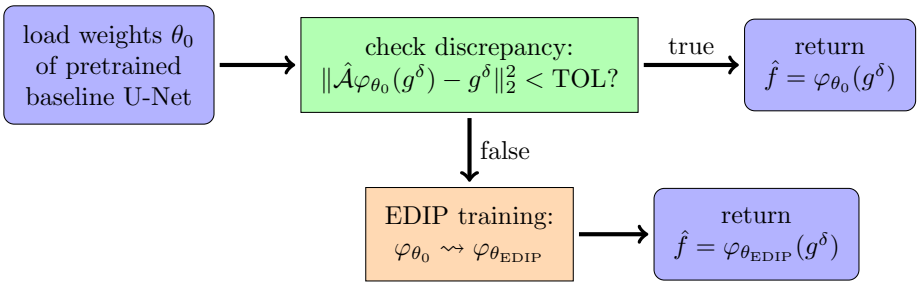


FIGURE 4. Strategy of the educated DIP (EDIP).

We observed that EDIP is most of the time not able to improve the reconstructions of the baseline U-Net on the text images but results in better looking reconstructions on the sanity check images (as can be seen in tables 1 and 2). In particular, EDIP is capable of overcoming the problem that reconstructions on the sanity check images contain text-like structures. Therefore, the EDIP approach can be understood as a means to improve the generalization ability of the U-Net without significantly changing the reconstruction quality on the text images. Consequently, we chose the tolerance TOL of the data discrepancy (above which the EDIP is trained) to be slightly larger than the data discrepancy of the U-Net reconstructions on the training text images. To be more precise, the tolerances for the data discrepancies were set to 0.007 for step 4, 0.015 for step 9 and step 14 and

0.035 for step 19. We would like to stress that this choice leads to a training of EDIP on nearly all sanity check images of step 4, 9 and 14 except step 19. In step 19, we found that the data discrepancy of the U-Net reconstructions of the sanity check images is lower than that of the text images, which is why EDIP is not used in this step.

We found two possible explanations for the observation that EDIP is not able to improve the reconstructions of the U-Net on the text images. First of all, EDIP is not aware that the targets are text images as it is not trained using ground truth images. This did not seem to be a limitation in the classical use case of natural images, see [19]. However, the prior of the EDIP architecture may not be beneficial for text images without relevant texture and with non-smooth edges. Secondly, the forward operator is a learned model that may be inexact. As the forward operator directly influences the loss function of EDIP this may introduce errors in the reconstruction.

4.3. **Learned Gradient Descent.** The idea of learned iterative methods is to unroll commonly used iterative algorithms for a fixed number of steps and replace parts of the resulting scheme by neural networks [1]. We use a learned gradient descent (LGD), which results in a learned reconstruction operator $f^{(L)} = \mathcal{R}_\theta(g^\delta)$ with

$$f^{(l)} = \Lambda_{\theta_l}(f^{(l-1)} - \lambda_l \bar{\mathcal{A}}^*(\bar{\mathcal{A}} f^{(l-1)} - g^\delta)) \tag{5}$$

for $l = 1, \dots, L$ and an initial value $f^{(0)}$ and the neural networks Λ_{θ_l} are parametrized as convolutional neural networks. The resulting reconstruction operator $\mathcal{R}_\theta$ with $\theta = [\theta_1, \dots, \theta_L]$ is then trained end-to-end by minimizing the empirical mean-squared error (MSE) i.e.

$$\hat{\theta} \in \arg\min_\theta \frac{1}{N} \sum_{i=1}^{N} \|\mathcal{R}_\theta(g_i^\delta) - f_i\|_2^2 \,. \tag{6}$$

These unrolled iterative methods offer a simple way of integrating the model-based knowledge into a learned reconstruction approach. Furthermore, learned iterative approaches seem to be more data efficient in contrast to fully learned approaches, i.e. one needs fewer data pairs for training [3]. In general, one would use the exact forward operator $\mathcal{A}$ in the network (5). However, as the exact forward operator is unknown, we use the disk blur operator $\bar{\mathcal{A}} : X \to X$ defined in Section 3.1. It would be possible to use the fully learned forward operator from Section 3.2. However, the LGD also needs access to the adjoint of the forward operator. We noticed some artefacts in the adjoint of the fully learned forward operator and therefore opted to use the model-based disk blur operator instead. Lunz et. al. [13] note that these artefacts could be avoided if one also learned the adjoint operator. Using an approximation of the forward operator in learned iterative schemes was first explored by Hauptmann et al. [9]. In this context the neural networks Λ_{θ_l} can be thought of as implicit corrections of the operator. Training such an iterative method end-to-end as in (6) can be computationally very expensive. Especially for the high-dimensional images in this challenge, the number of unrolling steps was limited by the available GPU memory. To reduce the memory footprint we use a multi-scale approach as proposed in [10]. We define a sequence of discretizations $X_1, \dots, X_5$ with $\dim(X_1) < \cdots < \dim(X_5)$ of the image space with $X = X_5$ as the original image. For each subsequent X_i the pixel resolution of the image is halved. We set the discretizations to have a pixel resolution of 1460×2360 for X_5, 730×1180

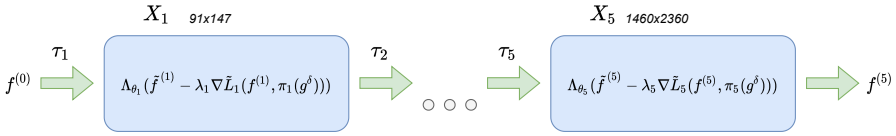


FIGURE 5. An outline of the learned iterative architecture. After each update on scale l the output is upsampled and passed to the next scale.

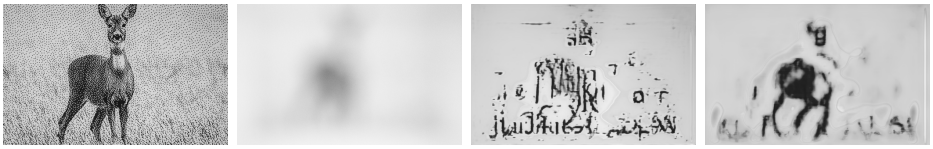
for X_4, 368×590 for X_3, 182×295 for X_2 and 91×147 for X_1. We introduce nearest-neighbour upsampling operators $\tau_l : X_{l-1} \to X_l$ and bilinear downsampling operators $\pi_l : X \to X_l$. For each discretization X_l we use a downsampled forward operator $\bar{\mathcal{A}}_i : X_l \to X_l$. Another trick to improve the performance is to use a filtered gradient for the update in (5) as the adjoint often produces smooth gradients [10]. In our work, we use the Wiener filter $\bar{\mathcal{A}}_l^\dagger$ [24] instead of the adjoint $\bar{\mathcal{A}}_l^*$ at scale l. This results in a filtered gradient step with $\nabla \bar{A}_l(\tilde{f}^{(l)}, \pi_l(g^\delta)) := \bar{\mathcal{A}}_l{}^\dagger(\bar{\mathcal{A}}_l \tilde{f}^{(l)} - \pi_l(g^\delta))$. Integrating both the multi-scale approach and the filtered gradient we get a final reconstruction operator $f^L = \mathcal{R}_\theta(g^\delta)$ with

$$\begin{aligned} \tilde{f}^{(l)} &= \tau_l(f^{(l-1)}) \\ f^{(l)} &= \Lambda_{\theta_l}(\tilde{f}^{(l)} - \lambda_l \bar{\mathcal{A}}_l{}^\dagger(\bar{\mathcal{A}}_l \tilde{f}^{(l)} - \pi_l(g^\delta))) \end{aligned} \tag{7}$$

for $l = 1, \dots, L$ with a zero initialization $f^{(0)} = 0 \in X_0 = X_1$ and $L = 4$. Note that the initial $\tau_1 = \mathrm{Id}$ is defined as the identity. An outline of this network is illustrated in Figure 5.

In contrast to our original submission we made two important changes. First, we adopt the multi-scale architecture as proposed by Hauptmann et al. [10]. This allowed us to increase the size of the subnetworks Λ_l. As a second step we use a filtered gradient update using the Wiener filter instead of the adjoint of the blurring operator. This further improved the performance of this approach.

Implementation details. As per the rules of the challenge, the same architecture, training setup and hyperparameters were chosen for all blurring levels. We choose the architecture such that we got an acceptable performance for the high blurring levels. This results in a model with 14.5M parameters. We observed that much smaller models would suffice for a good OCR accuracy for the smaller blurring levels. An obvious extension would be to tune the architecture for each blurring level. Each subnetwork Λ_{θ_l} uses the U-Net architecture [14] with varying depths. The downsampling is performed using max pooling and the upsampling used nearest neighbour interpolation. The size of the model limits the use of large batch sizes. Therefore instead of batch normalization we use group normalization [25] after each convolutional layer. As activation functions we use LeakyReLU [28] with a negative slope of 0.2. We train the network using the Adam optimizer [12] with a learning rate of 10^{-4} for 150 epochs on the synthetic images and for another 150 epochs on the real images. The step sizes λ_l in the gradient step are fixed to $[1.1, 0.8, 0.8, 0.7, 0.3]$. It would be possible to also learn these step sizes together with the weights of the networks. The Wiener deconvolution used at every scale needs the signal-to-noise ratio as a hyperparameter. This was empirically chosen as [5e-3, 6e-3, 4e-3, 5e-3, 4e-3].



Sharp image | Blurred image | Without pretraining | With pretraining

FIGURE 6. Reconstruction of the LGD for one image from the sanity check. The model without pretraining on Urban100 shows how the model extrapolated text in the reconstruction.

For the LGD we used an additional pretraining strategy using a hand-crafted variation of the Urban100 dataset [11]. To make this dataset more similar to the HDC2021 data, we first extracted 200×300 px patches[6] and used K-means with 2 cluster centers to create a segmentation map for each patch. The values of the binary segmentation maps were transformed to match the black text and the white background of the HDC2021 images respectively. In a next step this image was upsampled using a nearest neighbour method to equal the size of the original training images of HDC2021. The combined blur and distortion model of Section 3.2 was then used to simulate blurry measurements. This pretraining was used to steer the LGD model towards a better generalization performance. Without this pretraining the model repeatedly extrapolated text on the highest blurring levels when applied to blurry non-text images as can be seen in Figure 6.

4.4. **StepNet.** For the StepNet approach, we follow the idea to split the deblurring problem into separate steps. The network consists of a concatenation of 20 U-Nets – one for each blur level. The task of a single U-Net is to reduce the level of its respective input by one. We use the first $l+1$ U-Nets to produce the reconstruction for an input image at blur level $l \in \{0, 1, \dots, 19\}$

$$\tilde{f} = \text{StepNet}_\theta\left(g^\delta, l\right) := \text{U-Net}_{\theta[l]} \circ \text{U-Net}_{\theta[l-1]} \circ \cdots \circ \text{U-Net}_{\theta[0]}\left(g^\delta\right). \tag{8}$$

Here, the U-Nets differ in their parameterization $\theta[\cdot]$, but not in their architecture. The StepNet can thus be understood as an unrolling via the blur stages.

For training, we start with images from the first blur level. This includes pretraining on simulated data. Following the definition in (8), only the first U-Net is involved in this case. The parameters of the network $\theta[0]$ are trained for 150 epochs. Afterward, we freeze the weights and continue training with the second level and U-Net. In general, training at blur level l is defined as a minimization problem

$$\hat{\theta}[l] \in \arg\min_{\theta[l]} \frac{1}{N} \sum_{i=1}^{N} \left\| \text{StepNet}_\theta\left(g_i^\delta, l\right) - f_i \right\|_2^2.$$

The Adam optimizer [12] in combination wit a learning rate of 10^{-4} is used for the minimization.

Freezing all other parameters, apart from $\theta[l]$, provides two advantages. The first one is that we can apply the exact same StepNet to blurry images of all possible levels. In addition, the effort required for training is reduced. Training all parameters of the network in each step would require too much computing resources and memory, especially in high levels. Further reduction of the computational requirements

[6] This patch size was chosen to match the resolution of the e-ink display used in the HDC2021.

is achieved through 4× downsampling of the input images by average pooling. In the end, the output of the StepNet is upscaled to the original image size by nearest neighbor interpolation.

Besides downsampling of the input, we also apply an undistort operation, which reverses the learned distortion model introduced in Section 3.2. The goal is to reduce the influence of elements outside the field of view in the ground truth image, which are present in the blurry image due to distortion.

5. Results.

We focus on the blur levels 4, 9, 14, and 19 for the analysis of the results. These levels were also used by the challenge organizers to show visual results of the competing approaches. The code for the models and all experiments is available[7].

5.1. Evaluation Procedures.

A total of four different metrics are used for the assessment. The optical character recognition (OCR) accuracy provides inside into the text reconstruction capability of each method. In addition, we use an adapted OCR score called OCRI, which we explain in the next paragraph. These scores are calculated on the test set of the text images with the help of the Tesseract software and the Levenshtein distance. Furthermore, we are interested in the overall reconstruction quality on the sanity set. Here, we use the PSNR (9) and SSIM (10) metrics. Both metrics are also calculated for the reconstructed text images as a reference.

OCR and OCRI. We follow the procedure and provided code of the HDC2021 challenge to evaluate the performance of the text reconstruction task [20, Sec. 5.1]. The Tesseract software is used to extract text from the reconstructed image. Next, the Levenshtein distance between the extracted text $\mathrm{T}_{\tilde{f}}$ of the prediction $\tilde{f}$ and the provided text T_{GT} of the ground truth image f is determined. The Levenshtein distance is defined as

$$\mathrm{lev}[\mathrm{T}_{GT}, \mathrm{T}_{\tilde{f}}] := \begin{cases} |\mathrm{T}_{GT}|, & |\mathrm{T}_{\tilde{f}}| = 0 \\ |\mathrm{T}_{\tilde{f}}|, & |\mathrm{T}_{GT}| = 0 \\ \mathrm{lev}[\mathrm{tail}(\mathrm{T}_{GT}), \mathrm{tail}(\mathrm{T}_{\tilde{f}})], & \mathrm{T}_{GT}[0] = \mathrm{T}_{\tilde{f}}[0] \\ 1 + \min \begin{cases} \mathrm{lev}[\mathrm{tail}(\mathrm{T}_{GT}), \mathrm{T}_{\tilde{f}}] \\ \mathrm{lev}[\mathrm{T}_{GT}, \mathrm{tail}(\mathrm{T}_{\tilde{f}})] \\ \mathrm{lev}[\mathrm{tail}(\mathrm{T}_{GT}), \mathrm{tail}(\mathrm{T}_{\tilde{f}})] \end{cases}, & \text{otherwise} \end{cases}.$$

Here, $|\cdot|$ is the length of the string, [0] refers to the first character, and tail truncates the first character from the string. The Levenshtein distance is solely calculated for the middle line of the text. The final OCR score in the range $[0, 100]$ is calculated by the FuzzyWuzzy python library[8]. If less than 3 lines were detected by the Tesseract software during the reconstruction, an OCR score of 0.0 is assigned to the image.

During our experiments, we observed that the Tesseract software was unable to perfectly extract the text even for the in-focus images. Therefore, we calculated a second score, called OCRI. Here, we replace the ground truth text T_{GT} by the text T_f which is extracted from the ground truth in-focus image f. Again, we use the Levenshtein distance to assign a score to the deblurred image.

[7] https://github.com/alexdenker/Hybrid-deep-learning-approaches-to-the-HDC2021-challenge
[8] https://pypi.org/project/fuzzywuzzy/

Peak Signal-to-Noise Ratio. The PSNR expresses the ratio between the maximum possible image intensity and the distorting noise, measured by the mean squared error (MSE),

$$\mathrm{PSNR}\left(\tilde{f}, f\right) := 10 \log_{10} \left(\frac{L^2}{\mathrm{MSE}\left(\tilde{f}, f\right)} \right). \tag{9}$$

Here f is the ground truth image, and $\tilde{f}$ is the reconstruction. Higher PSNR values are an indication of a better reconstruction. We choose $L = 1$, which is the maximum range of the normalized grayscale values.

Structural Similarity. Based on assumptions about human visual perception, SSIM compares the overall image structure of ground truth and reconstruction. Results lie in the range $[0, 1]$, with higher values being better. The SSIM is computed through a sliding window at M locations

$$\mathrm{SSIM}\left(\tilde{f}, f\right) := \frac{1}{M} \sum_{j=1}^{M} \frac{\left(2\tilde{\mu}_j \mu_j + C_1\right)\left(2\Sigma_j + C_2\right)}{\left(\tilde{\mu}_j^2 + \mu_j^2 + C_1\right)\left(\tilde{\sigma}_j^2 + \sigma_j^2 + C_2\right)}, \tag{10}$$

where $\tilde{\mu}_j$ and μ_j are the average pixel intensities, $\tilde{\sigma}_j$ and σ_j are the variances, and Σ_j is the covariance of $\tilde{f}$ and f at the j-th local window. Constants $C_1 = (K_1 L)^2$ and $C_2 = (K_2 L)^2$ stabilize the division. Like Wang et al. [31], we choose $K_1 = 0.01$ and $K_2 = 0.03$. The window size is 7×7 and $L = 1$.

5.2. **Text Dataset.** In this section, we evaluate our models on the test set of the HDC2021 and compare the performance with the official challenge results[9]. The full results are given in Table 1 and qualitative examples are shown in Figure 7. The corresponding inference times are provided in Table 3. Here, we assessed the OCR performance for the two fonts Times and Verdana separately. The test set includes the same number of images for both fonts, so the full OCR accuracy for the test set can be calculated by taking the mean of the results for Times and Verdana. In almost all cases the OCR performance is better for Verdana than for Times. One possible explanation could be, that Verdana consists of clearer characters without serifs. This can be seen in Figure 1, where letters and numbers of the two font types are shown with the same font size.

The results in Table 1 show that the performance of all proposed methods is more or less similar for blurring levels 4 and 9, while LGD outperforms all other methods for blurring levels 14 and 19. Moreover, comparing the OCR scores of the methods presented in this paper with those we submitted, it can be seen that the EDIP and StepNet versions result in similar accuracies, whereas the modifications for the LGD approach lead to better accuraccies. To be more precise, the LGD submission to the challenge got OCR scores of 43.88% for step 14 and 0.68% for step 19 compared to 78.80% for step 14 and 52.68% for step 19 with our adapted LGD version. However the differences between the methods could also be attributed to the size of the model. Whereas the EDIP uses a small U-Net architecture with 0.7M parameters the LGD method uses 14.5M parameters. For blur level l, the StepNet consist of $l + 1$ U-Nets with 4.1M parameters, i.e., it has $(l + 1) \cdot 4.1$M parameters.

Examples of reconstructions of the different models for the fonts Times and Verdana are depicted in Figure 7. It can be seen that all methods are able to extract

[9] https://www.fips.fi/HDCresults.php

text even if it is impossible for a human to read the text. This is particularly striking at blurring level 19, where the proposed models are still able to extract characters. Moreover, the reconstructions for the font Verdana look better than those for Times with LGD resulting in the best looking images. These observations are in line with the OCR and OCRI scores described in the previous paragraph.

Times		**Level 4**	**Level 9**	**Level 14**	**Level 19**
OCR	U-Net	91.00 ± 21.42	81.15 ± 15.43	38.55 ± 13.62	19.30 ± 11.45
	LGD	85.05 ± 21.72	81.80 ± 14.43	71.15 ± 17.54	42.95 ± 17.66
	EDIP	91.00 ± 21.42	81.15 ± 15.43	38.55 ± 13.62	19.30 ± 11.45
	StepNet	87.50 ± 8.96	65.00 ± 14.43	51.30 ± 13.27	24.15 ± 13.20
OCRI	U-Net	92.00 ± 21.70	98.00 ± 4.00	68.75 ± 13.69	21.40 ± 12.34
	LGD	96.75 ± 3.96	94.50 ± 6.69	85.95 ± 16.80	63.65 ± 14.98
	EDIP	92.00 ± 21.70	98.00 ± 4.00	68.75 ± 13.69	21.40 ± 12.34
	StepNet	97.25 ± 5.12	94.75 ± 6.61	84.15 ± 14.31	23.85 ± 10.37

Verdana		**Level 4**	**Level 9**	**Level 14**	**Level 19**
OCR	U-Net	91.25 ± 21.90	96.75 ± 4.55	68.65 ± 13.31	20.90 ± 11.34
	LGD	96.00 ± 5.62	94.25 ± 5.76	86.45 ± 15.15	62.40 ± 15.22
	EDIP	91.25 ± 21.90	96.75 ± 4.55	68.65 ± 13.31	20.90 ± 11.34
	StepNet	96.00 ± 6.44	94.00 ± 7.35	80.15 ± 16.09	22.95 ± 9.89
OCRI	U-Net	92.20 ± 21.70	98.00 ± 4.00	68.75 ± 13.69	21.40 ± 12.34
	LGD	96.75 ± 3.96	94.50 ± 6.69	85.95 ± 16.80	63.65 ± 14.98
	EDIP	92.20 ± 21.70	98.00 ± 4.00	68.75 ± 13.69	21.40 ± 12.34
	StepNet	97.25 ± 5.12	94.75 ± 6.61	84.15 ± 14.31	23.85 ± 10.37

Combined		**Level 4**	**Level 9**	**Level 14**	**Level 19**
OCR	U-Net	91.13 ± 21.66	88.95 ± 13.79	53.60 ± 20.08	20.10 ± 11.42
	LGD	90.53 ± 16.78	88.03 ± 12.63	78.80 ± 18.09	52.68 ± 19.14
	EDIP	91.13 ± 21.66	88.95 ± 13.79	53.60 ± 20.08	20.10 ± 11.42
	StepNet	91.75 ± 8.88	79.50 ± 18.48	65.73 ± 20.63	23.55 ± 11.68
OCRI	U-Net	92.50 ± 21.68	90.28 ± 13.20	54.90 ± 19.31	20.58 ± 12.02
	LGD	92.20 ± 16.72	88.43 ± 13.43	80.48 ± 18.11	53.75 ± 19.26
	EDIP	92.50 ± 21.68	90.28 ± 13.20	54.90 ± 19.31	20.58 ± 12.02
	StepNet	94.20 ± 7.62	80.65 ± 18.03	68.53 ± 19.99	23.95 ± 11.81

TABLE 1. Results for U-Net, LGD, EDIP and StepNet on the test set for the two font types on four selected blur levels. The OCR accuracy is calculated w.r.t. the middle row of the reconstruction. We report the mean and standard deviation calculated over the 20 test images in each set.

5.3. **Sanity Check.** In this section, we evaluate the PSNR and SSIM of our models on the sanity set of 16 images of the HDC2021. Additionally, we show reconstructions for different blurring levels in Figure 8.

The PSNR and SSIM results for the different blurring levels are depicted in Table 2. It can be seen that the EDIP outperforms the StepNet and the LGD on the sanity images for all blurring levels except for blurring level 19 where the LGD performs best. This shows that the training strategy of the EDIP explained

FIGURE 7. Four examples with different blur levels from the Times and Verdana test set. From left to right: Ground truth, blurry measurement, U-Net/EDIP, LGD, and StepNet reconstruction.

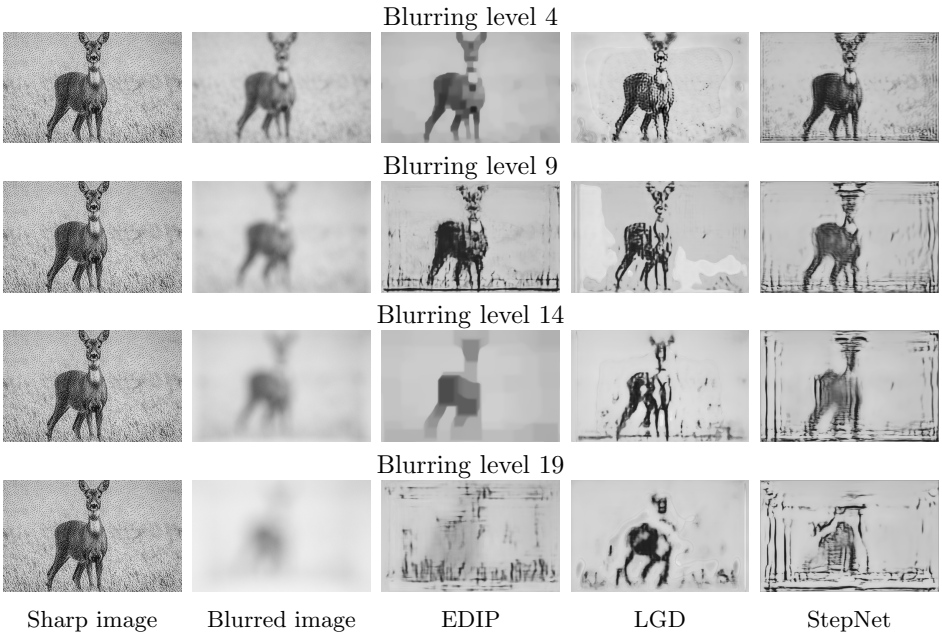


FIGURE 8. Reconstructions on one sanity test image. From left to right: Ground truth, blurry measurement, U-Net/EDIP, LGD, and StepNet reconstruction.

in Section 4.2 is able to improve the generalization ability of the U-Net without degrading the results on the text images. Consequently, applying EDIP in cases where the data discrepancy does not fall below a specified tolerance can make a network trained on a limited dataset more robust. However, it is necessary to choose the weight for the TV term in the EDIP loss. For the purpose of the challenge, this weight was chosen to achieve a good performance on the text images. Therefore, the weight may not be optimal for the natural images in the sanity set. For blurring levels 4 and 14 the EPID reconstruction contains the typical TV staircase artifacts, cf. Figure 8.

Reconstructions for one sanity test image are shown in Figure 8. It is immediately apparent that all methods are not perfectly suited for natural-looking images. This is not surprising as they are trained for the recognition of letters. Nevertheless, the reconstructions do not contain any characters. Comparing the images of the different methods, it can be seen that they are of similar quality in the first two blurring levels, while LGD results in better looking reconstructions especially in blurring level 19. This is consistent with the PSNR and SSIM values described in the last paragraph.

6. **Conclusion.** In this work, we reported on our submission to the HDC2021 and evaluated some modifications. In particular we demonstrated three ways in which the blind deblurring problem can be addressed. These methods rely on the availability of suitable training data to estimate the forward model. Our methods highlight three different philosophies of how such a learned forward operator can be used in a reconstruction method. To overcome the limited size of the training data,

Level 4	**Text**		**Sanity**	
	PSNR	**SSIM**	**PSNR**	**SSIM**
Blurred	16.29 ± 0.40	0.403 ± 0.008	13.42 ± 0.98	0.188 ± 0.073
U-Net	25.19 ± 2.10	0.630 ± 0.013	11.41 ± 2.56	0.320 ± 0.124
LGD	23.12 ± 0.38	0.605 ± 0.004	13.44 ± 1.25	0.311 ± 0.101
EDIP	25.19 ± 2.10	0.630 ± 0.013	14.21 ± 1.27	0.324 ± 0.115
StepNet	23.40 ± 0.88	0.605 ± 0.007	13.91 ± 1.78	0.310 ± 0.116

Level 9	**Text**		**Sanity**	
	PSNR	**SSIM**	**PSNR**	**SSIM**
Blurred	14.38 ± 0.42	0.442 ± 0.008	11.98 ± 0.76	0.193 ± 0.077
U-Net	22.18 ± 0.83	0.607 ± 0.006	12.40 ± 1.03	0.307 ± 0.098
LGD	23.82 ± 0.88	0.612 ± 0.006	12.63 ± 1.01	0.299 ± 0.102
EDIP	22.18 ± 0.83	0.607 ± 0.006	13.23 ± 0.92	0.312 ± 0.107
StepNet	22.13 ± 0.71	0.596 ± 0.005	12.51 ± 1.04	0.262 ± 0.099

Level 14	**Text**		**Sanity**	
	PSNR	**SSIM**	**PSNR**	**SSIM**
Blurred	13.97 ± 0.37	0.458 ± 0.009	11.15 ± 0.80	0.193 ± 0.078
U-Net	19.50 ± 0.57	0.597 ± 0.007	11.43 ± 1.01	0.290 ± 0.094
LGD	21.72 ± 0.57	0.609 ± 0.006	11.72 ± 0.90	0.296 ± 0.097
EDIP	19.50 ± 0.57	0.597 ± 0.007	12.21 ± 0.70	0.300 ± 0.097
StepNet	20.41 ± 0.43	0.600 ± 0.007	11.66 ± 1.10	0.256 ± 0.099

Level 19	**Text**		**Sanity**	
	PSNR	**SSIM**	**PSNR**	**SSIM**
Blurred	12.48 ± 0.33	0.548 ± 0.008	8.65 ± 1.17	0.228 ± 0.103
U-Net	17.61 ± 0.46	0.583 ± 0.007	9.83 ± 1.82	0.264 ± 0.105
LGD	19.06 ± 0.68	0.599 ± 0.007	10.39 ± 1.47	0.279 ± 0.103
EDIP	17.61 ± 0.46	0.583 ± 0.007	9.83 ± 1.82	0.264 ± 0.105
StepNet	17.35 ± 0.44	0.582 ± 0.007	10.06 ± 1.40	0.248 ± 0.105

TABLE 2. Mean and standard deviation for the U-Net, LGD, EDIP and StepNet on the test set (text images of both fonts) and on the the sanity images. The PSNR and SSIM are calculated using a data range of 1.

we simulated additional data pairs using the estimated forward model. It can be seen that relatively simple methods, like the baseline U-Net, already perform quite well with a pretraining on high quality synthetic data. However, for high blurring levels it seems to be important to introduce more knowledge into the neural network. As an example, the LGD performs best on the highest blurring level 19.

Improving the generalization ability of a neural network is an important open problem. In this work, we showed that the EDIP is able to enhance the reconstruction quality of a network on previously unseen image types without requiring an additional training dataset. In general, for the performance of our methods, it was necessary to introduce additional knowledge into the network architecture or the training setup, either by simulating appropriate training data or by incorporating an estimated forward model directly in the architecture.

	Level 4	**Level 9**	**Level 14**	**Level 19**
U-Net	$2.345 \pm 0.950\,\mathrm{s}$			
LGD	$0.198 \pm 0.017\,\mathrm{s}$			
EDIP	$199.5 \pm 2.5\,\mathrm{min}$	$385.3 \pm 12.6\,\mathrm{min}$	$476.0 \pm 5.7\,\mathrm{min}$	$476.0 \pm 5.7\,\mathrm{min}$
StepNet	$0.075 \pm 0.045\mathrm{s}$	$0.151 \pm 0.042\mathrm{s}$	$0.437 \pm 0.061\mathrm{s}$	$0.869 \pm 0.12\mathrm{s}$

TABLE 3. Inference time of the different methods. For EDIP the inference time is provided in the case that the data error of the initial U-Net was higher than the threshold. All computations were done on a GeForce RTX 3090 with 24GB memory.

Acknowledgments. The authors thank Sören Dittmer and Richard Schmähl for their support and helpful comments. Alexander Denker, Johannes Leuschner, and Maximilian Schmidt acknowledge the support by the Deutsche Forschungsgemeinschaft (DFG) within the framework of GRK 2224/1 "π^3: Parameter Identification – Analysis, Algorithms, Applications". Alexander Denker further acknowledges support from the Klaus Tschira Stiftung via the project MALDISTAR (project number 00.010.2019). Judith Nickel is funded by the 'Europäischer Fond für regionale Entwicklung' (EFRE) via the project ML-X-RAY (funding reference FUE0642B). Gael Rigaud is supported by the Stuttgart Center for Simulation Science (SimTech).

REFERENCES

[1] J. Adler and O. Öktem, Solving ill-posed inverse problems using iterative deep neural networks, *Inverse Problems*, **33** (2017).
[2] S. Arridge, P. Maass, O. Öktem and C.-B. Schönlieb, Solving inverse problems using data-driven models, *Acta Numerica*, **28** (2019), 1–174.
[3] D. O. Baguer, J. Leuschner and M. Schmidt, Computed tomography reconstruction using deep image prior and learned reconstruction methods, *Inverse Problems*, 36.9 (2020): 094004.
[4] R. Barbano, J. Leuschner, M. Schmidt, A. Denker et al., Is Deep Image Prior in Need of a Good Education? preprint, `arXiv:2111.11926` (2021).
[5] A. Beck and M. Teboulle, Fast Gradient-Based Algorithms for Constrained Total Variation Image Denoising and Deblurring Problems, *IEEE Transactions on Image Processing*, **18** (2009), 2419–2434.
[6] D. C. Brown, Decentering distortion of lenses, *Photogrammetric Engineering*, **32** (1996), 444–464.
[7] A. W. Fitzgibbon, Simultaneous linear estimation of multiple view geometry and lens distortion, *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR). IEEE.*, (2001).
[8] P. C. Hansen, J. G. Nagy and D. P. O'leary, Deblurring images: matrices, spectra, and filtering, *Society for Industrial and Applied Mathematics*, (2006).
[9] A. Hauptmann, B. Cox, F. Lucka, N. Huynh et al., Approximate k-Space models and deep learning for fast photoacoustic reconstruction, *International Workshop on Machine Learning for Medical Image Reconstruction*, Springer, Cham, 2018.
[10] A. Hauptmann, J. Adler, S. Arridge and O. Öktem, Multi-scale learned iterative reconstruction, *IEEE transactions on computational imaging*, 6 (2020): 843–856.
[11] J. Huang, A. Singh and N. Ahuja, Single image super-resolution from transformed self-exemplars *Proceedings of the IEEE conference on computer vision and pattern recognition*, (2015), pp. 5197–5206.
[12] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization *arXiv preprint arXiv:1412.6980*, (2014).
[13] S. Lunz, A. Hauptmann, T. Tarvainen, C.-B. Schonlieb and S. Arridge *SIAM Journal on Imaging Sciences*, (2021), 92-127.

[14] O. Ronneberger, P. Fischer and T. Brox, U-Net: convolutional networks for biomedical image segmentation, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, **9351** (2015), Springer.
[15] L. I. Rudin, S. Osher and E. Fatemi, Nonlinear total variation based noise removal algorithms *Physica D: Nonlinear Phenomena*, **60** (1992) 259–268.
[16] O. Scherzer, M. Grasmair, H. Grossauer, M. Haltmeier et al., Variational methods in imaging *Springer*, (2009).
[17] M.I. Sezan, G. Pavlovic, A. M. Tekalp and A. T. Erdem, On modeling the focus blur in image restoration, *ICASSP*, **91** (1991), pp. 2485–2488
[18] E. Y. Sidky and X. Pan, Report on the AAPM deep-learning sparse-view CT (DL-sparse-view CT) Grand Challenge, *Medical Physics*, 2021.
[19] D. Ulyanov, A. Vedaldi and V. Lempitsky, Deep image prior, *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, 9446–9454.
[20] M. Juvonen, S. Siltanen and F. S. de Moura, Helsinki Deblur Challenge 2021: description of photographic data, preprint, `arXiv:2105.10233` (2021).
[21] J. Zbontar, F. Knoll, A. Sriram, T. Murrell et al., fastMRI: an open dataset and benchmarks for accelerated MRI, preprint, `arXiv:1811.08839` (2018).
[22] F. Knoll, J. Zbontar, A. Sriram, M. J. Muckley et al., fastMRI: a publicly available raw k-Space and DICOM dataset of knee images for accelerated MR image reconstruction using machine learning, *Radiology: Artificial Intelligence*, **2** (2020), e190007.
[23] J. Leuschner, M. Schmidt, D. O. Baguer and P. Maass, LoDoPaB-CT, a benchmark dataset for low-dose computed tomography reconstruction, *Scientific Data*, **8** (2021).
[24] R. C. Gonzalez and R. E. Woods, Digital image processing (3rd edition), *Prentice-Hall, Inc.*, (2006).
[25] Y. Wu and K. He, Group normalization, *Proceedings of the European conference on computer vision (ECCV)*, (2018), 3–19.
[26] L. Xu and J. Jia, Two-phase kernel estimation for robust motion deblurring, *European conference on computer vision*, (2010).
[27] L. Xu, J. S. Ren, C. Liu and J. Jia, Deep convolutional neural network for image deconvolution, *Advances in neural information processing systems*, 27 (2014).
[28] B. Xu, N. Wang, T. Chen and M. Li, Empirical evaluation of rectified activations in convolutional network *arXiv preprint arXiv:1505.00853*, (2015).
[29] Z. Zhang, A flexible new technique for camera calibration, *IEEE Transactions on pattern analysis and machine intelligence*, 22, 11 (2000): 1330–1334.
[30] K. Zhang, W. Zuo, S. Gu, L. Zhang et al., Learning deep CNN denoiser prior for image restoration, *Proceedings of the IEEE conference on computer vision and pattern recognition*, (2017), 3929–3938.
[31] 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*, **13** (2004), 600–612.
[32] A. Kay, Tesseract: An Open-Source Optical Character Recognition Engine, *Linux J.*, **2007** (2007).
[33] International Organization for Standardization, ISO 12232:2019 - photography - digital still cameras - determination of exposure index, ISO speed ratings, standard output sensitivity, and recommended exposure index, *ICS 37.040.10 Photographic equipment. Projectors*, **3** (2019).

Received xxxx 20xx; revised xxxx 20xx; early access xxxx 20xx.

Applied Mathematics for Modern Challenges
Vol. 1, No. 2, December 2023, pp. 87-104
doi:10.3934/ammc.2023007

MODEL-BASED DEEP LEARNING APPROACHES TO THE HELSINKI TOMOGRAPHY CHALLENGE 2022

Clemens Arndt[✉1], Alexander Denker[✉1], Sören Dittmer[✉1,2], Johannes Leuschner[✉1], Judith Nickel[✉*1] and Maximilian Schmidt[✉1]

[1]Center for Industrial Mathematics, University of Bremen, Germany

[2]Department of Applied Mathematics and Theoretical Physics
University of Cambridge, United Kingdom

Abstract. The Finnish Inverse Problems Society organized the Helsinki Tomography Challenge (HTC) in 2022 to reconstruct an image with limited-angle measurements. We participated in this challenge and developed two methods: an Edge Inpainting method and a Learned Primal-Dual (LPD) network. The Edge Inpainting method involves multiple stages, including classical reconstruction using Perona-Malik, detection of visible edges, inpainting invisible edges using a U-Net, and final segmentation using a U-Net. The LPD approach adapts the classical LPD by using large U-Nets in the primal update and replacing the adjoint with the filtered back projection (FBP). Since the challenge only provided five samples, we generated synthetic data to train the networks. The Edge Inpainting Method performed well for viewing ranges above 70 degrees, while the LPD approach performed well across all viewing ranges and ranked second overall in the challenge.

1. **Introduction.** The task in computed tomography reconstruction is to recover an image x from measurements y given by the Radon transform [14]

$$y(\varphi, s) = A[x](\varphi, s) = \int_{L(\varphi,s)} x(t)\,\mathrm{d}t. \tag{1}$$

Each measurement results from an integral over a straight line L parameterized by a distance $s \in \mathbb{R}$ and an angle $\varphi \in [0, \pi]$.

When we sample s and φ sparsely, this becomes a challenging inverse problem. The goal of the HTC 2022 was to recover the shapes of 2D phantoms from limited-angle sinogram data. In limited-angle CT, the goal is to recover x while only having measurements for angles from some small interval $[\varphi_{\min}, \varphi_{\max}]$ – we denote these measurements by $A|_{[\varphi_{\min},\varphi_{\max}]}[x]$.

The challenge has two phases. In the first phase, the organizers provided a dataset of five 512×512 pixels phantoms with full-angle sinograms, filtered back projection reconstructions, segmentation masks, and information about the measurement geometry. The target phantoms are homogeneous discs with differently shaped holes. We used this data set to develop this paper's algorithms. The second phase consisted of evaluating the algorithms; thus, no changes to the submitted methods were allowed at this stage of the challenge. The evaluation was split into

2020 *Mathematics Subject Classification.* Primary: 94A08.
Key words and phrases. Limited angle tomography, deep learning, inverse problems.
*Corresponding author: junickel(at)uni-bremen.de. Alphabetical author order.

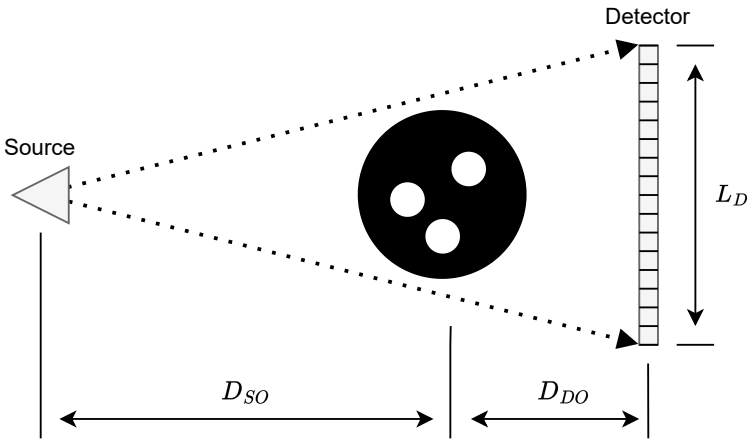


FIGURE 1. 2D Fan-beam geometry used for data collection. Adapted from https://fips.fi/HTC2022.php.

7 levels, starting with a view range of 90° in level 1 and decreasing by 10° per level down to 30° in level 7. The performance of the models is evaluated based on their ability to segment the reconstructed phantoms into material and air accurately.

We tackle this challenge with two different deep learning approaches: an adapted Learned Primal-Dual (LPD) method in Section 3.1 and an Edge Inpainting approach in Section 3.2. We trained both models using synthetic data with simulated measurements; see Section 2. The results are discussed in Section 4.

2. Dataset.

A significant difficulty of the challenge was the need for more data. The challenge organizers provided a small dataset of only 5 pairs of full view sinograms and phantoms [12]. Hence we built methods to generate many pairs of synthetic phantoms and simulated measurements similar to the five provided. We then used this synthetic data to train our two data-driven approaches.

2.1. Modeling of the forward operator.

The challenge data was collected using the University of Helsinki's in-house cone-beam computed tomography scanner. The measurement parameters, e.g., the distance of the source to origin D_{SO}, distance of detector to origin D_{D0}, number of detector pixels, and pixel size, were provided by the organizers. Using the number of detector pixels and the pixel size, one can calculate the length of the detector L_D. The data given was already background and flat-field corrected, and the attenuation data has already undergone log transformation. We used the provided measurement parameters to define a 2D fan beam ray transform operator in ODL [1] using the ASTRA [17] backend. We used this approximated forward operator to generate the new measurements from the phantom created in Section 2.2 and in the LPD model in Section 3.1.

2.2. Data generation.

All of our synthetic phantoms share the following features. They have binary values, which correspond to air and material, respectively. The material always has the same homogeneous density. The discs have a circular shape and fixed size. The center of the disc is positioned randomly around the center of the squared images.

We used four different methods to generate four different types of holes inside these discs (see Figure 2), most of them inspired by the given data.

The first method generates discs with a random number (up to 15) of circular holes. The sizes and locations of the holes are also random, and the holes are not allowed to intersect each other or the boundary.

The second method works analogously but uses randomly drawn polygons instead of circles as holes.

The third method creates holes by drawing a grid of lines inside the disc. At first, a big circular hole, almost the size of the disc itself, is set into the middle of the disc. Then, a grid of randomly drawn lines separates the big hole into smaller holes. The lines of the grid are not perfectly straight, and the orientation of the grid is random. Thus, the shapes of the resulting holes vary significantly.

The fourth method uses a Gaussian mixture model to generate holes. At first, several Gaussian functions, which define a mixture model, are initialized randomly inside the disc. Then, we use a sublevel set defined by the 70% percentile of the pixel values of these combined functions to define where material is present; the remaining parts are the holes.

These methods are fast enough to be used during training to generate training data on-the-fly.

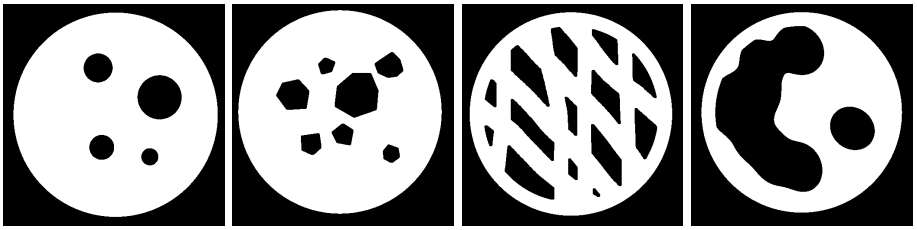


FIGURE 2. Examples of generated synthetic data samples. The different methods for data generation from left to right: circular holes, polygons, grid of lines, and Gaussian mixture.

3. **Methods.** We develop two deep learning approaches: an adapted version of the LPD method [3] trained directly for segmentation and a sequentially trained Edge Inpainting model. In the following exposition, we drop the subscript $[\varphi_{\min}, \varphi_{\max}]$ and denote the limited-angle Radon transform by A.

3.1. **Learned Primal-Dual.** In deep learning, selecting an appropriate architecture is crucial to any successful approach. In particular, traditional iterative algorithms, e.g., proximal gradient descent and primal-dual hybrid gradient [8], inspire the architecture class for *learned iterative methods* for inverse problems. One creates these powerful architectures by replacing parts of the iterative algorithms with neural networks – typically the proximal mappings. One can then train these architectures end-to-end, e.g., in a supervised manner [6]. We want to point out that while these learned iterative methods are motivated by classical iterative methods, they do not possess the same theoretical guarantees. A powerful example of a learned iterative method is the Learned Primal-Dual (LPD) proposed by Adler et al. [3]. As our first method, we build upon the LPD architecture by incorporating some common modifications [9].

The LPD swaps the proximal mappings in the dual and primal update with convolutional neural networks $F_{\theta_k} : Y^{N_{\text{dual}}} \times Y \times Y \to Y^{N_{\text{dual}}}$ and $G_{\theta_k} : X^{N_{\text{primal}}} \times X \to X^{N_{\text{primal}}}$. The networks in each unrolling step k have separate weights. We implemented the dual network F_{θ_k} as small residual convolutional networks and the primal networks G_{θ_k} as U-Nets. The use of U-Nets was motivated by the effectiveness of multi-scale architectures when working with images. Further, instead of using the adjoint A^* in the primal update, we use the filtered back projection (FBP) $A^\dagger$ using the Hann filter and a frequency cutoff of 0.75. This choice is motivated by Hauptmann et al. [9], where the authors show that this change can increase performance. The application of a filtered gradient is reminiscent of the Newton method, which has a higher convergence speed than traditional gradient descent. In the challenge, the performance was judged not by the reconstruction but by a downstream segmentation task. We address this by placing a U-Net $T_\theta : X^{N_{\text{primal}}} \to X$ after the last iteration of the LPD. In addition, we normalized the sinograms such that the model is invariant to intensity shifts. Similar to [2], we trained the resulting model directly for the segmentation task by minimizing the binary cross-entropy between the output and the provided segmentation masks. The pseudocode for the full model is given in Algorithm 1. We used $K = 4$ unrolling steps and memory channels $N_{\text{primal}} = 4$ and $N_{\text{dual}} = 2$. In total, the network has 2.4M parameters. The full model is denoted by R_Θ with $\Theta = (\theta, \theta_1, \dots, \theta_K)$ denoting all trainable parameters. The U-Nets used in the primal update and in the segmentation step consist of 4 scales with skip connection on all scales. The specific details of the implementation are given in Table 1.

Applying the forward operator and FBP during each unrolling step produces significant memory requirements during training, limiting our batch size to 6. To address this small batch size, we used group normalization [18] instead of batch normalization.

As with most deep learning models, most of the computational effort is spent in training the network. Once trained, a forward pass through the network requires only 4 evaluations of the forward operator and 5 evaluations of the FBP, which has a similar computational complexity as the adjoint. Thus, the evaluation of the LPD is cheaper than classical iterative methods which may require hundreds of iterations to create a suitable solution.

The evaluation phase of the HTC 2022 was split into 7 levels with decreasing angular ranges. Using the same training configuration, we trained one LPD model instance for each level. The specific angular subset $[\varphi_a, \varphi_b]$ was not fixed and only became known during testing. To robustify the model, we shift the sinograms by $-\varphi_a$ to the angular range $[0°, \varphi_b - \varphi_a]$, input this shifted sinogram to the LPD network, and rotate the output by φ_a to get the initial orientation back. Using this pre- and postprocessing, we can restrict the training of the LPD model to sinograms with a 0° starting angle.

We submitted three variants of this modified LPD model: trained only on the synthetic data[1], additional finetuning on the 5 challenge phantoms[2] and an additional equivariance constraint in the evaluation process[3].

For the first variant, we trained the network instances for 41 666 steps with a batch size of 6 on the synthetic phantoms with simulated measurements having

[1] https://github.com/alexdenker/htc2022_LPD2
[2] https://github.com/alexdenker/htc2022_LPD
[3] https://github.com/alexdenker/htc2022_LPD3

Algorithm 1 Modified Learned Primal-Dual

$x_0 \in X^{N_{\text{primal}}}, h_0 \in Y^{N_{\text{dual}}}$
for $k = 1, \dots, K$ **do**
  $h_k = F_{\theta_k}(h_{k-1}, Ax_{k-1}^{(1)}, y^\delta)$
  $x_k = G_{\theta_k}(x_{k-1}, A^\dagger h_k^{(0)})$
end for
$\hat{x} = T_\theta(x_K)$

TABLE 1. The implementation details of the primal, dual, and segmentation network in the LPD model.

	Primal U-Net G_{θ_k}	Segmentation U-Net T_θ
scales	4	4
channels	16, 32, 64, 64	16, 32, 64, 128
skip channels	16, 32, 32	8,8,8
activation function	Leaky ReLU	Leaky ReLU
downsampling	max pooling	max pooling
upsampling	nearest neighbor	nearest neighbor
kernel size	3	3
	Dual CNN F_{θ_k}	
number of layers	4	
channels	64	
activation function	LeakyReLU	
kernel size	3	

1% relative additive Gaussian noise. In total 250 000 synthetic data samples were generated. We used the Adam optimizer [11] with a batch size of 6 and an initial learning rate of 1×10^{-4}. We used a step learning rate scheduler which decayed the learning rate by 25% every 4166 gradient updates.

In the second variant, we additionally fine-tuned for 2000 steps on random angular subsets of the 5 challenge phantoms. Again, we used the Adam optimizer with a fixed learning rate of 5×10^{-6} and a batch size of 5. In the last variant, we experimented with an equivariance constraint during evaluation. This variant used the same weights as the fine-tuned LPD. We first compute the reconstruction given the limited-angle sinogram, i.e., $\hat{x} = R_\Theta(y)$. We fix a number of angles $0 \leq \alpha_1, \dots, \alpha_T \leq \pi$, rotate the output of the network and simulate new measurements $y_{\alpha_i} = AT_{\alpha_i}\hat{x}$ with T_{α_i} denoting a rotation matrix by the angle α_i. We then compute reconstructions from these rotated measurements $\hat{x}_{\alpha_i} = T_{-\alpha_i} R_\Theta(y_{\alpha_i})$. We then compute the final reconstruction as the mean over the rotations:

$$x = \frac{1}{T} \sum_{i=1}^{T} \hat{x}_{\alpha_i}. \tag{2}$$

In our implementation, we choose $T = 50$. We observed a slight performance increase on our synthetic data and the 5 challenge phantoms.

3.2. **Edge inpainting.** We base our second approach on the work by Bubba et. al. [7], which also deals with limited angle computed tomography. Their reconstruction

method aims to be consistent with the measurement data and reliable in that they use neural networks only for subtasks that model-based methods fail to solve. They use Quinto's fundamental visibility analysis of limited angle CT [15] to realize these objectives. The visibility analysis shows that the reliable recovery of a generalized function f's edges is only possible if the edges are tangent to a line contained in the measured data. An edge not tangent to any measured line is impossible to reconstruct.

Following [7], we call – according to the visibility analysis – recoverable edges 'visible' and all others 'invisible.' Note that we know the measurement geometry prior to reconstruction, and it fully determines edges' (in-)visibility.

In order to calculate the visible and invisible edges of a generalized function f, one can use the wavefront set of f, which is the set of positions of the singular support of f and their unsmooth directions. We refer the reader to [10] for a detailed discussion. Bubba et. al. [7] use the shearlet transform to determine the wavefront set and, in particular, the (in-)visible singularities of a function. To be more precise, they apply a variational approach with an ℓ^1-penalty in the shearlet domain to solve a sparse regularization problem and determine the visible singularities of the target function. They then use the visible coefficients of the shearlet transform as input to a neural network, which they train to estimate the invisible shearlet coefficients. In the final step, they combine the visible and invisible coefficients predicted by the classical method and the neural network, respectively, to obtain a reconstruction of the target via the inverse shearlet transform.

The setting in [7] is rather general as the algorithm can reconstruct any objective function f. In contrast, the objective functions in the challenge have a specific structure, and the goal was not to reconstruct the objective function but to create a segmentation of it. Therefore, we adapted the approach of [7] to the setting in the challenge. The main difference is that we do not use the shearlet transform but estimate the wavefront set using gradients. Similarly, we start with a variational approach using Perona-Malik and a W-shaped functional for regularization. We then use the resulting reconstruction to estimate the visible edges by calculating the gradient field and discarding all gradients corresponding to invisible edges. The visible edges are used as input to a neural network with the task of inpainting the invisible edges. In the final step of our pipeline, we use a second neural network to create a segmentation mask from this inpainted output.

Once the full pipeline is trained, both the inpainting and segmentation network can be evaluated quite fast. The main computational complexity lies in obtaining the visible edges via a variational regularization as this is done with an iterative algorithm with many evaluations of the forward operator and its adjoint. Thus, the complexity of our method is comparable to classical reconstruction methods.

In the following paragraphs, we discuss the pipeline's individual steps; see also Figure 3. The complete reconstruction scheme and all the weights of the networks are available on GitHub[4].

Classical reconstruction. First, we normalize the input sinogram data to make the reconstruction method invariant to changes in the intensity of the phantoms. We then define our variational approach via

$$\min_{x \in X} \frac{1}{2}\|Ax - y\|^2 + R(x), \tag{3}$$

[4] https://github.com/arndt-c/htc2022_edge_inpainting

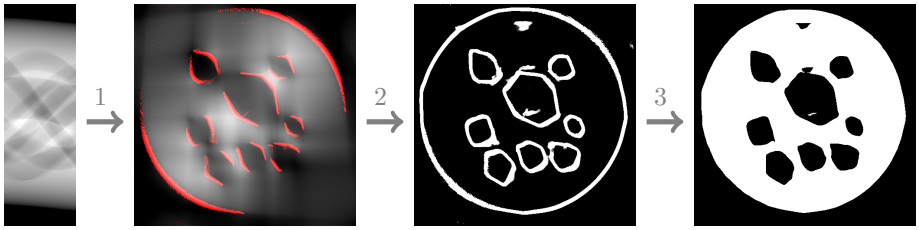


FIGURE 3. Visualization of the steps of the Edge Inpainting method: (1) variational reconstruction and extraction of visible edges, (2) inpainting of invisible edges, (3) segmentation.

where the penalty term R is a combination of the Perona-Malik function [5]

$$G(s) = \frac{T^2}{2}\left(1 - \exp\left(-\frac{s^2}{T^2}\right)\right) \quad \text{for } s = \|\nabla x_{i,j}\|_2 \tag{4}$$

and a W-shaped functional

$$W(s) = \frac{(s-a)^2(s-b)^2}{(b-a)^3} \quad \text{for } s = x_{i,j}. \tag{5}$$

Here, a and b are the air and material intensity levels, respectively, estimated according to the 5 given challenge phantoms.

We use Perona-Malik to support the subsequent edge extraction, as it sharpens the edges and smooth areas between them. The binary nature of the phantoms motivates the W-shaped functional. The choice $R(x) = \sum_{i,j} 2000\, G(\|\nabla x_{i,j}\|_2) + W(x_{i,j})$ is a suitable weighting of the penalty terms.

We solve the minimization problem (3) via alternating gradient steps w.r.t. the data discrepancy (the gradient is $A^*(Ax - y)$) and the regularization terms implemented using automatic differentiation in PyTorch. We also include optimization steps w.r.t. the values a and b to make the penalty term adaptive. Due to the high computational cost of the Radon transformation and its adjoint, we restrict the number of iteration steps to 40.

Alternatively, one could use the filtered back projection (FBP) instead of the variational method for the reconstruction. Despite its computational complexity, we chose the variational approach over the FBP as it demonstrates significantly higher robustness to noise (see Figure 4). Similar observations were also made in [7].

Estimation of visible edges. Since the overall goal is a segmented image, we only need information about the edges of the reconstruction. To obtain these, we compute the gradient field of the reconstruction from the previous step using a convolution with the Laplace filter. We then use a threshold to reject gradients with a small magnitude; all remaining gradients correspond to edges in the image. That is why Perona-Malik regularization, which prefers sharp edges and smooth areas in between, is beneficial.

The gradients also contain information about the corresponding edges' orientation (an angle between zero and 360°). If the angular range of the CT measurements is $[\varphi_{\min}, \varphi_{\max}]$, all gradients whose orientation is in one of the intervals $[\varphi_{\min} - 90, \varphi_{\max} - 90]$ or $[\varphi_{\min} + 90, \varphi_{\max} + 90]$ correspond to visible edges in a

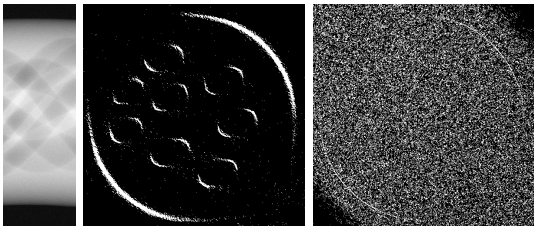


FIGURE 4. Here we compare the extracted visible edges via the variational method (middle) and the FBP (right). Both methods use the noisy sinogram on the left with 3% relative additive Gaussian noise.

parallel beam geometry. For the given fan beam geometry, we shrink the intervals of visible angles symmetrically by 10° to avoid unwanted invisible edges.

Inpainting of invisible edges with a U-Net. After extracting visible edges, we use a neural network to predict the invisible edges. The network's inputs consist of binary images, where we encode the visible edges via '1's. The output is also a binary image with visible and invisible edges encoded by '1's. The network consists of a U-Net [16] of depth 6 with skip connections on all scales. For an overview of the training and architecture parameters, see Table 2. As discussed above, we apply group normalization after each convolution due to the small batch size. The activation function is set to LeakyReLU with a negative slope of 0.2 at all layers except the last one, where we use sigmoid to enforce the output to be in the interval $[0, 1]$. In the decoder part of the U-Net, we use nearest neighbor interpolation to upsample the input of the respective blocks as it produces non-smooth images, which is beneficial given that the output should be binary.

We train the network for 16 000 steps using the Adam optimizer [11] with a learning rate of 2×10^{-5} and a batch size of 4 due to memory constraints. In total we used 64 000 samples for training. As in the LPD case, we train separate instances for each angular range, resulting in 7 different sets of network weights. We chose the weighted binary cross entropy (BCE) as our loss function with a weight on edges, i.e., areas with value '1', cf. [19].

The training data consists of on-the-fly generated synthetic phantoms, see Section 2.2, where we calculate visible and invisible edges (input and target, respectively) using the approach described in the preceding step. We want to stress that we do not use the variational approach to calculate the visible edges as it would slow down the training tremendously. We expect the overall results to be slightly better when training with visible edges extracted from reconstructions of the variational approach. However, this would require a fixed dataset, which we do not consider beneficial in this setting.

Further, we want to point out that the network can alter the extracted visible edges; hence our method, unlike the one presented in [7], does not provide any real guarantees. We opted for this approach as it simplifies the pipeline, and we observed that the inpainting network did not change the input edges significantly.

Segmentation with a U-Net. The last step of our pipeline consists of segmenting the output of the edge inpainting network. For this, we use the same U-Net architecture

TABLE 2. Details of the training setup and U-Net's architecture for the inpainting and segmentation task.

Kernel size:	9×9	Channels:	16, 32, 64, 128, 256, 256
Scales:	6	Skip channels:	16, 32, 64, 128, 256
Parameters:	≈ 34M	Downsampling:	max pooling
Optimizer:	Adam	Upsampling:	nearest neighbor
Batch size:	4	Activation:	LeakyReLU
Loss function:	weighted BCE	Overall gradient steps:	16000

as for the inpainting network and train the network using the synthetic data of Section 2.2 with the same training parameters except for the learning rate of 1×10^{-5} (see Table 2). The segmentation task does not depend on the angular range of the sinogram. However, we train the network for each angular range separately as we use the output of the previous edge inpainting network as input, which depends on the angular range.

4. **Results.** To evaluate our models, we show results on the test data of the challenge and out-of-distribution data. We detail in which cases the models show a good performance and the characteristics of the reconstruction errors – occurring, especially in more difficult levels. An overview of all results (scores and reconstructions from all methods of all participating teams) can be found on the challenge website[5].

4.1. **Challenge data.** The test set of the challenge consists of three different phantoms per angular range. The complexity of the phantoms increases as the viewing angle decreases, i.e., the phantoms contain more holes of different shapes (see Figures 7 - 10). For evaluating the different methods, the challenge organizers use the Matthews correlation coefficient (MCC) between ground truth segmentation masks I_t and segmentation calculated from the reconstruction I_r. One defines the MCC as

$$S = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

where TP denote the number of true positives, TN the number of true negatives, FP the number of false positives and FN the number of false negatives. The score S is a number between -1 and 1 where 1 indicates a perfect match and -1 indicates a total mismatch between I_r and I_t. For comparing the performance of the different methods on each level, the organizers use the overall score defined as the sum of the scores of the reconstructions on three different phantoms A, B, C for each level, i.e.,

$$S_N = S_N^A + S_N^B + S_N^C \quad \text{with } N \in \{1, \dots, 7\}.$$

For more details on the challenge's scoring system, we refer the reader to the challenge website[6].

Our best performing method on the test data was the fine-tuning LPD method without equivariance postprocessing. We depict the overall scores of the best LPD variant, the Edge Inpainting method, and the method of the challenge's winning team (provided by the challenge organizers) in Figure 5. LPD results in the best

[5] https://www.fips.fi/HTCresults.php
[6] https://www.fips.fi/HTC2022.php

scores up to level 5 and is merely slightly worse on the last two levels than the winning team's method. The Edge Inpainting method scores well on the first three levels, but the performance drops significantly from level 4 onwards.

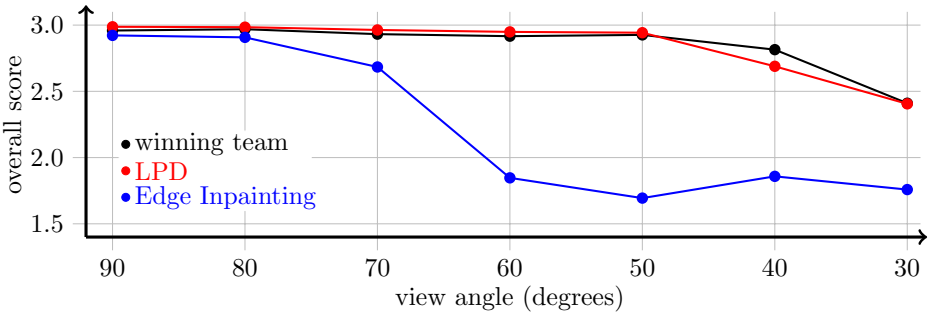


FIGURE 5. The official overall scores of our main methods in the challenge on the different levels in comparison with the method of the winning team. Accessed at https://www.fips.fi/HTCresults.php.

Figure 6 shows the overall scores of the different variants of the LPD method. One can observe that LPD's fine-tuning on the provided training data significantly boosted performance. Focusing on the fine-tuned model with equivariance post-processing, one can see that the post-processing step decreases the model's performance. With the equivariance modification, we aimed to increase the robustness of the model. However, as this modification only affects the inference of the model, it was not clear at the beginning whether this post-processing step is beneficial.

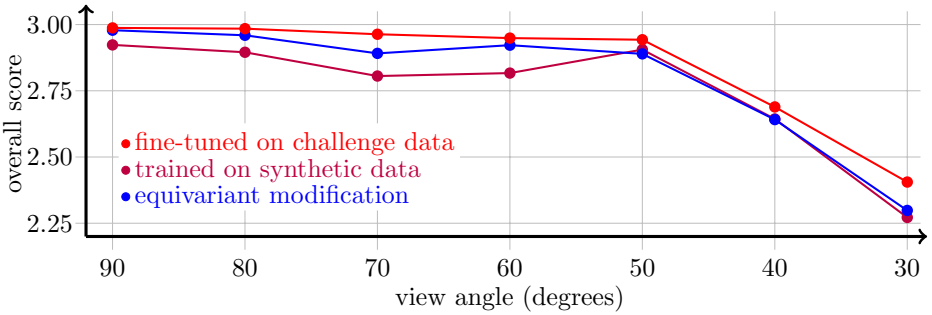


FIGURE 6. The official scores of the different variants of the LPD in the challenge.

We depict the reconstructions of the Edge Inpainting method and the fine-tuned LPD method for the three different phantoms of level 1, 3, 5, and 7 in Figures 7, 8, 9 and 10. Focusing on the reconstructions of the Edge Inpainting method, one can observe that in levels 1 and 3, the method can approximate most of the holes correctly. Nevertheless, some reconstruction errors are visible; still, they have a relatively small impact on the score. In levels 5 and 7, the inpainting method cannot correctly reconstruct the holes and shapes, which is consistent with the

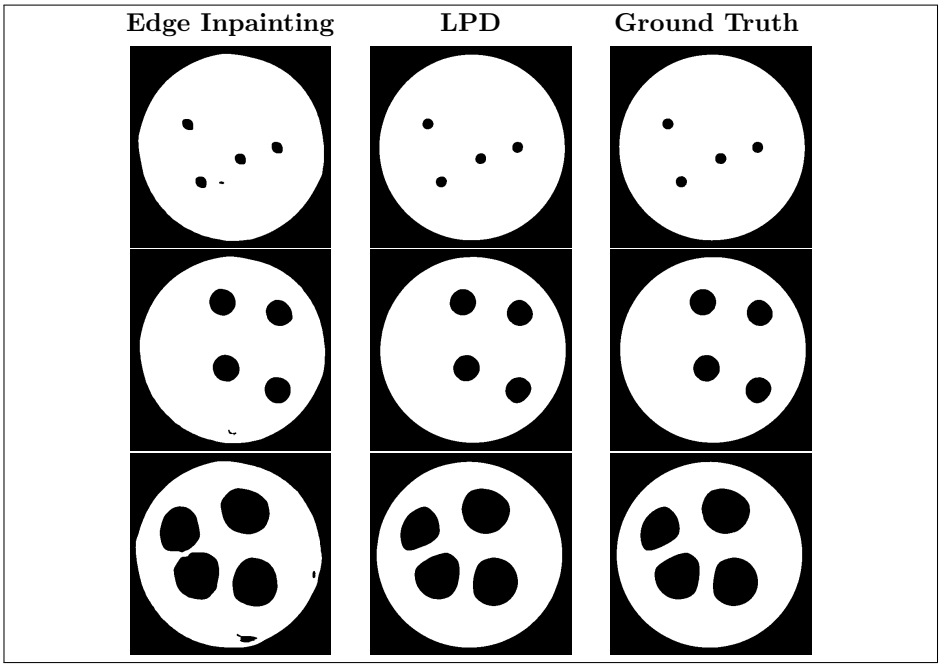


FIGURE 7. Reconstructions of the Edge Inpainting method and of LPD (fine-tuned variant) in **level 1** (90°).

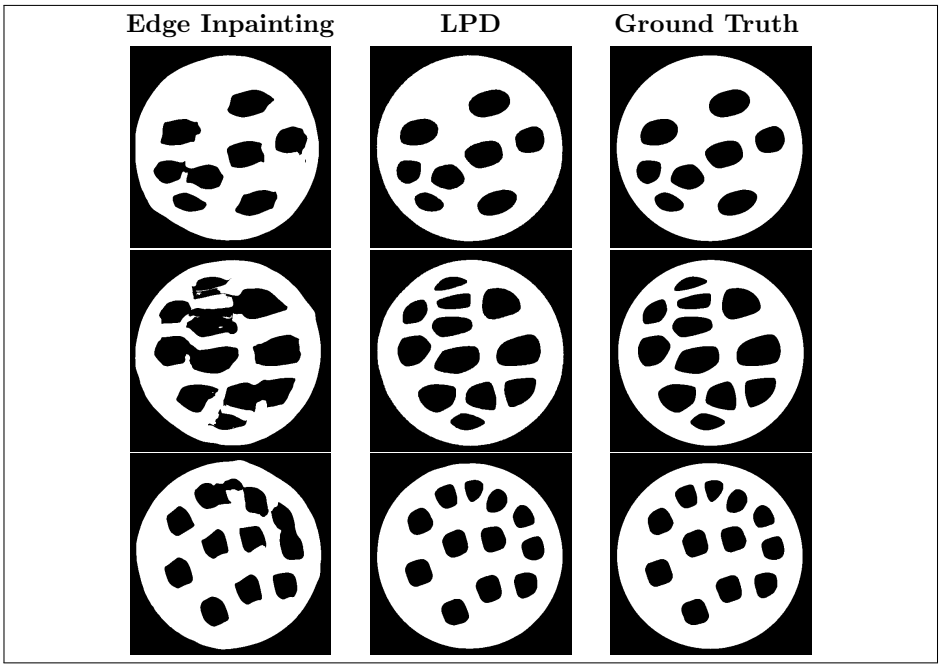


FIGURE 8. Reconstructions of the Edge Inpainting method and of LPD (fine-tuned variant) in **level 3** (70°).

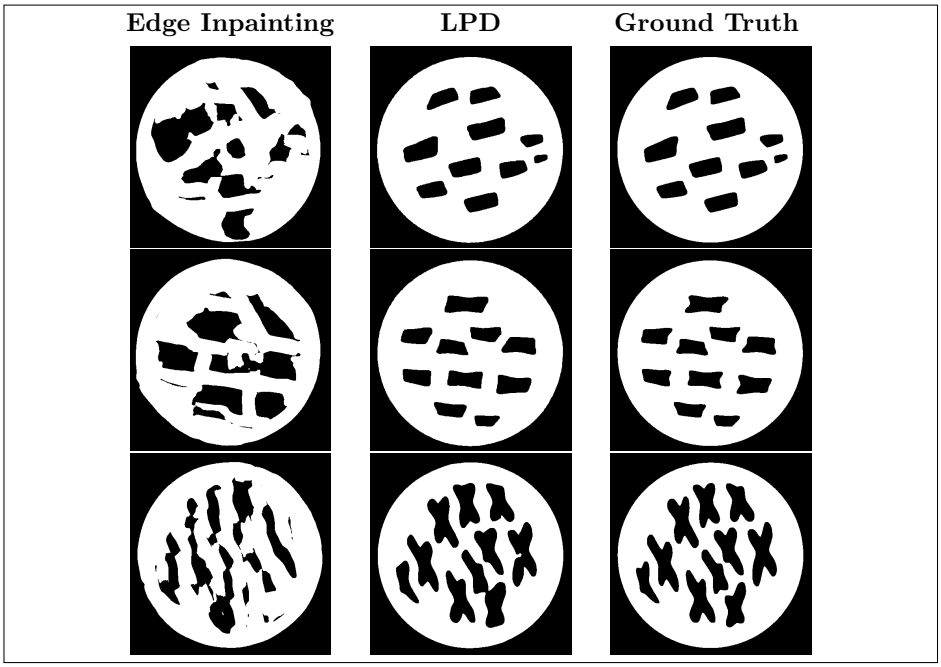


FIGURE 9. Reconstructions of the Edge Inpainting method and of LPD (fine-tuned variant) in **level 5** (50°).

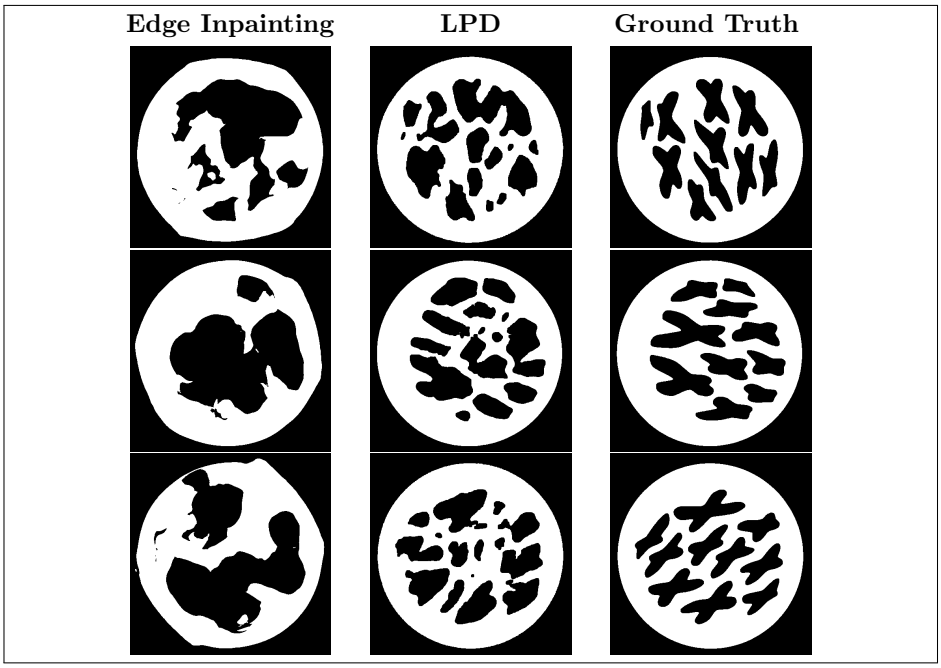


FIGURE 10. Reconstructions of the Edge Inpainting method and of LPD (fine-tuned variant) in **level 7** (30°).

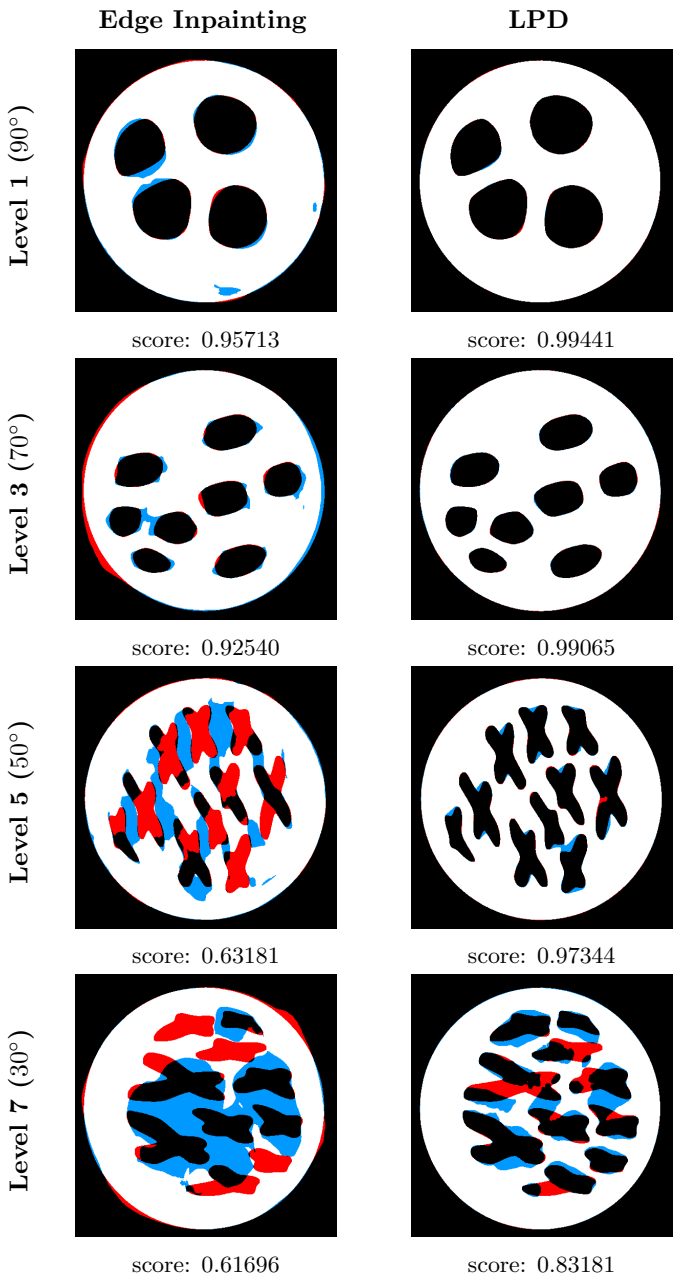


FIGURE 11. Visualization of the reconstruction errors for the Edge Inpainting method and LPD (fine-tuned variant) in **level 1** (90°)**, 3** (70°)**, 5** (50°)**, 7** (30°): white = true positive (material), black = true negative (air), red = false positive, blue = false negative.

results of Figure 5. Moreover, we can see that at all levels, the reconstructed outer discs are often not perfectly circular. With a suitable fine-tuning of the method, this error should, in principle, be avoidable since the data contains only circular disks. However, as the challenge's organizers did not officially specify the disk, we opted for a less restrictive method.

In contrast to the Edge Inpainting method, the fine-tuned LPD method results in nearly perfect reconstructions up to level 5 with very few visible reconstruction errors. In level 7, LPD can roughly estimate the holes' location but can no longer correctly reconstruct the shapes and the borders between them. Moreover, comparing the reconstructions of the different shapes, one can observe that LPD tends to fail to reconstruct non-convex holes. This is probably due to the underrepresentation of similar holes in the synthetic data.

We further illustrate the results and observations of the previous sections in Figures 11 and 12. The figure exemplifies our methods' true positives, true negatives, false positives, and false negatives at levels 1, 3, 5, and 7, respectively.

(A) score: 0.89855 (B) score: 0.84846 (C) score: 0.88876

FIGURE 12. Visualization of the reconstruction errors for the LPD variants (A) fine-tuned, (B) pre-trained, (C) fine-tuned and equivariance) in level 6 (40°): white = true positive (material), black = true negative (air), red = false positive, blue = false negative.

4.2. **Out-of-distribution data.** To test the generalization ability of the networks, we created three different out-of-distribution phantoms depicted in Figure 13.

We visualize the reconstructions, and the results of the intermediate steps of the Edge Inpainting method on level 1 (90°) in Figure 14. One can observe that

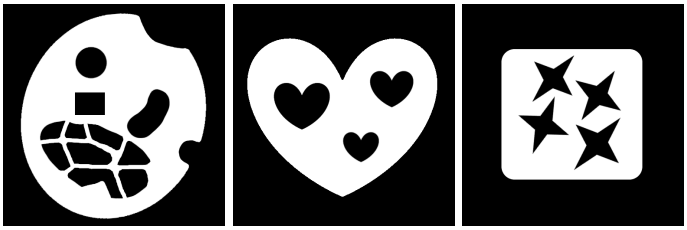


FIGURE 13. Example phantoms, which look significantly different from the challenge phantoms for testing the models on out-of-distribution data.

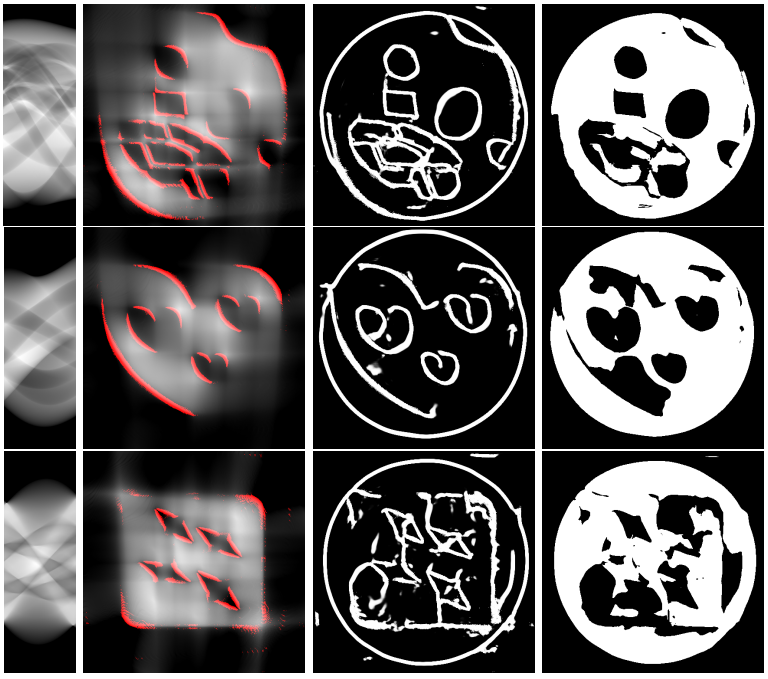


FIGURE 14. Reconstructions and intermediate steps of the Edge Inpainting method on out-of-distribution data with angular range of 90°.

the visible edges are at least visually correctly estimated, and the Edge Inpainting method does not significantly alter them. However, the Edge Inpainting method fails to connect the different shapes correctly and draws a circle in every image. This makes image segmentation exceptionally difficult, which is visible in the segmented reconstructions. These results show that the networks unsurprisingly completely adapted to the training data. Moreover, the combination of two networks might not be beneficial in this setting as the errors made by the edge inpainting network seem to get amplified in the segmentation network resulting in a loss of visible edges. Therefore, joint training of the full segmentation pipeline might lead to better results.

The LPD methods result in nearly perfect reconstructions at level 1 (90°) but deteriorate at level 2 (80°), which is why we only show the reconstructions at level 2 in Figure 15. The LPD methods can approximate the phantom's outer shape but attempt to form circles in regions with invisible edges. In addition, the network trained on synthetic data only (not fine-tuned) leads to the best-looking reconstructions except for the stars in the third phantom. Thus, there seems to be a trade-off between a high fitting on the challenge data and a good generalization ability.

5. **Discussion.** The results and observations from Section 4 motivate some ideas for further improvements of the methods.

Two phenomenons can explain many reconstruction errors of the Edge Inpainting method. First, the inpainting network connects the wrong visible edges or fails to

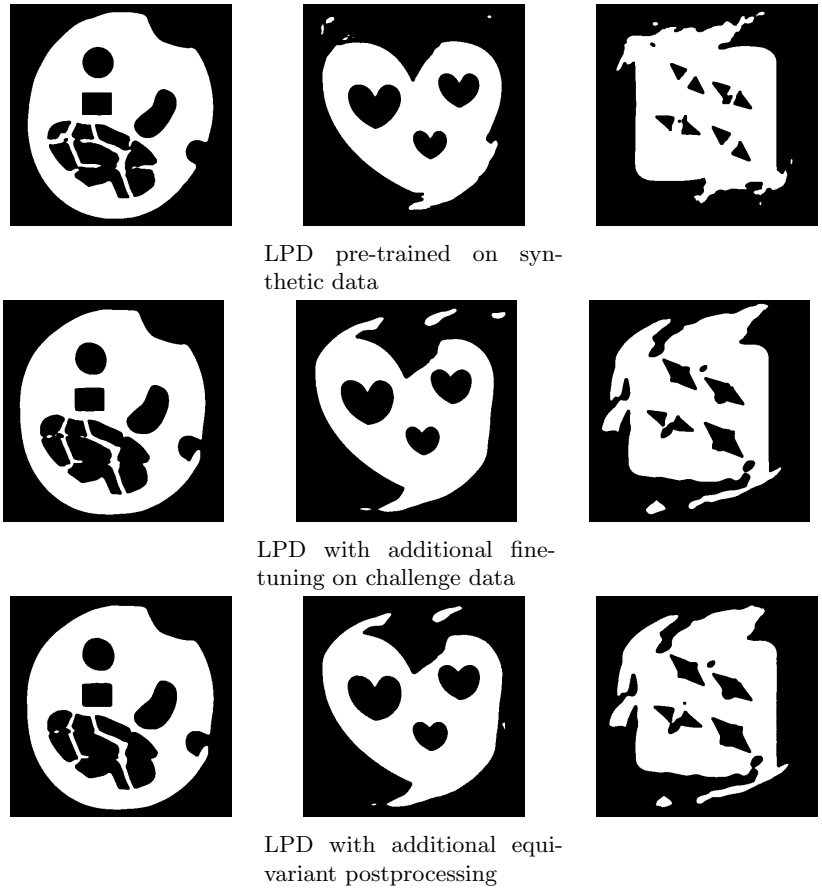


FIGURE 15. Reconstructions of the LPD variants on out-of-distribution data with angular range of 80°.

connect them. Second, the segmentation network labels the wrong areas as holes. One possibility to avoid this is to provide the networks with information about the orientation of the edges [4].

Besides, the sequential training of the two U-Nets in the Edge Inpainting method might not be beneficial. Instead, one could test an end-to-end approach or joint training of both networks; see, e.g., [2]. One could also test the approach of [7] (explained at the beginning of Section 3.2) on the challenge data and compare it with our simplified method.

For the LPD approach, the results show that this method performs very well. Nevertheless, it does not provide any reconstruction guarantees, which can be in real-world scenarios. One could combine the LPD method and the concept of visible edges to overcome this shortcoming. One could estimate the uncertainty as a first step by comparing the edges in the LPD reconstruction with the visible edges in a classical reconstruction.

If the visible edges between the classical reconstruction and the LPD approach do not align, one could correct the LPD reconstruction using the correct visible edges

of the classical reconstruction. For further insights into reconstruction guarantees in Deep Learning, we refer the reader to [13].

6. **Conclusion.** We show that it is possible to apply data-driven methods in a small data setting using a suitable chosen simulated dataset. However, the choice of simulated data is crucial, and this is only possible if a reliable approximation of the data distribution is possible in advance. Further, the generalization to new data is still an open question, as could be observed in the OOD experiment in Section 4.2. Combining model-based reconstruction with data-driven components is a promising research direction, as it ensures consistency with measured data while still providing great flexibility.

Acknowledgments. Alexander Denker and Johannes Leuschner acknowledge the support by the Deutsche Forschungsgemeinschaft (DFG) within the framework of GRK 2224/1 "π^3: Parameter Identification – Analysis, Algorithms, Applications". Maximilian Schmidt acknowledges the financial support by the German Federal Ministry for Economic Affairs and Climate Action (BMWK) and the European Social Fund (ESF) within the EXIST Transfer of Research project "aisencia".

REFERENCES

[1] J. Adler, H. Kohr and O. Öktem, Operator discretization library (ODL), *Zenodo*, (2017).
[2] J. Adler, S. Lunz, O. Verdier, C.-B. Schönlieb and O. Öktem, Task adapted reconstruction for inverse problems, *Inverse Problems*, **38**, (2022), Paper No. 075006, 21 pp.
[3] J. Adler and O. Öktem, Learned primal-dual reconstruction, *IEEE Transactions on Medical Imaging*, **37**, (2018).
[4] H. Andrade-Loarca, G. Kutyniok, O. Öktem and P. Petersen, Deep microlocal reconstruction for limited-angle tomography, *Applied and Computational Harmonic Analysis*, **59** (2022), 155-197.
[5] S. R. Arridge, M. M. Betcke and L. Harhanen, Iterated preconditioned LSQR method for inverse problems on unstructured grids, *Inverse Problems*, **30** (2014), 075009, 27 pp.
[6] S. Arridge, P. Maass, O. Öktem and C.-B. Schönlieb, Solving inverse problems using data-driven models, *Acta Numerica*, **28** (2019), 1-174.
[7] T. A. Bubba, G Kutyniok, M. Lassa, M. März, W. Samek, S. Siltanen and V. Srinivasan, Learning the invisible: A hybrid deep learning-shearlet framework for limited angle computed tomography, *Inverse Problems*, **35** (2019), 064002, 38 pp.
[8] A. Chambolle and T. Pock, An introduction to continuous optimization for imaging, *Actra Numerica*, Cambridge University Press, **25** (2016), 161-319.
[9] A. Hauptmann, J. Adler, S. Arrdige and O. Öktem, Multi-scale learned iterative reconstruction, *IEEE Transactions on Computational Imaging*, **6**, (2020).
[10] L. Hörmander, *The Analysis of Linear Partial Differential Operators. I. Distribution Theory and Fourier Analysis (reprint of the 2nd edn 1990)*, Springer, Berlin, 2003.
[11] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, arXiv preprint, `arXiv:1412.6980`, (2014).
[12] A. Meaney, F. Silva de Moura and S. Siltanen, Helsinki Tomography Challenge 2022 open tomographic dataset (HTC 2022), *Zenodo*, 2022.
[13] S. Mukherjee, A. Hauptmann, O. Öktem, M. Pereyra and C.-B. Schönlieb, Learned reconstruction methods with convergence guarantees: A survey of concepts and applications, *IEEE Signal Processing Magazine*, **40** (2023), 164-182.
[14] F. Natterer, *The Mathematics of Computerized Tomography*, SIAM, 2001.
[15] E. T. Quinto, Singularities of the X-ray transform and limited data tomography in $\mathbb{R}^2$ and $\mathbb{R}^3$, *SIAM J. Math. Anal.*, **24**, (1993), 1215-1225.
[16] O. Ronneberger, P. Fischer and T. Brox, U-net: Convolutional networks for biomedical image segmentation, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, Springer, **9351** (2015), 234-241.

[17] W. van Aarle, W. J. Palenstijn, J. De Beenhouwer, T. Altantzis, S. Bals, K. Joost Batenburg and J. Sijbers, The ASTRA toolbox: A platform for advanced algorithm development in electron tomography, *Ultramicroscopy*, **157** (2015), 35-47. .

[18] Y. Wu and K. He, Group normalization, *Proceedings of the European Conference on Computer Vision (ECCV)*, (2018), 3-19.

[19] S. Xie and Z. Tu, Holistically-nested edge detection, *IEEE International Conference on Computer Vision (ICCV), Santiago, Chile*, (2015), 1395-1403.

Received June 2023; revised October 2023; early access October 2023.