Nr. 67
16. April 2020

universität innsbruck
Institut für Mathematik

Preprint-Series: Department of Mathematics - Applied Mathematics

# Data-consistent neural networks for solving nonlinear inverse problems

Y. E. Boink, M. Haltmeier, S. Holman, J. Schwab

APPLIEDMATHEMATICS

Technikerstraße 13 - 6020 Innsbruck - Austria
Tel.: +43 512 507 53803 Fax: +43 512 507 53898
https://applied-math.uibk.ac.at

# Data-consistent neural networks for solving nonlinear inverse problems

Yoeri E. Boink

Department of Applied Mathematics, University of Twente
Multi-Modality Medical Imaging group, University of Twente
Drienerlolaan 5, 7522 NB Enschede, Netherlands
E-mail: y.e.boink@utwente.nl

Markus Haltmeier

Department of Mathematics, University of Innsbruck
Technikerstrasse 13, 6020 Innsbruck, Austria
E-mail: markus.haltmeier@uibk.ac.at

Sean Holman

School of Mathematics, University of Manchester
Oxford Road M13 9PL
E-mail: sean.holman@manchester.ac.uk

Johannes Schwab

Department of Mathematics, University of Innsbruck
Technikerstrasse 13, 6020 Innsbruck, Austria
E-mail: johannes.schwab@uibk.ac.at

March 25, 2020

**Abstract**

Data assisted reconstruction algorithms, incorporating trained neural networks, are a novel paradigm for solving inverse problems. One approach is to first apply a classical reconstruction method and then apply a neural network to improve its solution. Empirical evidence shows that such two-step methods provide high-quality reconstructions, but they lack a convergence analysis. In this paper we formalize the use of such two-step approaches with classical regularization theory. We propose data-consistent neural networks that we combine with classical regularization methods. This yields a data-driven regularization method for which we provide a full convergence analysis with respect to noise. Numerical simulations show that

compared to standard two-step deep learning methods, our approach provides better stability with respect to structural changes in the test set, while performing similarly on test data similar to the training set. Our method provides a stable solution of inverse problems that exploits both the known nonlinear forward model as well as the desired solution manifold from data.

# 1 Introduction

Let $(\mathbb{X}, \|\cdot\|)$ and $(\mathbb{Y}, \|\cdot\|)$ be Banach spaces [1] and let $\mathbf{F}\colon \mathbb{D} \subseteq \mathbb{X} \to \mathbb{Y}$ be a continuous, possibly nonlinear, mapping. In this paper we study the stable solution of the inverse problem

$$\text{Recover } x \in \mathbb{D} \text{ from } \mathbf{F}(x) = y\,, \tag{1.1}$$

where $y \in \mathbf{F}(\mathbb{D})$ are exact data. We are especially interested in the noisy data case where data $y^\delta \in \mathbb{Y}$ are given with $\|y - y^\delta\| \leq \delta$. For solving problems of this form we introduce and analyze convergent regularization methods comprising deep neural networks.

An inversion method for exact data is a right inverse $\mathbf{G}_0\colon \mathbf{F}(\mathbb{D}) \to \mathbb{D}$ for $\mathbf{F}$,

$$\forall y \in \mathbf{F}(\mathbb{D})\colon \quad \mathbf{F}(\mathbf{G}_0(y)) = y\,. \tag{1.2}$$

The inversion method $\mathbf{G}_0$ therefore recovers elements in $\mathbf{G}_0\mathbf{F}(\mathbb{D}) = \mathrm{Fix}(\mathbf{G}_0\mathbf{F})$, the set of fixed points of $\mathbf{G}_0\mathbf{F}\colon \mathbb{D} \to \mathbb{D}$. We are mainly interested in the case where (1.1) is ill-posed, where no continuous right inverse $\mathbf{G}_0$ exists. If noisy data $y^\delta \in \mathbb{Y}$ are given with $\|\mathbf{F}(x) - y^\delta\| \leq \delta$ for $x \in \mathbf{G}_0\mathbf{F}(\mathbb{D})$, then $\mathbf{G}_0(y^\delta)$ is either not well defined or arbitrary far away from $x$. In this case one has to apply regularization methods to the data, which are stable approximations to $\mathbf{G}_0$ and continuous on all of $\mathbb{Y}$.

## Background

Well established regularization methods are quadratic Tikhonov regularization [8, 21, 29] and iterative regularization [4, 12]. Both methods are designed to approximate minimum norm solutions $\mathbf{G}_0(y) \in \arg\min\{\|x - x_0\| \mid \mathbf{F}(x) = y\}$ with fixed $x_0 \in \mathbb{X}$. However, for most applications minimum norm solutions are not the desired ones. One way to overcome this issue is by convex variational regularization [25], where one takes

$$\mathbf{G}_\alpha(y^\delta) \in \arg\min\left\{\frac{1}{2}\|\mathbf{F}(x) - y^\delta\|^2 + \alpha\mathcal{R}(x) \mid x \in \mathbb{D}\right\}, \tag{1.3}$$

which approximate $\mathcal{R}$-minimizing solutions $\mathbf{G}_0(y) \in \arg\min\{\mathcal{R}(x) \mid \mathbf{F}(x) = y\}$. Here the regularization functional $\mathcal{R}\colon \mathbb{X} \to [0, \infty]$ incorporates a-priori information and takes the role of the norm. There are still several challenges related to variational regularization techniques. First, computing $\mathcal{R}$-minimizing solutions requires time consuming iterative minimization schemes. Second, finding a regularization functional that well models solutions of interest is a difficult issue. Typical choices such as total variation or

[1] You can take $\mathbb{X}$ and $\mathbb{Y}$ as finite dimensional spaces $\mathbb{R}^n$ and $\mathbb{R}^q$ with the Euclidian norm if you are not familiar with infinite dimensional Banach spaces.

the $\ell^q$-norm with respect to some frame enforce strong handcrafted prior assumptions that are often not met in practical applications. One solution for this problem was proposed in [18, 19], where a regularization term in the form of a neural network is learned before it is applied in the classical setting.

Recently, several deep learning methods to solve inverse problems were proposed. Some approaches apply iterative methods, which alternate between data consitency steps and neural network updates, [1, 22, 13, 28, 14, 7], while others aim for a fully learned reconstruction scheme [30, 6]. Further, a popular approach for imaging problems is using a neural network as a second step after some initial reconstruction. Several such post processing methods have been considered in the literature [10, 11, 13, 23]. In these two-step approaches, the reconstruction network takes the form $\mathbf{R} = \mathbf{\Phi}\mathbf{G}$ where $\mathbf{G}\colon \mathbb{Y} \to \mathbb{X}$ maps the data to the reconstruction space (reconstruction layer or backprojection) and $\mathbf{\Phi}\colon \mathbb{X} \to \mathbb{X}$ is a neural network (NN) whose free parameters are adjusted to the training data. In particular, so called residual networks $\mathbf{\Phi} = \mathrm{Id}_{\mathbb{X}} + \mathbf{U}$, where only the residual part $\mathbf{U}$ is trained [9, 11, 15, 2] show very accurate results for solving inverse problems. Due to the huge amount of recent research in this field our discussion of related work is necessarily incomplete, although we have tried to mention the foundational works.

In order to address the ill-posedness of linear inverse problems, regularizing networks of the form $\mathbf{R}_\alpha = \mathbf{\Phi}_\alpha \mathbf{G}_\alpha$ were introduced in [27, 26]. Here $\mathbf{G}_\alpha\colon \mathbb{Y} \to \mathbb{X}$ defines any regularization and $\mathbf{\Phi}_\alpha\colon \mathbb{X} \to \mathbb{X}$ are trained neural networks approximating data-consistent networks. In the linear case these networks are called nullspace networks because they only add parts in the kernel of $\mathbf{F}$ as proposed in [20, 26]. In this paper we derive convergence and convergence rates for data consistent network families $(\mathbf{R}_\alpha)_{\alpha>0}$ for nonlinear problems and provide some numerical examples.

## Regularizing networks for nonlinear problems

Let $\mathbf{G}_0$ be any right inverse of $\mathbf{F}$ and $(\mathbf{G}_\alpha)_{\alpha>0}$ a regularization of $\mathbf{G}_0$, for example classical Tikhonov regularization. Let $(\mathbf{\Phi}_\alpha)_{\alpha>0}$ be a family of Lipschitz continuous mappings $\mathbf{\Phi}_\alpha\colon \mathbb{X} \to \mathbb{X}$. In this paper we show that under suitable assumptions the reconstruction networks

$$\mathbf{R}_\alpha := \mathbf{\Phi}_\alpha \mathbf{G}_\alpha\colon \mathbb{Y} \to \mathbb{X}, \tag{1.4}$$

define a convergent regularization method. Additionally, we derive convergence rates (quantitative error estimates) for the reconstruction error.

A main condition for these results is that $\mathbf{\Phi}_\alpha \mathbf{G}_\alpha \mathbf{F}$ converges pointwise to a network of the form $\mathbf{\Phi}_0 \mathbf{G}_0 \mathbf{F}$ where $\mathbf{\Phi}_0\colon \mathbb{X} \to \mathbb{X}$ is data-consistent in the sense that $\mathbf{F}\mathbf{\Phi}_0 z = \mathbf{F}z$ for $z \in \mathbf{G}_0\mathbf{F}(\mathbb{D})$. The latter property implies that $\mathbf{\Phi}_0$ preserves data consistency of $\mathbf{G}_0$, meaning that if $\mathbf{G}_0(y)$ is a solution of (1.1), then $\mathbf{\Phi}_0\mathbf{G}_0(y)$ is a solution of (1.1) too. Hence the goal of the learned mapping $\mathbf{\Phi}_0$ is to improve solutions of the inverse problem, while keeping data-consistency of the initial solution. We prove that the family of reconstruction networks $(\mathbf{\Phi}_\alpha \mathbf{G}_\alpha)_{\alpha>0}$ is a convergent regularization method, and we provide convergence rates.

The benefits of $(\mathbf{\Phi}_\alpha \mathbf{G}_\alpha)_{\alpha>0}$ over $(\mathbf{G}_\alpha)_{\alpha>0}$ are twofold. First, in the limit $\alpha \to 0$, the network $\mathbf{\Phi}_0\mathbf{G}_0\mathbf{F}$ selects solutions in $\mathbf{\Phi}_0\mathbf{G}_0\mathbf{F}(\mathbb{D})$ that can be trained to better reflect the desired image class than $\mathbf{G}_0\mathbf{F}(\mathbb{D})$. Second, for $\alpha > 0$, the networks $\mathbf{\Phi}_\alpha$ can be trained to

undo the smoothing effect of $\mathbf{G}_\alpha\mathbf{F}$ and thereby allow for obtaining convergence rates for less regular elements than the original regularization method $(\mathbf{G}_\alpha)_{\alpha>0}$. The operator $\mathbf{\Phi}_0\mathbf{G}_0$ can be seen as a right inverse that is learned from a suitable class of training data.

### Outline

The paper is organized as follows. In section 2 we describe the regularization of nonlinear inverse problems and we define the proposed two-step data-driven regularization method. We introduce data-consistent networks, which allow to define the regularization method called *regularizing networks*, which approximate a data-driven right inverse. We investigate under which assumptions these networks generate a convergent regularization method and we give examples of how such regularizing networks can be constructed. In section 3 we present a convergence analysis and derive convergence rates for the proposed method. Section 4 presents the mathematical description of the inverse problems considered in the numerical simulations. These simulations are explained in detail in section 5, after which results are shown in section 6. Additional simulation results can be found in the appendices. The paper concludes with a short summary of the established theory and the numerical simulations.

## 2 Convergence of regularizing networks

Throughout the rest of this paper, let $\mathbf{F}\colon \mathbb{D} \subseteq \mathbb{X} \to \mathbb{Y}$ be a continuous mapping between Banach spaces $\mathbb{X}$ and $\mathbb{Y}$. We study the stable solution of the inverse problem (1.1). In this section we introduce the regularizing networks and present the convergence analysis.

### 2.1 Regularization of inverse problems

Let $\mathbf{G}_0\colon \mathbf{F}(\mathbb{D}) \to \mathbb{D}$ be any right inverse for $\mathbf{F}$. If (1.1) is ill-posed and $y^\delta$ are noisy data with $\|\mathbf{F}(x) - y^\delta\| \leq \delta$, then the reconstruction method $\mathbf{G}_0(y^\delta)$ is unstable, meaning arbitrarily far away from $\mathbf{G}_0\mathbf{F}(x)$ or not defined. To obtain meaningful approximations of $\mathbf{G}_0\mathbf{F}(x)$, one has to apply regularization methods defined as follows.

**Definition 2.1** (Regularization method). Let $(\mathbf{G}_\alpha)_{\alpha>0}$ be a family of continuous mappings $\mathbf{G}_\alpha\colon \mathbb{Y} \to \mathbb{X}$. If for all $x \in \mathbf{G}_0\mathbf{F}(\mathbb{D})$ there exists a parameter choice function $\alpha^*\colon (0,\infty) \times \mathbb{Y} \to (0,\infty)$ such that

$$
\begin{aligned}
0 &= \lim_{\delta\to 0} \sup\{\alpha^*(\delta, y^\delta) \mid y^\delta \in B_\delta(\mathbf{F}(x))\} \\
0 &= \lim_{\delta\to 0} \sup\{\|x - \mathbf{G}_{\alpha^*(\delta,y^\delta)}(y^\delta)\| \mid y^\delta \in B_\delta(\mathbf{F}(x))\},
\end{aligned}
$$

where $B_\delta(\mathbf{F}(x))$ is the ball with radius $\delta$ around $\mathbf{F}(x)$, we call $((\mathbf{G}_\alpha)_{\alpha>0}, \alpha^*)$ a regularization method for $\mathbf{G}_\alpha$. If $((\mathbf{G}_\alpha)_{\alpha>0}, \alpha^*)$ is a regularization method for $\mathbf{G}_0$, we call $(\mathbf{G}_\alpha)_{\alpha>0}$ a regularization of $\mathbf{G}_0$ and $\alpha^*$ an admissible parameter choice.

Probably the best known regularization is quadratic Tikhonov regularization in Hilbert spaces [8]. Under the assumption that $\mathbf{F}$ is weakly sequentially closed, one shows that

there exist solutions of (1.1) with minimal distance to a given point $x_0 \in \mathbb{X}$ and that

$$\mathcal{T}_{\alpha,y^\delta}(x) := \frac{1}{2}\|\mathbf{F}(x) - y^\delta\|^2 + \frac{\alpha}{2}\|x - x_0\|^2 \quad \text{for } x \in \mathbb{D} \tag{2.1}$$

has at least one minimizer. We can define $\mathbf{G}_\alpha(y^\delta)$ as any minimizer of $\mathcal{T}_{\alpha,y^\delta}$. If the solution of (1.1) with minimal distance to $x_0$ is unique and denoted by $\mathbf{G}_0(y)$, then $((\mathbf{G}_\alpha)_{\alpha>0}, \alpha^*)$ with a parameter choice satisfying $\delta^2/\alpha^* \to 0$ and $\alpha^* \to 0$ as $\delta \to 0$ is a regularization method for $\mathbf{G}_0$ [8, 21].

Research indicates that solutions with minimal distance to a fixed initial guess $x_0 \in \mathbb{X}$ are too simple in many applications. The use of non-quadratic penalties has demonstrated to often give better results. Recently, deep learning methods showed outstanding performance. Here solutions are defined by a neural network that maps the given data to a desired solution.

## 2.2 Data-consistent networks

The first ingredient for constructing regularizing two-step networks are data-consistent networks.

**Definition 2.2** (Data-consistent network). We call $\mathbf{\Phi}_0 \colon \mathbb{X} \to \mathbb{X}$ a data-consistent network if $\mathbf{\Phi}_0$ is Lipschitz continuous and $\forall z \in \mathbf{G}_0\mathbf{F}(\mathbb{D})\colon \mathbf{F}\mathbf{\Phi}_0(z) = \mathbf{F}(z)$.

In data-consistent networks, if $z \in \mathbf{G}_0\mathbf{F}(\mathbb{D})$ is a solution of (1.1), then $\mathbf{\Phi}_0(z)$ is solution of (1.1) too. In particular, $\mathbf{\Phi}_0\mathbf{G}_0$ is a right inverse for $\mathbf{F}$ with solution set $\mathbf{\Phi}_0\mathbf{G}_0\mathbf{F}(\mathbb{D}) = \mathrm{Fix}(\mathbf{\Phi}_0\mathbf{G}_0\mathbf{F})$. Data-consistent networks can be constructed by

$$\mathbf{\Phi}_0(z) = \mathbf{P}_{z,0}(\mathbf{U}(z)), \tag{2.2}$$

where $\mathbf{U}\colon \mathbb{X} \to \mathbb{X}$ is a Lipschitz continuous trained neural network, and $\mathbf{P}_{z,0}\colon \mathbb{X} \to \mathbb{X}$ a Lipschitz continuous mapping with $\mathbf{P}_{z,0}(x) \in \mathbf{F}^{-1}(\mathbf{F}(z)) = \{x \in \mathbb{D} \mid \mathbf{F}(x) = \mathbf{F}(z)\}$. The mapping $\mathbf{P}_{z,0}$ can be seen as a generalized projection on $\mathbf{F}^{-1}(\mathbf{F}(z))$. In the special case where $\mathbf{F}$ is a linear mapping, $\mathbf{\Phi}_0(z)$ can be chosen as $\mathbf{\Phi}_0(z) = z + \mathbf{P}_{\ker(\mathbf{F})}\mathbf{U}(z)$, where $\mathbf{P}_{\ker}$ is the projection on the kernel of $\mathbf{F}$ [26].

**Definition 2.3** (Regularizing networks). We call $(\mathbf{R}_\alpha\colon \mathbb{Y} \to \mathbb{X})_{\alpha>0}$ defined by $\mathbf{R}_\alpha := \mathbf{\Phi}_\alpha \circ \mathbf{G}_\alpha$ a family of regularizing networks if the following hold:

(R1) $(\mathbf{G}_\alpha\colon \mathbb{Y} \to \mathbb{X})_{\alpha>0}$ is a regularization of $\mathbf{G}_0$.

(R2) $(\mathbf{\Phi}_\alpha\colon \mathbb{X} \to \mathbb{X})_{\alpha>0}$ are uniformly $L$-Lipschitz continuous mappings.

(R3) For some data-consistent network $\mathbf{\Phi}_0\colon \mathbb{X} \to \mathbb{X}$ we have

$$\forall x \in \mathbb{D}\colon \lim_{\alpha\to 0} \mathbf{\Phi}_\alpha\mathbf{G}_\alpha\mathbf{F}(x) = \mathbf{\Phi}_0\mathbf{G}_0\mathbf{F}(x). \tag{2.3}$$

In practice an important issue is to design networks that converge to a data-consistent limiting network as the noise level goes to zero. Next we give examples for a possible strategy to train such networks.

**Example 2.4.** *Let $x_1, \dots, x_N$ be training signals and $y_i = \mathbf{F}(x_i)$ and respectively $y_i^\delta$ the corresponding data. Further define the vectors of reconstructions $v := (v_1, \dots, v_N)$ and $v^\alpha := (v_1^\alpha, \dots, v_N^\alpha)$ where $v_i = \mathbf{G}_0(y_i)$ and $v_i^\alpha = \mathbf{G}_\alpha(y_i^\delta)$. The weights of the neural network are denoted by $\theta \in \Theta$. We write $\mathbf{\Phi}^\theta$ to depict a neural network with fixed architecture, whose weights have not yet been fixed.*

*(a) One possible simple approach is to take the networks $\mathbf{\Phi}_\alpha := \mathbf{\Phi}_0$ for all $\alpha > 0$, where $\mathbf{\Phi}_0$ is the network obtained by minimizing the functional*

$$\min_\theta \sum_i^N \|\mathbf{\Phi}^\theta(v_i) - x_i\|^2 + \mathcal{R}(\theta)\,. \tag{2.4}$$

*Here $\mathcal{R}$ denotes some regularization functional for the weights $\theta$ that may be used to ensure a small Lipschitz constant. Clearly, since $\mathbf{\Phi}_0$ is Lipschitz continuous and $(\mathbf{G}_\alpha)_{\alpha>0}$ is a regularization method, we have the desired limit in 2.3 for all $x \in \mathbb{D}$. Now if the data consistency is incorporated in the network architecture, the condition (R3) is satisfied.*

*(b) A more sophisticated approach is to choose the sequence of networks depending on the regularization parameter $\alpha$. Here the networks $\mathbf{\Phi}_\alpha$ are obtained by minimizing*

$$\min_\theta \sum_i^N \|\mathbf{\Phi}^\theta(v_i^\alpha) - x_i\|^2 + \mathcal{R}(\theta).$$

*To enforce the data consistency of the limiting network $\mathbf{\Phi}_0$ one could either choose the network architecture to be data-consistent, meaning $\forall \theta \in \Theta\ \forall x \in \mathbb{D}$: $\mathbf{F}(\mathbf{\Phi}^\theta(x)) = \mathbf{F}(x)$, or taking networks increasingly data-consistent of the form*

$$\mathbf{\Phi}_\alpha(z) = \mathbf{P}_{z,\alpha}\mathbf{U}_\alpha(z)\,. \tag{2.5}$$

*Here $\mathbf{U}_\alpha \colon \mathbb{X} \to \mathbb{X}$ is a trained networks and $\mathbf{P}_{z,\alpha}$ is a Lipschitz continuous mapping with $\operatorname{Im}(\mathbf{P}_{z,\alpha}) \subseteq E_{\alpha,z} := \{x \mid \|\mathbf{F}(x) - \mathbf{F}(z)\| \leq r(\alpha)\}$ with $\lim_{\alpha\to 0} r(\alpha) = 0$. Data-consistency is obtained in the limit. One example for $\mathbf{P}_{z,\alpha}$ is the metric projection on $E_{\alpha,z}$ which is Lipschitz continuous if $E_{\alpha,z}$ is convex.*

*Note that in (2.5) there are no restrictions on the particular choice of the architecture of the networks $\mathbf{U}_\alpha$.*

*(c) Another network architecture guaranteeing data consistency is given by*

$$\mathbf{\Phi}_\alpha(z) = \mathbf{\Phi}^{\text{dec}}\left(\mathbf{S}_0 + \alpha\mathbf{S}_1\right)\mathbf{\Phi}^{\text{enc}}(z)\,. \tag{2.6}$$

*Here $\mathbf{\Phi}^{\text{enc}}$ and $\mathbf{\Phi}^{\text{dec}}$ denote an encoder and decoder network respectively, $\mathbf{S}_0$ denotes an $\alpha$-independent network and $\mathbf{S}_1$ denotes a network that is allowed to depend on $\alpha$.*

## 2.3 Convergence analysis

**Theorem 2.5** (Regularizing networks). *Any family of regularizing networks $(\mathbf{R}_\alpha = \mathbf{\Phi}_\alpha\mathbf{G}_\alpha)_{\alpha>0}$ (see Definition 2.3) is a regularization for $\mathbf{\Phi}_0\mathbf{G}_0$ in the sense of Definition 2.1.*

*Proof.* Let $x \in \mathbf{\Phi}_0\mathbf{G}_0\mathbf{F}(\mathbb{D})$, $y^\delta \in \mathbb{Y}$ with $\|\mathbf{F}(x) - y^\delta\| \leq \delta$ and set $x_\alpha^\delta := \mathbf{\Phi}_\alpha\mathbf{G}_\alpha(y^\delta)$. Then

$$\begin{aligned}\|x - x_\alpha^\delta\| &= \|\mathbf{\Phi}_0\mathbf{G}_0\mathbf{F}(x) - \mathbf{\Phi}_\alpha\mathbf{G}_\alpha y^\delta\| \\ &\leq \|\mathbf{\Phi}_0\mathbf{G}_0\mathbf{F}(x) - \mathbf{\Phi}_\alpha\mathbf{G}_\alpha\mathbf{F}(x)\| + \|\mathbf{\Phi}_\alpha\mathbf{G}_\alpha\mathbf{F}(x) - \mathbf{\Phi}_\alpha\mathbf{G}_\alpha y^\delta\| \\ &\leq \|\mathbf{\Phi}_\alpha\mathbf{G}_\alpha\mathbf{F}(x) - \mathbf{\Phi}_0\mathbf{G}_0\mathbf{F}(x)\| + L\|\mathbf{G}_\alpha\mathbf{F}(x) - \mathbf{G}_\alpha y^\delta\|\,.\end{aligned}$$

Now if $\alpha^*(\delta, y^\delta)$ is an admissible parameter choice for $(\mathbf{G}_\alpha)_{\alpha>0}$, then

$$\begin{aligned}\sup\{\|x - \mathbf{R}_{\alpha^*(\delta,y^\delta)}(y^\delta)\| \mid y^\delta \in B_\delta(\mathbf{F}(x))\} &\leq \|\mathbf{\Phi}_{\alpha^*(\delta,y^\delta)}\mathbf{G}_{\alpha^*(\delta,y^\delta)}\mathbf{F}(x) - \mathbf{\Phi}_0\mathbf{G}_0\mathbf{F}(x)\| \\ &\quad + L\sup\{\|\mathbf{G}_{\alpha^*(\delta,y^\delta)}\mathbf{F}(x) - \mathbf{G}_{\alpha^*(\delta,y^\delta)}y^\delta\| \mid y^\delta \in B_\delta(\mathbf{F}(x))\}\,. \quad (2.7)\end{aligned}$$

According to (R3) in Definition 2.3, the first term converges to zero and because of

$$\|\mathbf{G}_{\alpha^*(\delta,y^\delta)}\mathbf{F}(x) - \mathbf{G}_{\alpha^*(\delta,y^\delta)}y^\delta\| \leq \|\mathbf{G}_{\alpha^*(\delta,y^\delta)}\mathbf{F}(x) - \mathbf{G}_0\mathbf{F}(x)\| + \|\mathbf{G}_0\mathbf{F}(x) - \mathbf{G}_{\alpha^*(\delta,y^\delta)}y^\delta\|$$

and the fact that $(\mathbf{G}_\alpha)_{\alpha>0}$ with $\alpha = \alpha^*(\delta, y^\delta)$ is a regularization method for $\mathbf{G}_0$, the second term converges to zero as $\delta \to 0$. □

# 3 Convergence rates

Another important issue is the rate of approximation. This means specifically that there exists a decreasing function $f\colon (0,\infty) \to (0,\infty)$ such that $\lim_{\delta\to 0} f(\delta) = 0$ and $\|\mathbf{R}_{\alpha^*(\delta,y^\delta)}(y^\delta) - x\| \leq f(\delta)$ uniformly for all $y^\delta \in \mathbb{Y}$ with $\|\mathbf{F}(x) - y^\delta\| \leq \delta$.

## 3.1 Reconstruction algorithms and convergence rates

**Definition 3.1** (Reconstruction error of an algorithm). Let $\mathbb{X}_0 \subseteq \mathbb{X}$, $\delta > 0$ and $\mathbf{G}\colon \mathbb{Y} \to \mathbb{X}$ be a reconstruction algorithm. We call

$$\mathcal{E}(\mathbf{G}, \delta, \mathbb{X}_0) = \sup\{\|x - \mathbf{G}(y^\delta)\| \mid x \in \mathbb{X}_0 \wedge y^\delta \in \overline{B_\delta(\mathbf{F}(x))}\} \quad (3.1)$$

the reconstruction error of $\mathbf{G}$ over $\mathbb{X}_0$.

**Definition 3.2** (Convergence rate of an algorithm). Let $\mathbb{X}_0 \subseteq \mathbb{X}$, $r \in (0, 1]$ and for any $\delta > 0$, let $\mathbf{G}^\delta$ be a reconstruction algorithm. We say that $(\mathbf{G}^\delta)_{\delta>0}$ converges at rate $\delta^r$ over $\mathbb{X}_0$ if $\mathcal{E}(\mathbf{G}^\delta, \delta, \mathbb{X}_0) = \mathcal{O}(\delta^r)$ as $\delta \to 0$.

The concept of convergence rates in particular applies for reconstruction algorithms defined by regularization methods. In general, no convergence rate over $\mathbf{G}_0\mathbf{F}(\mathbb{D})$ is possible; they require restricting to proper subsets $\mathbb{X}_0 \subsetneq \mathbf{G}_0\mathbf{F}(\mathbb{D})$ [8, Proposition 3.11].

Source conditions define suitable sets $\mathbb{X}_0$ for classical Tikhonov regularization and related methods based on minimal norm solutions. We investigate the source conditions (transformed source sets) and convergence rates for regularizing networks where $(\mathbf{G}_\alpha)_{\alpha>0}$ is Tikhonov regularization in Example 3.5.

### 3.2 Rates for the regularizing networks

Our aim is to prove a convergence rate for $\mathbf{R}_{\alpha^*(\delta,y^\delta)}$ assuming a convergence rate for $\mathbf{G}_{\alpha^*(\delta,y^\delta)}$. Let $(\mathbf{\Phi}_\alpha \mathbf{G}_\alpha)_{\alpha>0}$ be a regularizing network and $\alpha^*$ a parameter choice function. For any $\delta > 0$ we define the reconstruction algorithms $\mathbf{G}^\delta, \mathbf{R}^\delta \colon \mathbb{Y} \to \mathbb{X}$ and $\mathbf{\Phi}^\delta \colon \mathbb{X} \to \mathbb{X}$ by

$$\begin{aligned}\mathbf{G}^\delta(z) &:= \mathbf{G}_{\alpha^*(\delta,z)}(z),\\ \mathbf{\Phi}^{\delta,y^\delta}(x) &:= \mathbf{\Phi}_{\alpha^*(\delta,y^\delta)}(x),\\ \mathbf{R}^\delta(z) &:= \mathbf{\Phi}_{\alpha^*(\delta,z)}\mathbf{G}_{\alpha^*(\delta,z)}(z),\end{aligned}$$

for $x \in \mathbb{X}$ and $z \in \mathbb{Y}$.

**Assumption 3.3** (Convergence rate conditions). *Let $\mathbb{X}_0 \subseteq \mathbf{G}_0\mathbf{F}(\mathbb{D})$ satisfy the following for some $r \in (0, 1]$*

*(R1)* $\mathcal{E}(\mathbf{G}^\delta, \delta, \mathbb{X}_0) = \mathcal{O}(\delta^r)$ *as* $\delta \to 0$.

*(R2)* $\sup\{\|\mathbf{G}^\delta(y^\delta) - \mathbf{G}^\delta\mathbf{F}(x)\| \mid x \in \mathbb{X}_0 \wedge y^\delta \in \overline{B_\delta(\mathbf{F}(x))}\} = \mathcal{O}(\delta^r)$.

*(R3)* $\sup\{\|\mathbf{\Phi}^{\delta,y^\delta}(x) - \mathbf{\Phi}_0(x)\| \mid x \in \mathbb{X}_0 \wedge y^\delta \in \overline{B_\delta(\mathbf{F}(x))}\} = \mathcal{O}(\delta^r)$.

The first condition (R1) means that $(\mathbf{G}_{\alpha^*(\delta,y^\delta)})_{\delta>0}$ converges at rate $\delta^r$. Condition (R2) is a stability estimate for $\mathbf{G}_{\alpha^*(\delta,y^\delta)}$. Condition (R3) gives a relation between $\mathbf{\Phi}_{\alpha^*(\delta,y^\delta)}$, applied in noisy cases, and $\mathbf{\Phi}_0$, applied in the noiseless case.

**Theorem 3.4** (Convergence rate for regularizing networks). *Let $\mathbb{M}_0 = \mathbf{\Phi}_0(\mathbb{X}_0)$. Under Assumption 3.3 we have $\mathcal{E}(\mathbf{R}^\delta, \delta, \mathbb{M}_0) = \mathcal{O}(\delta^r)$.*

*Proof.* Let $x \in \mathbb{M}_0$, $\|\mathbf{F}(x) - y^\delta\| \le \delta$, and $z \in \mathbb{X}_0$ s.t. $\mathbf{\Phi}_0(z) = x$. Then

$$\begin{aligned}\|\mathbf{R}^\delta(y^\delta) - x\| &\le \|\mathbf{R}^\delta(y^\delta) - \mathbf{R}^\delta\mathbf{F}(x)\| + \|\mathbf{R}^\delta\mathbf{F}(x) - x\|\\ &\le L\|\mathbf{G}^\delta(y^\delta) - \mathbf{G}^\delta\mathbf{F}(x)\| + \|\mathbf{\Phi}_0\mathbf{G}^\delta\mathbf{F}(z) - \mathbf{\Phi}_0(z)\|\\ &\qquad + \|\mathbf{R}^\delta\mathbf{F}(x) - \mathbf{\Phi}_0\mathbf{G}^\delta\mathbf{F}(z)\|\\ &\le L\|\mathbf{G}^\delta(y^\delta) - \mathbf{G}^\delta\mathbf{F}(x)\| + L\|\mathbf{G}^\delta\mathbf{F}(z) - z\|\\ &\qquad + \|\mathbf{\Phi}^{\delta,\mathbf{F}(x)}\mathbf{G}^\delta\mathbf{F}(x) - \mathbf{\Phi}_0\mathbf{G}^\delta\mathbf{F}(x)\|\\ &\le L\|\mathbf{G}^\delta(y^\delta) - \mathbf{G}^\delta\mathbf{F}(x)\| + L\|\mathbf{G}^\delta\mathbf{F}(z) - z\|\\ &\qquad + \|\mathbf{\Phi}^{\delta,\mathbf{F}(x)}\mathbf{G}^\delta\mathbf{F}(x) - \mathbf{\Phi}^{\delta,\mathbf{F}(x)}\mathbf{G}_0\mathbf{F}(x)\|\\ &\qquad + \|\mathbf{\Phi}^{\delta,\mathbf{F}(x)}\mathbf{G}_0\mathbf{F}(x) - \mathbf{\Phi}_0\mathbf{G}_0\mathbf{F}(x)\|\\ &\qquad + \|\mathbf{\Phi}_0\mathbf{G}_0\mathbf{F}(x) - \mathbf{\Phi}_0\mathbf{G}^\delta\mathbf{F}(x)\|\\ &\le L\|\mathbf{G}^\delta(y^\delta) - \mathbf{G}^\delta\mathbf{F}(x)\| + L\|\mathbf{G}^\delta\mathbf{F}(z) - z\|\\ &\qquad + \|\mathbf{\Phi}^{\delta,\mathbf{F}(x)}\mathbf{G}_0\mathbf{F}(x) - \mathbf{\Phi}_0\mathbf{G}_0\mathbf{F}(x)\| + 2L\|\mathbf{G}^\delta\mathbf{F}(x) - \mathbf{G}_0\mathbf{F}(x)\|.\end{aligned}$$

Each of the above terms are $\mathcal{O}(\delta^r)$: the first term due to the stability estimate (R2), the second term due to (R1), and the third term due to (R3). For the fourth term we use that $\mathbf{G}_0\mathbf{F}(x) = \mathbf{G}_0\mathbf{F}(z) = \mathbf{G}_0\mathbf{F}\mathbf{G}_0\mathbf{F}(w) = \mathbf{G}_0\mathbf{F}(w) = z \in \mathbb{X}_0$ for some $w \in \mathbb{D}$. This implies that the fourth term is $\mathcal{O}(\delta^r)$, due to (R1) again. □

In the following we give an explicit example of a classical regularization method combined with a sequence of regularizing networks, where Assumption 3.3 is satisfied and therefore Theorem 3.4 can be applied.

**Example 3.5** (Regularizing networks combined with Tikhonov regularization). *Given a Gâteaux differentiable forward operator* $\mathbf{F}$ *we consider* $(\mathbf{G}_\alpha)_{\alpha>0}$ *defined by classical Tikhonov regularization* (2.1)*, a data-consistent network* $\mathbf{\Phi}_0 \colon \mathbb{X} \to \mathbb{X}$ *and a sequence of regularizing networks* $(\mathbf{\Phi}_\alpha \colon \mathbb{X} \to \mathbb{X})_{\alpha>0}$ *satisfying (R3) for* $r = 1/2$*.*

**Corollary 3.6.** *If we consider the set* $\mathbb{M}_0 := \mathbf{\Phi}_0(\mathbb{X}_0)$ *where* $\mathbb{X}_0$ *is the source set of classical Tikhonov reguarlization and we assume that the networks* $\mathbf{\Phi}_{\alpha^*(\delta,y^\delta)}$ *converge pointwise to* $\mathbf{\Phi}_0$ *on* $\mathbb{X}$ *at rate* $\mathcal{O}(\delta^{1/2})$ *as* $\delta \to 0$*, then* $\mathcal{E}(\mathbf{R}^\delta, \delta, \mathbb{M}_0) = \mathcal{O}(\delta^{1/2})$

*Proof.* The convergence rate condition (R1) holds according to [8]. Since for $x = \mathbf{\Phi}_0(z) \in \mathbb{M}_0$ we have $\mathbf{F}(x) = \mathbf{F}(z)$ (Definition 2.2) and because of the stability estimate for Tikhonov regularization [8] we have

$$\begin{aligned} &\sup\{\|\mathbf{G}^\delta(y^\delta) - \mathbf{G}^\delta(\mathbf{F}(x))\| \mid x \in \mathbb{M}_0 \wedge y^\delta \in \overline{B_\delta(\mathbf{F}(x))}\} \\ &\qquad = \sup\{\|\mathbf{G}^\delta(y^\delta) - \mathbf{G}^\delta(\mathbf{F}(z))\| \mid z \in \mathbb{X}_0 \wedge y^\delta \in \overline{B_\delta(\mathbf{F}(x))}\} = \mathcal{O}(\delta^{1/2}), \end{aligned} \tag{3.2}$$

which shows (R2). Finally (R3) holds by assumption and therefore the conditions of Theorem 3.4 are satisfied for $r = 1/2$. □

This shows one of the main benefits of the concept of regularizing networks, namely transforming the set $\mathbb{X}_0$ on which the basic regularization converges at a certain rate, to a different data dependent set $\mathbf{\Phi}_0(\mathbb{X}_0)$ with possibly less regularity, while preserving the convergence rate.

# 4 Considered inverse problems

In this section we provide the general mathematical description of inverse problems that are considered in the simulation experiments. We define nonlinear mappings $\mathbf{F}$ and derive formulations of right inverses $\mathbf{G}_0$ and data-consistent networks $\mathbf{\Phi}_0$.

## 4.1 Projection on convex set

We consider the nonlinear inverse problem, where $\mathbf{F} := \mathbf{P}_C \colon \mathbb{D} \to C$ is a metric projection on a closed convex set $C \subseteq \mathbb{D}$, i.e.

$$y = \mathbf{P}_C(x) := \operatorname*{arg\,min}_{\bar{x} \in C} \{\|\bar{x} - x\|\} \,. \tag{4.1}$$

The affine normal cone to $C$ at $x$ is defined as

$$\mathcal{N}_C(x) := \{\tilde{x} \in \mathbb{D} \mid \mathbf{P}_C(\tilde{x}) = x\} \,.$$

It is easily shown that any mapping that maps $x \in C$ to an element in the normal cone $\mathcal{N}_C(x)$ is a right inverse of $\mathbf{P}_C$. In particular, the projection $\mathbf{P}_{\mathcal{N}_C(x)} \colon \mathbb{D} \to \mathbb{D}$,

$$\mathbf{P}_{\mathcal{N}_C(x)}(\hat{x}) := \underset{\tilde{x} \in \mathcal{N}_C(x) \cap \mathbb{D}}{\arg\min} \{\|\tilde{x} - \hat{x}\|\}, \tag{4.2}$$

combined with any function $\mathbf{H}_0 \colon C \to \mathbb{D}$ defines a right inverse $\mathbf{G}_0 \colon C \to \mathbb{D} \colon x \mapsto \mathbf{P}_{\mathcal{N}_C(x)}(\mathbf{H}_0(x))$. This follows because $\forall x \in C \colon \mathbf{P}_C(\mathbf{P}_{\mathcal{N}_C(x)}(\mathbf{H}_0(x))) = x$. We assume, that (4.2) is well defined for all $x \in C$ and $\hat{x} \in \mathbb{D}$. According to Definition 2.2, a data-consistent network $\mathbf{\Phi}_0$ satisfies $\forall z \in \mathbf{G}_0\mathbf{F}(\mathbb{D}) \colon \mathbf{F}\mathbf{\Phi}_0(z) = \mathbf{F}(z)$. We define

$$\mathbf{\Phi}_0(z) := \mathbf{P}_{\mathcal{N}_C(\mathbf{P}_C(z))}(\mathbf{U}(z)), \tag{4.3}$$

so this requirement is satisfied. Here $\mathbf{U} \colon \mathbb{X} \to \mathbb{X}$ is any Lipschitz continuous trained neural network (c.f. definition 2.2). See Figure 4.1 for a visual illustration.

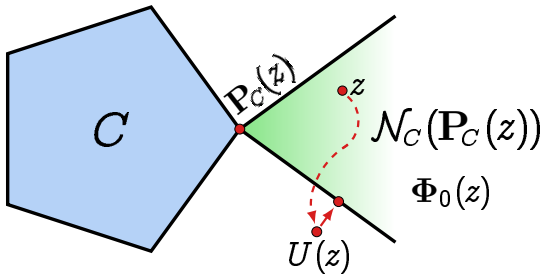



Figure 4.1: Visualization of the data-consistent network for the projection problem. It can be seen that $\mathbf{P}_C(\mathbf{\Phi}_0(z)) = \mathbf{P}_C(z)$, as required by the definition of a data-consistent network.

## 4.2 Composition of mappings

As a second inverse problem, we consider the mapping $\mathbf{F} \colon \mathbb{D} \to \mathbb{Y}$ that is defined as a composition of two (possibly nonlinear) mappings:

$$\mathbf{F}(x) := \mathbf{F}_2(\mathbf{F}_1(x)), \qquad \text{where } \mathbf{F}_1 \colon \mathbb{D} \to \mathbb{E} \text{ and } \mathbf{F}_2 \colon \mathbb{E} \to \mathbb{Y},$$

where $\mathbb{E}$ is a Banach space. Furthermore we impose the restriction that the mapping $\mathbf{F}_2$ provides a data-consistent network that can be written as a projection

$$\mathbf{\Phi}_0^{(2)}(z) := \mathbf{P}_{S \cap \mathrm{Im}(\mathbf{F}_1)}(\mathbf{U}_2(z)),$$

where $S \subseteq \mathbb{E}$. In particular, this is true for the mapping described in Section 4.1, when the projection on a normal cone also maps into the range of the operator $\mathbf{F}_1$. The projection onto the intersection in the data-consistent network can be implemented by an alternating projection algorithm 5.2. If we assume that $S \cap \mathrm{Im}(\mathbf{F}_1) \neq \emptyset$, then we define the data-consistent network $\mathbf{\Phi}_0$ for the full mapping as

$$\mathbf{\Phi}_0(z) = \mathbf{\Phi}_0^{(1)}\mathbf{G}_0^{(1)}\mathbf{\Phi}_0^{(2)}\mathbf{G}_0^{(2)}\mathbf{F}_2\mathbf{F}_1(z) \tag{4.4}$$

where $\mathbf{\Phi}_0^{(i)}$ and $\mathbf{G}_0^{(i)}$ are defined as the data-consistent network for $\mathbf{F}_i$ and the right inverse for $\mathbf{F}_i$ respectively. We check the data-consistent property (Definition 2.2) by

$$\begin{aligned}\mathbf{F}\mathbf{\Phi}_0(z) &= \mathbf{F}_2\mathbf{F}_1\mathbf{\Phi}_0^{(1)}\mathbf{G}_0^{(1)}\mathbf{\Phi}_0^{(2)}\mathbf{G}_0^{(2)}\mathbf{F}_2\mathbf{F}_1(z)\\ &= \mathbf{F}_2\mathbf{F}_1\mathbf{G}_0^{(1)}\mathbf{\Phi}_0^{(2)}\mathbf{G}_0^{(2)}\mathbf{F}_2\mathbf{F}_1(z)\\ &= \mathbf{F}_2\mathbf{\Phi}_0^{(2)}\mathbf{G}_0^{(2)}\mathbf{F}_2\mathbf{F}_1(z)\\ &= \mathbf{F}_2\mathbf{G}_0^{(2)}\mathbf{F}_2\mathbf{F}_1(z)\\ &= \mathbf{F}_2\mathbf{F}_1(z)\\ &= \mathbf{F}(z),\end{aligned}$$

where we used in order: the data-consistent property of $\mathbf{\Phi}_0^{(1)}$; the definition of a right-inverse $\mathbf{G}_0^{(1)}$ in combination with the projection on the range of $\mathbf{F}_1$; the data-consistent property of $\mathbf{\Phi}_0^{(2)}$; and the definition of the right-inverse $\mathbf{G}_0^{(2)}$.

We note that this is not the only data-consistent network possible for such an inverse problem: one could also design a network that only makes use of either $\mathbf{\Phi}_0^{(1)}$ or $\mathbf{\Phi}_0^{(2)}$. However, (4.4) provides a network that is intuitively clear: an initial solution $z$ is obtained by a classical regularization method, after which first a better 'guess' is made by applying a neural network on $\mathbb{E}$, followed by a neural network that makes a better guess on the reconstruction space $\mathbb{D}$, while keeping the solutions data-consistent throughout.

# 5 Simulation experiments

In this section, we first specify two examples of the inverse problems described in section 4. After that, we explain all neural networks that will be compared for these examples, among which are the derived data-consistent networks. Finally, we provide the implementation details for all simulation experiments.

## 5.1 Spatially dependent saturation of multivariate Gaussians

In the first simulation experiment, we consider the inverse problem of recovering images of multivariate Gaussians which have been nonhomogeneously saturated. Formally we define the domain $\mathbb{D} = \mathbb{X} := \ell^2(\Omega)$ and we define the saturation mapping as a projection on a convex set, as described in section 4.1. This means $\mathbf{F}(x) := \mathbf{P}_C(x)$, where

$$C := \left\{ x \in \ell^2(\Omega) \mid x(r) \leq M(r), \quad \forall r \in \Omega, \right\}$$

where $M(r) \geq 0$ is the saturation value at location $r$. The corresponding right inverse for $y \in C$ is defined as $\mathbf{G}_0(y) = \mathbf{P}_{\mathcal{N}_C(y)}(y) = \mathrm{Id}(y)$. The projection defined by (4.1) is explicitly given by

$$[\mathbf{P}_C(x)](r) := \min\{x(r), M(r)\}.$$

Since $\mathcal{N}_C(\mathbf{P}_C(z)) = \{x \mid \mathbf{P}_C(x) = \mathbf{P}_C(z)\}$, (4.3) can be written pointwise as

$$[\mathbf{\Phi}_0(z)](r) = \begin{cases} z(r) & \text{for } z(r) < M(r), \\ \max\{[\mathbf{U}(z)](r), M(r)\} & \text{for } z(r) \geq M(r). \end{cases} \tag{5.1}$$

We consider the square domain $\Omega := [-1, 1] \times [-1, 1]$. The spatially dependent saturation function is defined

$$M(r) := \begin{cases} 0.6 & \text{if } \|r\| \leq \frac{1}{2}, \\ 0 & \text{if } \|r\| > \frac{1}{2}. \end{cases} \tag{5.2}$$

Each image in the training or test set contains one centered multivariate Gaussian with diagonal covariance matrix, having standard deviations $(\sigma_1, \sigma_2)$ independently randomly chosen in the interval $[0.24, 0.32]$. All images in the training and test set are scaled to obtain maximum values randomly chosen in the interval $[0.75, 1]$. Opposed to standard networks, one of the benefits of using a data-consistent network is that it is more robust to changes in the data. For this reason, a modified test set has been created, where the Gaussians have standard deviations in the interval $[0.12, 0.20]$ with maximum intensities in the interval $[0.6, 0.8]$. For the numerical implementation we consider the discretized domain $\bar{\Omega} := \mathbb{R}^{128 \times 128}$ as discretization of $\Omega$.

The data-consistent network $\mathbf{\Phi}_0(z)$, as described in (5.1), is compared with the neural network $\mathbf{U}(z)$ without data-consistency. We compare reconstruction quality for both the regular test set and modified test set. A description of the neural network architecture and training details are provided in section 5.3.

## 5.2 Saturation of Radon transformed human chest images

In the second simulation experiment, we consider the inverse problem of reconstructing images of the human chest, from saturated and highly limited angle Radon measurements. We consider the composition of two mappings as described in section 4.2, where $\mathbf{F}_1$ is a linear mapping that acts as the discrete Radon transform and $\mathbf{F}_2$ is a nonlinear saturation mapping that saturates the Radon signals at a constant value $M$. For conciseness, we define our domains and mappings in the discretized setup: $\mathbb{D} := \mathbb{R}^{n_x \times n_x}$, $\mathbb{Y} = \mathbb{R}^{n_\alpha \times \frac{3}{2} n_x}$, where $n_x = 192$ is the number of pixels in each direction of the image and $n_\alpha = 8$ is the number of angles in the Radon transform, uniformly sampled in the interval $[0, \pi]$.

We now define all elements that are needed to obtain the data-consistent network (4.4), which we repeat here for completeness:

$$\mathbf{\Phi}_0(z) = \mathbf{\Phi}_0^{(1)} \mathbf{G}_0^{(1)} \mathbf{\Phi}_0^{(2)} \mathbf{G}_0^{(2)} \mathbf{F}_2 \mathbf{F}_1(z). \tag{5.3}$$

A matrix representation $\mathbf{F}_1 \in \mathbb{R}^{n_\alpha \cdot \frac{3}{2} n_x \times n_x^2}$ of the Radon transform is obtained as described in [17]. Its right inverse is taken as the pseudo-inverse of $\mathbf{F}_1$, i.e. $\mathbf{G}_0^{(1)} := \mathbf{F}_1^\dagger$. Since the mapping is linear, the corresponding data-consistent network is a null-space network [26], i.e. $\mathbf{\Phi}_0^{(1)}(z) = z + \mathbf{P}_{\ker(\mathbf{F}_1)} \mathbf{U}(z)$, where $\mathbf{P}_{\ker(\mathbf{F}_1)} = \mathrm{Id} - \mathbf{F}_1^\dagger \mathbf{F}_1$. The saturation mapping $\mathbf{F}_2 = \mathbf{P}_C$, its right inverse $\mathbf{G}_0^{(2)}$ and the data-consistent network $\mathbf{\Phi}_0^{(2)}$

are chosen as described in section 5.1, this time with constant saturation level $M = 8$. Finally, for this particular choice of $\mathbf{F}_2$, the projection on the intersection of convex sets reads $\mathbf{P}_{\mathcal{N}_C(\mathbf{P}_C(z))\cap \mathrm{Im}(\mathbf{F}_1)}$, which can be achieved by the 'projection onto convex sets' (POCS) algorithm [5]: by alternatingly performing $\mathbf{P}_{\mathcal{N}_C(\mathbf{P}_C(z))}$ and $\mathbf{P}_{\mathrm{Im}(\mathbf{F}_1)} = \mathbf{F}_1\mathbf{F}_1^\dagger$, the resulting iteration converges linearly to a point on the intersection.

Training and test images were obtained from the LoDoPaB-CT dataset [16], which on its turn makes use of the LIDC/IDRI dataset [3]. In our work, we only make use of the high quality CT reconstructions in the LoDoPaB-CT dataset that we use as 'ground truth' for our setup. The images are scaled to $192 \times 192$ pixels, after which the mappings $\mathbf{F}_1$ and $\mathbf{F}_2$ are applied to obtain simulated sinograms. After that, pseudo-inverses $\mathbf{G}_0^{(2)}$ and $\mathbf{G}_0^{(1)}$ are applied to obtain the input for our data-consistent network. For this simulation experiment we have also created a modified test set to investigate how the trained networks generalize towards slightly modified data. For conciseness, the procedure to get from the regular test data to the modified test data is not explained in full detail. In short, the test set consists of images in the range of $\mathbf{F}_1^\dagger$ that produce sinograms that have a maximum below or around the saturation level. This means that the saturation mapping $\mathbf{F}_2$ will not have a big effect on the unsaturated sinograms. Images in the modified test set look very similar to the ones in the regular test set, but they often show a small gradient at locations where the regular images show a piecewise constant structure. Some samples from the modified test set are shown in Figure 6.4 and appendix B.2.

The data-consistent network $\mathbf{\Phi}_0(z)$, as described in (4.4), is compared with two other networks: the first one applies a single neural network to the pseudo-inverse reconstruction; the second one first applies a neural network in the sinogram domain, then applies the pseudo-inverse of the Radon-transform, followed by a neural network in the image domain. For completeness, we summarize the three networks below:

- One neural network: $\mathbf{N}_1(z) = \mathbf{U}_1(z)$.
- Two neural networks: $\mathbf{N}_2(z) = \mathbf{U}_1\mathbf{G}_0^{(1)}\mathbf{U}_2\mathbf{F}_1(z)$.
- Data-consistent network: $\mathbf{\Phi}_0(z) = \mathbf{\Phi}_0^{(1)}\mathbf{G}_0^{(1)}\mathbf{\Phi}_0^{(2)}\mathbf{G}_0^{(2)}\mathbf{F}_2\mathbf{F}_1(z)$.

We emphasize that the data-consistent networks, in terms of architecture, make use of the same neural networks $\mathbf{U}_1$ and $\mathbf{U}_2$ as the first two networks, i.e. $\mathbf{\Phi}_0^{(1)}(z) = z + \mathbf{P}_{\ker(\mathbf{F}_1)}\mathbf{U}_1(z)$ and $\mathbf{\Phi}_0^{(2)}$ makes use of $\mathbf{U}_2$ as defined in (5.1). A more detailed description of the neural network architecture and training details are provided in section 5.3.

Ideally, the data-consistent network is trained 'end-to-end', meaning that both $\mathbf{U}_1$ which is used in $\mathbf{\Phi}_0^{(1)}$, and $\mathbf{U}_2$ which is used in $\mathbf{\Phi}_0^{(2)}$, are trained at the same time. However, the application of the POCS algorithm is computationally intensive, since it requires iterative application of the mappings $\mathbf{F}_1$ and $\mathbf{F}_1^\dagger$. For this reason, we have chosen to first train $\mathbf{\Phi}_0^{(2)}$ to output the unsaturated sinogram, then perform the POCS algorithm and finally train $\mathbf{\Phi}_0^{(1)}$ to output the reconstructed image.

### 5.3 Neural network architecture and training details

In this work, the popular U-Net [24, 11] is implemented as a neural network. By using standard nonlinearities such as rectified linear units (ReLUs) and convolutions, the network is Lipschitz continuous, which is a requirement as described in Definition 2.2. The Lipschitz constant can be controlled by weight regularization, such as adding an $L^2$-loss on the weights in the loss function.

| | Exp. 1 ($\mathbf{U}/\mathbf{\Phi}_0$): image domain | Exp. 2 ($\mathbf{U}_1/\mathbf{\Phi}_0^{(1)}$): image domain | Exp. 2 ($\mathbf{U}_2/\mathbf{\Phi}_0^{(2)}$): sinogram domain |
|---|---|---|---|
| #training samples | 1024 | 35584 | 35584 |
| #validation samples | 256 | 3522 | 3522 |
| #test samples | 1024 | 3553 | 3553 |
| depth | 4 | 4 | 4 |
| width | 2 | 2 | 2 |
| #channels in top layer | 8 | 16 | 16 |
| convolution size | $3 \times 3$ | $3 \times 3$ | $3 \times 3$ |
| nonlinearity | ReLU | ReLU | ReLU |
| start learning rate | $10^{-3}$ | $10^{-3}$ | $10^{-3}$ |
| final learning rate | $10^{-4}$ | $2 \cdot 10^{-4}$ | $2 \cdot 10^{-4}$ |
| batch size | 64 | 32 | 32 |
| #epochs | 1000 | 25 | 25 |

Table 1: U-Net parameter details for all simulation experiments.

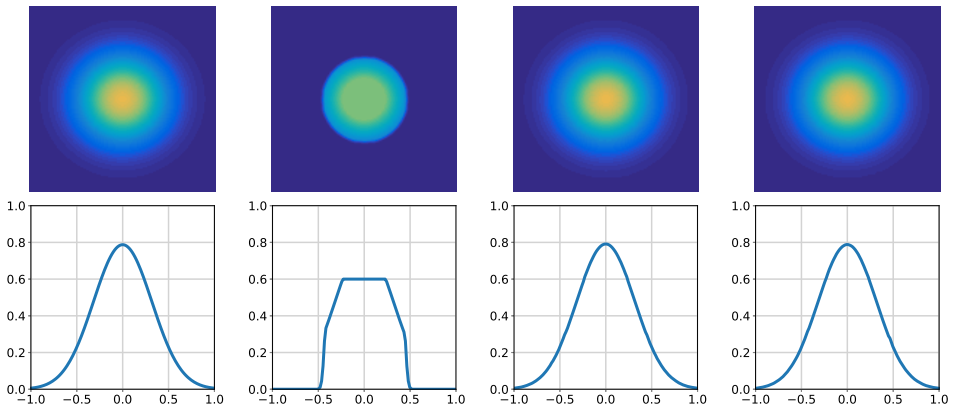
The U-Net was implemented as described in [11], although for each experiment some parameters were chosen slightly different to obtain optimal results. For all experiments, the network has a 'depth' of four, meaning four times max-pooling and upsampling. The U-Nets in the image domain perform the regular max-pooling and upsampling in two directions, while the U-Net in the sinogram domain performs these only in one direction, leaving the number of angles constant at 8. This is done because the neighboring angles in the sinogram show very little resemblance to each other and there are only 8. The 'width', or the amount of convolutions at every depth is chosen to be two. As in [11] the number of convolution channels doubles after each max-pooling; the number of channels at the start is stated in Table 1, since this was chosen differently for every simulation experiment. In all experiments a residual structure that is also apparent in [11] is used. The U-Net uses $3 \times 3$ convolutions with biases and applied a ReLU-activation after each convolution, except the last one. In all experiments, an $L^2$-loss function on the difference between output and ground truth is minimized. For optimization, the ADAM optimizer with exponentially decaying learning rate is chosen. The learning rates and batch sizes are stated in Table 1.

# 6 Numerical results

In this section, the reconstruction quality of data-consistent networks is compared with the U-Nets that are not data-consistent. Besides visual comparison, the quality will be compared by means of peak signal-to-noise ratio (PSNR) and structural similarity (SSIM). For both experiments this will be done for the regular test set as well as the modified test set in order to investigate the generalization capacity of the networks.

## 6.1 Spatially dependent saturation of multivariate Gaussians

Multivariate Gaussians are saturated with a spatially dependent saturation function (5.2), as described in section 5.1. Results for one sample from the regular test set are shown in Figure 6.1. Here it can be seen that both U-Net and the data-consistent network provide a very accurate reconstruction. This is also reflected in the PSNR and SSIM values shown in Table 2. The pseudo-inverse reconstruction, which in this case is just the measurement, is not a good one, since a lot of information is lost by applying the saturation mapping.

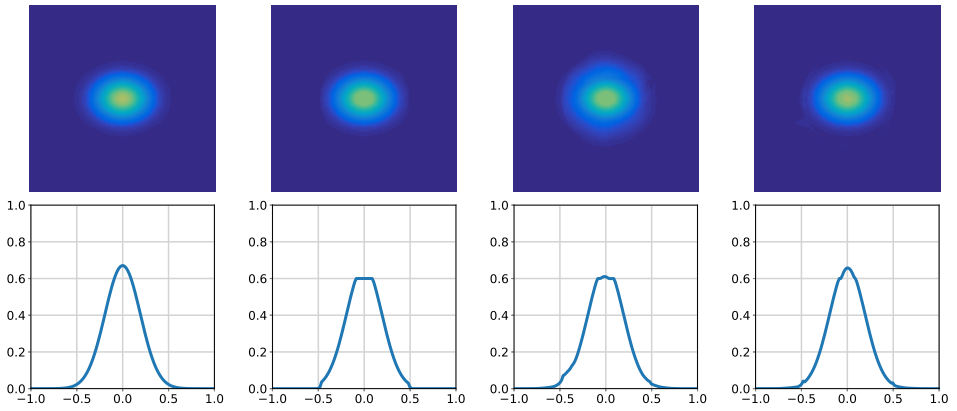


(a) Ground truth (b) Saturated (c) U-Net (d) Data-consistent

Figure 6.1: Reconstructions of a sample from the regular test set. In the bottom the horizontal central slice is shown. Both U-Net and data-consistent network provide an almost perfect reconstruction.

In Figure 6.2 the results for one sample are shown for the modified test set, which contains smaller Gaussians with a slightly lower intensity. Both U-Net and the data-consistent network are not perfectly able to fill in the missing information in the small Gaussians. This can be expected, since Gaussians of this size were not included in the training set. However, the data-consistent network does not deform the Gaussian at the location where it is not saturated, while U-Net does this slightly; for instance around $-0.5$ in the slice plot. This behaviour is also reflected in the PSNR and SSIM values in Table 2. Interestingly, the pseudo-inverse behaves very well if we just look at the values in the table, because the saturation mapping did not destroy a lot of the information in the Gaussian. Visual results of three more samples in the modified test set are shown in appendix A. It can be seen that U-Net tends to widen the Gaussians, since it was trained on Gaussians in the training set that were wider. Although the modified test

set shows a very specific modification, it illustrates that a data-consistent network is beneficial over using an arbitrary neural network: by making use of the information that we have from the mapping $\mathbf{F}$, we obtain generalization capacity.



(a) Ground truth (b) Saturated (c) U-Net (d) Data-consistent

Figure 6.2: Reconstructions of a sample from the modified test set. In the bottom the horizontal central slice is shown. Data consistency makes sure that intensity is only changed above the saturation level.

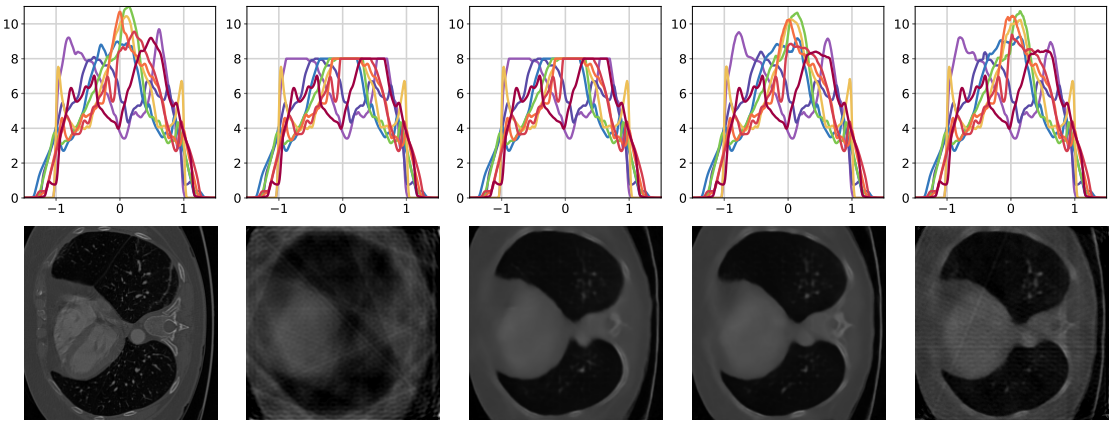
| | PSNR | | | SSIM | | |
|---|---|---|---|---|---|---|
| | Pseudo-inverse | U-Net | Data-consistent | Pseudo-inverse | U-Net | Data-consistent |
| Regular set | $24.2 \pm 2.2$ | $60.6 \pm 2.1$ | $66.7 \pm 1.6$ | $0.56 \pm 0.08$ | $1.00 \pm 0.00$ | $1.00 \pm 0.00$ |
| Modified set | $48.0 \pm 7.8$ | $36.9 \pm 2.9$ | $48.0 \pm 4.4$ | $0.99 \pm 0.01$ | $0.92 \pm 0.03$ | $0.97 \pm 0.01$ |

Table 2: Comparison of PSNR and SSIM for all reconstruction methods.

## 6.2 Saturation of Radon transformed human chest images

For one selected sample in the regular test set and one in the modified test set, all reconstructions are shown in Figures 6.3 and 6.4. These specific samples were selected because their PSNR values for the U-Nets and the data-consistent network show a similar relation to each other as the average PSNR values of the whole test set (c.f. Table 3).

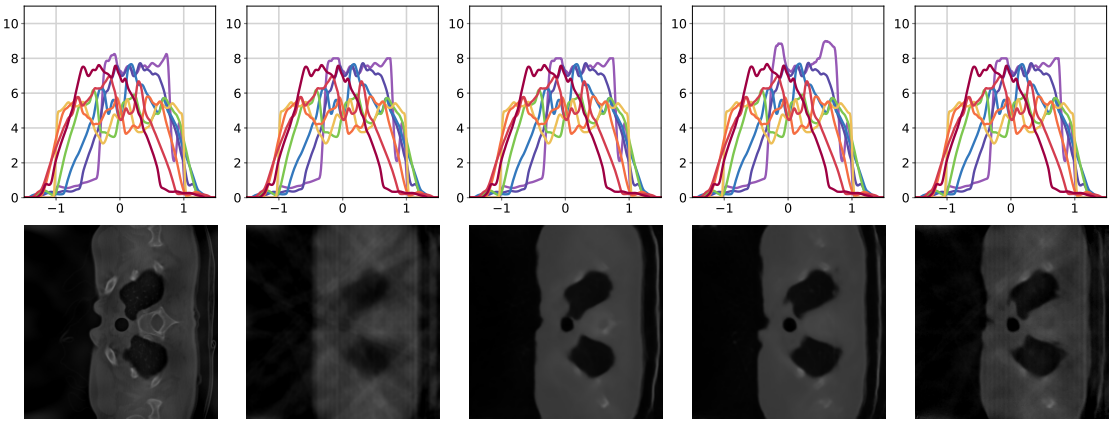
In the top of Figure 6.3, the inputs of the right inverse $\mathbf{G}_0^{(1)}$ are shown. This corresponds to the saturated sinograms in case no or only one neural network is trained, and this corresponds to the output of the neural network in the sinogram domain in case two neural networks are trained. The sinogram signals are plotted in a different color for each angle. The data-consistent network does not change the values of the sinogram that are below the saturation level ($M = 8$). Interestingly, the U-Net in the sinogram domain has learned not to do this as well to a large extent: some values just below the saturation level are changed (for instance the purple line around $-0.5$), but values much lower are not changed at all. In the bottom of Figure 6.3, all reconstructions



(a) Ground truth (b) Pseudo-inverse PSNR = 22.80 (c) One U-Net PSNR = 27.31 (d) Two U-Nets PSNR = 27.70 (e) Data-consistent PSNR = 27.01

Figure 6.3: Reconstructions of a typical sample from the regular test set. Top: reconstructed sinograms with all 8 angles in different colors. Bottom: reconstructed images.

for this sample are shown. It can be seen that although PSNR values are similar, there is a clear visual difference: the data-consistent network shows more artefacts that are typical for limited-angle Radon reconstructions, while the standard U-Nets provide over-smoothed reconstructions. In other words, the data-consistent reconstruction does not smooth out potential details, while the other networks do. However, all networks perform rather similarly, since there are no clear structures that can be seen in one of the reconstructions and not in the others. This is also reflected in Table 3. Moreover, since only 8 angles were used in the Radon transform, only larger structures can be reconstructed. Some extreme samples, for which the PSNR value of the data-consistent solution is high, similar or low compared to the U-Net solutions, are shown in Appendix B.1.



(a) Ground truth (b) Pseudo-inverse PSNR = 30.02 (c) One U-Net PSNR = 28.29 (d) Two U-Nets PSNR = 29.08 (e) Data-consistent PSNR = 30.83

Figure 6.4: Reconstructions of a typical sample from the modified test set. Top: reconstructed sinograms with all 8 angles in different colors. Bottom: reconstructed images.

In the top of Figure 6.4, again all sinogram reconstructions are plotted, now for the

modified test set. As the modified sinograms contain many values just below or around the saturation level ($M = 8$), this is challenging for the regular U-Net. Indeed it can be seen that the data-consistent network does not change the saturated sinogram much, while the U-Net on the sinogram increases the purple line well beyond the saturation level. In the bottom of Figure 6.3, it can be seen that the ground truth possesses some 'smooth' regions, especially in the background on the left and right of the two dark inclusions. While both U-Nets create piecewise constant reconstructions that completely remove this gradient, the data-consistent network keeps these smooth regions: it generalizes better to test images that are not found in the training set by making use of the information in the operator. More extreme samples, where the PSNR value of the data-consistent solution is high, similar or low compared to the U-Net solutions, are shown in Appendix B.2. The same effect on smooth regions can be seen in these visualizations. In Table 3, it can be seen that for all networks the quality drops when the regular test set is replaced by the modified test set; however, this drop is only very small for the data-consistent network and is much bigger for the regular U-Nets. Note that the pseudo-inverse gives an increased PSNR for the modified test set, because the modified images were constructed to lie in the range of $\mathbf{F}$.

| | PSNR | | | |
|---|---|---|---|---|
| | Pseudo-inverse | One U-Net | Two U-Nets | Data-consistent |
| Regular set | $23.1 \pm 2.3$ | $30.5 \pm 1.5$ | $31.0 \pm 1.5$ | $30.1 \pm 1.9$ |
| Modified set | $29.1 \pm 1.6$ | $27.5 \pm 1.7$ | $28.3 \pm 1.4$ | $29.9 \pm 1.2$ |
| | SSIM | | | |
| | Pseudo-inverse | One U-Net | Two U-Nets | Data-consistent |
| Regular set | $0.50 \pm 0.07$ | $0.82 \pm 0.04$ | $0.83 \pm 0.04$ | $0.74 \pm 0.07$ |
| Modified set | $0.71 \pm 0.07$ | $0.74 \pm 0.05$ | $0.73 \pm 0.05$ | $0.75 \pm 0.05$ |
| | Data-fidelity | | | |
| | Pseudo-inverse | One U-Net | Two U-Nets | Data-consistent |
| Regular set | $6.1 \pm 3.3$ | $4.8 \pm 1.5$ | $3.9 \pm 1.1$ | $0.9 \pm 0.4$ |
| Modified set | $0.4 \pm 0.2$ | $11.9 \pm 5.4$ | $8.5 \pm 2.8$ | $0.6 \pm 0.2$ |

Table 3: Comparison of PSNR, SSIM and data-fidelity for all reconstruction methods.

Finally in Table 3 we check the data-fidelity of the solutions from all networks by computing $\|\mathbf{F}(\tilde{x}) - \mathbf{F}(x)\|$, where $\tilde{x}$ is the solution of the respective reconstruction method and $x$ is the ground truth. Ideally, the data-fidelity should be zero for the pseudo-inverse and the data-consistent network. It can be seen that indeed the data-fidelity is much lower for these solutions than for the U-Nets, although not completely zero. This is most probably due to numerical issues (especially for the pseudo-inverse of the regular set) and due to the fact that we needed to save and load images from disc while training because of the size of the data set.

# 7 Conclusion

In this paper we introduced data consistent networks for nonlinear inverse problems. We presented a convergent regularization method by combining deep neural networks that converge to a data-consistent network with classical regularization methods. With the proposed data-driven regularization methods we are able to preserve convergence rates of classical methods over a transformed source set, which is adapted to some data set. This yields improved reconstructions for elements close to the training set, but at the same time data-consistent networks make use of the information from the forward mapping $\mathbf{F}$, which provides increased generalization capacity. This is particularly useful when the physics process is understood, but exact knowledge on real data is not available or when it is not possible to create a training set that is similar to the real data. We showed that on a test set similar to the training set, our approach shows reconstruction results comparable to a classical post-processing network, whereas for instances not represented in the training set, the loss of performance is much less present. This demonstrates the generalization ability of our approach.

# References


[1] J. Adler and O. Öktem. Solving ill-posed inverse problems using iterative deep neural networks. *Inverse Probl.*, 33(12):124007, 2017.

[2] S. Antholzer, M. Haltmeier, and J. Schwab. Deep learning for photoacoustic tomography from sparse data. *Inverse problems in science and engineering*, 27(7):987–1005, 2019.

[3] S. G. Armato III, G. McLennan, L. Bidaut, M. F. McNitt-Gray, C. R. Meyer, A. P. Reeves, B. Zhao, D. R. Aberle, C. I. Henschke, E. A. Hoffman, et al. The lung image database consortium (lidc) and image database resource initiative (idri): a completed reference database of lung nodules on ct scans. *Medical physics*, 38(2):915–931, 2011.

[4] A. B. Bakushinsky and M. Y. Kokurin. *Iterative Methods for Approximate Solution of Inverse Problems*, volume 577 of *Mathematics and Its Applications*. Springer, Dordrecht, 2004.

[5] H. H. Bauschke and J. M. Borwein. On projection algorithms for solving convex feasibility problems. *SIAM review*, 38(3):367–426, 1996.

[6] Y. E. Boink and C. Brune. Learned svd: solving inverse problems via hybrid autoencoding. *arXiv:1912.10840*, 2019.

[7] Y. E. Boink, S. Manohar, and C. Brune. A partially learned algorithm for joint photoacoustic reconstruction and segmentation. *IEEE Trans. Med. Imag.*, 2019.

[8] H. W. Engl, M. Hanke, and A. Neubauer. *Regularization of inverse problems*, volume 375. Kluwer Academic Publishers Group, Dordrecht, 1996.

[9] S. J. Hamilton and A. Hauptmann. Deep D-bar: Real time electrical impedance tomography imaging with deep neural networks. *IEEE Trans. Med. Imag.*, pages 2367–2377, 2018.

[10] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[11] K. H. Jin, M. T. McCann, E. Froustey, and M. Unser. Deep convolutional neural network for inverse problems in imaging. *IEEE Trans. Image Process.*, 26(9):4509–4522, 2017.

[12] B. Kaltenbacher, A. Neubauer, and O. Scherzer. *Iterative regularization methods for nonlinear ill-posed problems*, volume 6 of *Radon Series on Computational and Applied Mathematics*. Walter de Gruyter GmbH & Co. KG, Berlin, 2008.

[13] E. Kobler, T. Klatzer, K. Hammernik, and T. Pock. Variational networks: connecting variational methods and deep learning. In *German Conference on Pattern Recognition*, pages 281–293. Springer, 2017.

[14] Andreas Kofler, Markus Haltmeier, Christoph Kolbitsch, Marc Kachelrieß, and Marc Dewey. A u-nets cascade for sparse view computed tomography. In *International Workshop on Machine Learning for Medical Image Reconstruction*, pages 91–99. Springer, 2018.

[15] D. Lee, J. Yoo, and J. C. Ye. Deep residual learning for compressed sensing MRI. In *Biomedical Imaging (ISBI 2017), IEEE 14th International Symposium on*, pages 15–18. IEEE, 2017.

[16] J. Leuschner, M. Schmidt, D. O. Baguer, and P. Maaß. The lodopab-ct dataset: A benchmark dataset for low-dose ct reconstruction methods. *arXiv:1910.01113*, 2019.

[17] R. M. Lewitt. Multidimensional digital image representations using generalized kaiser–bessel window functions. *JOSA A*, 7(10):1834–1846, 1990.

[18] H. Li, J. Schwab, S. Antholzer, and M. Haltmeier. Nett: Solving inverse problems with deep neural networks. *Inverse Problems, online first*, 2020.

[19] S. Lunz, O. Öktem, and C.-B. Schönlieb. Adversarial regularizers in inverse problems. In *Advances in Neural Information Processing Systems*, pages 8507–8516, 2018.

[20] M. Mardani, E. Gong, J. Y. Cheng, S. Vasanawala, G. Zaharchuk, M. Alley, N. Thakur, W. Han, S.and Dally, J. M. Pauly, et al. Deep generative adversarial networks for compressed sensing automates MRI. *arXiv:1706.00051*, 2017.

[21] V. A. Morozov. *Methods for Solving Incorrectly Posed Problems*. Springer Verlag, New York, Berlin, Heidelberg, 1984.

[22] JH Rick Chang, C.-L. Li, B. Poczos, BVK Vijaya, K., and A. C. Sankaranarayanan. One network to solve them all–solving linear inverse problems using deep projection

models. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5888–5897, 2017.

[23] Y. Rivenson, H. Göröcs, Z.and Günaydin, Y. Zhang, H. Wang, and A. Ozcan. Deep learning microscopy. *Optica*, 4(11):1437–1443, 2017.

[24] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 234–241, 2015.

[25] O. Scherzer, M. Grasmair, H. Grossauer, M. Haltmeier, and F. Lenzen. *Variational methods in imaging*, volume 167 of *Applied Mathematical Sciences*. Springer, New York, 2009.

[26] J. Schwab, S. Antholzer, and M. Haltmeier. Deep null space learning for inverse problems: convergence analysis and rates. *Inverse Problems*, 35(2):025008, 2019.

[27] J. Schwab, S. Antholzer, and M. Haltmeier. Big in japan: Regularizing networks for solving inverse problems. *Journal of Mathematical Imaging and Vision, online first*, 2020.

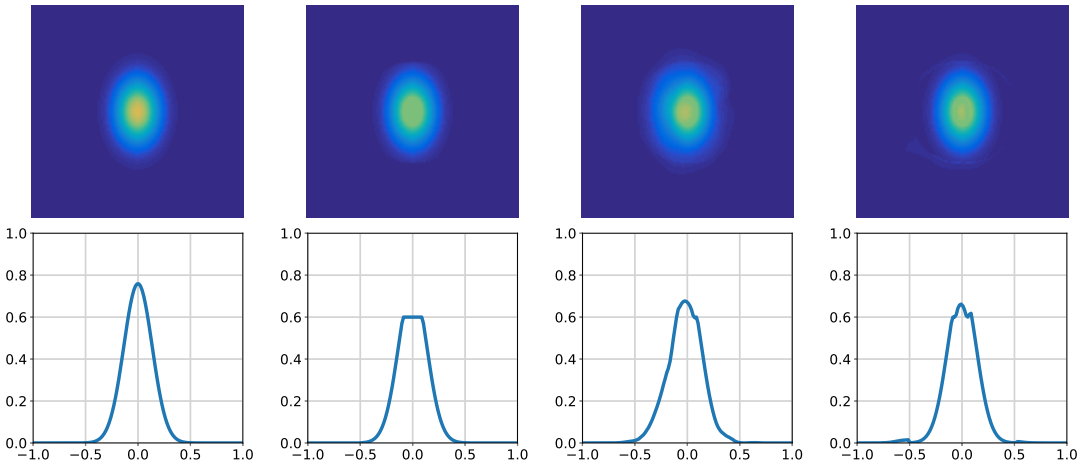
[28] Jian Sun, Huibin Li, Zongben Xu, et al. Deep admm-net for compressive sensing mri. In *Advances in neural information processing systems*, pages 10–18, 2016.

[29] A. N. Tikhonov and V. Y. Arsenin. *Solutions of Ill-Posed Problems*. John Wiley & Sons, Washington, D.C., 1977.

[30] B. Zhu, J. Z. Liu, S. F. Cauley, B. R. Rosen, and M. S. Rosen. Image reconstruction by domain-transform manifold learning. *Nature*, 555(7697):487–492, 2018.

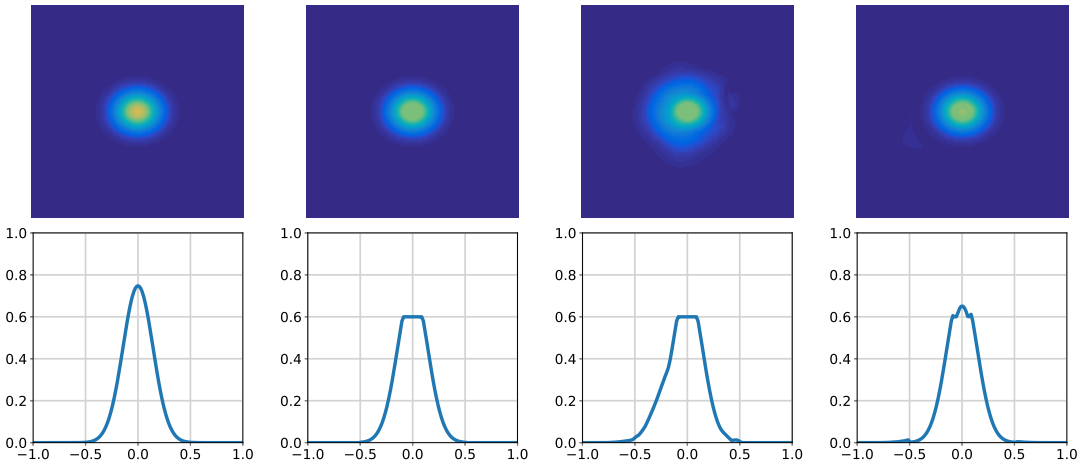
# A Additional results for saturated Gaussians

In this appendix, three more samples from the modified test set of the saturated Gaussians are shown. It can be seen in Figures A.1 and A.2 that U-Net tends to slightly widen the Gaussian in some cases, since this was necessary for the wider Gaussians in the training set. Moreover it can be seen in Figures A.2 and A.3 that both U-Net and the data-consistent network sometimes fail to restore the top of the Gaussian adequately: the training on wider Gaussians is not directly generalised for smaller Gaussians.



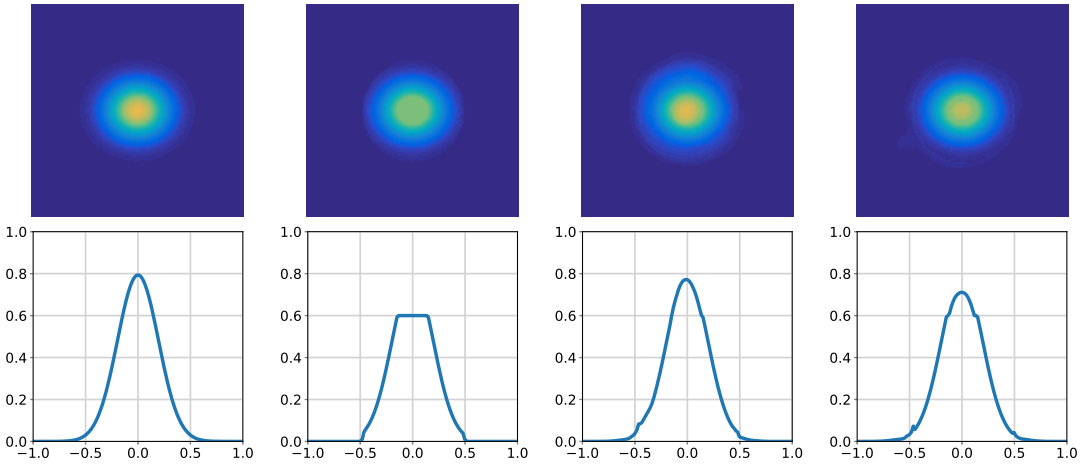
(a) Ground truth (b) Saturated (c) U-Net (d) Data-consistent

Figure A.1



(a) Ground truth (b) Saturated (c) U-Net (d) Data-consistent

Figure A.2



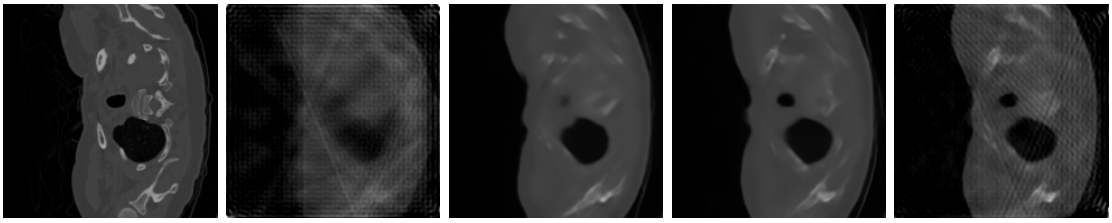

(a) Ground truth (b) Saturated (c) U-Net (d) Data-consistent

Figure A.3

# B Additional results for saturated Radon transform

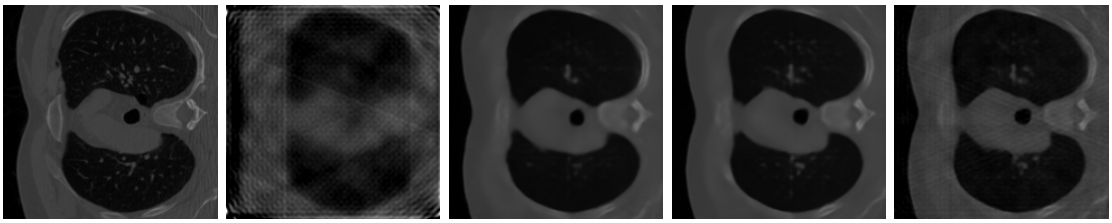
In this appendix, three additional samples from the regular and the modified test set of human chest images are shown. The samples have been selected based on their PSNR values: we show the samples in the test set for which the data-consistent network yields the highest relative PSNR value (Figures B.1 and B.4), a similar PSNR value (Figures B.2 and B.5), and the lowest relative PSNR value (Figures B.3 and B.6) when compared to the U-Nets.

## B.1 Regular test set



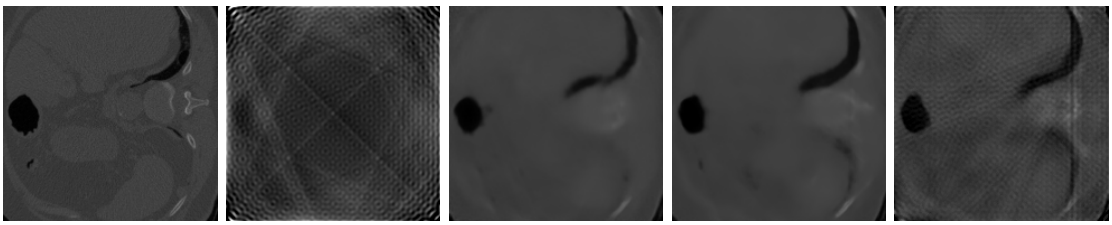
(a) Ground truth (b) Pseudo-inverse PSNR = 29.01 (c) One U-Net PSNR = 31.50 (d) Two U-Nets PSNR = 32.29 (e) Data-consistent PSNR = 34.10

Figure B.1: Sample for which the data-consistent PSNR value is relatively high compared to the U-Net PSNR values.



(a) Ground truth (b) Pseudo-inverse PSNR = 25.33 (c) One U-Net PSNR = 30.09 (d) Two U-Nets PSNR = 29.94 (e) Data-consistent PSNR = 30.09

Figure B.2: Sample for which the data-consistent PSNR value is approximately the same as the U-Net PSNR values.

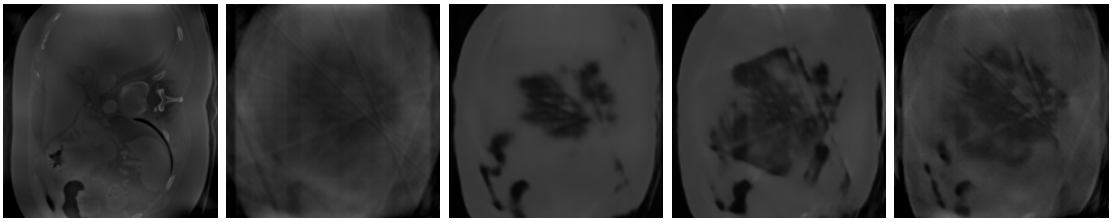


(a) Ground truth (b) Pseudo-inverse PSNR = 22.87 (c) One U-Net PSNR = 29.78 (d) Two U-Nets PSNR = 28.36 (e) Data-consistent PSNR = 24.52

Figure B.3: Sample for which the data-invariant PSNR value is relatively low compared to the U-Net PSNR values.

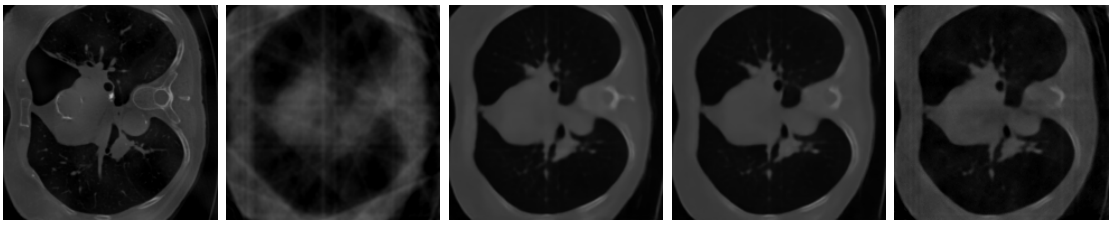
## B.2 Modified test set

The ground truth images in the modified test set contain more regions which are non-constant, as opposed to the regular test set, which consists of piecewise constant images. The data-consistent network is better able to deal with these modifications in the images, as can be seen particularly well in Figure B.4: The U-Nets create a piecewise constant dark structure in the middle, while the data-consistent network keeps it more smooth.



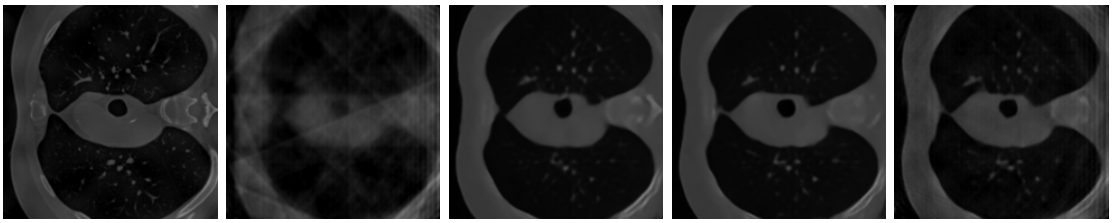
(a) Ground truth (b) Pseudo-inverse PSNR = 34.76 (c) One U-Net PSNR = 25.96 (d) Two U-Nets PSNR = 27.61 (e) Data-consistent PSNR = 33.61

Figure B.4: Sample for which the data-invariant PSNR value is relatively high compared to the U-Net PSNR values.



(a) Ground truth (b) Pseudo-inverse PSNR = 28.43 (c) One U-Net PSNR = 31.50 (d) Two U-Nets PSNR = 31.52 (e) Data-consistent PSNR = 31.47

Figure B.5: Sample for which the data-invariant PSNR value is approximately the same as the U-Net PSNR values.



(a) Ground truth (b) Pseudo-inverse PSNR = 28.62 (c) One U-Net PSNR = 31.05 (d) Two U-Nets PSNR = 31.30 (e) Data-consistent PSNR = 30.89

Figure B.6: Sample for which the data-invariant PSNR value is relatively low compared to the U-Net PSNR values.